Computer-Aided Conceptual Building Design

Kenechukwu C.U. Meniru

A Thesis

in the

Centre for Building Studies

Department of Building, Civil and Environmental Engineering

Presented in Partial Fulfilment of the Requirements

for the degree of

Ph.D. in Building Engineering

at Concordia University

Montréal, Québec Canada

Your file Votre référence
ISBN: 0-494-04053-X
Our file Notre référence
ISBN: 0-494-04053-X

Abstract

Computer-Aided Conceptual Building Design

Kene Meniru, Ph.D. candidate

Concordia University. March, 2005

Decisions made at the early stages of design are critical in relation to the later stages of the building design and construction process. This is becoming more evident as the design process and a variety of building techniques/materials become more complex. Computers are currently ubiquitous in later stages while absent in the earlier design stages which are still dominated by manual techniques. One of the consequences of the situation is the need to translate manual design solutions into a format that can be used by computers at the later stages, thus increasing the probability for errors and incompatibilities.

This research project provides a clear description of the early building design process in a way that makes it possible to formulate basic requirements for successfully supporting the process. These basic requirements are then used to compare with existing computer tools in order to determine what is the current available support. An in-depth study, using the technique of protocol analysis, of eight designers during early building design sessions is carried out in order to identify the features that are needed in a computer environment. Specifications are defined based on this study to guide the creation of a system to support the early building design process using computers.

The research project formulates concepts for computer-supported early building design and uses these to create a prototype that satisfies a subset of the specifications. This prototype is described and then tested in a way that favorably compares it to a manual design session showing superiority in design management and interpretation of ideas while reducing efforts.

Table of Contents

List of Figures

List of Tables

Acknowledgment

IT HAS BEEN a long road and I am glad to say that I have reached yet another significant milestone. For that, I am grateful. Many, along the way, have been instrumental in my arrival. I wish to thank every single one of them and hope that they have gained, however small, from being a part of my life as I have from being a part of theirs. Here are some that I must mention.

It has been very helpful having the assurance from my family especially my parents Augustine and Ada Meniru. They have made my life not only possible but very rewarding in many different ways. My heartfelt wishes.

I have had the good fortune to meet a few professionals that have filled the cracks that would never be accounted for by just a formal education. The bulk of this gratitude goes to doctors Hugues Rivard and Claude Bédard. I sincerely would not have managed quite as well without your teaching, support, advise and friendship. It was with your patience and fortitude that I found it possible to put my ideas to words that matter.

To Günter Schmitz, I wish to thank you for the encouragement to pursue further specialization. It started because you were there.

To life, I have waited a long time just to say, in my present capacity... "Hi"!

Chapter 1

Introduction

DESIGN IS A process that begins with a poorly defined problem or need. It undergoes an early phase that resembles an exploration of ideas in which parts of the problem are interactively modified and updated. In the early days buildings were simple and were realized in one stage – construction. Over the years the discovery of new materials and the need to migrate to new environments forced people to seek new building tools, materials and methods. Buildings and their design became a very costly and complex process divided into many phases. It became critical to find the right tools, materials and methods that will support such an activity and minimize errors in accuracy and coordination of all building aspects.

The exploration of ideas in the early building design process is typically handled by a minimum number of designers [Howard *et al.*, 1989; Larsson and Pope, 1999] and the decisions made form the basis for all work done for the life of the project. The success of the final solution therefore depends on how successful this exploratory phase is and how the decisions made can be easily interpreted or transferred to other parts of the design team [Bédard and Gowri, 1990]. One way of integrating the building design effort so that data is consistent and available is through the use of computers or computer-based systems [Rivard, 1999]. However, few of these systems are designed to support the exploratory nature of the building design process prevalent in the early stages. This thesis will investigate the probability of providing such support.

1.1 The Design Process

The building life-cycle consists of the following phases: feasibility, design, construction, operation, renovation and demolition. Of these, the first two phases represent the moment when the building is conceived and described in detail as is appropriate for its realization [Guthrie, 1995; Rivard *et al.*, 1995]. These phases are defined below.

1.1.1 Feasibility Phase

This phase of the building life-cycle establishes the need for the building, and the owner or representative to seek the services of a designer. A list of requirements is compiled through research, interviews and otherwise appointments with the local building authority. Steps in this phase include:

- Recognition of needs/Programming;
- Budget for the project;
- Assembling the design team.

1.1.2 Design Phase

During this phase, design professionals begin to consider and gather information regarding the site such as site analysis, selection and development including environmental studies and reports. The work in this phase can be divided into the following stages:

- Conceptual [Schematic] Design: Here the designer makes major decisions regarding the building type, form and structural system for the proposed building. This activity is characterized by a flurry of sketches as the designer explores the problem for appropriate solutions [Goldschmidt, 1994]. The goal at this stage is to obtain a design concept [Hernan and Goldschmidt, 1999]. A large number of decisions are made which can result in alternate design concepts. The larger the number of alternatives the more chances for an optimal building solution. The process is rife with speculations and projected ideas with no emphasis on specificity or precision. This stage constitutes the focus of this thesis.

- Preliminary [Development] Design: If there are more than one design concepts resulting from the conceptual design stage, a choice is made which is then developed further. Precise and visually realistic methods are used to portray the building in a finished state. This validates or exposes the decisions made in the conceptual stage and forces an elimination or consolidation action. The goal at this stage is to obtain one primary design solution from all existing alternatives. This elaboration also helps to communicate the design decisions in an easily recognizable format with reasonable precision.
- Detailed Design [Construction Documentation]: A choice is made at the end of the preliminary design stage on whether to go ahead with the selected design concept or not. When this decision is taken, the selected design concept is then detailed for construction. This requires ample details and precision in drawings and specifications of materials and methods for construction including scheduling and costing.

1.2 Industry Trend

Computers have become ubiquitous tools. Many software applications have been developed to do a variety of tasks. One of the earliest computer tools used for drawing can be traced back to the introduction of SketchPad in 1963 [Sutherland, 1980]. As the use of these systems increased, the focus of the software creators remained mainly on the detailed design activities and only recently have incorporated the activities of preliminary design. There is still however a lack of tools that support the conceptual design activity

[McGown *et al.*, 1998; Rivard, 1999].

Recently better access to computers and the proliferation of CAD packages have exposed building professionals to the potential of computer tools for conceptual design especially regarding the ability to interact, record and recall data [Rivard, 1997; Rivard *et al.*, 1998]. This in turn has also underlined the problems of complete integration throughout the building life-cycle as the stages beyond conceptual design are becoming more computerized, thus making the coordination of data used/produced during the design phase a large and complex effort [Marx, 1998]. More importantly however, conceptual design is crucial in the overall design process as all later decisions are based on the decisions that are made earlier during this process and much effort must be made to provide effective tools for addressing the early stage [Bédard, 1988; Bermudez and King, 1998].

1.3 Research Problem

Traditionally, building designers work independently in the beginning of a project and with the help of the client, determine the main building characteristics: massing, orientation and character. This is problematic as invariably the engineers and other specialists get locked out of critical initial design decisions. By the time they become involved in later stages when the architect is possibly locked in a sub-optimal solution, there is less flexibility left to provide remedies to problems that were not accounted for in the beginning.

There is a need for an approach that will allow the designer to work in any chosen condition and provide an environment that encourages the consideration of the necessary items as well as the contribution of other members of the building design team [Akin, 1986; Fazio, 1990; Bédard *et al.*, 1991; Rush, 1986]. Computers present one of the possibilities to achieve this goal however there are still fundamental problems to overcome [Bédard, 1988; Marx, 1998].

On the one hand, computer programmers who build the software that designers use do not generally understand the early design process. This is partly due to the lack of theory or formal representation of the early design process in a form that can be translated to computer software directives [Cross *et al.*, 1996]. On the other hand there are currently no tools that can be used to manage the vague and imprecise manner in which designers work at the early stage. Software systems that are available for the design process are systems that were originally conceived for the later stages of design and therefore do not address the type of data available in the conceptual design activity [Eastman, 1975; Weinzapfel and Handel, 1975; Cross, 1977; Björk, 1999].

The problem therefore is that designers continue to carry out early design processes manually, in relative isolation, which produce work that has to be translated in order to be used in later stages now supported by computers. In addition the process continues to be difficult and costly because consideration must be made for changes that the designer could not foresee without input from the other design team members or the errors that arise from faulty translations or exchange of knowledge [Bédard, 1988; Fazio, 1990; Bédard *et al.*, 1991; Marx, 1998].

1.4 Research Objectives and Scope

This research project will address the problem in three steps. The first step is providing a formal representation of the process so that it is better understood. The second step is an in-depth analysis of the process to identify the required specifications for developing software for early design support. The third step is to use this analysis in the actual design and implementation of a prototype system to show how support for this process can be actually achieved.

Due to the limited resources and time available the research project is limited to residential buildings, the focus is on describing the actions of the designer based on what is happening to the drawings. The intent is not to discover what is happening in their minds or to devise a means to actively support their design thinking process. The intent is to help the management of the process in a way that reduces the effort needed and improves the accuracy and consistency of translations of work from the early stage to all other stages in the building design process.

1.5 Outline of Thesis

The thesis begins in chapter 2 with a demonstration of a typical conceptual design session, followed by an analysis combining knowledge from the sample session with a review of published literature on conceptual design. The outcome of this analysis provides the minimum requirements of early design that need to be supported to help designers. The chapter presents next a number of existing software systems used in the design process in order to ascertain how many of these requirements are currently supported. The chapter ends with a comparison between systems on the basis of the different requirements.

Chapter 3 begins with a presentation of the characteristics required in a tool that can capture the necessary parts of an early design product. It then outlines the methodology followed to understand better the early design process and develop an approach that will make it possible to satisfy the requirements determined in chapter 2.

Chapter 4 describes an in-depth study of eight designers at work on a common conceptual design problem. The study consists of an interview and an observation of the work process (protocol analysis). This produces specific features that should be provided in a software approach for supporting early design. Specifications are also extracted from the analysis of this study to guide the design and implementation of a prototype system in chapter 5. The prototype system is implemented and described in chapter 6 with its main functionalities for supporting early design and fulfilling the minimum requirements observed in chapter 2.

A validation chapter follows where the prototype system is used to solve a similar design problem as that shown in the sample session in chapter two for comparison. Remarks are made comparing the advantages of using the prototype against using the manual design approach at the end of chapter 7. The main contributions from this research project and the recommended future work then conclude the thesis in chapter 8.

Chapter 2

Early Building Design Process

THIS CHAPTER BEGINS with a sample session of the early design process followed by a review of works from published research on this process. The aim is to provide an overview of the current state of the art in order to determine what parts are important when computer support is being considered. Based on this knowledge a list of preliminary requirements is drafted against which a comprehensive review of available design tools is compared. This comparison will then be used to determine the type of support needed at the early design stage.

2.1 Traditional Early Design Session

This section presents a typical early design session. It starts with problem definition and is followed by the design process organized in steps. Each step is documented with a description, a comment and possibly an image of the designer's sketch. The comments try to call attention to certain issues regarding better support of the process.

The design of a building begins with the need for shelter. This need is expressed by the future user or owner of the building who then pass it to the architect in the form of a design brief. The sample design brief specifies the following.

> *A residential house is to be located on a plot of land (specified in the site plan) to accommodate a family of five. Requirements include: 3 bedrooms, 2 bathrooms, living room, family room (preferably on the second level), kitchen, storage areas, outside green (or play) area, a garage for two cars, office with an area for woodwork. The office is to have a separate entrance.*

The architect obtains additional data such as location (site) and budget for the building from the owner as well as other data such as local building codes and ordinances. The information collected gives the designer the basic knowledge about the design problem and how to begin the early design process.

In the following, a sample design session is organized in stepwise format using *Description* and *Comment* sections. The *Description* section contains a narrative of the activity being performed by the designer. The *Comment* section presents observations that hint at the possible support that computers could provide for the activity. The drawing sheets used by the designer are numbered using the format *Dwgxx* where *xx* is a sequence number.

2.1.1 Step 01

DESCRIPTION

The designer needs to collect data for the design process in one easily accessible location to minimize interruptions during the thought process while designing. Some of this data is embedded in letters, interviews and notes. The designer collects all the relevant information in a format that can be easily referenced. The method used here is a listing of the requirements which is pasted on a prominent surface near the work-surface. The designer also uses this list to check the progress of the design as considerations can be numerous and complex. Having the requirements accessible and in a form that can be manipulated facilitating the addition of notes, updating or amendment of items in the list, is important.

COMMENT

A design tool should allow the designer to enter, update, remove and check requirements in a repository. The tool should give direct access to this knowledge so as to assist the designer in tagging or presenting (to a second party) any solved and/or unsolved design issues.

2.1.2 Step 02

DESCRIPTION

The designer identifies parts of the site by writing names. Relevant data from the site plan is extracted onto drawing sheet Dwg01 (Figure 2.1). Notes and marks are made on the drawing to show the boundaries to the site as well as conditions that may influence resolution of the design problem. Some of these conditions are the north direction, the main entrance and any adjacent structures.

COMMENT

For the most part, this process appears to be a standard activity which can be generalized with input of certain data. For example the site boundaries can be automatically defined if the length of the sides and the orientation of one of them is available.

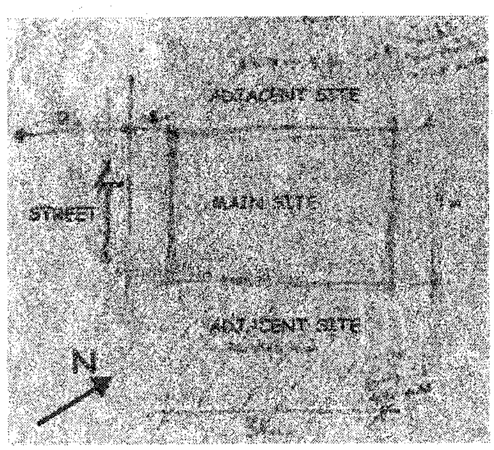

Figure 2.1: (Dwg01) Site sketch

2.1.3 Step 03

DESCRIPTION

Using Dwg01 (Figure 2.1) as an underlay, the designer traces the site on a new sheet to determine the maximum footprint on Dwg02 (Figure 2.2). The maximum footprint is the maximum space on the site that the proposed building can occupy. Many factors lead to its determination such as the presence of mature trees that the clients want to retain but generally, the main determinant is the setbacks. Setbacks are spaces at the perimeter of the site that provide a buffer between the site and its surroundings. Setbacks are regulated by the city as ordinances governing proximity to public or private spaces. Additional spaces such as for driveways and parking may also further affect the size of the maximum footprint.

COMMENT

The maximum footprint influences the designer's concept formulation. A large space may encourage fewer building levels and more open spaces while a smaller space will have the opposite effect. This step shows that maximum footprint can be determined from available requirements such as setbacks and so can be automated.

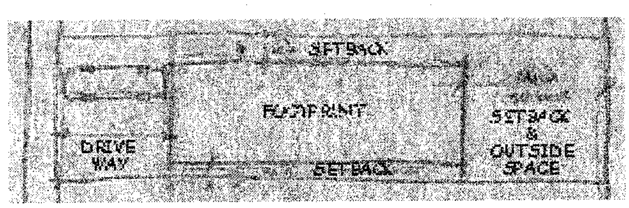

Figure 2.2: (Dwg02) Determining the proposed building footprint

2.1.4 Step 04

DESCRIPTION

Prior to this step, the designer has been working at a small scale similar to the scale of the original site plan. When the maximum footprint is obtained, the designer changes the scale so that the drawing is large enough to make it possible to play with spatial configurations but not so large as to require the resolution of unnecessary details.

Dwg03 (Figure 2.3) shows the initial design efforts as the designer begins work on the first level (ground floor) of the proposed building by configuring the garage. Certain standard measurements called "stencils" are used by designers to guide the configuration of spaces such as car sizes, human dimensions and other items that must be accommodated in buildings. Stencils are not often drawn in detail because it is easier and faster to consider the overall dimensions only (i.e. the outlines).

Minimum parking spaces for two cars are established. Considering the sun path and wind direction on Figure 2.1, doors and other openings are located to establish appropriate circulation. With the garage configured, the designer considers the posted list of requirements as an evaluation of the consequences of decisions so far taken. The designer makes the attempt to locate all required spaces for this floor. Straight flights for the staircase is proposed near the garage. Entrance locations for all parts of the structure are located roughly.

Although an attempt is being made to resolve functional spatial issues, the focus is partly on the shape of the building. An attempt is being made to arrive at an interesting and proportional overall shape of a floor plan – symmetric harmony. The session proceeds with the designer trying to resolve the most influential spaces in the specification, which in this level is the garage. This influence may be determined by either the owner's emphasis in the brief or the designer's priorities. After establishing the garage space, the designer skips the rest of the spaces for this level to tackle the next level. There seems to be no specific reason for this jump except that it seemed rather pressing to resolve the spaces on the upper level in order to complete the connection between the floors. With the connection resolved, the designer will be able to continue more confidently with the other spaces in the current level.

COMMENT

This step shows the use of stencils which allow the designer to quickly reproduce relatively accurate building design items with the right sizes or scales. In using stencils however, the designer pays more attention to the outlines or the length and width information as opposed to the overall form. For example, once the first car was drawn with some detail, subsequent use of the stencil only focused on the overall length and width with or without partially drawing it. The designer stops at intervals to consider the progress of the work by observing the spaces that have been considered. The designer frequently glanced at a visibly drawn north arrow even though this information did not change throughout the design process. The designer often marked-up parts of the design and came back later for a full consideration of the issues surrounding each mark.

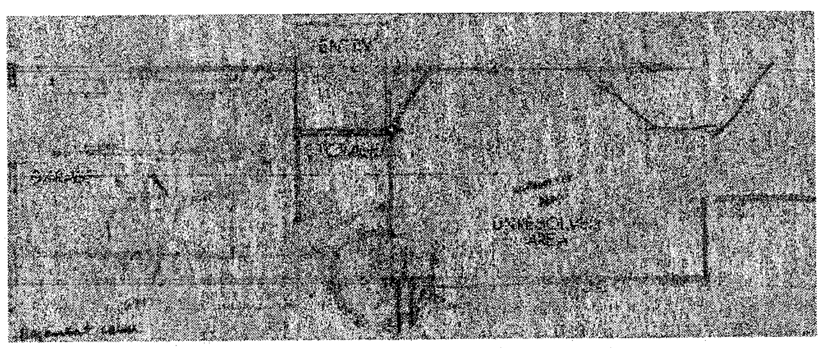

Figure 2.3: (Dwg03) Level 01 - initial layout

2.1.5 Step 05

DESCRIPTION

Using Dwg03 as an underlay, the designer begins the configuration of the second level by transferring the outline of the garage onto Dwg04 (Figure 2.4). On this level the living room and kitchen are roughly located with the labels "public" and "private" respectively on the drawing. The main entrance is also located and circulation around the established spaces is mentally visualized. The designer considers that the main entrance seems to be an unreasonable distance from the public spaces. The use of the straight flight of stairs seems to also provide little help. The current solution is halted and changes are made to incorporate a circular stair structure. More attempts are made as well to pull the entrance away from the private part of the design with little success.

The designer decides to re-examine the entire floor. Dwg04 is removed and posted for visual reference.

COMMENT

This step shows the designer backtracking on two issues. The first is that the staircase design changes from a previously proposed straight flight configuration to a circular one. The second is a lesser example of backtracking in which the designer decides to redraw a floor that was currently being configured in order to reconsider the location of the main entrance.

The staircase change shows the need for the designer to return to a previously decided option in order to reconsider it and make a major change. This step also shows that the design decisions made are not necessarily disposed of. In this way, certain decisions can be referenced or reused even though they were discarded.

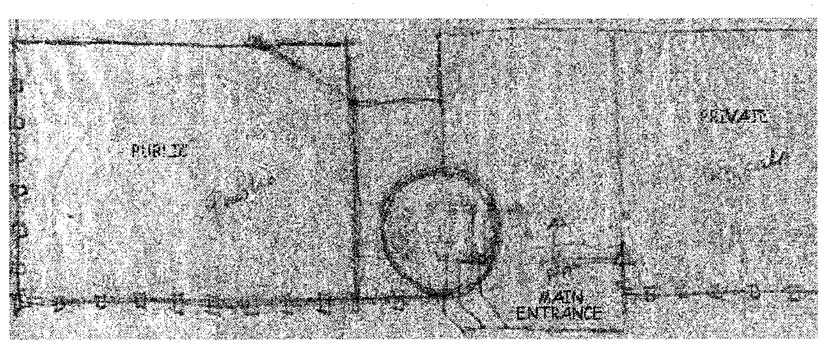

Figure 2.4: (Dwg04) Level 02 - initial layout

2.1.6 Step 06

DESCRIPTION

The designer starts a second attempt at resolving the spaces on the second level using a new sheet numbered Dwg05 (Figure 2.5). The main entrance is moved closer to the public space, the circular staircase becomes the accepted option. The arrow towards the right side of the circular staircase in the figure shows the possible point of egress being considered by the designer. The dining space is located to take advantage of the eastern sun (Figure 2.1) which is typical. More detail is added to the walls which show the use of shading devices. It provides openness while maintaining privacy and shelter. The kitchen space is also detailed by showing its boundary clearly. The designer also shows the intention of providing storage spaces within this boundary.

COMMENT

The use of new sheets over prior determined information allows the designer to collect already accepted ideas while including new ones. Less effort is spent on drawing information that exists. The designer works with simple forms (usually rectilinear or straight edges). These forms are kept simple to ease their manipulation and any consequent changes. During the configuration of circular staircases, the designer needs to know the point of egress in order to properly coordinate the circulation at the head of the stair.

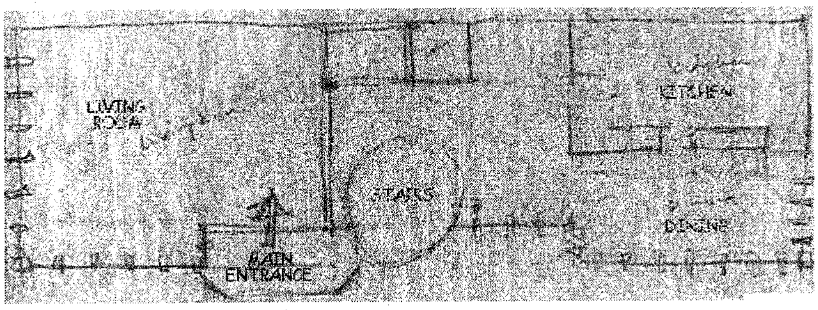

Figure 2.5: (Dwg05) Level 02 - revision 1

2.1.7 Step 07

DESCRIPTION

The designer continues the resolution of the first level. Figure 2.6 shows that the focus of the designer has now shifted to the uncompleted area in the first level. The spaces for office, library and workshop are incorporated into this level. A separate entrance for the office area is provided as specified by the brief. The designer considers issues such as noise from the workshop (which will be used for working on wood) and view or line-of-sight issues coming through the garage or through the office entrance during the resolution of these spaces.

COMMENT

This step shows the designer returning to a previously marked office space in order to resolve the design issues. It also shows the consideration of items such as noise sources and the psychological effect of views in the configuration of spaces.

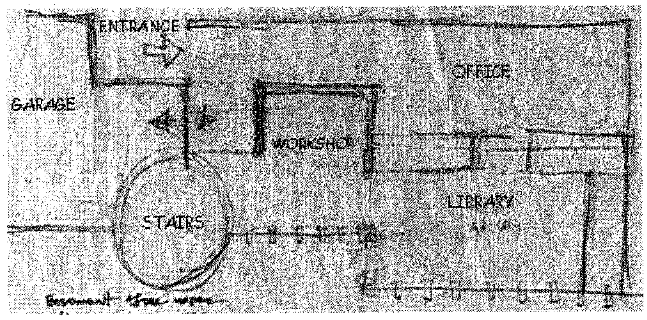

Figure 2.6: (Dwg06) Level 01 - revision 1

2.1.8 Step 08

DESCRIPTION

The designer copies the boundary (external wall) line of the third level from the second level onto Dwg07 (Figure 2.7). The bathrooms spaces are centrally located between the master bedroom and family room spaces. The master bedroom needs a vantage point while the family room needs an airy and visually-rich atmosphere. Both need separate bathrooms and an access to the stairs. At this point, the designer feels that levels 1 to 3 will significantly influence the fourth level (which will consist of the rest of the bedrooms). So there is a need to resolve the design elaborated so far using more detail before proceeding to the fourth level.

COMMENT

At the beginning of the design session the designer drew design items with little attention to the use of precision drawing methods but now, there seems to be an increase in the effort to draw more carefully and to measure and pay attention to the scale of items such as the thickness of walls and the sizes of doors/windows.

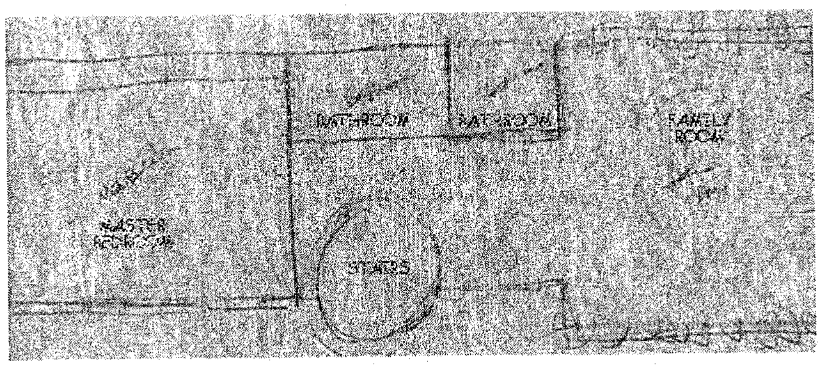

Figure 2.7: (Dwg07) Level 03 - initial layout

2.1.9 Step 09

DESCRIPTION

More precision in the drawings is used to provide a higher level of accuracy in the location and proximity of spaces. The designer traces level 01, Figure 2.6 on Dwg08 (Figure 2.8) paying attention to wall and door dimensions. Stencils are used while door and window sizes are shown as well as other relevant furniture. The aim is to check and make sure that adequate sizes for the spaces have been provided. The designer decides to reconfigure the first level around the workshop area.

COMMENT

When the resolution of design functions appears to be satisfactory, it is important to check the spacing and sizes so that there will be no significant changes later when more detail is added. Due to the manual effort involved in redrawing all items however the designer only concentrates on the main areas. In this case the designer redraws only the second level.

2.1.10 Step 10

DESCRIPTION

The designer rearranges the office to open up the woodwork space in Dwg09 (Figure 2.9). The goal of this step is to isolate the woodwork space from the rest of the first level

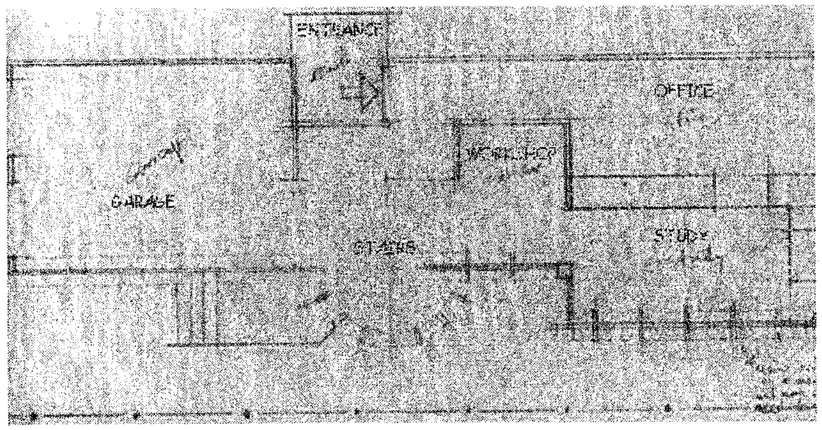

Figure 2.8: (Dwg08) Level 01 - revision 2

area. This is important as the woodwork area needs good ventilation for venting dust and fumes as well as possible noise reduction.

COMMENT

The designer uses several sheets of drawings to solve the design problem, an example of which is the separate sheet used in this step. There are two main reasons both of which depend on the transparent characteristic of the drawing sheets. First the transparency makes it possible to copy previously drawn items onto new sheets without having to redraw. This includes selectively copying parts that are needed while ignoring others. Secondly, manipulation of the drawn items is easy as the sheets can be combined in layers to show or present different solutions. However managing these sheets is tricky because they are transparent and easily destroyed. Therefore, although the early stage of design is critical and should be saved for many reasons, the designer often depends more on later stages beyond early design to record and preserve design decisions.

2.1.11 Highlights of the Design Session

The designer starts the early design process by collecting requirements for the design in a way that makes references to such requirements easy during the design process. The site is manipulated in order to extract the maximum footprint. This is accomplished by subtracting the setbacks and considering other site features all of which are obtained

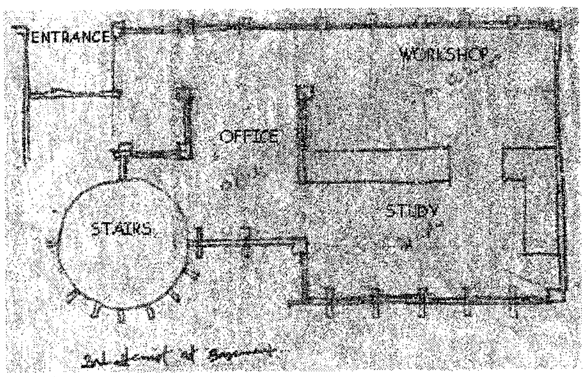

Figure 2.9: (Dwg09) Level 01 - revision 3

from the design requirements. The designer creates and manipulates simple forms or line figures which minimize the effort and increase the speed used in manipulating and editing them during the process. The designer does not necessarily complete all issues that affect a particular area before moving onto another. Usually marks (a combination of lines and labels) are created on these unfinished parts so that a return to them for further analysis is made easier. A return to these unfinished parts usually occurs when the designer pauses in the process for assessment. At these times, the requirements are often referenced as well as the north direction (orientation of the site) which has been visibly marked for such purposes. The designer often labels on the drawing sheet to show items that need to be considered for a variety of issues such as noise source, or to show items that have been considered or addressed such as spaces.

The use of measurements and other means of accurately scaling design objects is often avoided. Instead, the designer makes use of stencils which are predefined standard forms of design objects. Although the design starts with little attention to detail and accuracy in the forms created, the designer often tries to redraw certain important parts of the design towards the end of the design session. This is in preparation for the presentation of the ideas being proposed but more importantly it is to assure the designer that no glaring mistakes have been made about spatial arrangements so that no major changes will have to be made. Changes may undermine the functional resolutions achieved in the early design process and may lead to the rejection of the design solution.

All design items created and manipulated in this stage of design are organized using transparent sheets. These sheets make it possible to limit the amount of repetitive drawing actions because the designer places them over each other to add new or altered information without having to redraw previously drawn items. Also due to this transparency, it is possible to combine several drawings in layers to make a solution.

2.2 Review of Early Design Process

Bearing in mind findings from the previous section, this section reviews research data from published sources in order to further understand and compare the process as it affects most designers. The majority of publications reviewed are analyses and reports of findings from protocol studies of designers' sessions. Protocol study/analysis is an empirical, observational research method for understanding and analyzing design activity. It has become regarded as the most important method to bring out the "mysterious cognitive abilities of designers" [Cross *et al.*, 1996] as it relies on the actions as well as the verbal accounts of the designers as they describe their cognitive activities. Although verbally describing an activity while doing it could be seen to be a problem, as it may change the designer's behavior or cognitive performance, it is difficult to imagine how else to examine what is going on in a person's head without asking [Cross *et al.*, 1996]. In the next paragraphs, notes have been included where appropriate in brackets, that connect the reviews with particular findings from section 2.1.

Building design represents a complex process that succeeds when concerns from different specialty areas are addressed to produce a building solution [Howard *et al.*, 1989; Bédard *et al.*, 1991]. Traditionally, building design begins with an isolated person or office (architect) whose decisions set the basis upon which all other decisions are taken. The early stage of this process is crucial, being the foundation for all other activities. Its success is determined, to a large extent, by the availability of information from other participating members of the building team [Bédard and Gowri, 1990]. Successful early design processes are characterized by a constant clarification of requirements, on-going prioritizing of requirements at each stage of the process (as seen in Step 01), investigation of ideas without becoming fixated to a particular solution and production of design variations. All this is accomplished while the designer maintains an overview of the different parts of the design in order to be able to identify successful alternatives [Fricke, 1999].

Building designers need to produce numerous drawings which are used in clarifying the characteristics of the design solution, store ideas and reveal the mechanics of the design thinking process [Atman *et al.*, 1999; Casakin and Goldschmidt, 1999; Dörner, 1999] (this can be seen in the number of sheets used in the sample early design session). In particular, the sketch is perhaps the most important item [Lipson and Shpitalni, 2000] during early design and allows associations among hidden meanings in the designer's imagination. Its use enhances the designer's imagination because it captures abstract and unstructured pictorial representations of early design ideas [Purcell and Gero, 1998]. Its rough, simple and unfinished state invites the designer to forge ahead with the search for a design solution (as shown in the use of single-line forms at each step of the sample session). The use of simplistic drawings can be seen in specific studies called "protocol analysis" where the designer is observed and recorded while drawing [Goldschmidt, 1994] or iterating between sequences of externalization and criticism/analysis [Casakin and Goldschmidt, 1999; McGown *et al.*, 1998].

Externalizations are reinterpretations of images or ideas in the designer's mind. During externalization the designer sketches mostly without regard for technical details while during criticism/analysis the sketch is reviewed as the designer restructures or reinterprets their mental idea/picture [Gero and McNeil, 1998]. This continues in the form of a dialogue between the designer and the sketch until the designer moves on to other areas in the sketch [Purcell and Gero, 1998; Goldschmidt, 1991]. When this move occurs, the solution for the present problem area is not necessarily complete or satisfactory. The designer keeps an iterative development process, often leaving behind partial solutions to return to at a later time [Heylighen *et al.*, 1999] (this is illustrated in the change of focus from the second level to the first level in Step 05).

Although the designer's sketch may be thought of as consisting of a hierarchical arrangement of components, such change of focus leaves the impression of the design process as seemingly jumping from one point to the other, resembling an uncoordinated sequence that is very difficult to predict [Atman *et al.*, 1999; Cross, 1977, 1998]. This unpredictability applies even to seemingly finished sketches as the designer may come back with changes at a later time (this is illustrated in the return to the first level in Step 09 as well as the decision to redo the second level in Step 06. As the designer attempts to finish the design by adding detail to the first floor, it suddenly became necessary to make some changes to the office area). This behavior of jumping from one point to the other can be attributed to the designer's need to subdivide the problem into many different sub-categories, which are then examined at different levels i.e. from high-level overall views to low-level details of the problem [Gero and McNeil, 1998].

As the designer works, sketches undergo duplications and transformations [McGown *et al.*, 1998; Goel, 1995; Verstijnen *et al.*, 1998; Rodgers *et al.*, 2000]. **Duplications** allow the designer to reproduce ideas or sketches already existing in or outside the workspace (as illustrated in Step 08 where the designer copies the external envelope from a different building level). There are two types of transformations—vertical and lateral. **Vertical transformation** occurs in successive detailing of design data while **lateral transformation** allows for the creation or development of new data (step 04 illustrates vertical transformations as the designer develops the first building level by adding the garage, storage and staircase spaces. Step 05 shows an example of lateral transformation when a shift is made from the design of a straight staircase to a circular one).

Designers stimulate their imagination by viewing the design problem from different angles and possibly using different drawing renditions and types such as elevation, sections and 3D, sometimes simultaneously. In this way, emerging solutions can be extracted but quite as importantly, the design problem is thoroughly investigated for various possible solutions [Cross, 1998]. A greater understanding of the requirements, and consequently of the design problem, is thus attained [Günther and Ehrlenspiel, 1999].

One of the important characteristics of early design is the pursuit of more than one solution [Fricke, 1999]. This may happen consecutively or concurrently. It is not just a way to expand the possibilities of a successful solution but also to introduce 'spare parts' from which the designer can consolidate and refine desirable solutions into a single 'best' possible solution for that particular design session (for example in the change from straight to circular stairs, some knowledge which is common to both items will be reused such as riser and thread sizes).

2.3 Requirements for Support

The description of the early building design process from the sample design session and the review shows that the main item created and manipulated by the designer is the building space. This section provides a summary of the important issues that should be addressed when considering a computer-based approach for the support of early building design.

2.3.1 Interface Issues

The drawings created and manipulated during early design consist of simple graphical forms that represent building spaces. They are kept simple because it must be fast and easy to create and discard them as many times as necessary without much loss in the design time or effort. The production of numerous drawings is normal according to the literature review and the sample session shows an instance of redrawing a floor just so that the designer can think clearly through a problem of properly locating the main entry point for the second level in Step 06. The literature review makes it clear that the sketch is a very important part of the early design process. It is used to both understand and propose design solutions. There is a need to create a comfortable interface that will make it possible for the designer to concentrate on designing (the items being created) and not the drawing of them. The knowledge and the steps or actions required by the designer to effectively draw must be simple and intuitive [Raskin, 2000].

2.3.2 Design Item Recognition

The designer draws at the early design stage with little attention to specifics or detail in the depiction of design items. The review likens this to a dialogue between the designer and the sketch which consists of externalizations (by the designer) and reinterpretations (in the designer's mind) based on what the sketch represents. The outcome is not always predictable and reinterpretations are not always consistent. When there is a problem with the reinterpretation of an item, it can produce a conflict that can jeopardize the success of the final design solution [Fazio, 1990].

One way the designer provides reasons for creating design items is through the use of labels. This is shown in the sample session where the designer labels spaces that have been created using items from the list of requirements. The reasons behind the creation of a design item, such as a space, may not be final and may change later but should be interpreted consistently. Labels provide a reasonable solution but may require further explanations from the designer as labels may mean different things to different people and may not portray all the facts about the decision. For example in Step 06 the designer shows a proposed egress point on the circular staircase with an arrow. This is a crucial point in the designer's consequent design decisions for the rest of the floor, however this fact may not be correctly interpreted without the designer's explanations.

The digital environment presents advantages beyond simply labelling the spaces. It can understand the roles implied by the labels. In this way, it will provide assistance by offering additional information and reacting in certain ways to minimize the effort and time spent by the designer on drawing actions in the design process. In the sample design session the designer redraws the second level (in Step 08 and Step 09) in an effort to show wall thicknesses as well as sizes of doors and windows. The action of redrawing these walls in this fashion is redundant. The designer's energy would have been better spent on deciding whether the wall thickness is adequate. All walls have a certain thickness so the creation of a wall should imply a certain default thickness even though it may not be specified at the time of creation.

Design item recognition allows some knowledge to be saved within the item created at the time of creation so that its role and its interpretation is helpful and consistent throughout the design process.

2.3.3 Collection of Alternatives

The creation of more than one solution is essential in the early design process because it provides choices for the designer. The creation of more than one solution encourages the designer to test them and to be able to make more informed decisions. This is shown in the sample session when the designer creates a second circular staircase in Step 05, and further substantiated by the literature review which also gives clues to how alternatives can be captured. Sketches undergo duplications and transformations that create different versions of ideas and design components. These versions combine to form solution paths. Figure 2.10 illustrates two solution paths. A solution path is the combination of all design steps and items that are created in a vertical transformation (illustrated by combining "House", "AP01" and "AP02" in Figure 2.10). The second solution path is initiated by the lateral transformation from "AP01" to "SP01" which represent the transfer of design information such as number of steps and riser height. Lateral transformations show the creation of two parallel ideas that derive from the same parent ("House") but represent different ways of solving the same problem. Duplications present a means of reusing design items that have already been produced in the design process. This is demonstrated in the sample design session in Step 06 when the designer tries a second time to resolve the location of the entrance to the second level and is shown in Figure 2.10 by the diagonal arrow from "AP02" to "SP01". Tracking when transformations and duplications occur provides an early design tool with the opportunity to capture the alternatives created by the designer in a design session. This makes it possible to provide support by allowing the browsing of such captured solutions as a means of review and comparison. It also makes it possible to combine parts from different solution paths to form the most desirable solution for the design problem.

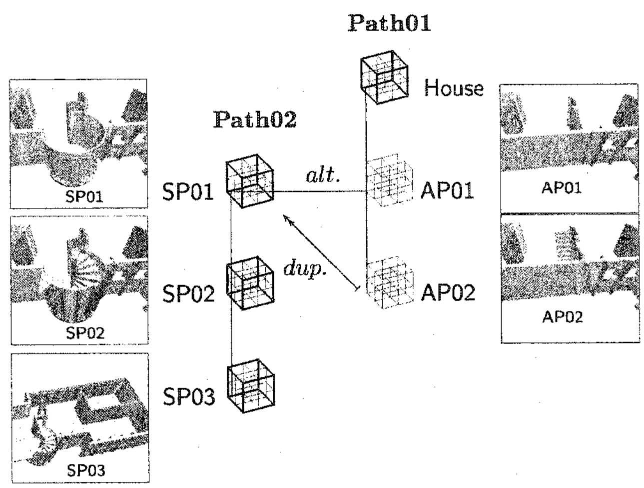

Figure 2.10: Solution paths

2.3.4 Design Decomposition

The designer's sketch can be seen as a hierarchical arrangement of components (see section 2.2) used to subdivide the work into many different sub-categories for easier management. This gives the designer a unique control and view of the items in the design solution in such a way that it is possible to seemingly jump from one item to the other according to where the opportunities present themselves for possible solutions. The designer is then able to move back and forth examining the design in context (high-level overall views) and in isolation (low-level details). The sample session shows an example of this type of move from an overall view to a detailed view in Step 09 and Step 10 when the focus went from the second level to the reconfiguration of the office space in the first level.

The use of hierarchies provide a natural representation of the connections between all sub-problems composing the main problem. Hierarchical decomposition of a building design represents the relationship of the items that make up the design [Rivard and Fenves, 2000]. A space is the main design item used by building designers during the early design process. When created, a space is an area in the sketch that is demarcated for a certain use. Spaces have parts as shown in the lower levels of the hierarchy in Figure 2.11 and

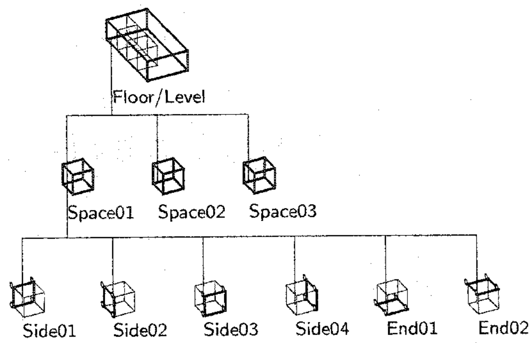

Figure 2.11: Decomposition of a space

can be aggregated to form floors or levels as shown in the upper tree items in Figure 2.11 and Figure 2.12.

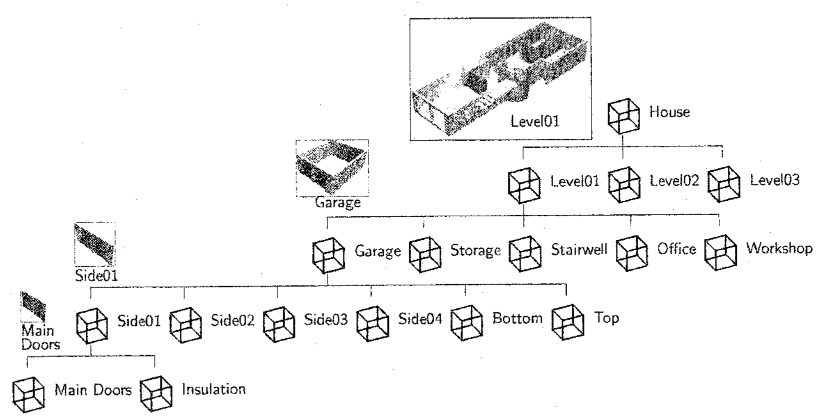

Figure 2.12: Decomposition of a floor

In turn, levels can be aggregated to show the building as shown in Figure 2.13. Use of

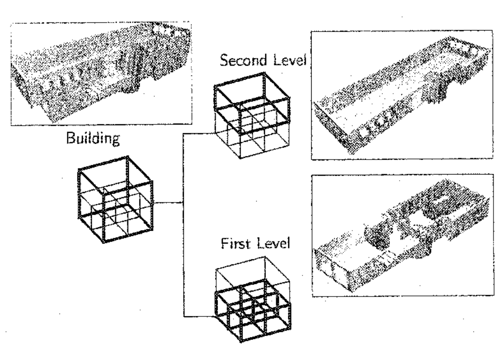

Figure 2.13: Decomposition of a building

hierarchies facilitate multiple views of the sketch which provides ample opportunity for interaction. This increased interaction as stated in the review in section 2.2, stimulates the designer's imagination to produce creative solutions for the design problem.

2.3.5 Knowledge Integration

The literature review explains that decisions made at this early stage provide a framework upon which all other decisions must be built [Cross *et al.*, 1996]. Therefore the process benefits when there is input from all professionals concerned regardless of the fact that the designer is often solely involved. This is often a problem because, due to the complexity and commitment in resources necessary in the early design process, designers seek to isolate themselves as much as possible in order to limit the interruption and inconvenience that comes with consulting other professionals or stopping the process in order to do additional research. An example of a situation that would warrant consultation or additional research is demonstrated in the sample design session during the configuration of the office space. The extent to which the spatial organization and structural composition in this space will change or conflict with later design proposals is unknown at the end of the design session. This is because the space for woodwork will likely produce ample noise, vibration and dust, none of which was considered during the creation of that space. This may not be the case for another designer but no single person is specialized to address all necessary aspects of every building.

To avoid this difficulty computer tools can assist by providing some knowledge. Computers present a unique environment in which knowledge can be stored and used in interactive ways. There is a need to integrate various types of design knowledges especially engineering rules-of-thumb that the designer can use in order to make acceptable decisions that provide a functional framework for the diverse building design process.

2.3.6 Summary of Requirements for Support

The sample design session and review of the early design process make it possible to draw requirements that must be considered in order to support the early building design process. The designer requires a simple and intuitive interface that does not take away focus from the design as opposed to the drawing actions. Design items should be recognized in such a way that their role in the design is correctly interpreted and applied consistently. The solutions created should be collected with little disruption of the designer's focus on the design activity. These solutions should be made available for review by the designer and possibly manipulated towards a preferred single solution. Decompositions should be available in various degrees (decomposition of the building, the floors and the spaces) so that the designer has a complete overview of the design solution as well as fast and easy access to its constituent parts. To assist in an early design process that provides a reasonable consideration for the set of diverse issues involved, there is a need to make more knowledge available that allows the designer to try "what if..." scenarios and to anticipate correctly the outcome of decisions that are being made.

With reference to these requirements, the next step in this thesis is to review available computer-based tools that provide assistance in the early design process to assess if the tools have taken these requirements into account.

2.4 Review of Existing Computer-Based Tools

Computers constitute an important component of the designer's chest of tools for building design but the assistance provided by computers so far has focused mostly on the later stages of the building design process. Computer-aided design tools were first introduced in the 1960s [Cross, 1977]. These tools were mostly rudimentary 2D drafting systems that did not offer substantial advantages over traditional manual methods [Novitski, 1998]. With the introduction of solid modeling, the method of design using these tools became complex and quite different from the normal processes used in the professions. However, due to the precision, information management and excellent rendition available in using these tools, building designers have endured or ignored the difficulties [Meniru and Schmitz, 1996]. This has ultimately reduced efficiency and discouraged further progress in developing appropriate methods for using computers effectively in building design [Bhavnani *et al.*, 1993].

This section presents a comprehensive review of available computer-based tools in an effort to discover what support is available for the design of buildings. The systems discussed do not represent an exhaustive list. The majority are commercially available and therefore accessible but others are academic or experimental systems that were not available for a full assessment. Some of the commercial systems are reviewed after substantial use in actual design sessions while others are reviewed based on a combination of a brief period of testing and access to published reviews from users. Most of the academic or experimental systems were not available for testing. As a result their review depends solely on published information or materials available at their web site.

This section reviews existing tools for computer-aided building design in six categories described below. In each category, the first section presents some general features available in the systems being reviewed in the category, then each system is listed with its unique features. In the end a section summarizing how the category fulfills the requirements of section 2.3 is provided.

- The category of "2D Systems" describe tools that assist in the creation and manipulation of 2D drawings, often emulating the capabilities of traditional pen and paper (section 2.4.1). 2D systems provide limited assistance in the type of support this research is proposing however many designers prefer to use these tools because they are relatively simple drawing tools with a relatively negligible learning curve. These systems are therefore reviewed in an effort to capture this simplicity in a drawing interface.
- "3D Systems" describe 2D tools with added capabilities for solid modeling and animation (section 2.4.2). Solid modeling is a term that refers to the addition of

heights to 2D (length and width) information.

- "Integrated Systems" describe computer tools that work with a building model (section 2.4.3). A building model captures all the information that describe a building, not just the geometric information. These systems are reviewed because they reduce the designer's drawing effort by automating the management of the drawings produced from the 3D models.
- "Virtual Reality Systems" describe tools that allow total or partial immersion of the designer in an electronic world where the design process occurs (section 2.4.4). These systems offer a new and interesting approach to working with computers.
- "Generative Systems" describe tools that accept certain parameters and automatically generate design alternatives to be reviewed by the designer (section 2.4.5).
- "Interactive Design Exploration Systems" describe tools that integrate recognition of building components in order to provide more relevant coordination of the design process in comparison to other reviewed systems (section 2.4.6). Such systems present an environment that recognizes the roles of the design items.

The last three categories describe tools that are mostly in the development stage and are not yet commercially available. In general, an attempt has been made to include at least two systems in each category in an effort to provide variety in the available capabilities.

2.4.1 2D Systems

2D systems allow the designer to create, edit and manage two dimensional objects. Sample tools in this category are SmartSketch by Intergraph, AutoCAD LT by Autodesk, QCAD by RobbinSoft.com and DESI-III [Mariën, 2003] which are all commercially available systems. These types of systems were the earliest ones introduced for assisting the drawing process. They emulated the traditional pen and paper on a drafting board but offered, in addition, several advantages such as consistency and accuracy in the drawing process. With such tools the designer is able to maintain a consistent look and style to drawings with the use of layers, line widths, line styles and editing functions such as copy, move and trim.

The possibility of attaining very accurate drawings is perhaps the most compelling reason for the use of such systems. Accuracy in measurement and placement of drawing items can be achieved with up to eight decimal places in any unit of measurement. Location of points, construction of lines, arcs and text as well as the ability to automatically provide dimensions are available including filters that make it possible to snap to exact positions or to measure exact sizes of drawn items.

2.4.1.1 SmartSketch

SmartSketch [Intergraph, 2003] is a commercial tool that uses modules to provide 2D functionalities. These modules are basically work-flow templates incorporating the appropriate standards, settings and symbols for different professional domains. Users can install some or all the modules.

The user interface is shown in Figure 2.14. Choosing the architectural template creates a default standard sheet and symbols for architectural drawings such as doors, windows, structural elements, stairs, furniture, etc. There is a number of 2D standard tools for drawing, modifying, labelling and dimensioning which are easy to use. For example to draw a line, the user starts at any point, moves in the desired direction and keys in the dimension and an angle if necessary.

2.4.1.2 AutoCAD LT

AutoCAD LT is a commercial 2D tool based on the mature AutoCAD family of products [Khemlani, 2003e]. It is a light version of AutoCAD (see section 2.4.2) without the 3D capability. It shares the same file format as AutoCAD and so is fully compatible with it. LT has limited customization capabilities, it cannot take advantage of the large number of third party applications for AutoCAD, and it does not allow the modification of layer attributes.

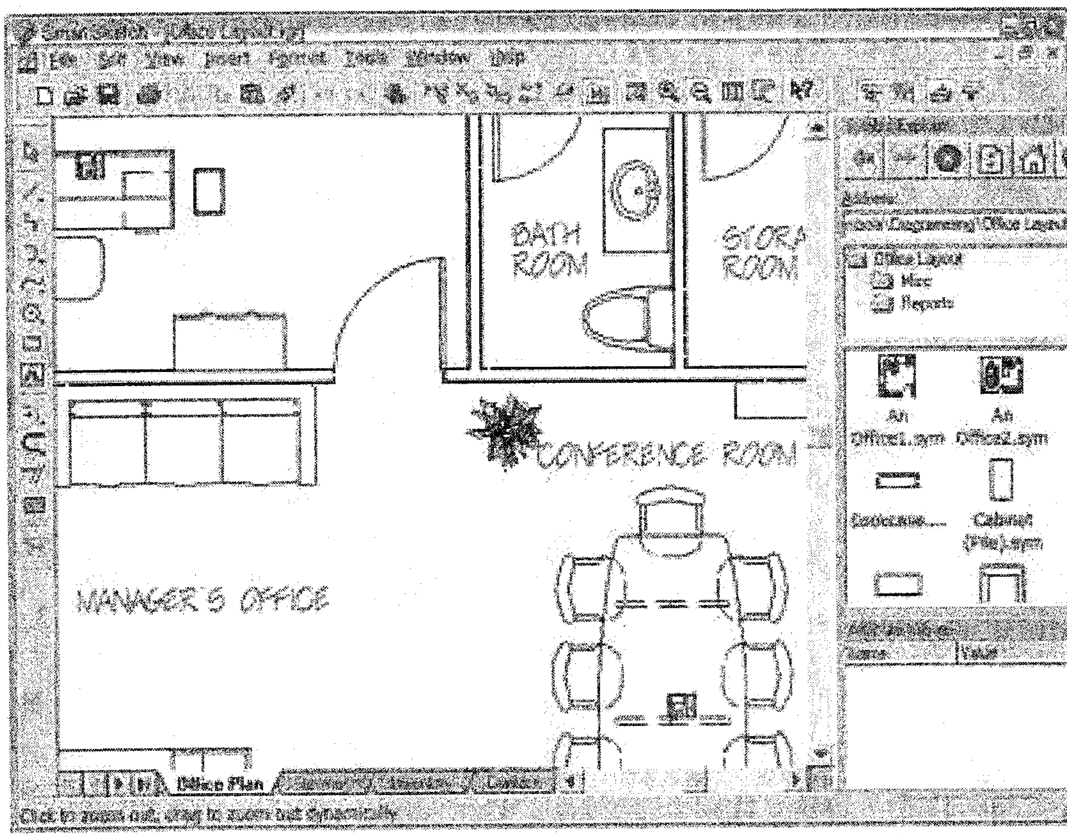

Figure 2.14: Intergraph SmartSketch
www.cadinfo.net/reviews/SmartSketch-3.htm

2.4.1.3 QCAD

QCAD, shown in Figure 2.15, is a commercial system that was extracted from a computer-aided manufacturing (CAM) system after the realization that the CAD functions embedded in the CAM tool were becoming too complex [Mustun *et al.*, 2003]. QCAD works with other CAD tools by exporting to popular file formats such as DWG and DXF.

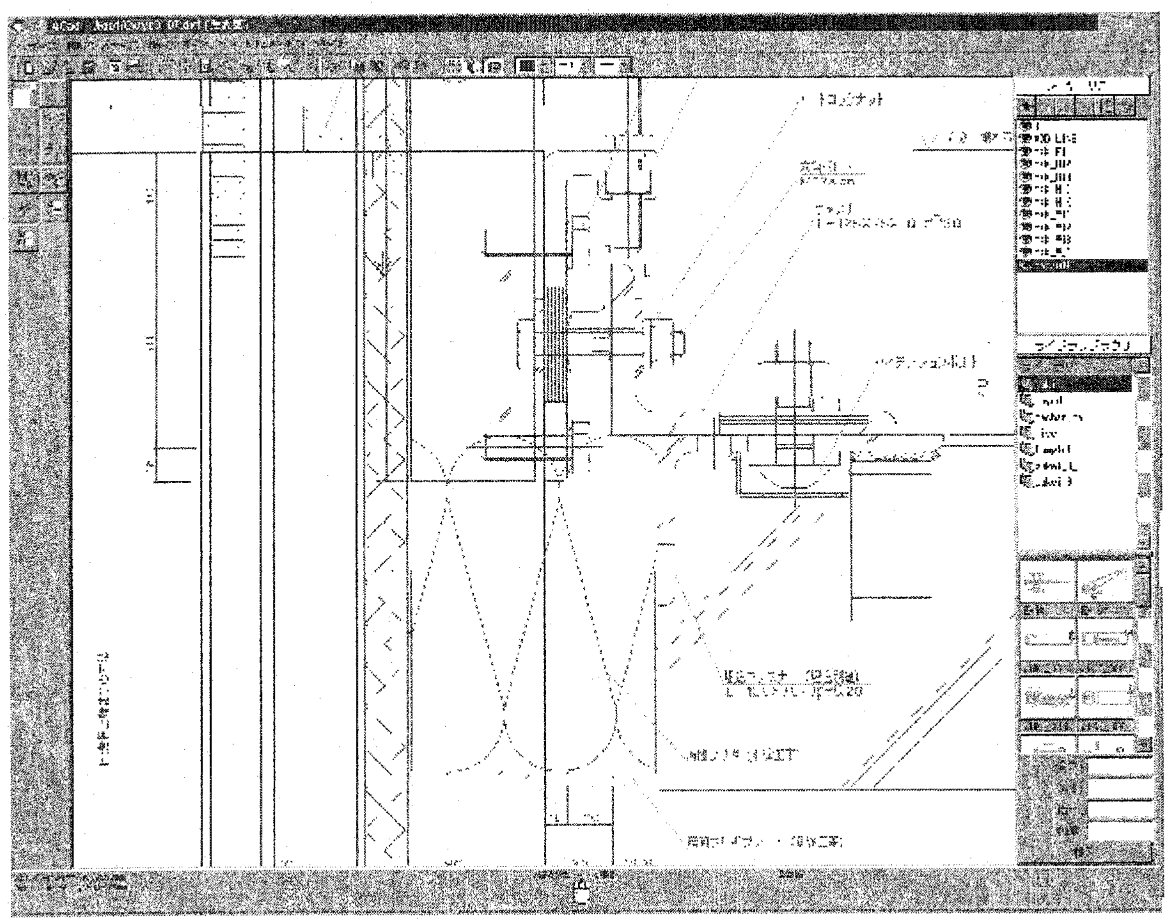

Figure 2.15: QCAD interface
www.qcad.org

2.4.1.4 2D Systems: Summary

2D systems provide an interface for handling simple forms, thus enabling the designer to manipulate these forms for the purpose of designing but the steps required to create them are too many and the precision required is too detailed to be useful in the early design process. These systems are used typically to refine early design solutions. There is no provision for identifying design items so roles cannot be determined. There is no support for the collection of alternate design solutions or the decomposition of the design in an effort to provide an effective overview. Due to the general lack of support for the early design process there is no applicable knowledge available in these systems either. Table 2.1 presents a summary of 2D systems.

Table 2.1: Summary of reviewed 2D systems

Keys: √ Substantial support | ⊙ Some support | – Limited support

	Type Of Support				
System	Interface Issues	Recognize Roles	Collect Design Alternatives	Design Overview	Knowledge Integration
2D Systems					
SmartSketch	⊙	–	–	–	–
AutoCAD LT	–	–	–	–	–
QCAD	–	–	–	–	–

2.4.2 3D Systems

3D systems improve on the capabilities of 2D systems by adding the ability to draw in the third dimension, to animate drawn objects and to render these object in a realistic manner. They are complex tools with regard to user interface and the variety of functions available, thus are relatively non-intuitive and require some dedication in their use. Typical systems in this category are Architectural Studio, AutoCAD and Autodesk VIZ by Autodesk Inc, form*Z by auto*des*sys, SketchUp by @Last and SKETCH.

They provide a means of importing images and text files to be used as overlays or references. They provide grids for guiding the placement of design items. Digital tools that emulate pencil or color pencil are available in conjunction with the ability to draw lines (free and constrained), rectilinear and rounded shapes. To satisfy solid modeling capabilities they provide a vast range of options some of which are for the creation of geometric primitives such as boxes, spheres, cones and cylinders and surface modeling tools such as NURBS (Non-Uniform Rational B Spline) for creating uneven surfaces such as that for organic forms. To round up creating items in 3D, these systems provide a range of tools for rendering and illustrating drawings such as tools that emulate realistic lights, textures and shadows. Special effects such as illumination, reflection and atmospheric effects such as fogs are also supported. These tools provide the ability to create complex shapes using constructive solid geometry (CSG). This includes three main techniques, the combining of primitives called a *union*, the subtraction of one or more solids from another called a *difference* and the carving of a common part of one or more solids called an *intersection*.

All of these tools provide a form of realistic rendering and animation technique for completing the 3D design and presentation. The following sections present the systems reviewed with their unique capabilities.

2.4.2.1 Architectural Studio 3

Architectural Studio 3 is a commercial tool that provides transparent windows for creating sketches or models. It is a system from the makers of AutoCAD that is meant to cater specifically to the architectural design process. The interface is shown in Figure 2.16 [Khemlani, 2003c]. Architectural Studio 3 tries to recreate the architect's physical desktop by providing a workspace in which 2D sketches and 3D models can be created. In addition, other drawings, images, text files, etc. can be imported and used as overlays or references. It provides a background grid and a set of tools categorized in four palettes. It allows the designer to draw freely on the screen which is closely modeled after the manual design drawing action.

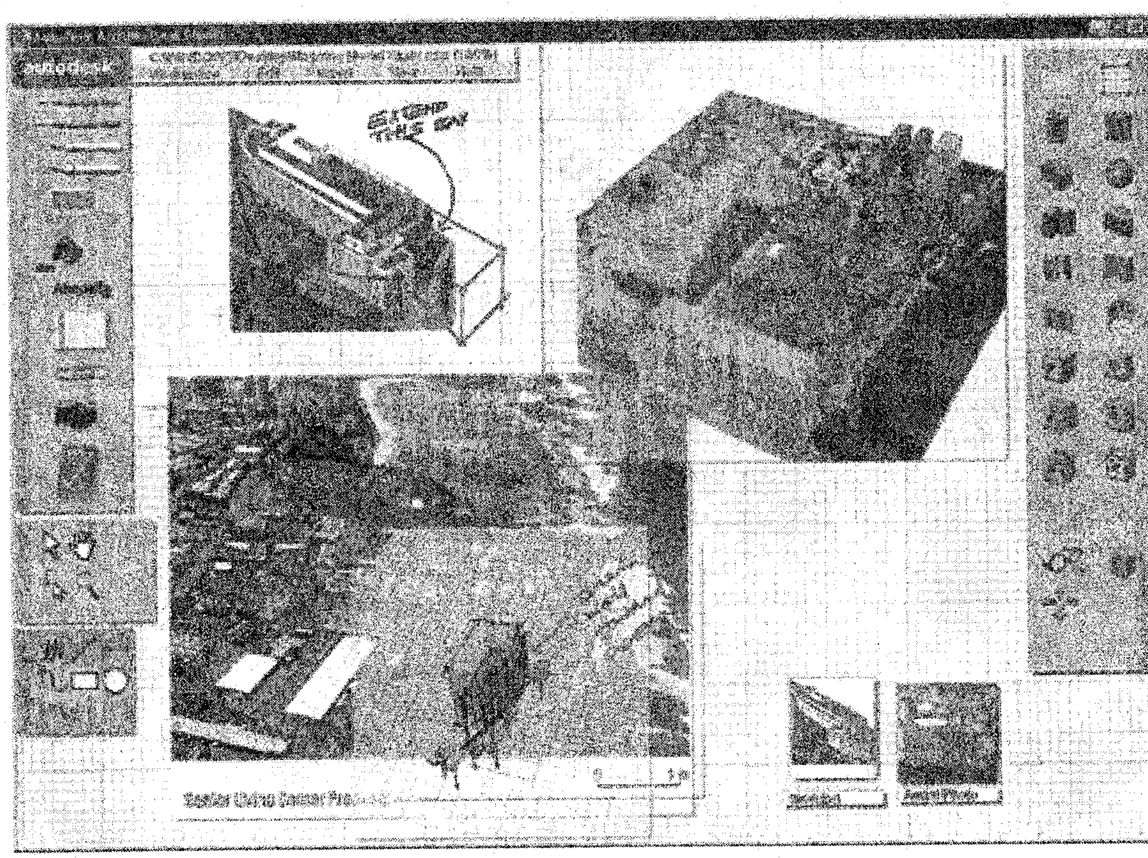

Figure 2.16: Architectural Studio interface
cadence.advanstar.com/2003/0503/pr0503_archstudio.html

2.4.2.2 AutoCAD 2004

AutoCAD is a commercial general-purpose 2D CAD system with added capabilities for solid modeling [Khemlani, 2003d]. The main mode of entering command into AutoCAD is through its command line interface even though in its latest version, the interface has been revamped to make it look more modern with, for example, more sophisticated icons as shown in Figure 2.17. AutoCAD files do not maintain backward compatibility with

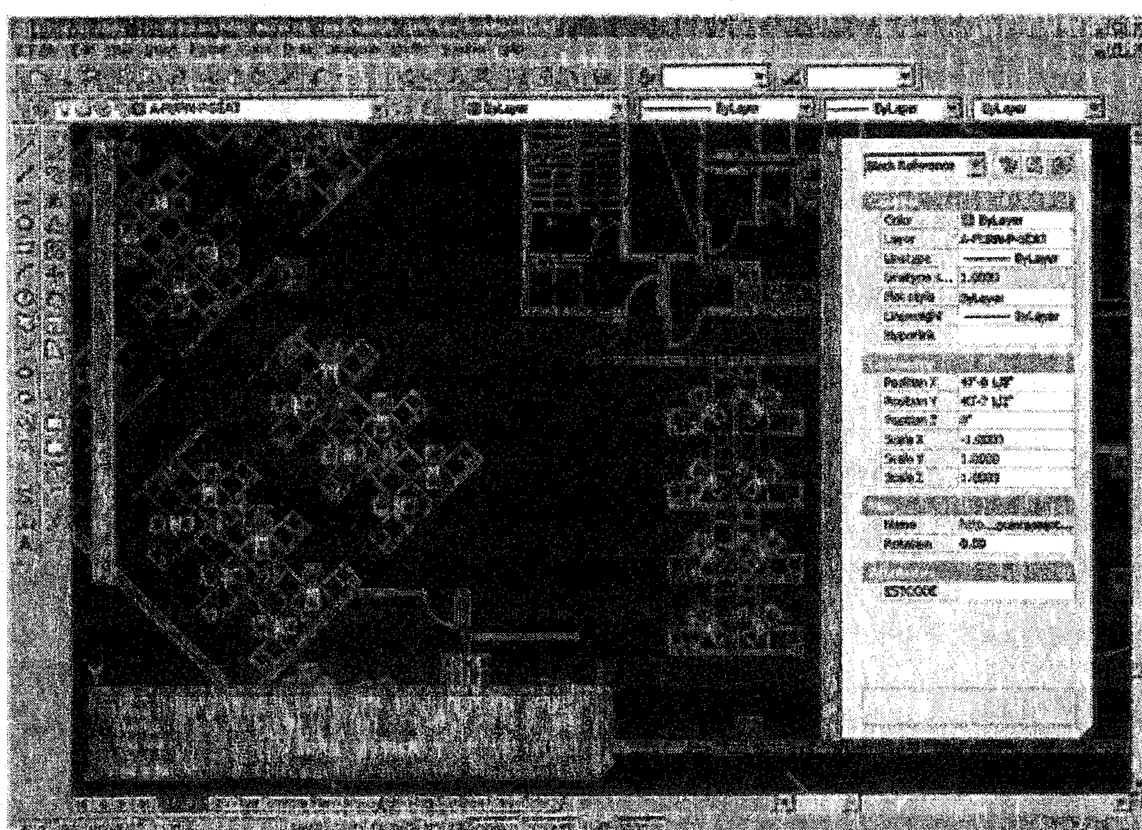

Figure 2.17: AutoCAD interface
cadence.advanstar.com/2003/0603/fr0603_autocad.html

prior versions of the software. The latest version of the drawing file called DWG file has been significantly changed to increase it efficiency and capability. As a result, it cannot be used by older versions of AutoCAD. It is however possible to save as DXF (drawing exchange file) which may cause some loss in drawing information but will make it possible for files to be opened in older versions. AutoCAD allows for drawings to be electronically signed and saved with passwords. As electronic copies of drawings are being accepted in bids, permits and tenders, this fulfills the need to authenticate drawings. AutoCAD has a difficult-to-use 3D functionality because there are many commands brought over from its older 2D only version and do not present an intuitive 3D environment. The focus, it seems, is still on using 2D capabilities.

2.4.2.3 AutoDesk VIZ 4

Autodesk VIZ is a commercial tool that started out as 3D Studio which was developed primarily for animation and video graphics [Khemlani, 2003a]. As AutoDesk VIZ, it has been specifically tailored as a visualization and design solution for architectural, industrial and product design. Figure 2.18 shows the interface for autodesk VIZ 4. Once objects are drawn in 3D, there are numerous tools for applying materials, providing lights, animating and rendering. The light distribution properties can be set manually or imported from a text file [Mottle, 2003].

2.4.2.4 Form*Z 4.0

Form*Z is a commercial general purpose system that came about as a result of findings in research performed at the Architecture Department of Ohio State University [Khemlani, 2002a, 2003f; auto*des*sys, 2003]. As a result, its use is particularly popular in the architectural fields. It comprises three separate programs including form*Z (for solid modeling), form*Z RenderZone (for lighting, texture mapping, rendering and animation) and form*Z RadioZity (for more advanced and realistically accurate lighting conditions). It uses a modular and open architecture that makes it possible for the authors or external programmers to create subsystems that perform specialized functions that are easily plugged into it, for example a Sketch Render display plug-in makes it possible to generate more artistic-looking images. Figure 2.19 shows the default interface for form*Z. The tools it provides are divided into two categories. The first is for solid modeling which

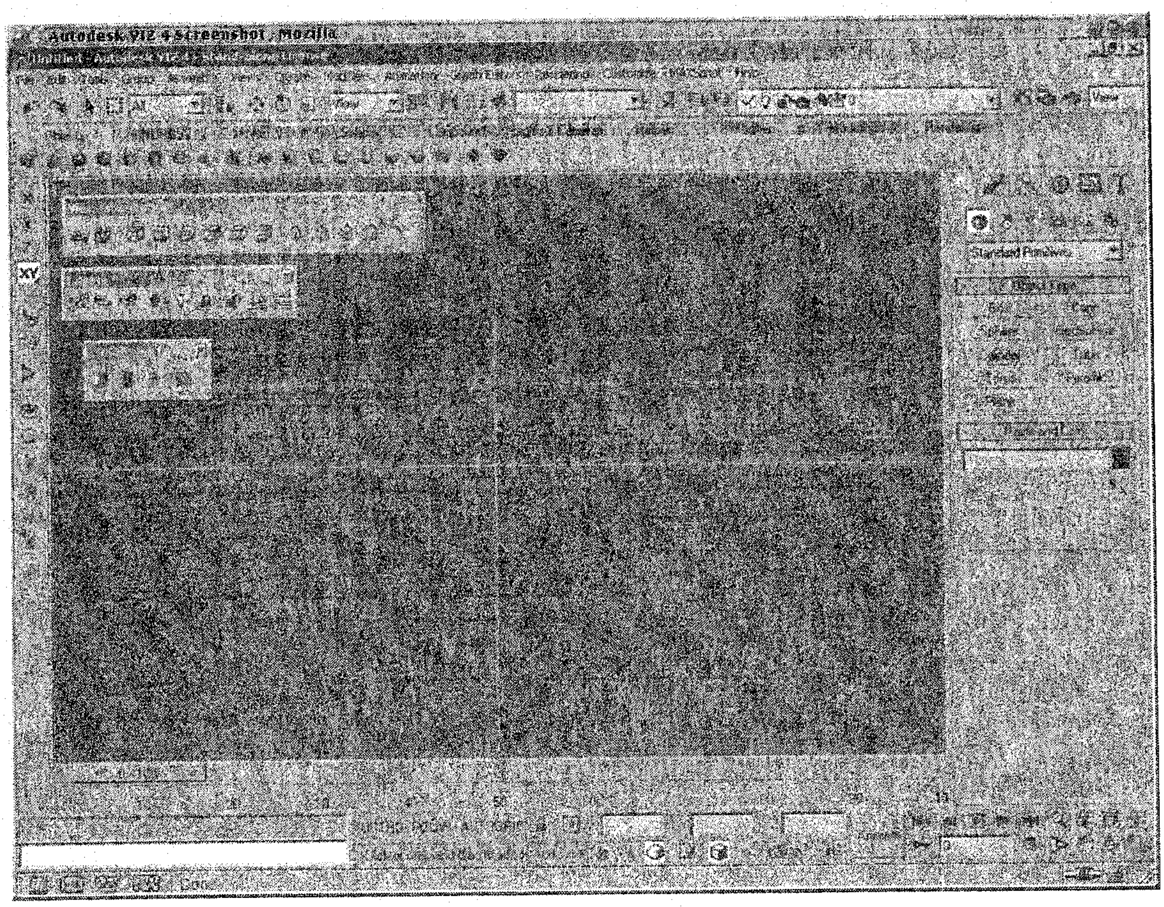

Figure 2.18: Autodesk VIZ interface
go.cadwire.net/?951,2,1

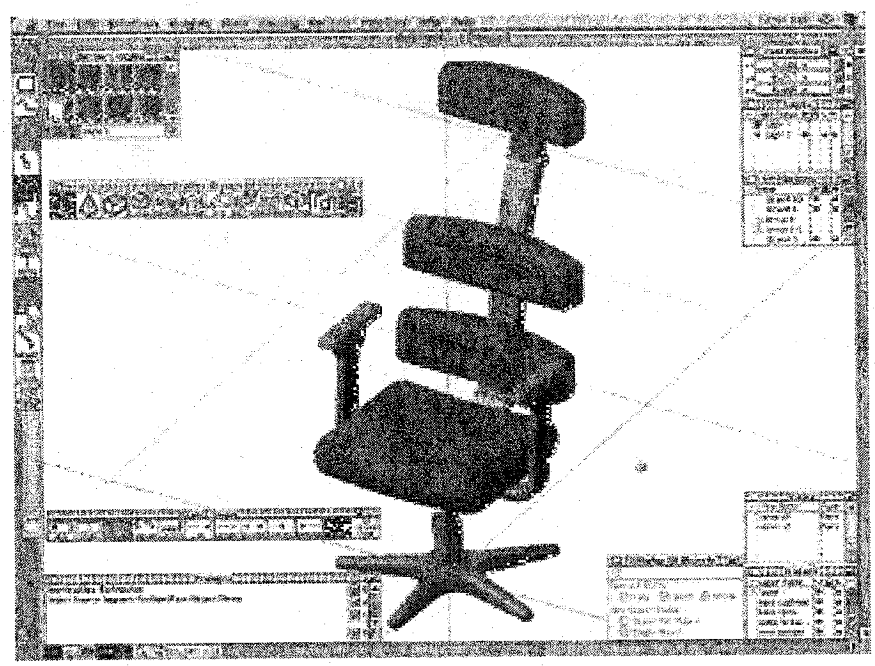

Figure 2.19: Form*Z interface
www.vgd.co.uk/pages/notebook/fzreview.html

provides commands for editing and manipulating entities. The second is for manipulating aspects of the display windows. Geometric primitives provided by 3D systems can be numerous for example autodesk VIZ has 17 different types. In contrast, form*Z uses creation modes and provides only five types which include 2D Surface, 2D Enclosure, 3D Extrusion, 3D Converged and 3D Enclosure. Lighting parameters that incorporate actual manufacturer-supplied data can be described and included as symbols which can be reused across multiple design projects.

2.4.2.5 SketchUp

SketchUp is a commercial 3D tool that tries to address issues at the early stages of the design process [Long, 2003] and is created for architects and designers that want to quickly create 3D ideas. It uses the concept of faces to create and manipulate design items. It provides only single views of the drawing space with a default view of the plan. This single view can be changed however into other views by spinning or rotating it into the desired angle.

The method of drawing in SketchUp intuitively allows coordination to achieve precision. For example to draw a parallel line, the user selects a starting point and points to an existing line. The system constrains the new line to match the angle of the existing line. Objects drawn automatically closeup to form shapes which can then be stretched into 3D objects. Lines that are drawn through shapes, slices them into separate parts. Figure 2.20 shows the interface for SketchUp. SketchUp uses a Push/Pull tool to extrude all shapes

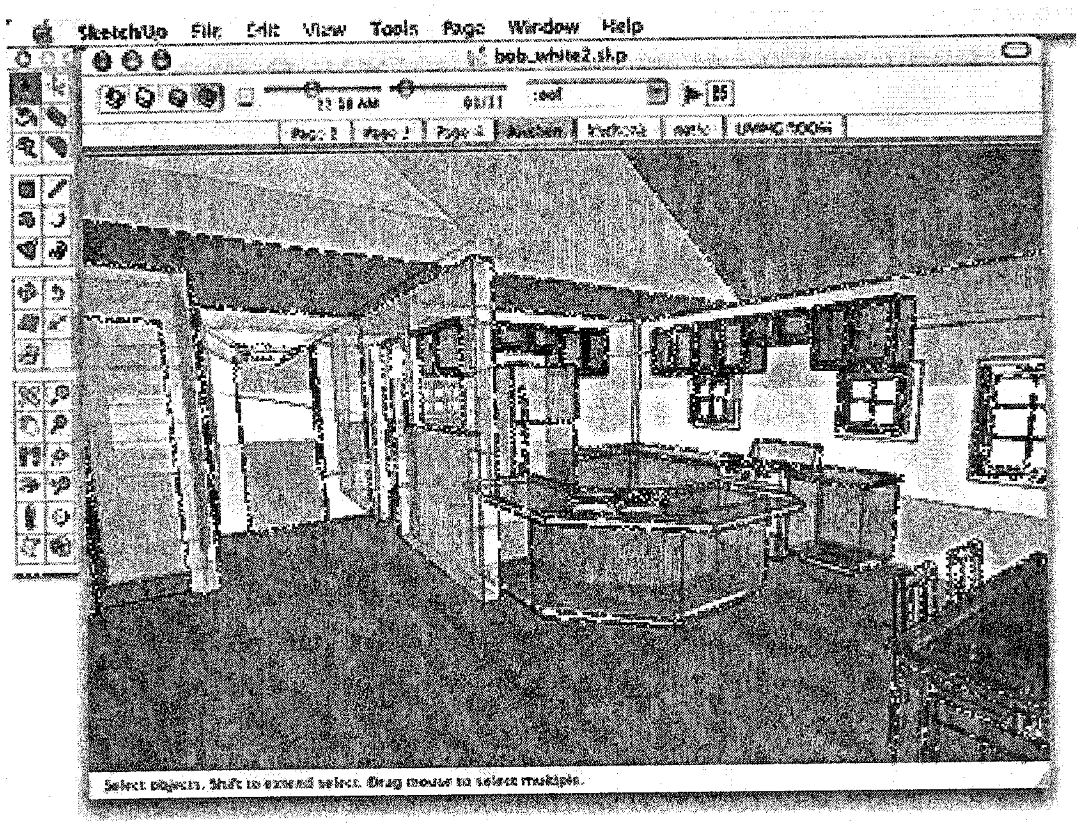

Figure 2.20: SketchUp interface
www.creativepro.com/story/review/18589.html

into 3D objects. In the same way, holes can be created in the 3D objects by drawing other shapes on them and using the Push/Pull tool to create indents and protrusions. SketchUp does not model organic complex curves.

2.4.2.6 SKETCH

SKETCH is a research system that aims to bridge the gap between hand sketches and computer solid modeling [Zeleznik *et al.*, 1996]. It combines some features of sketching and CAD to provide a light-weight gesture-based interface to 'approximate' 3D polyhedral modeling. It uses a gesture-based mode of input in which all operations are available in a 3D scene through a 3-button mouse.

When elements are created the system automatically groups them to make it easier to transform aggregates of geometry. The user has a limited set of geometric primitives: in order to achieve complex shapes, simpler ones must be grouped together. Rendering of objects is accomplished in a non photo realistic technique to maintain an unfinished look as the design is still in its early stages as shown in Figure 2.21. The SKETCH system

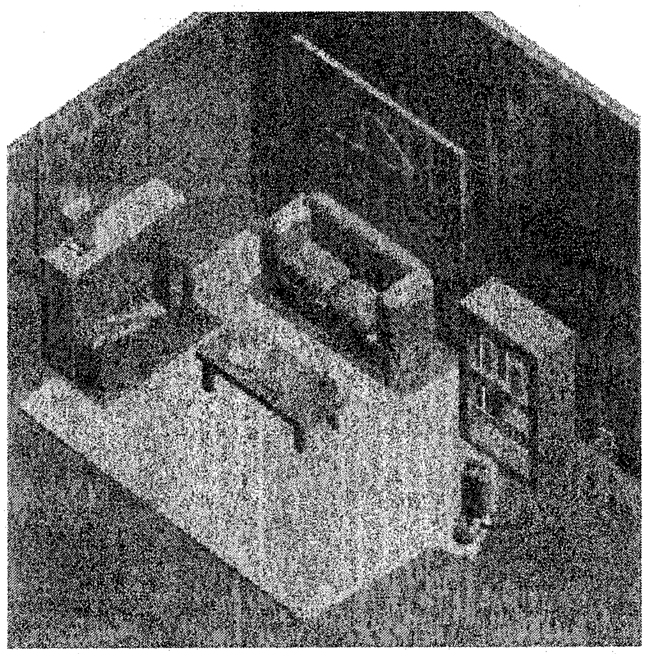

Figure 2.21: A room drawn with SKETCH
www.cs.brown.edu/research/graphics/research/sketch/home.html

provides different ways to render design items showing the incomplete status of work being done.

2.4.2.7 3D Systems: Summary

3D systems provide a similar interface to 2D systems but with the added complexity for drawing in 3D. This makes them less functional for early building design. There is some effort to address the need for a proper interface for early design but this still requires precision inputs to be useful in the design process. There is no provision for recognizing early design items and the solution is not organized in a way that provides an effective overview of the design. Alternatives to design solutions are not collected, however these systems provide some support for early building knowledge which typically is in the area of lighting and shadow studies. These are accomplished by using animation techniques which require an intricate setup process with much detail and effort. They do not provide feasible support at the stage of design that this research is addressing. This summary is illustrated in Table 2.2.

Table 2.2: Summary of reviewed 3D systems

Keys: √ Substantial support | ⊙ Some support | – Limited support

System	Type Of Support				
	Interface Issues	Recognize Roles	Collect Design Alternatives	Design Overview	Knowledge Integration
3D Systems					
Architectural Studio	⊙	–	–	–	⊙
AutoCAD	–	–	–	–	⊙
Autodesk VIZ 4	–	–	–	–	⊙
Form*Z	–	–	–	–	⊙
SketchUp	⊙	–	–	–	⊙
SKETCH	⊙	–	–	–	–

2.4.3 Integrated Systems

Due to the fragmentation rampant in the building industry [Bédard *et al.*, 1991], integrated systems were created to provide a specific building model from where all documentation and specifications can be extracted [Khemlani, 2003b]. The idea of a product model for storing building data is to structure and integrate in a single model all relevant data needed in all computer-supported phases of the life-cycle of the product [Björk, 1989]. These systems implement tools that make it possible for the designer to create a single

3D product model from which documentations and specifications necessary to construct the building are extracted. The systems reviewed in this category are ArchiCAD from Graphisoft, Revit from Autodesk, MicroStation Triforma from Bentley Systems and Visio from Microsoft.

These tools combine the interface of 2D systems and the added capability of 3D systems (except Visio) to create an environment that connects these two parts (2D and 3D data) using the single building model concept. When one is edited the changes are transfered to the other reducing the effort and potential errors in coordinating the two parts of the design. These systems also use building components which recognize and react to each other when being used by the designer such as "walls" and "doors". For example when a "door" is placed in a "wall", the former creates a hole in the later just enough to fit it as appropriate.

All the systems reviewed in this category provide support for the Industry Foundation Classes (IFC) standards. This is a standard used to describe all aspects of the building throughout its life cycle in an effort to encourage interoperability between discipline-specific applications. This standard is being created by an international AEC industry coalition led by a nonprofit organization called the International Alliance for Interoperability (IAI) [Khemlani, 2002c].

2.4.3.1 ArchiCAD v7.0

ArchiCAD is a commercial system and one of the earliest computer-aided building design systems to implement an integrated approach to drawing buildings with computers [Graphisoft, 2003]. Figure 2.22 shows the interface. It recognized building components such as wall, floors, doors and windows and allows the designer to edit or manipulate them in plan, section or perspective views. A single product model is used (stored in a single place in the network) which is then accessed by the design team. Team work is possible through the sharing of the product model either view an intranet or over the internet.

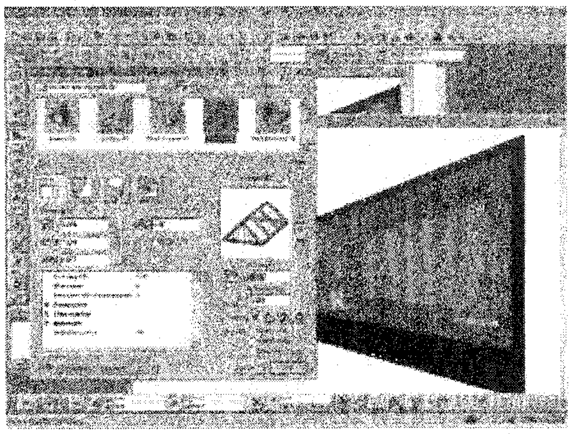

Figure 2.22: ArchiCAD interface
cadence.advanstar.com/2001/0801/cadoptions_archicad0801.html

2.4.3.2 AutoDesk Revit v5.0

In its bid to create an integrated solution for the building industry, Autodesk acquired the commercial software Revit by Revit Technology Corporation and turned it into AutoDesk Revit. Figure 2.23 shows the interface. When items are created, their dimensions are displayed interactively and floors can be created by defining their boundaries and slope, if any. Roofs are more complex than floors or walls and so are less intuitive to create. Revit uses a single product model similar to ArchiCAD, which is located in a single place in the network from where all designers access it.

2.4.3.3 MicroStation v8.0

MicroStation is a commercial system that started as a generic 2D/solid modeling system that recognized geometric attributes as opposed to building information in drawn items. Later an addition called Triforma that integrated with MicroStation was developed to enable the recognition of building information [Khemlani, 2002d; Bentley, 2003]. Triforma differs from ArchiCAD and Revit in the way the building model is made available to the designer. ArchiCAD and Revit use a single model that is centrally located in the network (or computer infrastructure) where all instances of the system access the data.

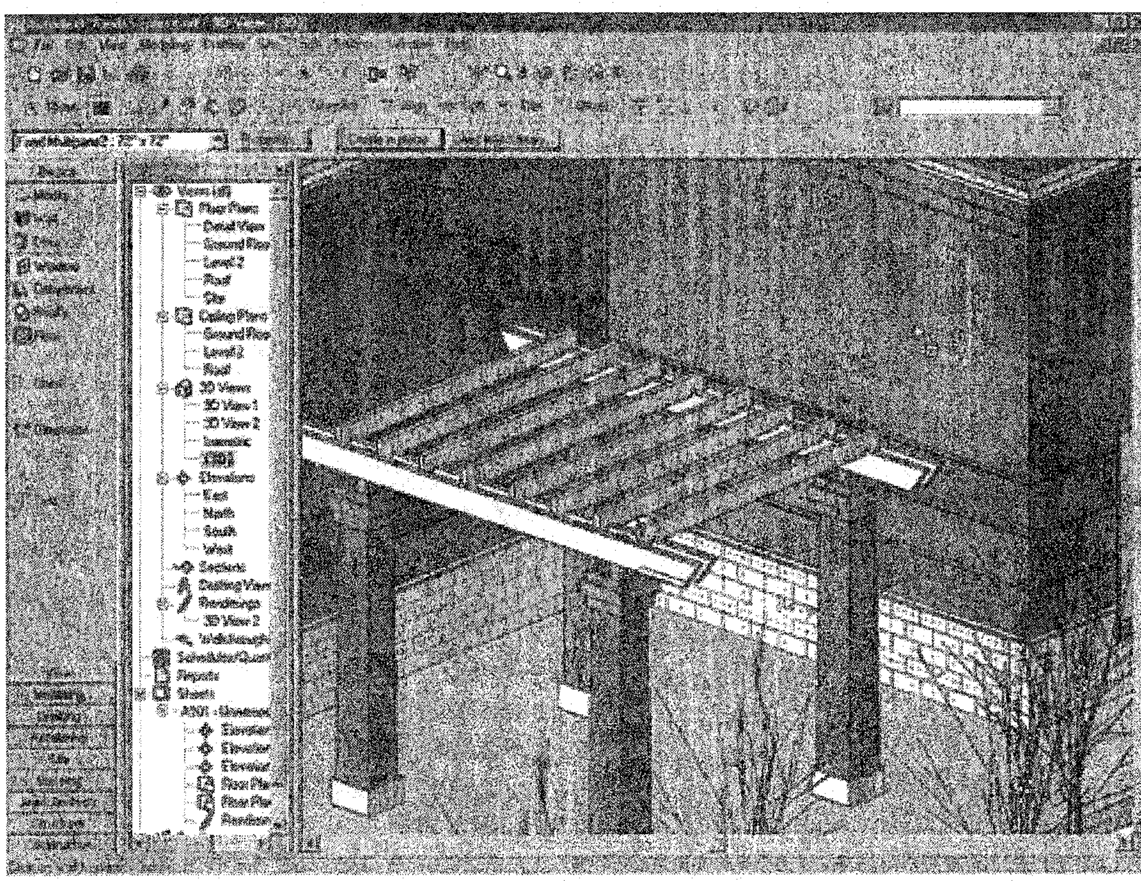

Figure 2.23: AutoDesk Revit interface
cadence.advanstar.com/2003/0503/pr0503_revita.html

MicroStation, on the other hand, uses a distributed system where instances of the system collects the parts that are needed and any changes get synchronized back to the main database. The approach is similar to a collaborative effort where different disciplines need specific data from the database so additions to the main program are provided to implement custom tools to manage the appropriate interface for that discipline. As work is done using these additions the changes, which are stored in a locally located model, are synchronized with the server model that contains all parts of the building model. Figure 2.24 shows the interface for Triforma.

Figure 2.24: MiscroStation Triforma interface
cadence.advanstar.com/2002/0702/fr0702.html

2.4.3.4 Microsoft Visio 2002 Professional

Visio is a commercial system that does not provide the same capability as the other reviewed tools in this category because it does not allow the drawing of 3D parameters and so does not position itself as a competitor, but it makes use of the integrated concept approach in assisting the building design process [Khemlani, 2002b]. It allows the designer to create diagrams using 2D graphic components and it uses a single model to collect all information about the building. It does not support the viewing of 3D parameters but 3D information is contained in its integrated model which can be extracted and used by other systems as required by the use of the IFC standard. Figure 2.25 shows its interface.

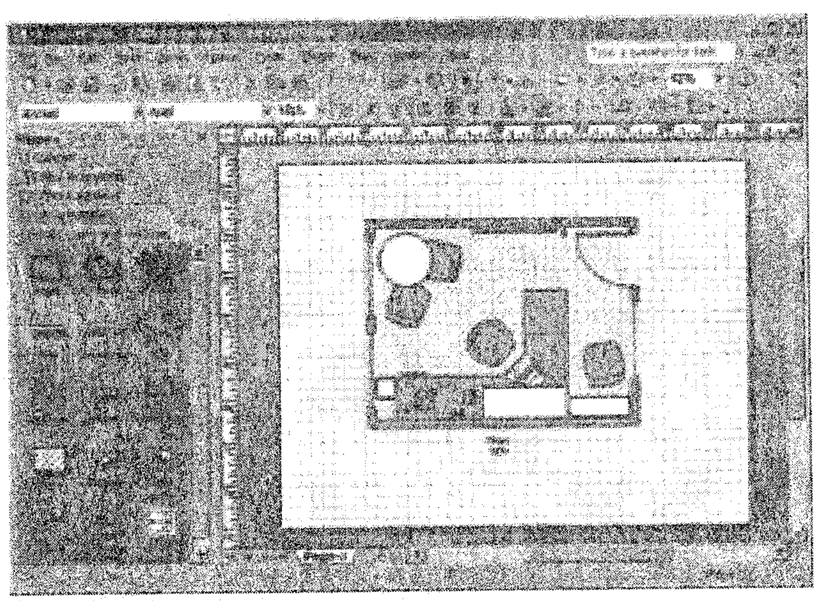

Figure 2.25: Microsoft Visio interface
cadence.advanstar.com/2001/1201/pr1201_visio.html

Visio started out as a general-purpose diagramming tool but now provides a comprehensive set of tools for several IT professionals such as architects, engineers and developers. It makes it possible to create detailed technical documents such as database diagrams, network diagrams, building plans and electrical layouts.

Its functionality is based on the concept of templates. It comes loaded with templates for various categories of drawing types such as flowcharts, maps, building plans, electrical-engineering drawings, etc. When using any template other relevant sub-templates may be used for example, using the building plan template makes it possible to choose between space plans, floor plans, reflected-ceiling plans, site plans, etc. When a template is chosen the floor plan menu palette changes to offer relevant tools such as walls, shells and structure. As a result of the large number of templates, drawings in Visio usually consist of "assembling" drawing items by dragging and dropping on the screen and modifying their properties. The interface is relatively easy to use and provides pages where the designer draws. Multiple pages can be created for a project and each can be set to a different size and scale.

2.4.3.5 Integrated Systems: Summary

Integrated systems provide a similar interface to 2D and 3D systems because they inherit their capabilities. They are able to recognize building components although these are not early building design items as emphasized in this research. There is no decomposition of design items in a way that provides an overview of the design and alternate solutions are not collected and made available for use by the designer. Just as the 3D systems, these systems also provide some knowledge integration but still not in a way that is favorable to the early design process. Table 2.3 illustrates the summary of integrated systems.

Table 2.3: Summary of reviewed integrated systems

Keys: | √ Substantial support | ⊙ Some support | – Limited support

	Type Of Support				
System	Interface Issues	Recognize Roles	Collect Design Alternatives	Design Overview	Knowledge Integration
	Integrated Systems				
ArchiCAD	–	⊙	–	–	⊙
Revit	–	⊙	–	–	⊙
Triforma	–	⊙	–	–	⊙
Visio	⊙	⊙	–	⊙	–

2.4.4 Virtual Reality (VR) Systems

Virtual Reality systems allow interaction between the designer and the design items in a computer-generated world. The interaction may take place with special equipment such as head gears and gloves that provide visual or tactile feedback and means of control to the designer. The main goal of VR is to experience space. They are mostly used for walk-through and fly-around of already designed spaces. The systems reviewed are Immersive Virtual Reality Aided Design (VRAD) and Sculptor.

2.4.4.1 VRAD

The VRAD system is a research derived system that immerses the designer in a 3D world where the design can be visualized using a head-mounted display and manipulated using gloves and a 3D menu tool [Donath and Regenbrecht, 1995, 1999]. The physical environment consists of a platform in an unconstrained space of size 4 x 4 x 2.5 meters. This platform is lit by two ambient lights and two optional positional lights. Figure 2.26 shows the environment for VRAD.

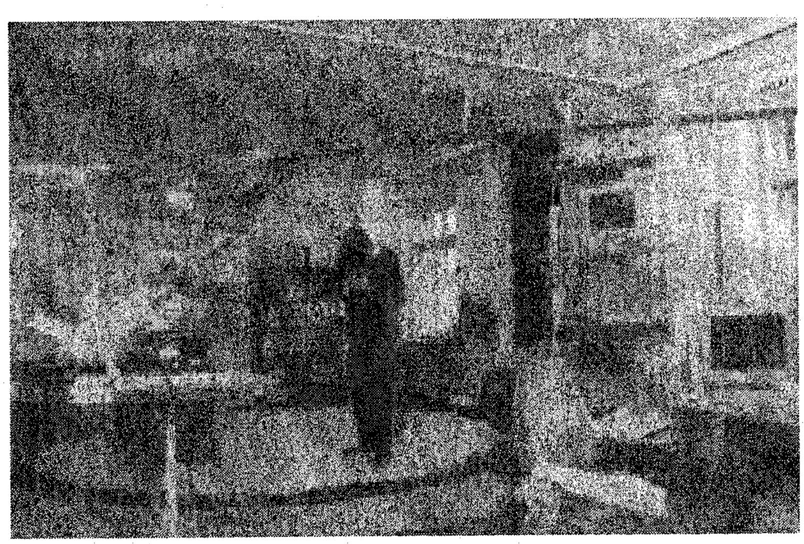

Figure 2.26: VRAD environment
[Donath and Regenbrecht, 1999]

This system implements the use of cubes of size 2.5 cubic centimeters which are then used to create more complex forms by grouping them together. The cubes have no textures but can be assigned any of eight colors, can be set, erased and objects or spaces created with them can be saved and loaded. A 3D menu can be pulled up by clicking or twisting the pointing device: it appears at the designer's location and upon selection, it disappears.

2.4.4.2 Sculptor

Sculptor is a research system that provides an interface for design which includes the elimination of buttons from the screen and windows as shown in Figure 2.27. Objects are

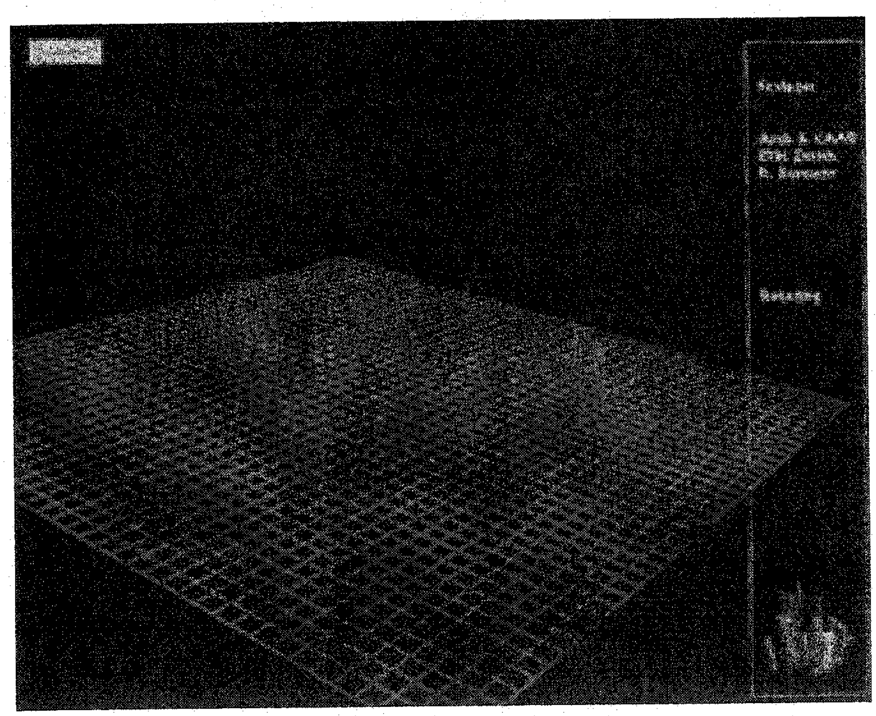

Figure 2.27: Sculptor interface
caad.arch.ethz.ch/ kurmann/sculptor

manipulated directly in real-time with the mouse or other pointing devices [Kurmann and Engeli, 1996; Kurmann, 1996, 1998; Schmitt *et al.*, 1996].

Sculptor uses positive and negative volume objects for the creation of 3D building items. As shown in Figure 2.28, positive volumes model solid items such as walls while negative volumes (or voids) model spaces. When negative volumes intersect with positive volumes, spaces or openings are created. The objects used in modeling scenes are embedded with some intelligence such as: gravity simulation (allows objects to fall on any designated plane or other objects), collision detection (causes objects to refuse to intersect with others), autonomous motion and transformation (allows a given movement and speed to be assigned to an element causing movement over time).

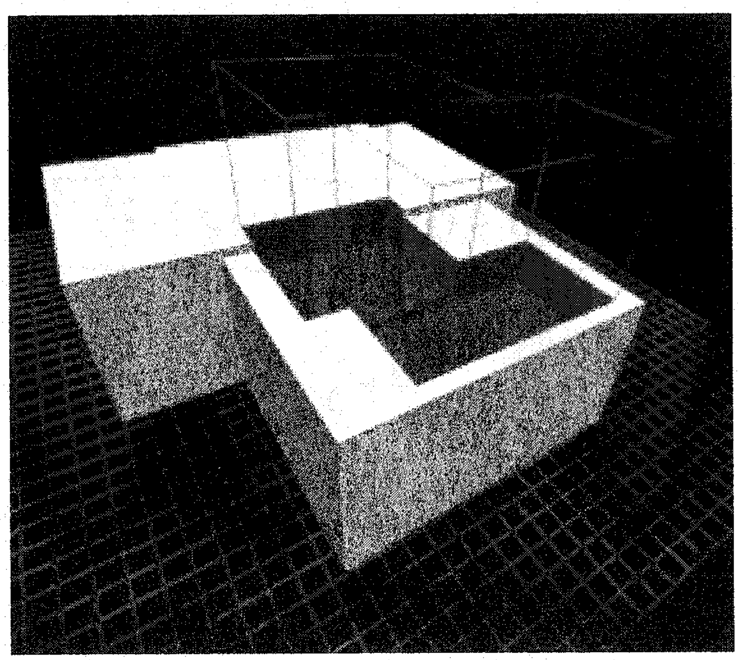

Figure 2.28: Sculptor–positive and negative volumes
caad.arch.ethz.ch/ kurmann/sculptor

The system also incorporates intelligent agents in the form of design assistants. These agents are specialized tools that act independently to address single issues like cost, circulation or the testing of design elements. These agents typically: contain knowledge, work autonomously, work on a specific task, act on behalf of the user and have the ability to learn. Some of these agents are:

1. Agents that enhance or analyze the virtual environment. For example the Navigator Agents use information from the design to generate graphs of possible connections or circulation between rooms.

2. Agents that execute tasks. For example the Cost Agents estimate the cost of the project by multiplying learned factors, given factors and the building cost index. The total from all rooms gives the estimate.

3. Agents that help to test the design e.g. Creature Agents. They populate the design and simulate evacuation procedures during emergencies.

Sculptor works with rectilinear forms as computation using more complex shapes is too intensive. Sculptor is a virtual system that does not immerse the designer in the design environment. In comparison to VRAD that requires the designer to be immersed in the design environment, this system is more advanced in its implementation and use because it eliminates the overhead of available equipment. It also provides better performance due to simpler installation and ease of use [Dorta and Lalande, 1998].

2.4.4.3 Virtual Reality Systems: Summary

Virtual Reality systems provide an interface that addresses issues that support the early design process because they encourage appropriate interactions with design items. They do not provide adequate recognition for building components and there is no provision for an overview or decomposition of the design space. Building solutions created are not collected automatically by the systems making it necessary for the designer to keep track just as in the manual design process. There is some provision of knowledge however especially in the system Sculptor. Table 2.4 illustrate the summary of the Virtual Reality system support for the early building design process.

Table 2.4: Summary of reviewed Virtual Reality systems

Keys: √ Substantial support | ⊙ Some support | – Limited support

	Type Of Support				
System	Interface Issues	Recognize Roles	Collect Design Alternatives	Design Overview	Knowledge Integration
	Virtual Reality Systems				
VRAD	√	–	–	–	–
Sculptor	√	⊙	–	–	√

2.4.5 Generative Systems

Generative systems assist the designer in producing alternative solutions for a design problem in a way that makes it possible for a large number of considerations to be applied [Flemming, 1990]. The systems reviewed are SG-CLIPS and Grid Sketcher.

2.4.5.1 SG-CLIPS

SG-CLIPS is a research derived computer tool that supports the automatic generation of designs from a set of design grammar rules [Chien *et al.*, 1998]. Shape grammars are a set of rules that define a language by which shapes can be generated that share common characteristics or conform to a certain style of design. It consists of a programming environment (where the rules are constructed), an inference engine (that executes the rules) and a graphical user interface (where the results are displayed). In addition to creating new rules, a predefined set of rules can also be loaded into the system. After the alternatives to a design problem have been generated, it allows the designer to manually explore them. It makes it possible for the designer to backtrack to a previously generated alternative.

2.4.5.2 The Grid Sketcher

The Grid Sketcher is a research derived tool that runs within AutoCAD for conceptual design processes [Gardner, 1998]. It allows the designer to derive a design concept by selecting parameters of size, scale, proportion and proximity in order to generate possible forms that can provide solutions to the design problem. The designer can then select growth algorithms which adjust the spatial relationships of these forms through the design session. Within the bounds of a grid structure, editing features allow the designer to analyze successive productions (generation) of shapes and forms as the exploration for a design concept progresses. Through creative manipulations of these algorithmic forms and shapes, the designer can eventually formalize ideas that represent an acceptable design concept.

2.4.5.3 Generative Systems: Summary

Generative systems exert a fair amount of control in the design process that limits the designer's role. As a result the interface is not appropriate for early building design. Design items are not recognized making it difficult to determine roles. These systems automatically collect design alternatives but they do not provide an overview of the design space. Knowledge integration is provided although this not appropriate for the purposes of early building design as they do not address the need for interactive reasoning and feedback. Table 2.5 presents a summary of this category of systems.

Table 2.5: Summary of reviewed generative systems

Keys: √ Substantial support | ⊙ Some support | – Limited support

System	Type Of Support: Interface Issues	Recognize Roles	Collect Design Alternatives	Design Overview	Knowledge Integration
Generative Systems					
SG-CLIPS	–	–	⊙	⊙	⊙
GRID Sketcher	–	–	⊙	–	⊙

2.4.6 Interactive Design Exploration (IDE) Systems

IDE systems are research derived tools that try to address the issues surrounding support for the early building design process. They present innovative ideas with regard to early design support. Typically some recognition of design items is provided which makes it possible for these systems to provide appropriate interaction with the designer. Computer hardware makes interfacing with designers difficult for example, using the mouse to draw is not very intuitive. So these systems are typically more advanced in other areas of support than the user interface. This category presents the systems that provide the most support for the early building design process.

The tools presented include the Electronic Napkin, Building Composer, SEED and EsQUIsE. All are experimental systems although in addition, Building Composer is used in assisting the facility design and installation in the US Army Corps of Engineers.

2.4.6.1 Electronic Napkin

The Electronic Napkin is a research derived pen-based system that provides support for the conceptual drawing process through the recognition, interpretation and management of drawings [Gross, 1996; Gross and Do, 1996]. It serves as an interface for knowledge-based critiquing, simulation and information retrieval processes. Figure 2.29 shows the main interface to the Electronic Napkin.

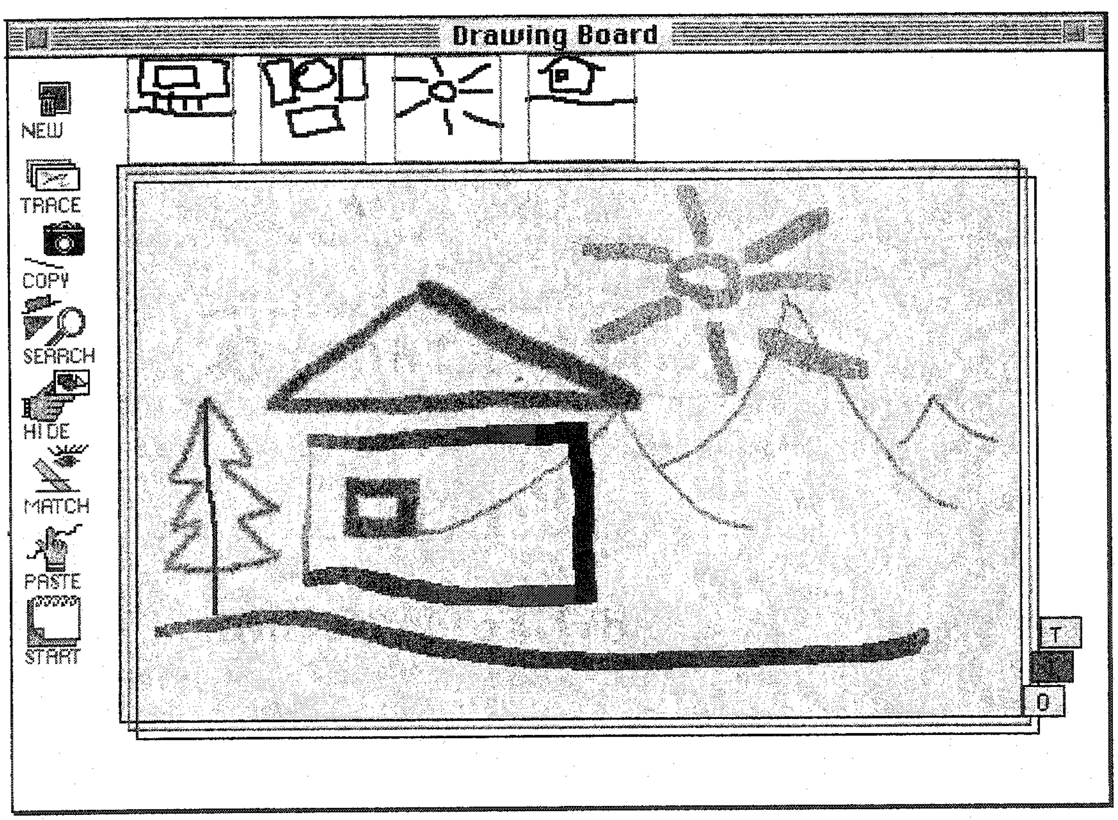

Figure 2.29: Main drawing interface
www.acm.org/sigchi/chi96/proceedings/demos/Gross/mdg_txt.htm

It uses semi-transparent overlays to simulate tracing papers used in manual early design sessions. It uses a simple on-the-fly recognition scheme to recognize the designer's pen strokes. It identifies elements and spatial relations and constructs a function to recognize instances of the elements configuration. As an alternative, the designer may designate a picture to replace a recognized configuration. Relations found in the drawings are automatically inferred and maintained by the system. A sketchbook interface is also provided for the designer to keep interesting drawings as shown in Figure 2.30. The

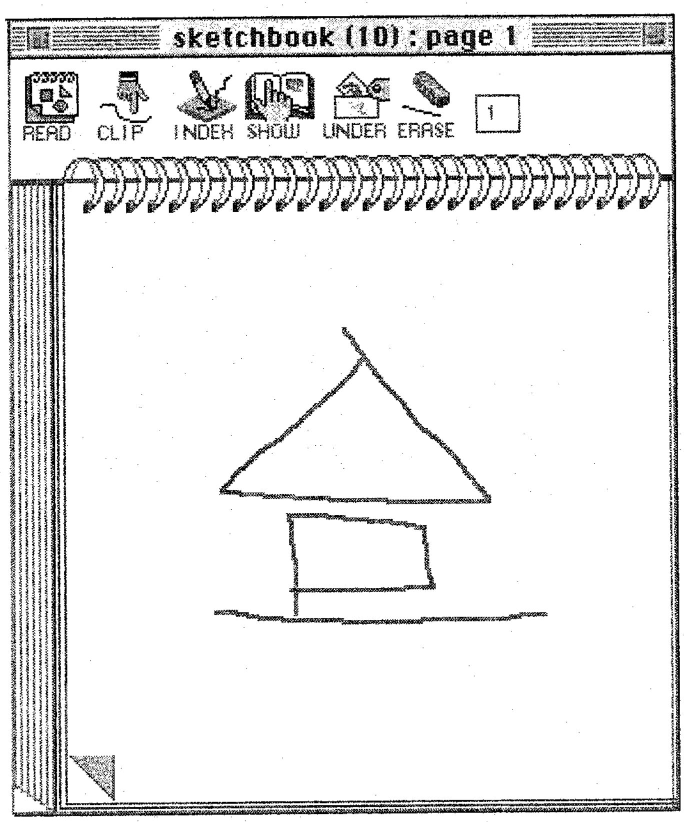

Figure 2.30: Sketch collection interface
www.acm.org/sigchi/chi96/proceedings/demos/Gross/mdg_txt.htm

Electronic Napkin system is used in saving and searching for solutions that can be reused in the design process.

2.4.6.2 Building Composer

Building Composer is a research derived system that provides a suite of tools for planners, designers and engineers in the early phases of facility planning and design [Brucker, 2002]. It implements the association of customer specification and computable criteria with the facility model or project. These associations can be made on different project elements at appropriate levels of detail: Project, Site, Building, Story, Function and Spaces. For example, a target cost can be associated to a Project or the specification that a masonry exterior wall and a steel structure be used in (or associated to) a Building.

Building Composer provides the following primary functions, some of which are illustrated in Figure 2.31:

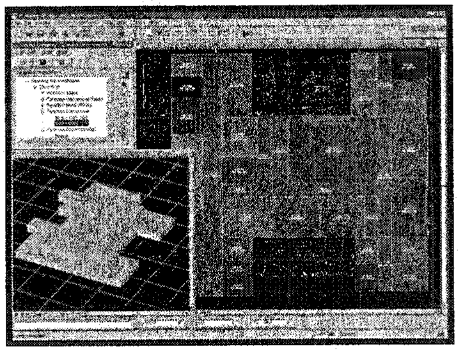

Figure 2.31: Building Composer interface
bc.cecer.army.mil/bc/lc.jsp

- Criteria Manager. A web based application that helps in the development of corporate and building specific libraries;
- Criteria Composer. Assists user in creating an architectural program and to set values for project specific criteria;
- Layout Composer. Provides an environment for the user to create 3D conceptual facility designs;
- Wizards. Supports various discipline specific issues and assists in the completion of individual design tasks and calculations.

Building composer works in conjunction with integrated systems such as MicroStation Triforma and AutoCAD Architectural Desktop. Spatial geometry created is recognized with the roles that they play in the design from the functions in the criteria manager. For example, the system recognizes a bathroom and also knows that it will require an exhaust air system while an office space will not. Building Composer is able to recognize building components however the interface relies on the support provided by the integrated system it is running on top of.

2.4.6.3 SEED

SEED is a research project of a multidisciplinary effort to create an environment to support the early phases in building design. SEED does not wish to rely on the traditional methods of form generation used in computers. It is the goal of the project to re-think the way forms are generated using computers during early design [Akin *et al.*, 1998].

SEED supports the early building design process in three main modules described as architectural programming (SEED-Pro), schematic layout design (SEED-Layout) and the generation of 3-dimensional configuration of physical building components like structure and enclosure (SEED-Config). These are in turn supported by a common object database, a standards usage support environment and additional modules that will be developed as the need arises. This review describes the three modules for programming, layout and 3D configuration. Figure 2.32 shows the relationships between the different modules of SEED.

SEED-Pro helps the designer develop an architectural program. It accepts input of design requirements from direct input, reads from projects in its database or from direct communication with another system or module. This information is then available to be used in the design process or exported for use outside the system for example as pre-formatted reports of the architectural programming.

SEED-Layout provides support for the generation and evaluation of schematic building layouts. It is only able to handle rectilinear forms within a given area without the need for

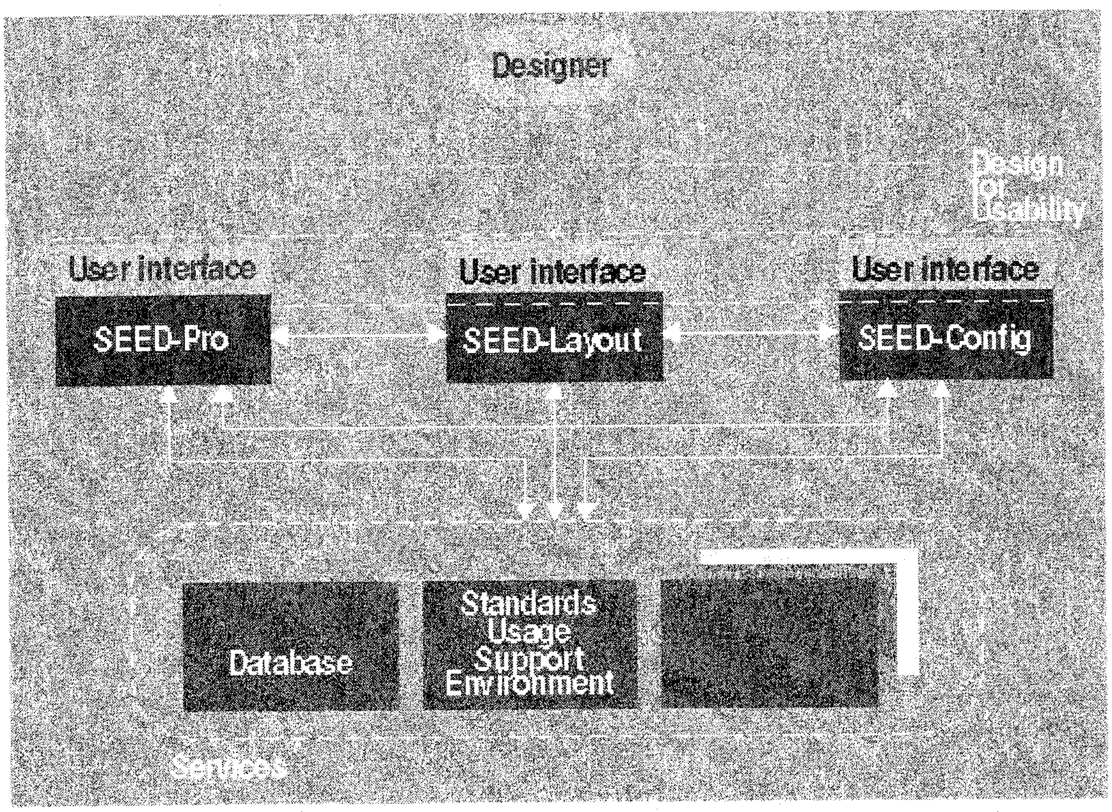

Figure 2.32: SEED modules
www.arch.usyd.edu.au/kcdc/journal/vol1/papers/flemming/toc.html

precise dimensions. The process involved in using SEED-Layout is shown in Figure 2.33. SEED-Layout is used to create and modify layouts that try to satisfy the requirements specified in the problem statement. It is also possible to work with a partial or complete layout with respect to the requirements [Chien, 2000].

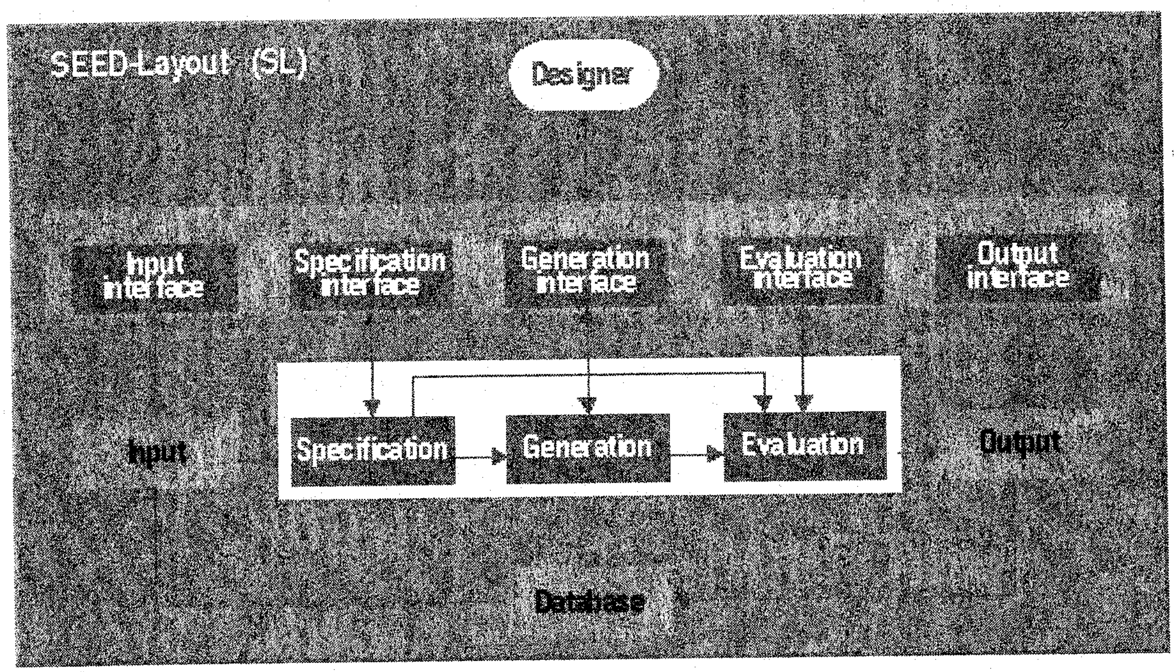

Figure 2.33: SEED-Layout components
www.arch.usyd.edu.au/kcdc/journal/vol1/papers/flemming/SL/arch.html

The designer is allowed to enter specifications using templates. Once accepted, the generative component of the system attempts to find a similar project from its case library matching the specifications. If no project is found, more information may be requested of the designer. For a given problem the system can generate two-dimensional layouts, allow the designer to manually generate them or it may allow the designer to interactively explore possible layouts. The system also assists the generation of alternative layouts. It also checks the validity of the designers actions and may ask for confirmation when certain items are set beyond their limits or permitted range.

The system evaluates the architectural program against the clients needs, preferences, functional requirements, costs and standards. In this way the system works in cycles of generate-and-test as the designer proceeds in the design process.

SEED-Config provides support by helping the designer to generate and evaluate schematic 3D building configurations in various levels of detail. It uses form generating rules and makes it possible for the designer to have automatic, manual or semi-manual control over the functions it provides. These functions include the creation and modification of problem statements, the creation and modification of configurations that try to satisfy the requirements specified in the problem statement, the creation and modification of technologies (form-generation rules) and the evaluation of a partial or complete configuration with respect to the requirements.

2.4.6.4 EsQUIsE

The principles behind most architectural support in software is to replace the traditional paper with an interface that is used to collect the graphic elements reflecting the designer's reasoning on paper. In the same way, 3D computer tools do the same utilizing a virtual representation of the designer's decisions in the computer. Nevertheless these are representations composed by the designer and do not represent appropriate early design assistance because they do not capture the reasons why those graphic items were created and so cannot provide appropriate support for the designer thinking process.

The EsQUIsE tool is a research derived prototype. It is a geometric interpreter of descriptive architectural sketches [Leclercq, 1999]. It is composed of two modules. The first interprets the graphic input of lines from the designer. These lines may have any of four colors to designate text, a degree of transparency, opacity and non-significant items like hatches or comments. Studying the contacts between lines the system is able to deduce spaces delimited by them. Figure 2.34 shows the capture window where the designer draws in different layers.

The second module interprets captions in order to recognize the role of the spaces being deduced by the first module. Figure 2.35 shows the result of analyzing the sketch in

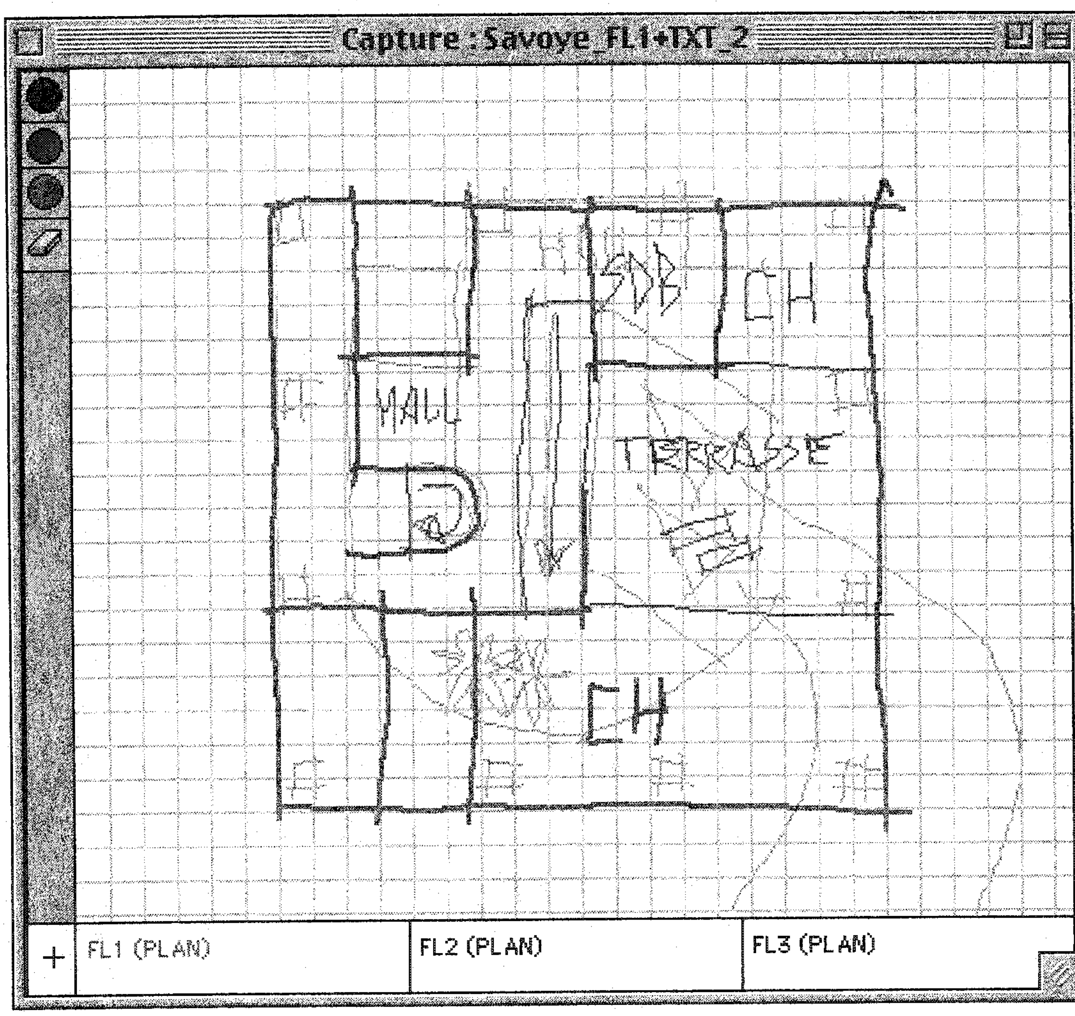

Figure 2.34: EsQUIsE - capturing spaces
www.lema.ulg.ac.be/tools/esquise/Esquise-Screenshots.html

Figure 2.34. The system implements the organization and recognition of functional spaces as shown in Figure 2.36. The system uses this recognition to define certain ancillary items (at this stage of design) such as thicknesses between spaces (walls).

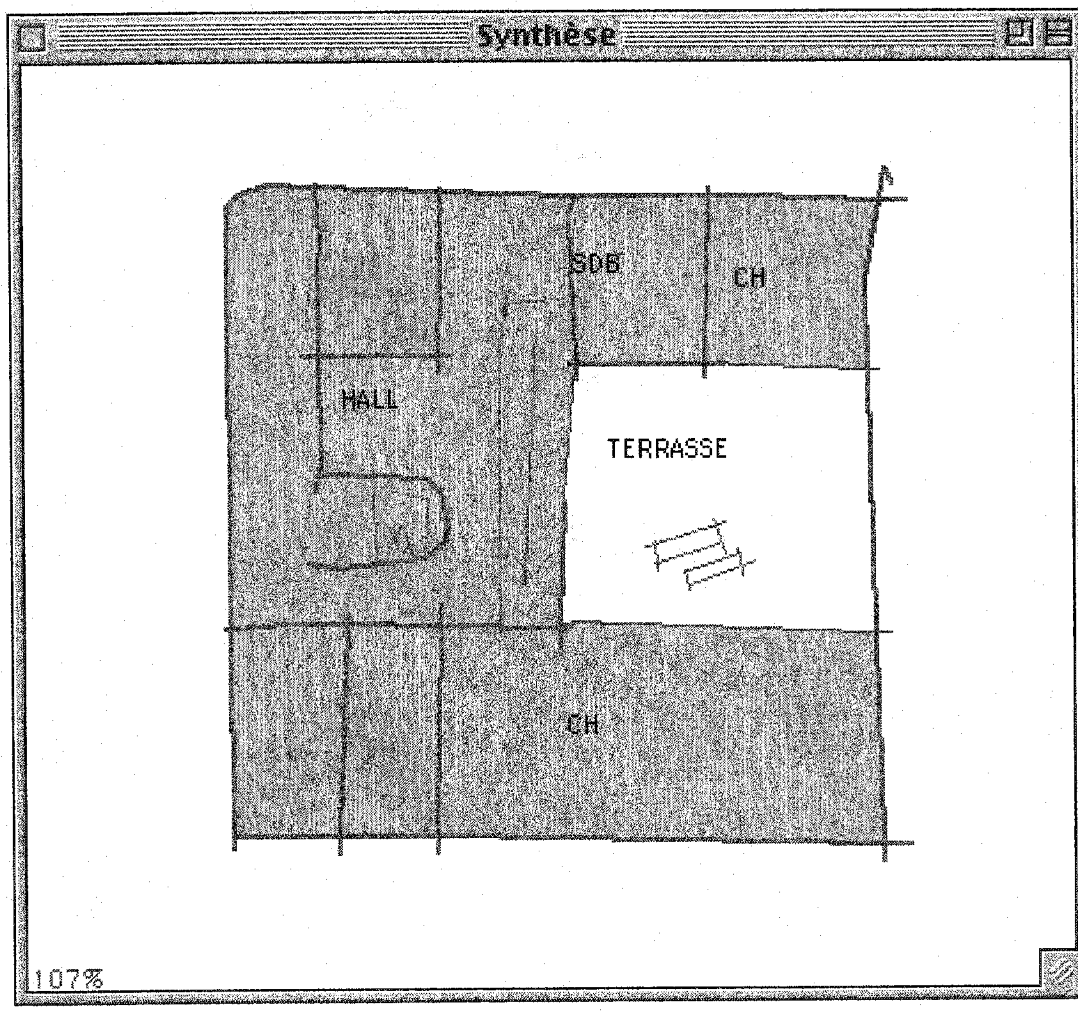

Figure 2.35: EsQUIsE - synthesizing sketches
www.lema.ulg.ac.be/tools/esquise/Esquise-Screenshots.html

The recognition of spaces in EsQUIsE is based on the recognition of the designer's labelling of the designated spaces. This will depend on the recognition algorithm or whether the language being used in the design process is supported. The designer must also be clear in labelling spaces and must draw attention from the design process to check that a label is accurately recognized.

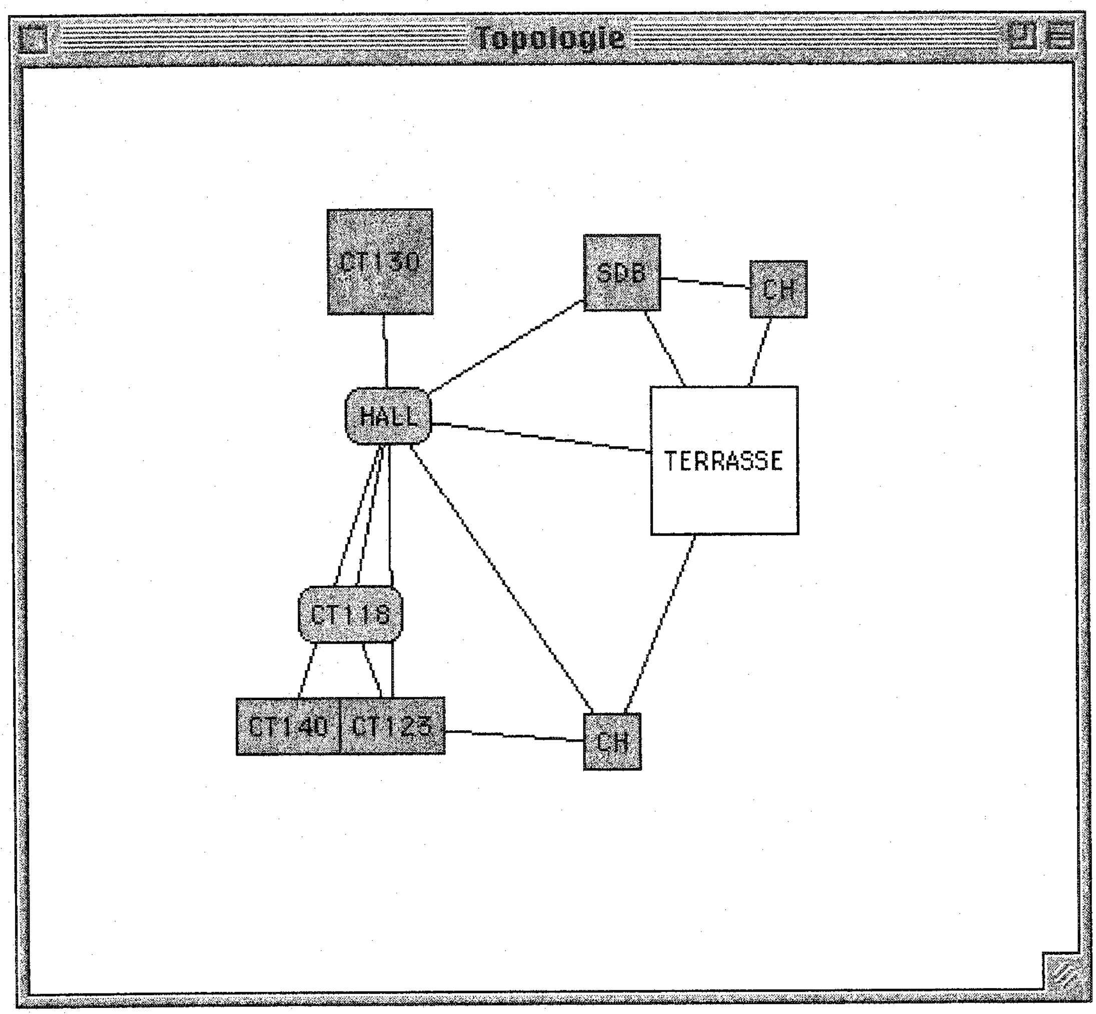

Figure 2.36: EsQUIsE - topology view
www.lema.ulg.ac.be/tools/esquise/Esquise-Screenshots.html

2.4.6.5 IDE Systems: Summary

IDE Systems provide tools that allow the designer to explore possible solutions to design problems. Some interface designs are more advanced than others but there is support for recognizing building components. Collection of design alternatives is present in all reviewed systems but this is not made available in a way the designer can easily interact with. Design alternatives are usually left in a database with no visual cue to how many or how they are linked to each other. The presentation of a design overview is another weak capability however there is support for knowledge integration appropriate for the early building design process. Table 2.6 illustrates the summary of this category of tools.

Table 2.6: Summary of reviewed IDE systems

Keys: √ Substantial support | ⊙ Some support | – Limited support

System	Type Of Support				
	Interface Issues	Recognize Roles	Collect Design Alternatives	Design Overview	Knowledge Integration
	Interactive Design Exploration Systems				
Electronic Napkin	√	–	–	–	√
Building Composer	⊙	√	–	√	√
SEED	–	√	–	⊙	√
EsQUIsE	√	√	–	⊙	√

2.5 Summary and Conclusions

Several systems, many of which are used in practice, have been reviewed in order to give an idea of their capabilities and limitations at early design tasks. This summary explicitly compares their capabilities as it relates to the requirements discussed in section 2.3 and establishes if any system provides the required early design support. Table 2.7 shows a summary of all systems reviewed and how much support they provide for the requirements discussed in section 2.3. Interface Issues (see section 2.3.1) refer to the need to draw without having to concentrate on the drawing action eliminating the need for precision input/manipulation or errors/interruptions from the system. The review does not consider the capability to just draw lines as addressing this issue. It must be possible to draw enclosed items that can be recognized or used as building spaces by the system.

Recognize Roles (see section 2.3.2) refers to the need to recognize the items created in the design. Such a recognition provides the system with the role or behavior of the item in the design. We do not accept the automatic *labeling* of items such as "walls" as a form of recognition and we have tried to identify these as providing limited support.

Collect Design Alternatives (see section 2.3.3) refers to the need to collect alternative solutions to any problem that is being explored by the designer. At the same time, these solutions must be made available, in real time, for use in obtaining a final most desired solution.

Table 2.7: Early building design support from reviewed systems

Keys: √ Substantial support | ⊙ Some support | – Limited support

	Type Of Support				
System	Interface Issues	Recognize Roles	Collect Design Alternatives	Design Overview	Knowledge Integration
			2D Systems		
SmartSketch	⊙	–	–	–	–
AutoCAD LT	–	–	–	–	–
QCAD	–	–	–	–	–
			3D Systems		
Architectural Studio	⊙	–	–	–	⊙
AutoCAD	–	–	–	–	⊙
Autodesk VIZ 4	–	–	–	–	⊙
Form*Z	–	–	–	–	⊙
SketchUp	⊙	–	–	–	⊙
SKETCH	⊙	–	–	–	–
			Integrated Systems		
ArchiCAD	–	⊙	–	–	⊙
Revit	–	⊙	–	–	⊙
Triforma	–	⊙	–	–	⊙
Visio	⊙	⊙	–	⊙	–
			Virtual Reality Systems		
VRAD	√	–	–	–	–
Sculptor	√	⊙	–	–	√
			Generative Systems		
SG-CLIPS	–	–	⊙	⊙	⊙
GRID Sketcher	–	–	⊙	–	⊙
			Interactive Design Exploration Systems		
Electronic Napkin	√	–	–	–	√
Building Composer	⊙	√	–	√	√
SEED	–	√	–	⊙	√
EsQUIsE	√	√	–	⊙	√

Design Overview (see section 2.3.4) refers to the need for the designer to orient themselves and observe the relationships between items in the design solution including having direct access with little effort. This limits frustration and unnecessary interruptions in the design sessions.

Knowledge Integration (see section 2.3.5) refers to the need to support certain actions or phenomena that help the designer make more informed decisions or make it possible to test issues that are under consideration.

It must be said that most of these systems are being improved constantly and our review may not be exhaustive of systems capabilities, especially if they are not made accessible to average users (in which case the system is not geared towards early design use anyway). As a result we have provided "soft" keys for indicating how much support is provided in the tables. As Table 2.7 shows, there is no single category that clearly provides all of the required support. Each category provides support that may be lacking in others, although the 2D systems category seems to lack most of the required support. This may be because development efforts are no longer focused on these systems. 3D systems provide some support for knowledge integration such as sun studies or animations that can be used to study movement or flow. These systems are marked as providing some support because they cannot be readily used in the early design process. There are numerous steps required to set up animations for example. There is also some support for interface issues but this support still relies on precision input in order to create useful forms.

Integrated systems apply knowledge just like systems in the 3D category but in addition they recognize building components. This recognition makes it possible for these systems to describe the same building component in many views and manage these views from one location. Such as updating the 2D views when the 3D view is updated and vice versa. This recognition makes it possible for the roles of certain building items to be determined such as the "doors" and "windows" creating holes in the "walls". This capability is however not fully supported for example a door can still be created even if there is no wall.

Virtual Reality systems are not a mature category but the main strength is in interface support. These systems can easily combine the capabilities of Integrated System to provide a more robust support but more progress has to be made in hardware cost and accessibility. This is a promising category for early design.

Generative systems provide good support for collecting design alternatives. This is necessary because the designer often cannot interact with the system. The system accepts some parameters and iterates to generate options which are considered by the designer before altering the operation of the system again. Some support for knowledge is provided as well but such systems do not provide an interface that follows the design process as described by this research project. Interactive Design Exploration Systems provide the most support in almost all requirements as they are systems that are geared towards use in the early design process. They do not fare very well in the collection of solutions however. This research project considers it important to collect these solutions in a way that keeps them visually available to the designer so that they are easy to browse or review. This encourages the ability to branch into explorations or to step back to a point in time in the process. To provide adequate support for the early building design process a tool must substantially address these five minimum requirements. Judging from Table 2.7, such a tool does not exist yet. The reason is that few analyses of design sessions are available in order to derive methods of design and implementation appropriate for the computer environment. To address this problem, there is a need to determine the appropriate means of representing early design data in computers. The next section discusses this issue and then provides a map for how this research project proceeds.

Chapter 3

Methodology

We have established the requirements that must be addressed in order to provide support for the early building design process in section 2.3. This constitutes the "what" issues surrounding early building design that need to be addressed in software assistance. An extensive review was performed of currently available computer systems (of research and commercial origins) that are used in the design process and arrived at the conclusion that no system shows adequate support for all requirements. This leads to the need to explore the "how" issues of making it possible to support these requirements.

To do this, an appropriate representation of early design data must be determined first to provide the type of interaction that is required at this stage of the design process. This chapter establishes this method of representation and then provides a map for how the rest of the research project will be undertaken.

3.1 Early Building Design Data Modeling

Successful approaches to support complex and vague processes like early building design depend on how information can be created, manipulated and stored. Early CAD systems enabled the production of construction drawings using the same layer technique as developed for manual drafting. Even with the advent of object-oriented methodologies in computers, the formalization of engineering knowledge and increasing computer hardware capabilities, these early CAD systems were still unable to support important and emerging areas of the design process such as early design and integration [Björk, 1989].

Conventional CAD tools, such as the commercial tools reviewed in the previous chapter, do not provide an appropriate means of representing or modeling information for the AEC domain [Rosenman and Gero, 1995]. CAD system providers have succeeded in extending the capabilities of their systems by providing domain-specific applications. These applications consist of a set of predefined and general graphic entities that perform a set of fixed functions such as rectilinear or circular shaped forms [Eastman, 1992]. This is not sufficient because early architectural design is a highly dynamic process which cannot be accomplished using a predefined set of graphic entities. For example, the form of a building space is only determined through exploration and cannot be predicted at the beginning [Eastman, 1992]. It is therefore necessary to provide a system that models data in a loose and therefore extensible way to allow for unforeseen ways of entity/data representation [Ekholm and Fridqvist, 1996]. A considerable amount of research has been carried out in this area and there seems to be an agreement on the use of object-oriented data modeling techniques [Ekholm, 1994]. Object-oriented data modeling refers to the use of objects to represent information used in data models.

In order to support the type of interactions envisioned in this thesis, an information model must provide the basic data structure and means to completely capture the conceptual design of a building—the detailed and abstract representations of the designer's reasoning in the workspace. Detailed representations are normally objects with factual properties such as representing spaces with actual length and width. Abstract representations are mental constructs with conceptual properties such as representing a desire (example proximity) [Ekholm and Fridqvist, 1995]. At various stages these representations refer to different kinds of information, thus different abstractions need to be used before a consistent representation emerges.

An appropriate information model is expected to have the following capabilities:

- To capture all data used by the designer including abstract information and the implications regarding the exploratory nature of early building design such as creation of more than one solution. To also capture relations desired by the designer that will provide a guide or basis for later decisions. The model must support the ability to present data in different ways such as using hierarchies (to show relationships), linear steps (to show overall design progression), abstract forms (to encourage change) and detail forms (for presentation and review).
- To support the designer's interaction in such a way as to decrease the drawing effort and increase the designer's capability for analysis and creativity. The model must not demand undue specificity, must be intuitive, must not require the designer to change their preferred drawing process and must provide many views of the design such as plans and model views.

The Building ENtity and Technology (BENT) information model provides some of these capabilities [Rivard and Fenves, 2000]. Entities, as shown in Figure 3.1, model building items at different levels of complexity by aggregating attribute-value pairs to represent functional, design and evaluation units.

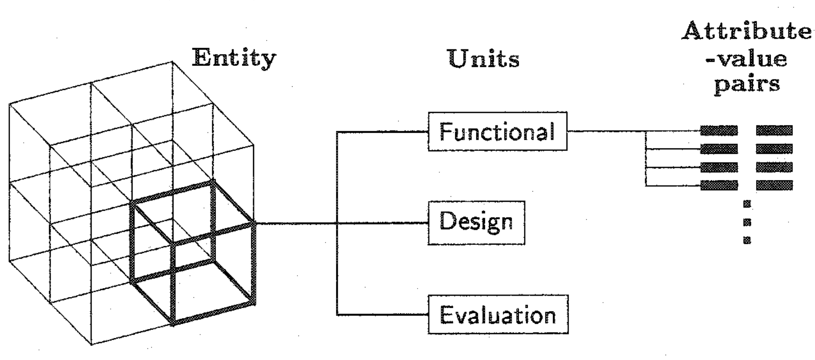

Figure 3.1: Entity model

3.2 Understanding the Early Building Design Process

Having established the "what" issues of early building design support in section 2.3, we begin to address the "how" issue by documenting complete design sessions and describing the actual workspace activity. This shall be accomplished using the method of observation and records called protocol analysis (see section 2.2). Other ways exist to study a designer's work process such as through interviews, observations, case studies, reflections and theorizing. However, protocol studies make it possible to reduce the complexity of the early design process by focusing on isolated parts of the activity [Baya and Leifer, 1996; Mazijoglou *et al.*, 1996; Dorst and Dijkhuis, 1996]. There are two main issues to be addressed by the protocol study. The first is to understand how data is added to a sketch so that this can be achieved in a digital environment. Secondly, it is important to derive an efficient digital interface to support the interactions and information produced between designer and tool at such an early stage of the design process [Lipson and Shpitalni, 2000; Baya and Leifer, 1996; Bhavnani *et al.*, 1993].

Knowledge about "how" the designer manipulates the sketch will provide the basis on which to specify the desired functionalities that can assist the early building design process in a digital environment. This is discussed and the specifications are presented in Chapter 4.

3.3 Planning to Support Early Building Design Process in Computers

Once a thorough investigation of the "how" issues is complete, the information obtained will be used to formulate methods that can be used in a digital environment which will adequately support the early design process. Based on these methods an object-oriented development process will be utilized to develop and create a prototype computer program.

An object is a representation of an entity, either real-world or conceptual. An object-oriented program is the use of groups of objects that communicate with one another to serve a purpose. An object-oriented development process is the turning of an idea or a problem into an object-oriented program [Quatrani, 1998; Richter, 2001].

The software development process shall be documented using a notation called the Unified Modeling Language (UML). UML provides a robust and 'de facto' notation that is able to grow from analysis into design. The behavior or functionality to be provided by the system is documented in "use case models". Use case models are made up of the intended functions (documented as "use cases") for the system, its surroundings (documented as "actors") and the relationship between use cases and actors (documented as "use case diagrams"). The structure of the system is documented using "classes". A class is a description of a group of objects that share a common property (attributes), behavior (operations), relationships to other objects and semantics. They form the bases for realizing the software system [Quatrani, 1998]. This development process is presented and discussed in Chapter 5.

3.4 Implementing and Validating a Solution

The development process provides a blueprint from which a system can be implemented. A prototype tool is implemented using this blueprint to show an example of the support for the early design process envisioned by this research. The prototype is described in Chapter 6 to address all the requirements described in section 2.3. In Chapter 7 the prototype is tested by comparing a design session accomplished using it to the traditional session described in section 2.1. This will validate whether the requirements for early design presented in section 2.3 are being supported and if this support makes any difference in the early building design process. The research project will be concluded in Chapter 8 where the highlights will be summarized including contributions and suggestions for future work.

Chapter 4

Protocol Study

As stated in section 2.2, a protocol study/analysis is an efficient way to reveal the intricacies of an activity such as the early building design process. This chapter begins with an introduction of the parameters initiating this study followed by some information on the designers studied as well as the procedure for the study, in section 4.1.1. This section also illustrates the methods with which the study is recorded. Section 4.1.2 presents the results of the study while section 4.1.3 discusses the analysis of results as well as other observations that are important in a successful early building session. A set of specifications that result from the analysis and observations is listed and described in section 4.2.

4.1 The Protocol Study

In order to investigate and to understand the "how" issues concerning developing a computer-based tool to assist designers during the early building design process, it is important first to observe how designers work in carefully controlled conditions. Such observations can be obtained by means of a protocol study performed on a varied group of designers at work. This protocol study of early design has focused on the actions that the designer initiates and performs on sketches (especially building spaces) in order to transform them into design solutions. Attention is focused also on how these sketches change during development in order to understand how the computer-based system should automatically capture and process such changes to successfully represent their transformations. This will also lead to appropriate user-interface considerations for the tool being proposed. The following sections describe the study of eight designers and an analysis of the data collected.

4.1.1 Procedure

The protocol study is divided into two sections consisting of an interview (15 minutes) and a design session (45 minutes), for a total of one hour. Both sections of the study are videotaped and transcribed. The study involves eight architects. Six are from Canada while two are from the United States. For the purposes of anonymity, each designer is referred to by code C0*x* where '*x*' ranges from 1 to 8. Their professional experience ranges from 3 to 20 years (see Table 4.1).

Table 4.1: Designers' experience

ID	Professional Experience (Years)
C01	3
C02	5
C03	10
C04	15
C05	10
C06	5
C07	20
C08	20

The study begins with a set of documents sent to the designer prior to the study date which includes an introduction, a consent form, a set of sample questions and the sample design problem. The introduction provides a description of the research project, its status, the goals of the study as well as information about the researchers and the funding agency. It informs the designer of the intent of the study to respect the designer's privacy with regard to the dissemination of knowledge obtained from the study. The consent form advises the designer of their right to ask questions or to discontinue at anytime during the study. The designer is required to sign the consent form to continue the study. The questions asked in the study are made available to prepare the designer.

In order to establish a basis for reference and comparison, the sample design problem description is similar to the sample design session described in section 2.1. It presents the need for a residence on a 9x31 m (30x100-feet) plot of land. Spatial requirements consist of a minimum of three bedrooms, two bathrooms, living and family rooms, kitchen, external play area, two-car garage, workshop and office. The structure can be up to three levels in height, typical in the neighbourhood. Other residences are located on all sides except the front, which serves as an entrance. The designer is provided with setbacks and required to separate entrances to the living and working areas. A budget and a site plan is provided as well as information about the client. At the end of the interview, questions that the designer may have regarding the documents are answered before proceeding to the design session.

The interview consists of a mixture of structured and unstructured question-and-answer meeting [Newman and Lamming, 1995]. Table 4.2 shows the questions which are formulated based on the goal of developing an early design tool according to the following headings (with the number corresponding to the question): enumerating requirements (1), work methods (2,3,4,5), updating design (6) and ideal tool (7).

The study is conducted in each designer's work environment as information relating to how they work is richest there [Holtzblatt and Jones, 1993]. They are allowed the freedom to elaborate on any comments they make and any interesting pointers are followed. This means that for each designer, not all questions may be covered and the discussion may veer off in totally unpredictable directions.

During the design session, the designer is advised to treat the interviewer as the client or representative in order to provide a design solution as they would normally do in such situations. Ignoring some possible drawbacks (see designer's comment, Figure 4.1), they

Table 4.2: Interview questions

1. How do you go about information gathering at the beginning of your design process? How do you organize or arrange this information for use in the design process?
2. Do you start with hand sketches or do you go straight to the computer?
   - If you start with hand sketches, when do you transfer to computers?
   - If you go straight to the computer, how do you use it to explore design?
   - What computer programs do you use and how does it help you in design exploration typical in early stages?
3. When you begin designing, is it important to have boundaries of minimum and maximum numeric data, for example, by consulting professional graphic standards?
   - How do you use these boundaries? Do you draw the boundary using grids, for example, or do you establish the code requirements and then begin projections of your ideas from that base point? In other words how do you begin to draw on paper with this numerical information?
   - If you were to use grids, would you constrain yourself to them or use them as visual cues to the flow of the design and as references to the space sizes and other measurements?
4. Do you typically work in more than one medium? Sometimes schematics are accomplished in traditional methods while visualization is completed using the computer, both occurring concurrently. If so, how does the computer model help you and how does software assist you?
5. How do you explore or generate design alternatives? How do you create these alternatives, record them and put them together for the final design solution?
6. If there is a new product or the clients comes up with new information/changes, how do you handle this or update your design?
7. If you could have the ultimate software for design, what would be the main tool or functionality that would make your life as a designer so much easier and productive? In other words, what would be the best assistance you could receive during your design process?

were requested to speak their thoughts aloud [Laurel, 1990; Cross *et al.*, 1996] as this will

Here [in this study] I did not try anything that i could not do... [because i was talking] I was very safe. If I tried something I could not do, I would concentrate so much I could not voice my thoughts out. I did not try to find an intent that would be at heart... solution that would be very involving. It is like teaching... you cannot try something you could not demonstrate. There is something emotional about designing, the problem has to mean something to you which is not the case here. *Except... all the steps are something I would do, these are exactly my drawings...* so I followed the design process but my creativity was limited because I was speaking out loud.

Figure 4.1: Designer's comment on speaking aloud

not only be crucial in transcribing the work but will throw some light on the designer's unique view of the workspace and how/what interactions are taking place. According to the designers, commenting aloud affected the results of the design session but not the goals of this research as alluded to by the comments in italics in Figure 4.1.

Table 4.3 shows a portion of the transcription of the interviews which is achieved using three columns. The first column records the sequence of the questions. Each question can

Table 4.3: Sample transcription of interview for C03

Sequence **Local** **Global**	**Questions**	**Designer's Answers**
012 0:11.07 2:15.45	If you were... I know you have not used computers to design but I am sure you have heard a lot about what they can do and you have seen some simulations. If you were to say something that is very crucial or that you feel is missing from computers. What would it be?	Maybe there is another way to extract stuff... I have got this shape and I draw this line and then I continue as I draw with tracing paper [I keep some parts and maybe I have other ideas but they] fade away but you still have them very faint [C03008RS]. I think it should be on the screen. I think that if you can say that this is a wall and this is a wall, then it gives you a perspective and then I have got these two walls, I continue to play with this perspective, I got this window here and it shows there... I just have never thought of that before. If you say this is 1:50, it should be easy to get other information about the environment, then you can get the perspective of the building and the street.

be referred to by a local or global time. Two six-hour video cassettes are used showing the tapes duration in minutes (global time). A local time (time in minutes of the duration of each designer's session) is then added to provide an easy reference for each designer. A second column records the questions asked by the interviewer while a third records the answers from the designer.

Table 4.4 shows a portion of the transcription of the design session which follows the same structure as the interviews in the first column but without the sequence numbering.

The second column records the designer's actions while the third records the designer's comments. The third column also records the interviewers comments or questions in bold text. During transcription, the designer's drawings were marked up in areas that were found to be relevant to the study. Each mark is labelled with a unique ID for that sheet. The ID 'C0301H' for example refers to mark H on sheet 01 from designer C03's design session. Each action references the point in the designer's comments when it was performed. An example of a reference is [*1]. This shows a reference to number 1 action in that transcription row.

Table 4.4: Sample transcription of design session for C03

Time: Local Global	**Designer's Actions**	**Designer's Comments**
0:20.11 2:24.49	1. Designer indicates C0301H. 2. Designer indicates C0301I. 3. Designer indicates C0301J.	**So at this point you are working only in section... or you are moving between plan and section?** Yes... this is a bubble diagram[*1], this is the plan[*2]... then I will start to make volumetric study[*3]. I come here with a balcony... • You see I play with forms and it could be something totally different at the end. • I have got the balcony here at the top. Then I go back, [to the plan] this is the master bedroom... and the three bedrooms... for fast, fast work like this I should [work in] 1:100, now I have too much detail to go into.
0:21.09 2:25.47	Designer is using bubbles to demarcate spaces and functions at C0301K	This is my second floor, ground floor... [C03012RS]. I [will] try to put my stairway here... maybe in three flights like that or even like that [*]. Here I have got the living room, the kitchen and the dining... here. I can go down and have that family room here. If they want a fireplace in the family room... depends on what they want... here is a very small backyard... it would be nice to have some light... maybe we could come... something like that. With the sunlight we can get light into the master bedroom in the morning.

A preliminary study of the transcripts was performed in which relevant features or methods regarding the design session are extracted by reading through the transcripts. These provide interesting highlights to be noted for later in a more in-depth study. They are identified with labels similar to [C03012RS]. The first part of the label identifies the designer in whose session it was discovered (C03), followed by a sequence number with respect to the overall features or methods obtained from that designer's session (012) and two letters to show that it is a possible required specification (RS). The recording of the feature/method 'C03012RS' is shown in Table 4.5. The first column shows the

Table 4.5: Sample features/methods extraction for C03

Sequence:	**Interesting Feature/Method**	**Inspiration**
012	Designer always makes a tally of all spaces or requirements completed. This is done preferably using graphics or diagrams but also can be checked off using text.	At 2:25.47 – see [C03012RS]
013	Needs a window for each sketch view (I am thinking of the screen real-estate possibilities and the need for the architect to see all views at the same time). What is the best means of navigation in a large drawing? Sketches are usually small though so we are not talking about really large but still... Definitely a view on each drawing. These views should hold steady on the screen but allow the designer to expand or decrease them. it should also be located relative to its actual position for orientation.	At 2:28.44 - see [C03013RS]

sequence in which the item was identified in the designer's session. The second column describes the item while the last column refers to the point in the transcripts where the feature/method was extracted. This is done by noting the time and the reference as explained above.

4.1.2 Results

The design sessions show designers going through stages in the design process identified as *activities* (effort towards an intermediate goal) and using actions at particular times within an activity identified as *events* that guided or influenced the activity. First, four main activities are identified and described as follows:

- **Design brief.** The designer tries to understand the nature of the client's needs and the resources that are available. This information is organized in an accessible manner depending on the designer's style and referred to numerous times during the process.

• **Site preparation.** The site is determined by establishing setbacks, adjacent construction, sun path and available footprint.

• **Building space.** The relationship between spaces and elements within the building are resolved often using simple bubble diagrams. This is where the designer produces possible solutions for the design problem. The designer combines several types of visual aides such as diagrammatic elements, plans, sections, elevations and volumetric studies. In addition to geometry this activity considers abstract items such as adjacency, circulation, relationships, perceptions, proximity and character.

• **Building elements.** This activity considers more tangible items such as walls, floors and furniture and places them on the sketch.

Table 4.6 describes the recognised events and their codes, while Table 4.7 shows how

Table 4.6: Event codes

Code	Description
3	Three-dimension drawings for studying or exploring ideas
A	Designer makes an alternate sketch
B	Designer backtracks to a previous point
E	Designer creates an elevation
G	Designer groups spaces or requirements using bubbles
L	Designer labels drawings
N	Designer begins drawing on a new sheet
S	Designer shades or renders drawings
X	Designer works in a section
˜	Event combination

Table 4.7: Example combinations of event codes

Code	Description
B˜N	Designer backtracks to a previous point using a new sheet
B˜N˜A	Designer backtracks, uses a new sheet and starts alternative idea

codes may be combined to show simultaneous or closely sequenced events. Summaries of each designer's session are now presented, with a diagram showing their activities and events.

C01 (see Figure 4.2) spends a few minutes tabulating the requirements provided in the design brief. These tabulations bring together all the important data and provide an easy reference during design. The designer quickly establishes the available footprint on the site before starting the building space activity. Building elements such as windows, doors and furniture are included at the end of the session and the designer makes two major changes in design development at about 25 and 35 minutes in the session.

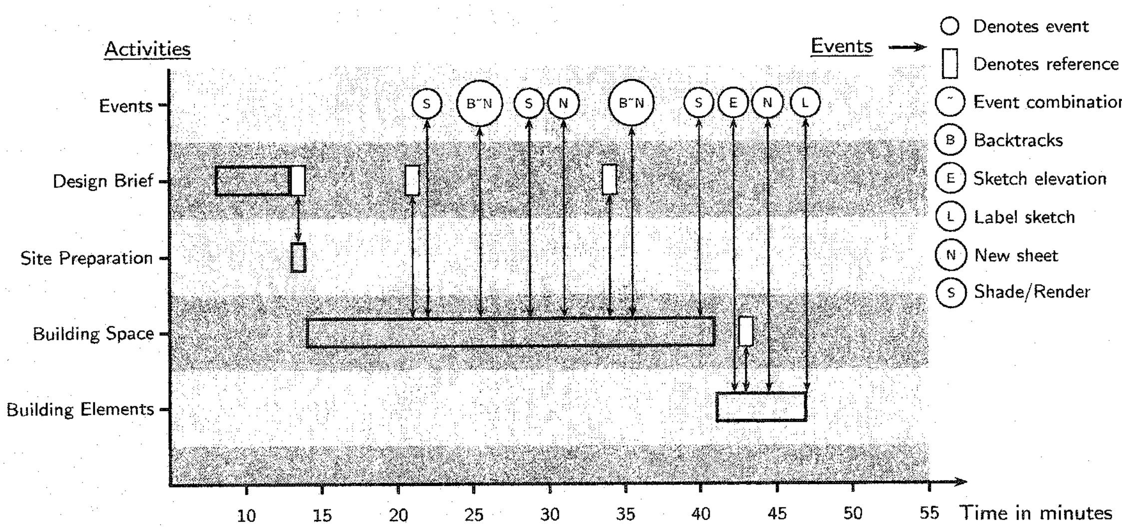

Figure 4.2: C01–Design session

C02 (see Figure 4.3) spends a few minutes asking questions about the design brief before going into building space activity. Some relevant data is jotted down on paper as a reminder, however these are random, uncoordinated events. Building space design proceeds in plan and section simultaneously and the designer spends time to configure the building elements in some of the spaces. Some time is spent at the end of the session on the overall elevation and its building elements such as doors and windows. The designer works on a single solution although there is one major change in development at about 33 minutes into the session. *C03 (see Figure 4.4)* reads the brief very quickly before begin-

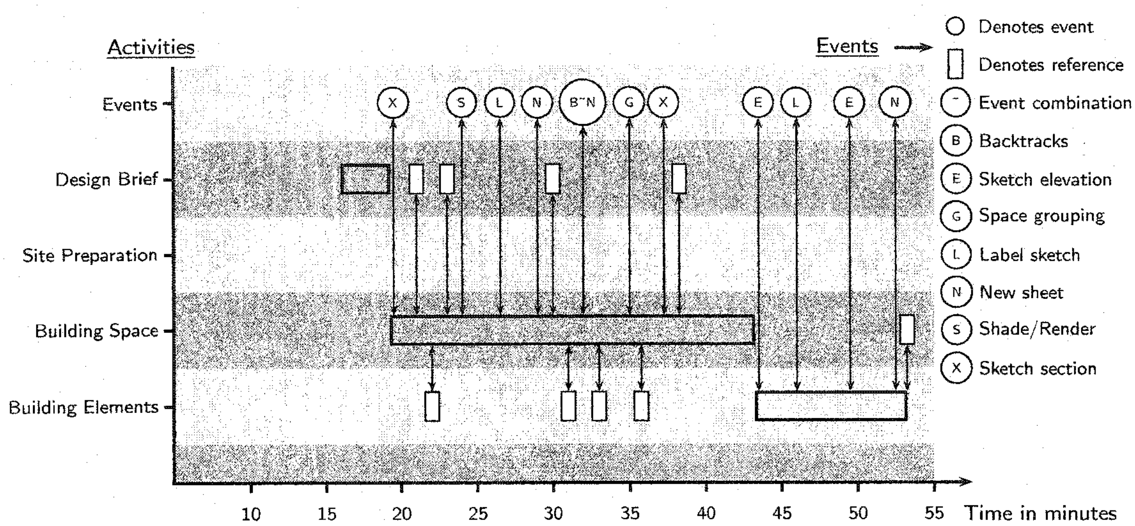

Figure 4.3: C02–Design session

ning to work on the site preparation. The brief is referenced continuously as the available building footprint and setbacks are established. The building space activity begins with plans, sections, grouping of spaces using bubbles in sectional diagrams and volumetric studies. After a satisfactory configuration of a particular space, the designer spends a little time elaborating on the building elements for that space. The designer works on a single solution although many options are created for different parts of the design.

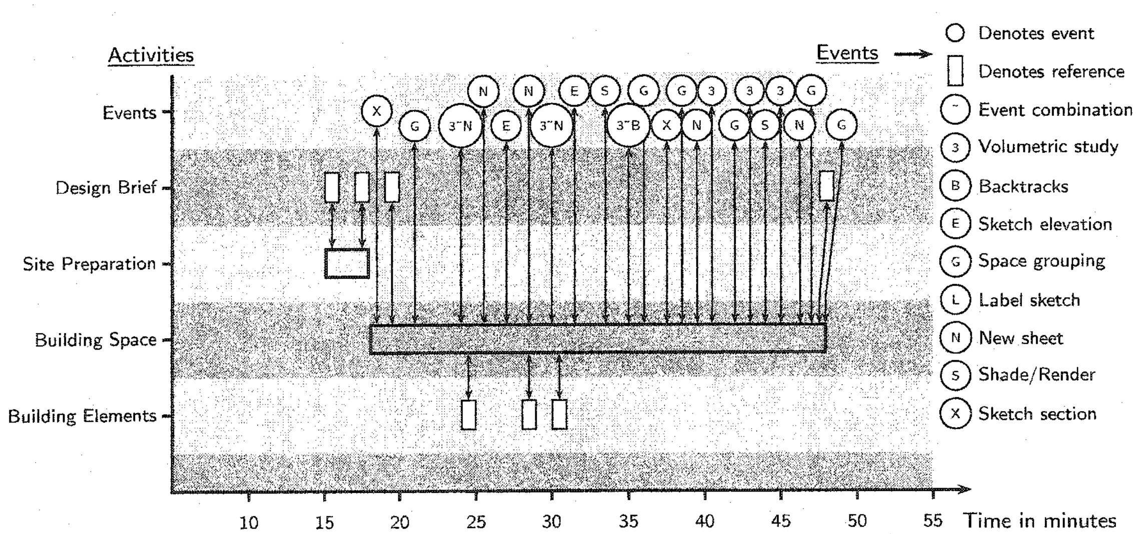

Figure 4.4: C03–Design session

C04 (see Figure 4.5) begins by reviewing the bylaws and the brief to collect requirements. Data from the brief is then used in the site preparation where the designer identifies the available footprint and renders the sun paths (winter and summer). After configuring each building space, the designer illustrates the building elements found in the spaces. This serves mainly to validate the decisions surrounding the building space design. The designer works on a new design alternative after the completion of the first alternative, starting at 38 minutes. The designer concludes the session with 3D volumetric study of facades.

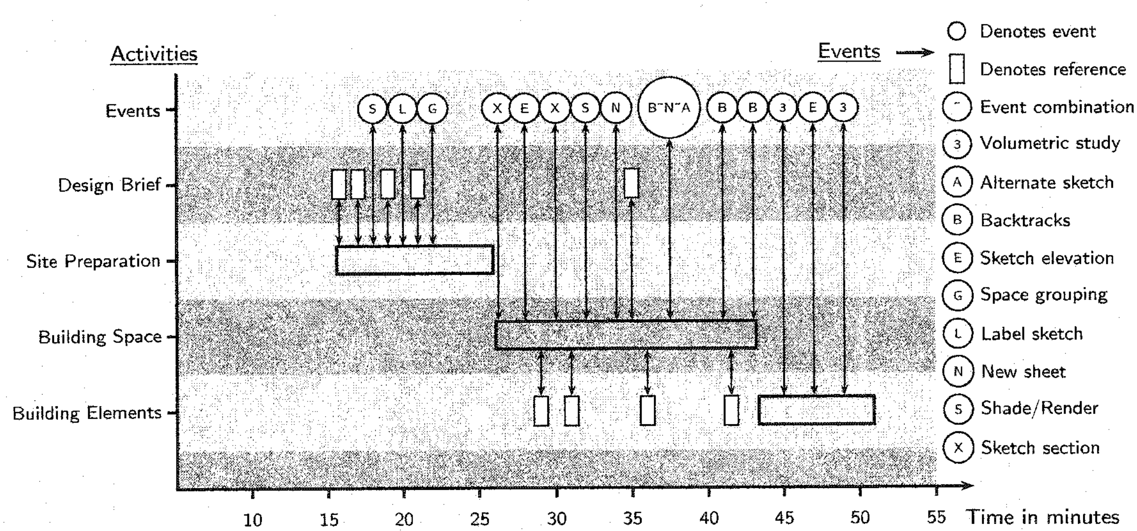

Figure 4.5: C04–Design session

C05 (see Figure 4.6) takes the time to understand the design brief and discusses the requirements thoroughly with the client (i.e. the researcher playing the client) using a combination of labels and bubbles. A quick site configuration is achieved with more bubbles and shades, which are elaborated in the building space activity. At certain times the designer designates or introduces building elements like a stair, a door or a window within the boundaries of a bubble. These act as icons or reminders for a later stage when the sketch will be drafted. This designer prefers a verbal and extremely abstract mode of initial design. Such a method starts with vague issues, which are detailed in subsequent iterations.

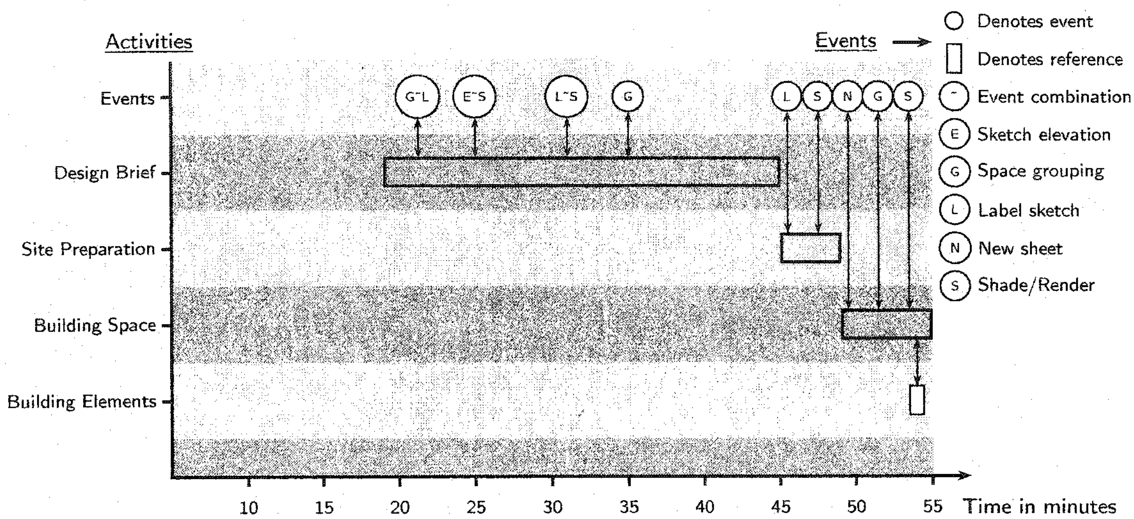

Figure 4.6: C05–Design session

C06 (see Figure 4.7) extracts data for the site preparation from the design brief and moves onto design space activity where most of the session time is spent. In the building space activity several references to building elements are made in an effort to elaborate the spaces with items such as furniture. The designer explores a second alternative at about 33 minutes in the session because of unresolved problems.

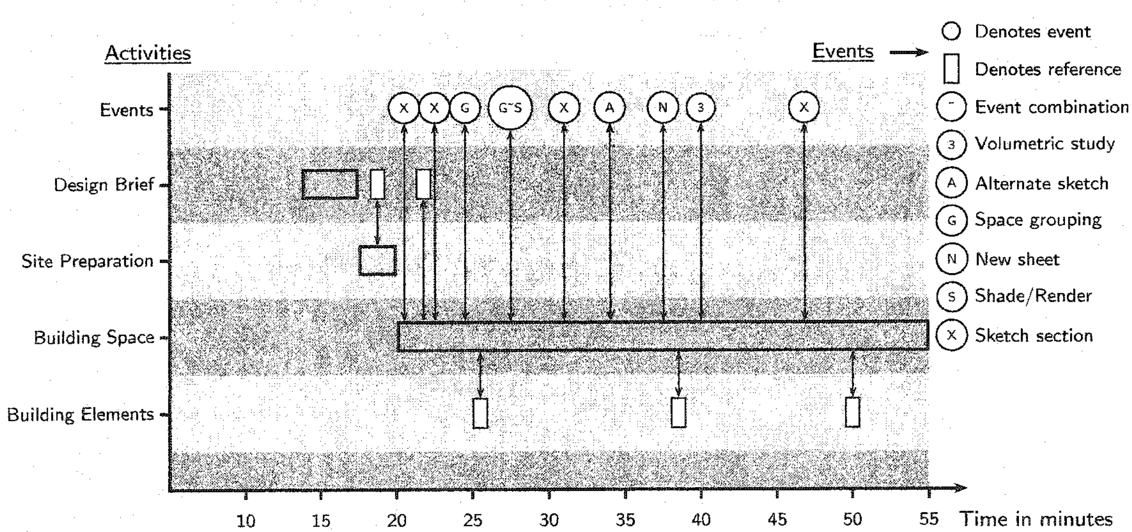

Figure 4.7: C06–Design session

C07 (see Figure 4.8) starts with a systematic analysis of the design brief. All requirements are tabulated and summarised. Maximum footprint as well as floor area, possible levels and total cost of building are estimated from the brief. The designer then provides isolated configurations for each building space by combining information obtained from queries to the client, references to the site and other requirements. This allows the designer to collect a snapshot of all spaces with their interior organisation and finishes. This also provides an early transition to the building elements activity in comparison to other designers in the study. These spaces are then abstracted for easier manipulation. The designer spends a few minutes illustrating the available footprint before starting the building space activity using the abstractions in the form of a combination of quadrilaterals, circles or ellipses. After a successful design is selected, the abstract forms are exchanged or substituted for their detailed version from the earlier isolated configurations. This illustrates a unique approach to all other designers (especially C05) and resembles a bottom-up design approach.

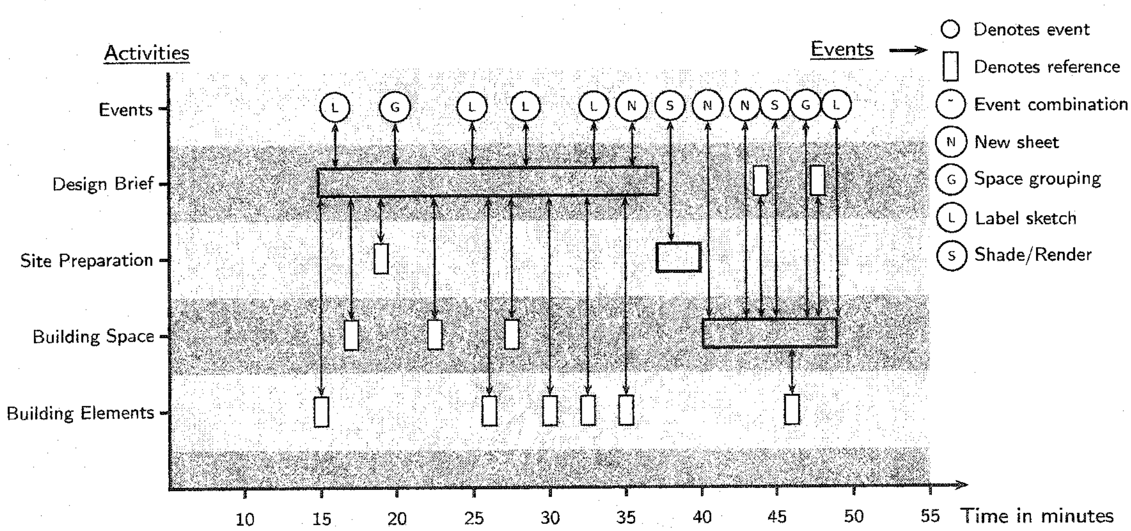

Figure 4.8: C07–Design session

C08 (see Figure 4.9) starts by studying the brief before preparing the site for building space configurations. The designer remains mainly in the building space activity but references to the building elements and design brief are made, as parts of the design are resolved.

The designer reverses direction due to a misunderstood requirement at about 24 minutes and creates an alternative to a part of the building at about 36 minutes. This designer has used the greatest number of new sheets compared to the others due to the preference of problem resolution using layers. This provides ample flexibility in creating alternatives and an easy way of introducing new ideas.

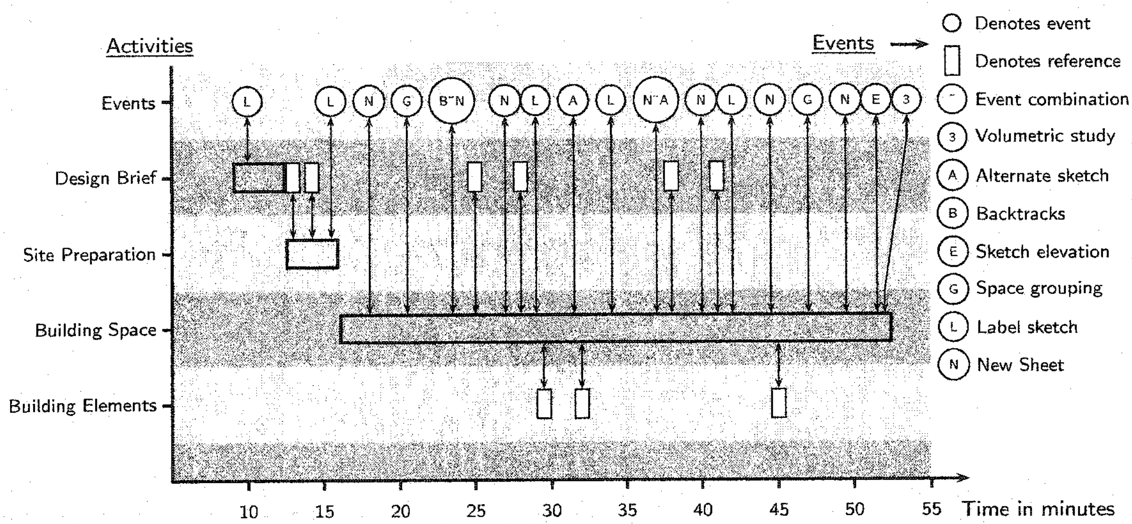

Figure 4.9: C08–Design session

4.1.3 Observations from Study

Using the transcriptions of the designers' actions during the different design sessions and their analysis, the following observations can be made.

1. The designers performed all four activities. For the first two (design brief and site preparation), the designer defines and sets the boundaries for the design problem. In the next two (building space and building elements), potential solutions are developed. The designer may refer to more than one activity in order to create a design item, such as looking at the site (site preparation) to determine the noise sources and the design brief for owners requirements when locating and configuring a bedroom space (building space).

Table 4.8 shows the number of references made to and from an activity. To obtain

Table 4.8: References between activities

Activity	References To [frequency (%)]	References From [frequency (%)]
Design brief	12 (50)	3 (13)
Site preparation	1 (4)	5 (22)
Building space	2 (13)	14 (61)
Building elements	8 (33)	1 (4)
Total	24 (100)	23 (100)

the numbers in the table each reference is counted only once with regards to the activity it is coming from. For example in Figure 4.9 there are only two references to the design brief activity, one each from the site preparation and building space activities respectively. Table 4.8 shows that most references are made to the design brief and to the building elements activities. This is consistent for all designers except C07 who did the opposite (i.e. most references were made to building space from design brief) showing a bottom-up approach.

2. The events show, in more detail, the actions the designer takes during the activities and may give hints to the type of tool or assistance that the designer requires to accomplish the activity. From Table 4.9, the 4 most frequent events are the use of new sheets (N), the grouping of spaces or requirements (G), use of labels on sketches (L) and shading or rendering of drawing element (S).

3. All designers (except C07) adopted a top-down design style i.e. starting with high level (abstract) design decisions, then providing details for these decisions as the design session progresses. C07, on the other hand, has used a bottom-up approach that begins with detailed decisions, which were then used as references or templates in the development of design options.

4. Five designers (C01, C04, C05, C07, C08) began by scrutinising the brief and usually rearranged or reorganised requirements into an accessible format for easy reference

Table 4.9: Occurrences of events

Event	Occurrences [frequency (%)]	Design Brief	Site Preparation	Building Space	Building Elements
Volumetric Study (3)	10 (9)	0	0	8	2
Alternate Sketch (A)	4 (3)	0	0	4	0
Backtracks (B)	8 (7)	0	0	8	0
Sketch Elevation (E)	9 (8)	1	0	4	4
Group Spacing (G)	18 (15)	3	1	14	0
Label Sketch (L)	17 (14)	7	3	5	2
New Sheet (N)	27 (23)	1	0	24	2
Shade/Render (S)	15 (13)	2	3	10	0
Sketch Section (X)	10 (9)	0	0	10	0
Total	118 (100)	14	7	87	10

during the design session.

5. All designers extracted the building footprint, by-laws and other parameters from the design brief to determine the orientation of the proposed structure.

6. All designers used multiple views for the configuration of the building spaces. The most common of these views are plans and sections. In addition four designers (C01, C02, C03, C04) used elevation drawings while two (C03, C04) explored solutions using 3D views.

7. All designers used simple diagrams to show and investigate relationships among design items such as 'bubble' diagrams.

8. All designers followed a design process that displayed several solution path branches. An example is explained in section 2.3.3 and shown in Figure 2.10 where the designer develops a straight staircase but later branches into a circular version.

9. Some designers (C04, C06, C08) relied on the development of alternative solutions, parts of which could be later merged to produce a successful solution. An example is explained in section 2.3.3 and shown in Figure 2.10 where the parts of one already

developed staircase, such as threads and risers, were used in configuring the alternate one.

10. Designers considered and undertook the design process differently [Eisentraut, 1999] because of perceptions and experience.

4.2 Specifications for Early Design Support

This section enumerates specifications for the development of a computer-based approach to assist designers in the early stages of building design. These specifications emerge from the protocol study described above as well as from conclusions drawn from the literature review in Chapter 2. The format followed presents a title for the specification, a summary of the surrounding issues and a set of itemized instructions. References are made to the previous section which provides summaries of observed characteristics as well as to relevant references from literature.

4.2.1 Requirements Repository

<u>Summary</u>: The design brief describes the design problem, goals and available resources. All designers rearrange this information for easy reference which is kept readily accessible throughout the entire design process [Fricke, 1999; Günther and Ehrlenspiel, 1999] (See items 1 and 4 in section 4.1.3).

<u>Specification</u>:

A computerized approach should:

- Provide the ability to enter requirements from the design brief, standards, codes and other sources.
- Allow designers to organise these requirements in a format that is convenient to them (e.g. as a list or a hierarchy).

- Provide easy access to the requirements so that the designer can refer to them throughout the design process.
- Allow the designer to establish a relationship between two or more requirements. For instance, the desire to maintain the same floor levels between two spaces.

4.2.2 Application of Requirements

Summary: Designers C01, C03, C04, C07 and C08 use information from the design brief to apply in the design. For instance C07 worked out minimum areas for each required space and used this information as a guide in creating the forms for both Site Preparation (available footprint) and Building Space activities. As such, adherence to the right scales and sizes is maintained during configuration without the use of detailed and cumbersome diagrams [Fricke, 1999; Günther and Ehrlenspiel, 1999] (See items 5 and 7 in section 4.1.3).

Specification:

A computerized approach should:

- Allow the designer to quickly extract the site parameters and isolate the minimum areas available for design. To do this, it should be possible for the designer to scan the site or input the boundaries in a convenient scale to which the system could apply information such as setbacks, automatically.
- Certain knowledge may be part of the requirements which may be set during the design problem definition (design brief) or site planning/analysis (site preparation). A general method for applying, checking or monitoring the requirements against the design such as preliminary total cost of materials or total size of circulation space should be developed.
- Requirements that change such as later addition or subtraction of spaces in the design, should dynamically be reported and updated in the requirement repository. Changes in the requirement repository should also reflect or be applied in the design.

4.2.3 Multiple Levels of Abstraction

Summary: C03 uses different design representations. For instance a plan view may be drawn with furniture while at other times empty spaces with only walls might be more adequate [Purcell and Gero, 1998; Goldschmidt, 1994; Eisentraut, 1999]. Also different designers like architects and engineers make use of different views. For instance, the structural engineer may want to focus on the structure only and abstract all other non-structural items out of the view. Extra information only tends to distract and complicate the design environment (See items 6 and 10 in section 4.1.3).
Specification:

- A computerized approach should provide the ability to present design views at different levels of abstraction, for example walls can be represented with single lines to reduce the drawing effort. At the same time it should be possible for the system to show the edges of the walls if the designer so desires. This is important in a situation where the architect only wants to see the spaces while the engineer wishes to see the location and perhaps size of the walls.

4.2.4 Solution Management

Summary: Designers C02, C03, C05, C07 and C08 keep records of the design progress so that they can backtrack [Atman *et al.*, 1999] to an earlier point in the design session or branch off towards a different goal. This action is desirable and creates alternative design solutions [McGown *et al.*, 1998; Verstijnen *et al.*, 1998]. Designers do not eliminate these alternatives but rather keep them so that these may be revisited at some later stages in the design session (See items 8 and 9 in section 4.1.3).
Specification:
A computerized approach should:

- Save every important step (or *version*) of the design evolution as well as any branching that occurred to facilitate backtracking.

- Provide a way to bookmark steps in the solution path for later reference, comparison or backtracking.

4.2.5 Element Interaction

Summary: All designers interact with design items in dynamic ways as they create, adjust and manipulate them. When the designer creates items in the early design stage, detailed considerations are not made in terms of technical details. For example when creating a window in a wall the designer does not necessarily pay attention to the details of how they should fit as the important issues at this stage are the position and relative size. These are interactions that must occur between the objects at this stage. At the same time there are interactions between the designer and the items being created. When an item is created there must be a certain feedback to the designer whether the parameters surrounding its creation are satisfactory as well as a reaction when the item is moved or when a relationship with another item does not go well. Finally, designer may want to set relationships between items on the fly such as two items maintaining a fixed distance from each other. (See items 2, 3 and 7 in section 4.1.3).

Specification:

A computerized approach should:

- Provide for how design items resolve their relationship in order to exist together, for example a wall accepting a window being placed in it.
- Provide for how usage issues and parameters are resolved for the design items, for example how designers manage the creation and manipulation of spaces.
- Finally, provide for how new relationships are addressed by the designer for example maintaining a particular line of sight between two building components.

4.2.6 Automatic Feedback

Summary: All designers remain alert to clues and opportunities in the design. They also try to anticipate limitations and problems that may hinder the success of the design. One way in which they do this is by checking the design at intervals and performing "what if..." scenarios every now and then [Heylighen *et al.*, 1999]. For example if the designer wishes to know how many items would occupy a particular surface area, it should be possible to automatically replicate these items visually. Designers can visualise scenarios in their mind but often prefer to make investigative sketches away from the main design area. These separate visualisations provide a richer feedback because of the isolation of the design item/scenario, the freedom to make new design items at new view angles and the possibility of using three-dimensional views. These visualisations are usually saved and situated in prominent places such as on walls, where the designer often refers to them as design continues (C03, C04, C05 and C07 during interviews).

Specification:

A computerized approach should:

- Provide an interface for performing "what if..." scenarios and integrating the results back into the design, or storing the results for later reference.
- Provide means through which the consequences of design decisions are shown, such as daylight values in a space given a window size, or the path of a shadow over the course of time.
- Provide direct tactile interaction with design items in order to enhance the designer's perception. For instance Haptic [Haptic Technologies, Inc, 1999] offers "the CAT® (Computer-Assisted-Touch)" force feedback technology which provides real-time feedback of the form, texture and reactions to external forces from digital objects.
- Provide the designer with non-intrusive feedback when errors or seemingly impossible actions or manipulations are detected.

4.2.7 Design Overview

Summary: All designers produce many sheets of drawings and refer to them frequently. Computer monitors provide smaller view screens in comparison to drawing sheets and designers often find it difficult to locate and orient their focus (See items 1 and 6 in section 4.1.3).

Specification:

- A computerized approach should provide the designer with a persistent bird's eye view of the design. The computer screen is limited in size but an overview could allow the designer to navigate to parts of the hidden screen without being disoriented or lost. The overview should be independent of the other view(s) used for active design.

4.2.8 Design Liability

Summary: Designers assume full responsibility for design decisions [Emkin, 1998] as these decisions originate from academic education, professional/personal experiences and style. No designer is willing to delegate professional responsibility to the computer according to the answers given by the designers to the last question of the interview (see Figure 4.2).

Specification:

- A computerized approach must not "design" for the designer. All changes or ideas it introduces automatically must be examined and approved by the designer before inclusion in the design.
- A computerized approach should provide a means for the designer to compile a list of "to-do" items possibly with extended notes.
- Designs are sometimes driven by experience and styles. A computerized approach should be customisable to allow the designer to specify how certain actions or details

should be presented or allowed to come together. In this way the designer's personal style and preferences may be preserved.

- A computerized approach should have an automatic logging process that keeps track of the designer's identity for any work or changes made. Access to these logs should be protected and the activity should be cumulative.

4.3 Summary

The sample design session and literature review of early design research carried out in Chapter 2 revealed certain requirements that were not adequately addressed in existing solutions. Such deficiencies can be viewed to represent the "what" issues concerning early design development. The protocol study and analysis presented in Chapter 4 make it possible to explore the "how" issues regarding possible support by a computerized approach.

Analysis of the protocol study shows that the early design process can be divided into four activities: design problem definition, site planning/analysis, building space configuration and building element configuration. The activities progressively contribute to design solutions using events. Activities suggest design environments or modules in which the designer interacts with the building while events suggest tools available in each environment or module for manipulating or directing the process.

The typical design process starts from problem definition (design brief) and proceeds as the building footprint is extracted during site planning/analysis (site preparation). The design problem is then explored for solutions in iterations during the building space configuration. Once a solution becomes more established, the designer begins to introduce building components in an attempt to convey a more finished view of the solution. Events are used in controlling this process such as those shown in Figures 4.2 to 4.9. However, there are others that could not be labelled or included in the graphs such as: the designers' dislike for interruption or distraction (such as having to talk or search for knowledge not immediately available from experience), need for free flowing drawing motions with little or no restrictions, need to use feint or vague initial lines or marks that can be emphasized or detailed later and the need to save drawn items that are not necessarily complete in what they depict. It should be mentioned that the most time spent in this process is in the building space activity which is also where the solution is first realized.

The information contained in this chapter describes features that should be provided in a computer tool to support the early design process. As a guide to this overall complex goal, a set of specifications is drafted based on the results of the protocol study/analysis. These specifications describe fundamental issues that are present in the early design sessions and which must be addressed in order to provide appropriate computer support. Following each description are directives that are meant to guide the development of the support.

The next chapter presents the use of a subset of these specifications to formulate computer-based support for design by means of the features discovered in this chapter and also to support the nature of the early building design process described so far in the research project.

Chapter 5

CoBL DT: Planning

HAVING ESTABLISHED THE necessary requirements in section 2.3 for supporting early building design in computers, relevant features to satisfy these requirements have been extracted in Chapter 4. This chapter utilizes a subset of the specifications described in section 4.2 to create a system called Conceptual Building Design Tool (CoBL DT) to support early building design.

This chapter begins by describing and justifying the specifications utilized. A general overview of the proposed system is presented in section 5.2 followed by a section on the formulation of the main functions of the system. These are presented with a set of illustrations which includes the introduction of the main objects (see section 3.3 for objects definition) that will make it possible to realize the illustrated support. Section 5.4 formulates the knowledge that is provided in the system which fulfills the knowledge integration requirement stated in section 2.3.5. The analysis and design of the functions provided in CoBL DT is provided in section 5.5 and a conclusion highlighting the main components and functions of CoBL DT is provided in section 5.6.

5.1 Specifications for CoBL DT

Due to the finite resources available to this research project in terms of money and time, it was necessary to impose limitations to our considerations and goals. One of these is the number of specifications utilized in the design of CoBL DT. In choosing the specifications to use, we considered the existing support for the requirements determined in section 2.3 as summarized in Table 2.7. It is immediately obvious, by looking at the Interactive Design Exploration System category, that there is a need to work out a solution for collecting design alternatives and providing proper design overview which are specified in Solution Management and Design Overview respectively (see section 4.2). The next requirement that is least supported according to the table is interface issues. To provide support for interface issues, the specifications that should be used are Application of Requirements and Element Interaction. We feel that Element Interaction will not be complete without the necessary hardware such as that provided by Haptic Technologies [Haptic Technologies, Inc, 1999]. However, such hardware was not accessible to this research project. Table 5.1 shows the list of specifications which were fully utilized in bold text.

Table 5.1: Use of specifications in designing CoBL DT

Utilized Specifications	
1. **Application of requirements**	Fully utilized
2. **Solution Management**	Fully utilized
3. **Design Overview**	Fully utilized
4. *Multiple Levels of Abstraction*	Partially utilized
5. *Element Interaction*	Partially utilized
6. *Automatic Feedback*	Partially utilized
7. Requirements Repository	Future consideration
8. Design Liability	Future consideration

There are other specifications that cannot be totally ignored. In designing to support the collection of solutions and the provision of appropriate overview, the system must be able to provide multiple levels of abstraction as specified. Also, element interaction is an important part of applying requirements to the design process while obtaining appropriate feedback is critical. So there is a need to partially handle specifications to increase the chances of providing a functional system although they have not been fully considered. These specifications are shown in Table 5.1 as partially utilized. The last two specifications are beyond our ability to make a contribution within this research project so have been left out for future consideration.

5.2 System Overview

COBL DT is divided into four parts as shown in Figure 5.1 and presented according to how these will interact with each other. The first part assists the designer in drawing

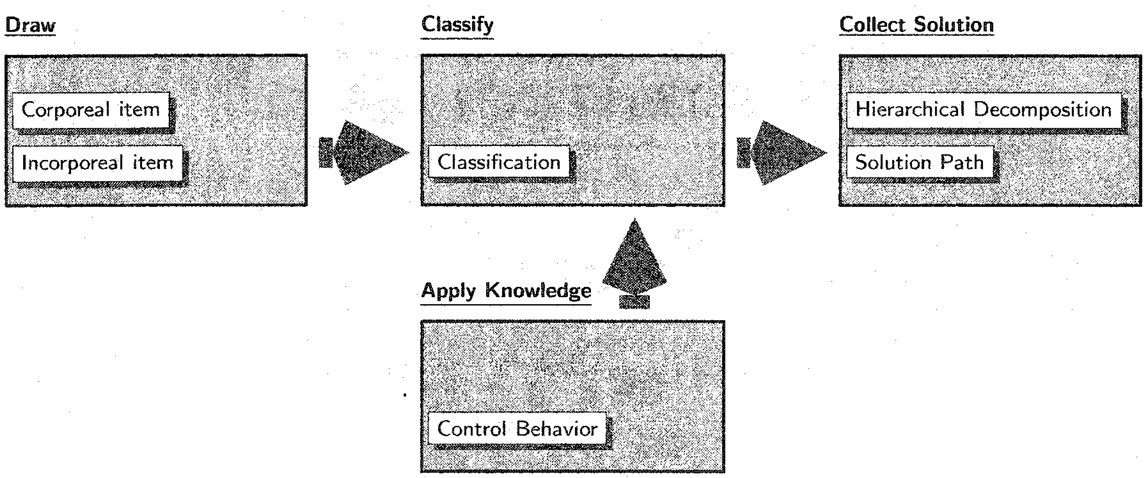

Figure 5.1: Overview of COBL DT functions

(**Draw**) a design item. The second part provides assistance in recognizing (**Classify**) and assigning a role to the item (see section 2.3.2 for a definition of role). The third part either provides additional knowledge (**Apply Knowledge**) to the item or allows the designer to establish additional knowledge or requirement. The fourth part captures the solution (**Collect Solution**) so that the designer can review or come back to it if desired.

The Draw part enables drawing by providing two types of objects. The first is called a CORPOREAL design item type. Objects of this type are used to denote usage of a space or a design feature. For example, it can denote a living room space or a stairwell (vertical circulation space). The second object type is called an INCORPOREAL design item. This item type is used to represent all other design items such as ordinary lines, marks or orientation/direction. As a result of the recognition that early design is a process in which the designer works mainly with spaces (see section 2.3), this research project concentrates only on corporeal item types as a further effort to limit the scope of the thesis. For the rest of this thesis Corporeal and Incorporeal design item types will be referred to as *design items*, or where necessary *corporeal item* and *incorporeal item*. A corporeal item is recognized as an enclosed shape by the system so if the designer does not complete the enclosure, the system will automatically close it. The four steps in Figure 5.2 show the creation of a sample corporeal item. Incorporeal items on the other

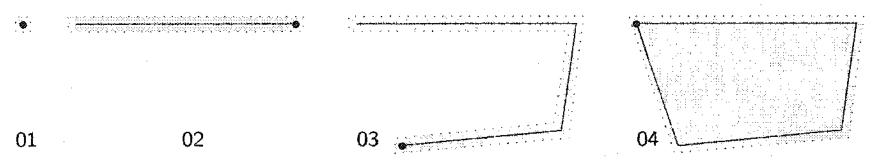

Figure 5.2: Creating a space

hand are not recognized as enclosed items even if an attempt is made to close them. They are in fact shown as single line items.

The Classify part makes it possible for the designer to accurately identify the design item created. This identification process is called CLASSIFICATION and is similar to the labeling action observed in the protocol study in section 4. When the designer wants to classify an item, the system presents a list of names (from the design project requirements) based on contextual information surrounding the item. The designer can then select the desired name. This name, such as "Living room" is used by the system to access appropriate characteristics and behavior necessary for a living room space to support its typical usage. This information is then transferred to the design item allowing it to participate appropriately in the design process.

Once a classification occurs, the designer can configure or add to the typical characteristics and behavior of a design item using the Apply Knowledge part. For example a typical Living Room has boundaries that define the space it is using for its function, using the Knowledge part the designer can remove one or more of these boundaries. For an example of behavior, the designer can ask the Living Room space to monitor its solar savings factor and warn the designer if it goes below or above a certain value. This type of behavior is discussed in more detail in section 5.4.

The Collect Solution part automatically captures design items in real time during the designer's design session. There are two functions provided in this part. The first is the HIERARCHICAL DECOMPOSITION of a solution so that the designer can have an appropriate overview and access to each design item in the solution (see section 2.3.4). It is possible to have more than one solution collected (which is discussed in section 5.3) but the decomposition is only applied to the current solution. The second function of this part is the actual collection of solutions which is accomplished using SOLUTION PATHS (see section 2.3.3). Solution paths monitor the designer's progress in the design session capturing all classified design items (as opposed to decomposition that includes unclassified design items), as explained in more detail in section 5.3.

The following sections provides information on how these parts are used by the designer in an early design session.

5.3 Formulation of Functions

This section describes the functionality of the system using views or windows of the data being manipulated as illustrations. Each window is a view of the functions illustrated in Figure 5.1 namely **Draw Window** (Draw function) abbreviated DW , **Classify Window** (Recognize function) abbreviated CW, **Hierarchy Window** (Collect Solution sub-function) abbreviated HW and **Path Window** (Collect Solution sub-function) abbreviated PW. The Apply Knowledge function is described in section 5.4.

The description is provided using steps that represent an actual design session. A written description of the events is given as well as graphical illustrations showing the windows the designer is interacting with.

5.3.1 Step 01

(Figure 5.3). At the beginning of the session, CoBL DT presents a blank screen in DW with a list of names for spaces that can be created in CW. The HW depicts the current

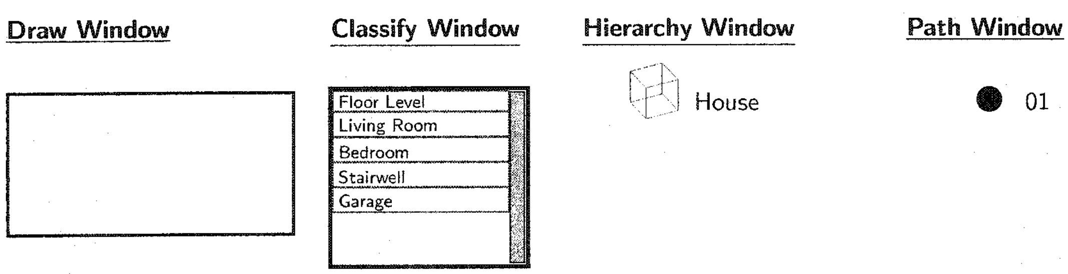

Figure 5.3: Step 01

state of the design session which is an empty space for solving a design problem which is a "House" . CoBL DT captures the state of the current solution as a progression of steps in PW as illustrated by the first dot in this Step 01.

5.3.2 Step 02

(Figure 5.4). The designer draws a shape in the DW at (A) and selects the item "Living room" for its classification at (B). The system automatically adds this classified item to the hierarchy tree as shown in the HW at (C) and this progress in the solution of the design problem is shown by the addition of a new step in the PW at (D).

5.3.3 Step 03

(Figure 5.5). The designer draws a second shape in DW at (A). Since this new shape is not contained in any other space, the space names offered in CW are similar to that in

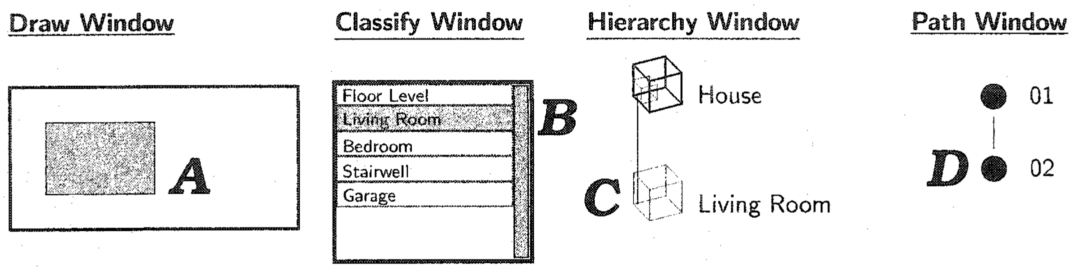

Figure 5.4: Step 02

Figure 5.4 at (B). This new space is added to the hierarchy window as shown in (C) but there is no change in the PW because this new item is not classified and does not provide any obvious improvement or additional feature to the current solution. Designers often draw shapes that may not be used in the final design solution.

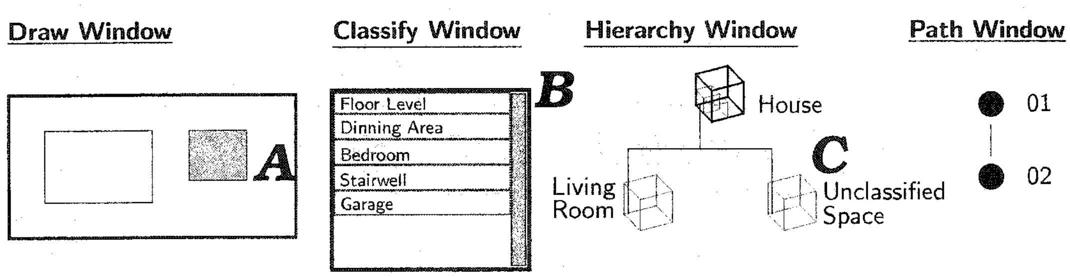

Figure 5.5: Step 03

5.3.4 Step 04

(Figure 5.6). A new space is drawn inside the living room space in DW at (A). The system presents space names in CW. These different choices of names are provided because the new space at (A) exists within the Living room space. The designer makes a choice at (B) causing the system to add the newly classified space in the HW at (C) within the living room space. Finally this progress in the design session is captured in PW at (D).

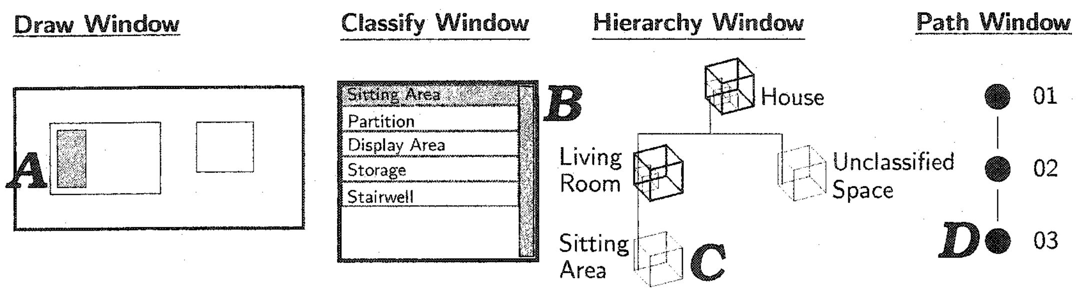

Figure 5.6: Step 04

5.3.5 Step 05

(Figure 5.7). The designer draws a space that encompasses some spaces in DW at (A) and classifies this new space in CW at (B). The system accepts this classification in HW at a higher level (C) than the existing rooms because it is a floor and it encompasses them. The solution path information is then updated in PW at (D).

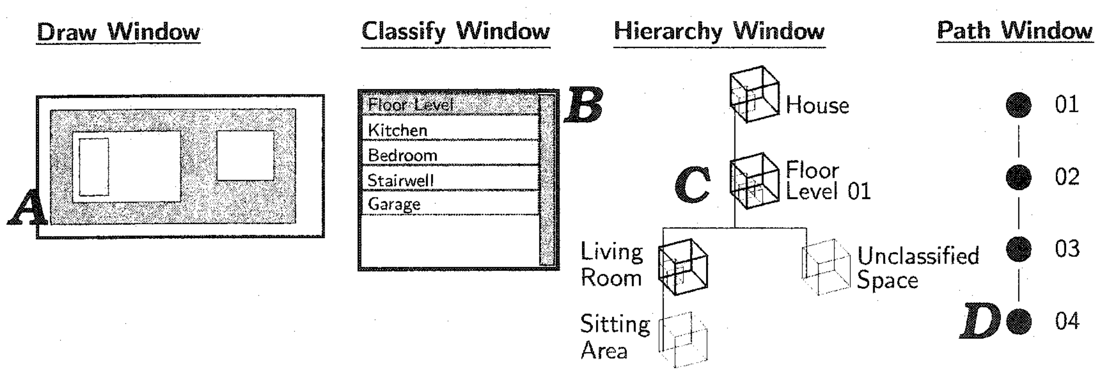

Figure 5.7: Step 05

5.3.6 Step 06

(Figure 5.8). The designer modifies the sitting area in DW at (A) by introducing an alternate version of the solution created in Step 04. No item is classified in this step and the data in HW does not change however the alternate solution is automatically recorded by the system in PW at (B). The first solution explored is recorded in PW by the path 01–04 while the alternate solution is recorded by the path 01–03 and 05.

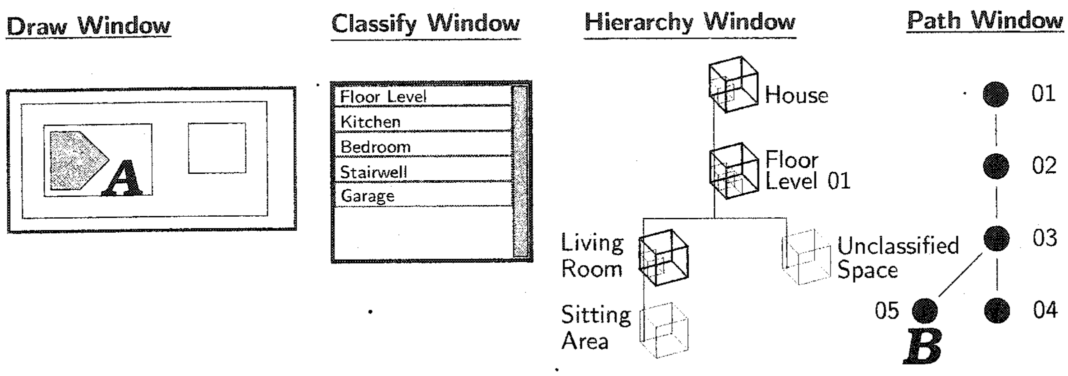

Figure 5.8: Step 06

5.4 Formulation of Knowledge

As observed in section section 4.3, designers tend to resist diverging their attention to issues not immediately within their domain or capability during the early design session with the intention of seeking this knowledge or advice later. CoBl DT will provide some assistance so that certain rules of thumb can be available without the need to stop the design session or ignore/postpone these important issues.

Knowledge can be normally applied through the role of a design item such as the "Garage" design item checking to make sure it has enough space to accommodate a standard car. Knowledge that affects two or more design items however is coordinated by a **Utility item**. Utility items, such as solar savings factor utility, are special purpose design items that are created to coordinate interactions between design items in order to obtain certain results for the designer.

Attributes or variables required by the knowledge is provided in a central repository file which is accessible to both design items and utility items. This separation makes it possible to freely update the attributes when needed without having to update the items used in CoBl DT. It also makes it possible to change the type of attributes as needed for each project. For example metric attributes may be changed for imperial without much difficulty.

There are two ways that knowledge can be used, which are for **checking requirements** and for **checking properties**. The following subsections provide explanations of how knowledge can be used in these two ways in COBL DT.

5.4.1 Checking Requirements

During the early building design process there are certain requirements that the designer establishes such as building codes, owner or designer's wishes and standards. Such requirements can be described in the beginning so that the design items can check or maintain such requirements during the design session. Examples of this type of knowledge in COBL DT is the checking of available space for specific activities (spatial conformance) and the use of standard templates for visually checking spatial design (use of stencils). The following two subsections explain how these are applied in COBL DT.

5.4.1.1 Spatial Conformance

Spaces may require minimum sizes to accommodate required equipment or activity. Building design standards provide such minimum, maximum and sometimes recommended spaces to accommodate versions of common equipment and activities.

When initially created (as corporeal items), spaces can serve any purpose and therefore do not have specific spatial requirements. However, as each space is classified, it acquires the knowledge about its minimum requirements especially its required size, through the assignment of roles during its recognition in CW (see section 5.3). While the knowledge to do this is embedded within the design item the attributes used in determining adequate size is provided in the repository. This information is then found by the design item which uses it to determine its minimum allowable size. This size is then compared to the current size. When the comparison is not satisfactory, COBL DT does not classify the space and reports this rejection to the designer.

All spaces check for this conformance. For instance, a stairwell is a connecting space between two levels or floors in a building. When a space is classified as a stairwell, it obtains attributes describing minimum stair sizes which includes risers, threads, etc. In most cases, stair design standards recommend a stair width that will accommodate one or two flights. Attributes needed to calculate minimum stairwell requirements are:

- Minimum access width to stairs;
- Minimum stair width;
- Minimum riser;
- Minimum thread.

With this information, the stairwell object calculates (and builds) a temporary stairwell geometry with minimum sizes which are compared to the available space to determine if the minimum requirements have been satisfied. These considerations were implemented and affect primarily the width, length and height of the stairwell as follows.

- When the width of the stairwell is equal to or less than double the minimum stair width, a straight flight is assumed.
- If the stairwell length does not make it possible to rise from one level of the building to the other, the classification is not possible.
- When the width of the stairwell is equal to or more than double the stair width, a double flight stair is assumed.
- If after two flights the total vertical distance traveled does not equal the floor to floor height, the classification is not allowed.

Spatial conformance makes it easier for the designer to create spaces that satisfy minimum sizes, for example when creating a garage to accommodate a single vehicle. In some cases however it may be necessary to create a garage to accommodate an arbitrary number of cars. This capability cannot be easily done using the method described in spatial conformance, but is possible using stencils.

5.4.1.2 Using Stencils

Stencils provide the designer with a graphical symbol of the actual requirement under consideration as shown in Figure 5.9 and allow the designer to interactively manipulate this symbol as desired. In the case of a garage, the designer is provided with the symbol of a car which can be placed in the design as needed. The designer then combines the overall sizes of the cars to obtain the minimum space to accommodate them.

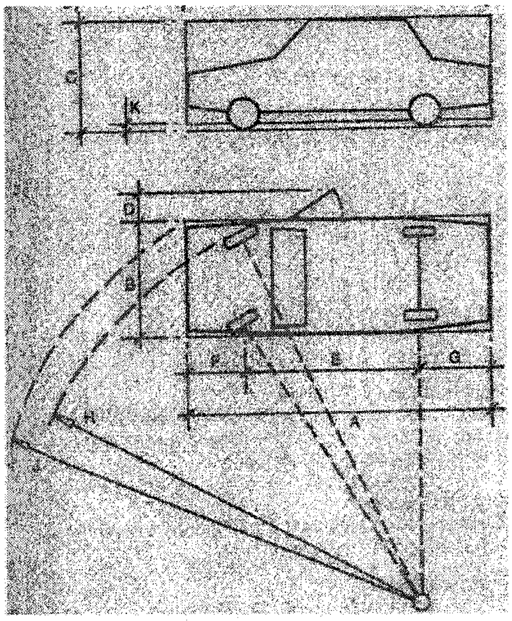

Figure 5.9: Car stencil
[Neufert, 1991]

Stencils are provided in CoBL DT by the stencil utility item (SUI). Each SUI is responsible for extracting the appropriate facts from the data repository and for combining them to create the appropriate stencil. For example, the Vehicle SUI can create a standard car, truck or bus, which can then be used by the designer. Typical information for creating vehicles are (refer to Figure 5.9):

- Average vehicle length (A);
- Average vehicle width (B);
- Average door opening allowance (for the sides and standing allowance for the ends) (D);
- Average vehicle height (C).

5.4.2 Checking Properties

Checking properties makes it possible for design items to consider certain parameters in order to solve for a specific problem such as calculating for solar saving factor (SSF). The SSF is a measure of solar conservation performance for a building when compared to an equivalent building that is not built to conserve solar energy. It compares the auxilliary heat needed by a solar-heated building to that needed by an energy neutral building that is otherwise similar. An energy neutral building is one with walls that have neither solar gain nor heat loss. It does not refer to the percentage of the solar building's heat but to a measure of the solar building conservation advantage [Stein and Reynolds, 1992].

The following is an example calculation of SSF using Table 5.2. The example shows SSF ratios for Toronto, Canada and Salem, Oregon for comparison purposes. For an office building located in Salem, Oregon, there is no night insulation on the windows. Preliminary design shows the total south window surface area to be 229 $sqft$ and the total floor surface area to be 1440 $sqft$. The ratio of total glass to total floor area is given by the equation:

$$\frac{\text{south glass area } 229\ ft^2}{\text{floor area } 1440\ ft^2} = 0.16 \tag{5.1}$$

First scenario: Assuming the building is in Toronto, Canada.

Looking in Table 5.2 a solar savings ratio for glass to floor ratio of 0.16 is too low immediately informing the designer to amend the design. To obtain the exact SSF value, the following calculation can be performed.

Table 5.2: Rules of thumb for passive solar glazing area [Stein and Reynolds, 1992]

Location	Area of Solar Glazing as Ratio of Floor Area		Approximate SSF Values (%)			
			No Night Insulation		With R9 Night Insulation	
	Low	**High**	**Low**	**High**	**Low**	**High**
	USA					
Salem, Oregon	0.12	0.24	21	32	37	59
	Canada					
Edmonton, Alberta	0.25	0.50	— NR —		54	72
Suffield, Alberta	0.25	0.50	28	30	67	85
Nanaimo, British Columbia	0.13	0.26	26	35	45	66
Vancouver, British Columbia	0.13	0.26	20	28	40	60
Winnipeg, Manitoba	0.25	0.50	— NR —		54	74
Dartmouth, Nova Scotia	0.14	0.28	17	24	45	70
Moosonee, Ontario	0.25	0.50	— NR —		48	67
Ottawa, Ontario	0.25	0.50	— NR —		59	80
Toronto, Ontario	0.18	0.36	17	23	44	68
Normandin, Quebec	0.25	0.50	— NR —		54	74

– *from design:*

glass/floor ratio $= 0.16$

– *from table:*

lowest ratio $= 0.18$

SSF range $(23 - 17)$ $= 6$

– *calculation:*

ratio difference $(0.18 - 0.16)$ $= -0.02$

SSF % to be added: $\frac{-0.02}{0.18} \times 6$ $= -0.7$

SSF % for building $(17 - 0.7)$ $= 16.3\%$ (With no night insulation)

The calculation for Toronto shows that with the glass to floor ratio of the building and no night insulation, the SSF advantage in Toronto would be low. The room is likely to cost more in heating because there will be little solar gain in the cold months.

Second scenario: Assume building is in Salem, Oregon.

Looking in Table 5.2, the same calculation is performed for a building in Salem, Oregon which shows a favorable performance of 25% (within reasonable limits as shown by the Table 5.2) as follows:

–*from design:*

glass/floor ratio $= 0.16$

–*from table:*

lowest ratio $= 0.12$

SSF range $(32 - 21)$ $= 11$

–*calculation:*

ratio difference $(0.16 - 0.12)$ $= 0.04$

SSF % to be added: $\frac{0.04}{0.12} \times 11$ $= 3.7$

SSF % for building $(21 + 3.7)$ $= 24.7\%$

Using expression 5.1, the utility object obtains the total glazing/floor ratio in relation to the orientation of each classified space. It then obtains the necessary data from Table 5.2 (using the default region as keyword) to calculate the SSF savings for the building. This calculation is automatically performed in the background and does not interfere with the designer's actions. The designer is simply informed whether the savings factor is low or acceptable. This assistance makes it easier for the designer to orient spaces as well as determine total south-facing glass surfaces necessary to keep the amount of auxilliary heating to a minimum while maintaining maximum thermal comfort.

5.5 Analysis and Design of Functions

The following sections describe the functions that make up the four main parts of CoBL DT as shown in Figure 5.1 and formulated in section 5.3. CoBL DT is an object-oriented software system and its description will be provided using UML as described in section 3.3. The use cases provided in CoBL DT can be found in the different parts illustrated in Figure 5.1. The first use case is *Draw*. This functionality makes it possible for the designer to create Corporeal and Incorporeal design items (see section 5.2). The second use case is *Classify* which makes it possible for design items to be recognized when they are introduced into the design. It is also a means through which roles are assigned to them in order to make the support they provide in a design session more relevant. The third use case is *Capture Entity* (see Collect Solution, *Hierarchical Decomposition* in Figure 5.1) which makes it possible for the system to put each design entity in context using a hierarchical tree showing relationships between all design items in the solution. Easy access is also provided to each individual design item because of the use of hierarchies. The fourth use case is *Capture Solution* (see Collect Solution, *Solution Path* in Figure 5.1) which makes it possible for the system to capture all the solutions created by the designer in the form of paths.

Figure 5.10 shows the typical relationship between these functionalities and the designer in terms of how each communicates with the other (see section 3.3). The designer typically communicates directly with the *Draw* use case to create design items. There is also a communication with *Classify* which makes it possible for the system to recognize and assign roles. The *Draw* use case typically communicates with the *Classify* use case when the designer classifies design items. It also communicates with the *Capture Entity* functionality in order to put the newly created design item in context with all others in the design. In the same way the *Classify* use case communicates with the *Capture Entity* and *Capture Solution* use cases when an item's identity is to be changed in the hierarchy tree or when it is to be captured in the solution path. The *Capture Solution* typically tells the *Capture Entity* to show all items in the current solution path.

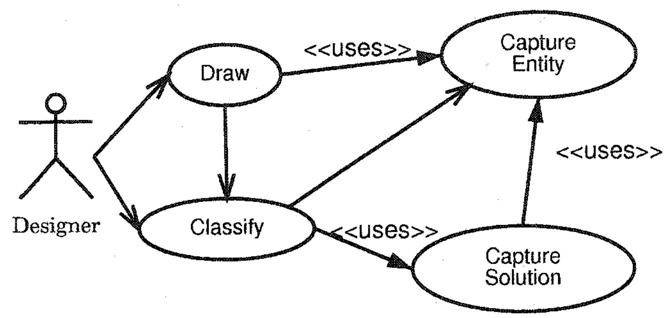

Figure 5.10: Main use case diagram

The analysis and design of the use cases for CoBLDT are presented next and also described in detail in Appendix B. The use cases are first introduced by providing a brief description of the functionality they provide and what they require in order to provide this functionality. This is followed by a section that describes the normal sequence of events for the use case which is performed when a communication is received from the designer or another use case. This sequence of events is called the main functionality and is supported by sub-functions that may not be used all the time but make it possible for the use case to provide the envisioned overall capability. These sub-functions are described following the description of the main functionality and it should be noted that sub-functions can be called from other use cases in order to provide a certain capability. The third section describes the classes or objects (see section 3.3) that have been identified to provide the functionality for this use case.

5.5.1 Draw Function

The draw functionality makes it possible for the designer to draw corporeal and incorporeal design items. Default settings that can be changed are provided for the color, pen size, height of space, building level, sheet number and orientation.

5.5.1.1 Main Function

This use case presents the designer with a menu choice for creating a Corporeal or Incorporeal design item but defaults to a corporeal item. The use case checks to make sure that the profile of the drawn shape is closed and automatically closes it otherwise. It then creates a corporeal object. The corporeal item is initialized and displayed with the numerical values of its area and volume and the graphic location of its boundary is drawn in a dashed outline. If the designer chooses to draw an incorporeal item, the system allows the creation of an open graphic item that does not form a closed shape. The coordinates of the endpoints are shown to the designer. When the item is created, the LIST CLASSIFICATIONS sub-function of the *Classify* use case and the INSERT ITEM sub-function of *Capture Entity* use case are executed.

5.5.1.2 Sub Functions

This use case provides the following sub-functions for managing the drawing activity,

Entity Height.

This sub-function makes it possible for the designer to provide a height for all new items being created. This height can be changed at any time but will apply to all new items following the change.

Line Color.

This sub-function makes it possible for the designer to designate line colors of the items created. This will apply to all new items.

Line Width.

This sub-function provides the designer with choices of line widths which can be applied to the objects being created. The line width chosen applies to all new items.

View Building Level.

This sub-function allows the designer to select a sheet for any building level to be displayed on the screen as a background to the current sheet. It makes it possible for the designer to reference other sheets while working on a given one.

New Building Level.

This sub-function creates a new building level and makes it the current level. It causes the NEW SHEET sub-function to execute automatically so that each new building level is created with a sheet. It collects the created building levels and allows the designer to view any of the existing levels.

New Sheet.

This sub-function creates a new sheet and makes it the current sheet. It makes it possible for the designer to organize data within building levels. Sheets are consecutively numbered when created and only one sheet can be worked on at any time. Only the sheets created in a particular level are available in that level.

Change Sheet Name.

This sub-function makes it possible for the designer to change the automatically generated name of a sheet to a custom name.

5.5.1.3 Classes

The designer draws an Entity object (see section 3.1) which can either be a **Corporeal** or an **Incorporeal** class. The created item is captured by **SolHierarchy** which is described in section 5.5.3. The object of class **SpecVClassify** is then called which prepares possible classifications for the newly drawn item. SpecVClassify class is described in the section 5.5.2. The class diagram is illustrated in Figure 5.11

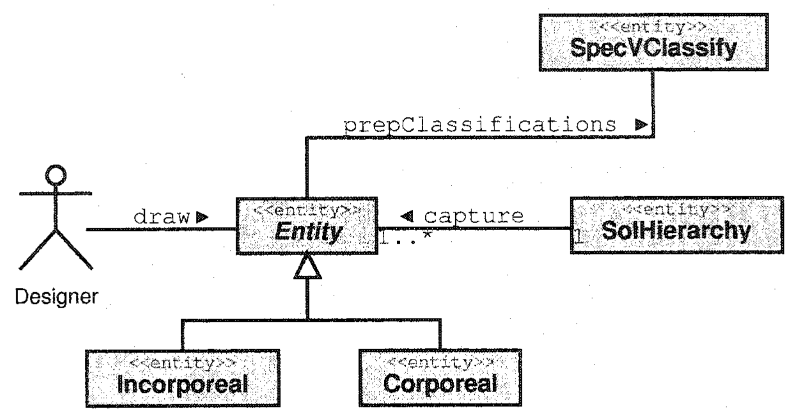

Figure 5.11: Draw class diagram

5.5.2 Classify Function

The Classify function makes it possible for the designer to designate what each entity created in the system is supposed to be thereby making it possible for the system to know more precisely what its role is in the design. It requires the orientation of the north direction.

5.5.2.1 Main Function

The function begins when design items are created. It prepares a set of types of building components for the designer to select which will designate the role of the drawn item. The choice of types to present for selection depends on its context such as whether it is created within another design item. When the designer makes a choice, the type of building component selected checks to make sure that the space provided by the drawn item is sufficient for its spatial needs. If this check fails, the classification is rejected and the designer is allowed to reclassify the item.

If the classification is accepted, specialized objects are initialized along the borders to provide enclosures ("walls") for the space being defined by the classification. CoBL DT includes these enclosure items as **Boundary** objects but only shows their presence by labelling them on screen. This makes it possible for the designer to configure them as and when needed. This design of boundaries using a different object from spaces is important because boundaries, such as walls, are in reality separate objects from the space.

In addition to the classification of corporeal and incorporeal design items, connectors are also available for classification. **Connector** objects are specialized corporeal items for establishing a relationship between two spaces such as a shared wall or a door. When classified a connector item is attached to the appropriate boundary of the corporeal concerned. The designer can then select this connector for connecting the corporeal item with another. When a connection is established between two corporeal items, it is maintained and the two spaces can communicate with each other as well as providing the designer the ability to adjust some of the object parameters through the link.

5.5.2.2 Sub Funtions

The following sub-functions describe the capability provided by classification.

Prepare Classifications.

This sub-function prepares the list of possible classifications after the designer draws a corporeal item and selects a corporeal or classified design item. The items in the list are determined by the design item within which the new corporeal is drawn because all classified corporeal items "know" what can be contained within them. For example if the designer draws a corporeal item within the kitchen space, it (the kitchen object) will not provide a bedroom as one of the options for classifying the new corporeal.

Apply Classification.

This sub-function provides the new corporeal with an identity after the designer has made a choice from the list. This type is used to obtain the necessary characteristics which are then applied to the new corporeal. The LABEL BOUNDARY sub-function is executed.

Label Boundary.

This sub-function creates the Boundary objects for each side of a recently classified item providing labels for accessing them. If the item is not a recently classified one it highlights the labels on screen for the designer. The BOUNDARY STATUS sub-function is executed.

Boundary Status.

This sub-function allows the designer to set the status of the boundary as "open" or "close". The default status is "close", however the designer can choose to switch it to "open". The BOUNDARY MATERIAL sub-function is executed.

Boundary Material.

This sub-function applies default materials to boundaries with the status of "close" using their labels. The designer is allowed to choose from a limited number of materials such as wood, which can be applied instead of the default after classification. If the boundary status is "open" the material is cleared and set to "none".

Remove Connection.

This sub-function removes a specified connector from a corporeal item and if there is already a connection in place, also removes it from the connected space.

Show Connection.

This sub-function makes it possible for a connector to be highlighted in the design to show the designer its location.

Adjust Connection.

This sub-function allows the designer to adjust the size of a connected space. Normally the designer creates spaces in CoBL DT with no assistance in precision. Using this function the corners of spaces or their overall sizes can be coordinated with some precision. See Chapter 6 section 6.2.4.2 for more information.

5.5.2.3 Classes

When the designer selects a corporeal for classification, the **SpecVClassify** object (see Figure 5.12) uses the selected label to obtain the appropriate characteristics from the **ClassDef** object. This object maintains a collection of types of building component objects (classes). The SpecVClassify object creates a new object (Kitchen) using these characteristics and the spatial specification provided in the corporeal item. It then replaces the corporeal item in the design. If any checks that must be made by the new object requires its orientation on the site, a request is made to the **CobIdtNorth** object which has the information regarding the north direction on the site. The new object automatically

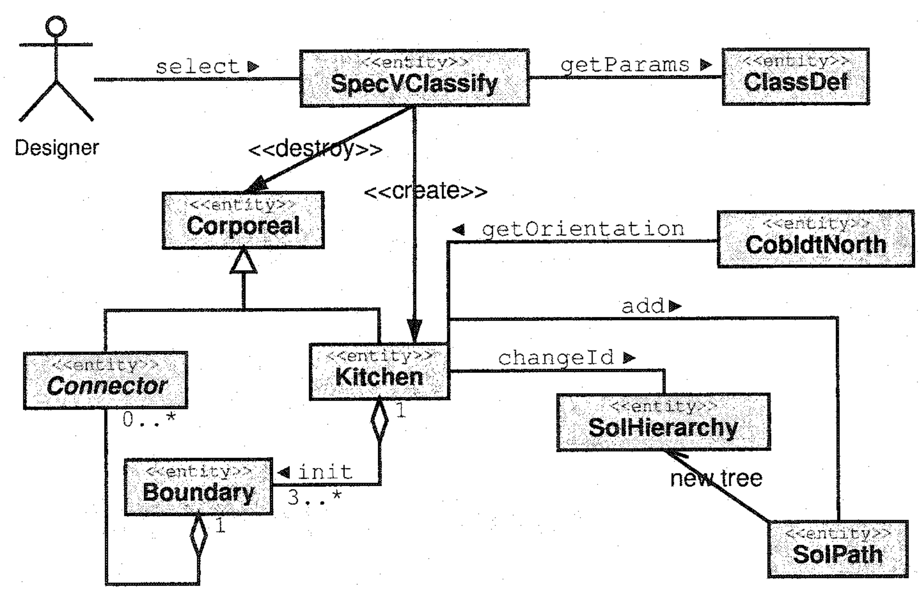

Figure 5.12: Classify class diagram

initializes its borders creating **Boundary** objects to be used by designer in configuring its edges or adding **Connector** objects later in the design process. Boundary objects specify how the space being defined by the corporeal is enclosed and connector objects provide the different ways that these boundaries can be modified in order to relate spaces with each other. The new object is then sent to *Capture Entity* (**SolHierarchy** object) and *Solution Path* (**SolPath** object) use cases. The following are the various Corporeal, Connector, Incorporeal and Stencil objects that have been implemented in the current version of CoBL DT.

Corporeal objects

Maximum Footprint	Living Room	Dining room

Kitchen	Bathroom	Bedroom
Master bedroom	Family room	Garage
Play area	Office	Sitting area
Straight Stair	Circular stair	Storage
Furniture	Kitchen appliance	Bathroom appliance
Work surface	Structural member	Pavement
Tree	Green space	Shrubs
Boundary (Walls)	Foyer	

Incorporeal objects	Connector objects	Stencil object
North direction	Door	Standard car
Wind direction	Window	
Sunlight	Hole	
Noise source	Space joiner	

5.5.3 Capture Entity Function

The Capture Entity function (see Collect Solution, *Hierarchical Decomposition* in Figure 5.1) makes it possible to collect and present each design item in such a way that its relationship with all other items in the design is presented as well. This is done using a hierarchical decomposition tree of all items in the current solution (see section 2.3.4). It requires that all entities have a unique identification and be able to record their parent's and children's identifications.

5.5.3.1 Main Function

This function begins when an item is created in the *draw* use case. It collects the unique id of the item and adds it to the hierarchical decomposition tree. If it is created inside an existing corporeal item, the unique id for the parent is collected and used to locate it in the tree. The object is then added as one of its children. The following sub-functions are provided by this functionality.

5.5.3.2 Sub Functions

Delete Entity.

This sub-function uses the unique id of an item to locate it in the tree. The item is then removed from the design. If it has a parent, the parent id is used to find it and update its list of children accordingly. The child along with any children it may have is then removed from the tree. The REMOVE sub-function of *solution path* function is executed.

Build Tree.

This sub-function builds a hierarchical decomposition tree of all design items in a solution. It needs all the unique identifications of all items in the top level of the tree. These items are collected from the design and each item is sent to the FIND CHILDREN sub-function. This sub-function uses the item's id to locate it in the tree then using the list of its children recursively collects them in a new list. The branch of each child is built until there are no more children to attach before the next child in the same level is processed. When all children for a top level item is completed, the next top level item in the list is processed until the tree is complete.

Insert Entity.

This sub-function inserts an entity into the hierarchy tree. It needs the unique identification of the entity. It uses the entity to check for a possible parent. If there is one it adds the item as a child but if there is no parent, it adds the item at the appropriate level in the tree.

5.5.3.3 Classes

The system captures entities using the **SolHierarchy** class which builds the hierarchy decomposition tree. Entity objects are added to the tree as soon as they are created. When classification occurs, the classified item is used to substitute, in the tree, the entity item it replaces in the design. SolHierarchy maintains a relationship with the **CobIdtDoc** class which records all the items created in the design. Figure 5.13 shows the relationship between these classes. Note that the class diagram uses only a corporeal object example which is a Kitchen.

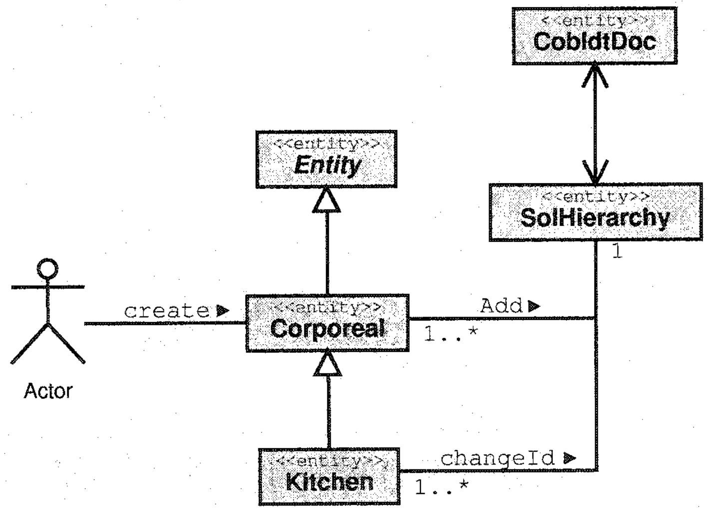

Figure 5.13: Capture entity class diagram

5.5.4 Capture Solution Function

The Capture Solution function (see Collect Solution, *Solution Path* in Figure 5.1) makes it possible to collect all steps taken to make up a particular solution. It preserves the sequence in which the entity objects were created and so makes it possible for the designer to browse and review both the solution and the process with which it was created. It requires that the entity objects have unique identifications and that they have a means of being designated as belonging to a specific solution path.

5.5.4.1 Main Functionality

This function begins when the designer classifies an entity object. The object's unique identification is collected as steps in the path and added to a list that represents the current solution path as illustrated in Figure 5.14. This list is incremented with every new classified item's unique id, representing the sequence in which they were classified. This

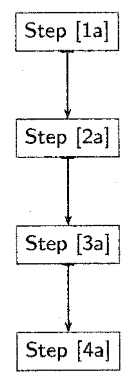

Figure 5.14: Illustrating a Solution Path

functionality provides the following sub-functions for managing the solution path.

5.5.4.2 Sub Functions

New Path.

This sub-function first calls the NO BRANCH POINT sub-function to establish if the designer can start a new path from the selected point. If it is not possible, the designer is informed and no new path is started. Otherwise, the system adds the current branch-off point (illustrated by Step [3a] in Figure 5.15) to the NO BRANCH POINT list and begins a new solution path. The following is a description of the creation of a new solution path as illustrated in Figures 5.15 and 5.16.

Assuming the designer selects Step [3a] in the current solution path as the beginning point for a new solution path, the system obtains the unique id represented by this step and locates and replicates the entity in the design. This entity represents the branch-off item in the current solution path. All other entity objects that lead to this branch-off

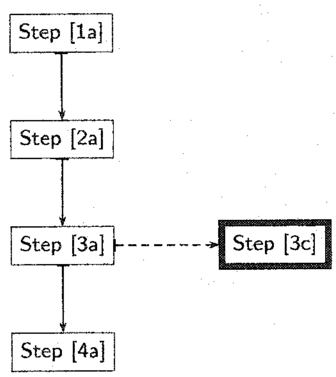

Figure 5.15: Branch-off Point

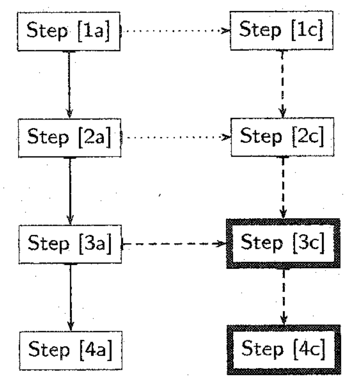

Figure 5.16: Replication

item, starting with the first entity in the current solution path (Step [1a] to Step [3a]), are all replicated and collected in a new list (Step [1c] to Step [3c]) as illustrated in Figure 5.16. The current solution path (Step [1a] to Step [4a]) is saved while the replicated items becomes the current solution path. Step [4c] in the replicated branch is the latest new item classified and added to the path by the designer.

Solution parts are independent from each other because design solutions are unique. Once a branch-off occurs the development of a new solution should not affect any previous ones. The aim of collecting solutions is to save or freeze previous efforts making it possible for the designer to be able to investigate new ideas and still be able to return and continue development. Replication makes this possible as the solutions are represented by copies that are not connected. For example the designer starts a new solution path consisting of Step [1c] to Step [3c] as shown in Figure 5.16. If the object represented by Step [2c] is edited or removed and the designer decides to go back to the first solution represented by Step [1a] to Step [4a], everything will be as it was because Step [2c] was a copy and all manipulations on it so far did not affect the state of Step [2a], its original version.

Change Path

This sub-function makes it possible for the designer to change to a previously created solution path. It is possible to browse or review all created solutions and it is also possible to continue the development of any solution while browsing or reviewing it using this sub-function. If the change of path is caused by the creation of a new path, the system increments the solution path number used in labelling new paths. Otherwise, the system changes to the path designated by the designer. The following is the description of how this occurs.

Items in the solution path are designated by numbers. The first is Solution Path 01 and subsequent ones are numbered consecutively. When the designer chooses to change path, which can take place from any one of the current solution path items, the system uses the selected item's number to retrieve the list containing the unique id of the entity objects in that path. Using this list, making sure to preserve its sequence, the objects are obtained from the system's document. The previous path is saved and the one chosen by the designer is made the current path. The list of entity objects are then sent to the BUILD TREE sub-function of the *capture entity* use case so that the hierarchical decomposition of objects in the current solution can be made accessible to the designer. Using this capability, different parts of different solutions can be merged to create a "best" possible solution for a design problem.

Remove

This sub-function requires the unique id of the removed entity. It marks this entity as removed in the solution path list. The items id is retained however and shown as part of the solution path although it is colored red to show its absent state.

No Branch Point

This sub-function manages the points from where the designer can begin a new solution path in the solution path tree. This is mostly a means of keeping the solution path tidy and also preventing any errors in building the tree. For example a previous branch off point in the tree cannot be selected for another branch off point. This function will check each time a branch off is requested and compare it to the list of branch off points already existing before allowing or disallowing the branch off request.

5.5.4.3 Classes

The solution path functionality is started when items are classified. Classification is a process that makes it possible for the system to recognize building components using the SpecVClassify class. The class diagram in Figure 5.17 shows the classes involved and their relationships. When a corporeal item is classified, it is added to the current solution path by the objects in the **SolPathDoc** class. This class coordinates with the **CobIdtDoc** class to implement the solution path functionality.

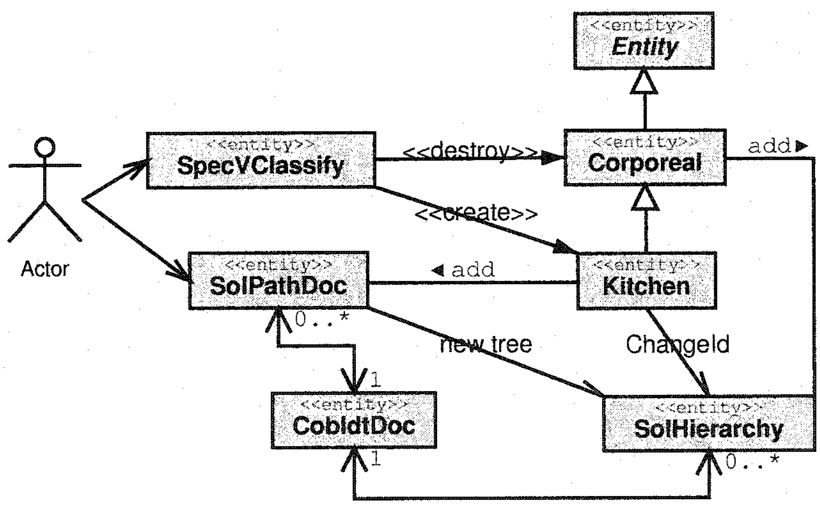

Figure 5.17: Capture solution path class diagram

5.6 Summary

This chapter defines the functionalities needed to support the requirements described in section 2.3. It has also designed objects and prescribed the type of interactions necessary between them in order to achieve this support.

We started by limiting the design considerations to only three specifications which include Collecting Design Solutions, Design Overview and Application of Requirements although some others are partially utilized. A system overview is described showing the main parts of the system. Formulations of details for these parts are described before in-depth analysis and design of the system was explored. The functionality in each part is divided into a main function and sub functions. These functions are coordinated by a set of objects that support the early building design process. The next step in this research project is to carry out an implementation based on these descriptions. This is described in the next chapter.

Chapter 6

CoBL DT: Implementation

THIS CHAPTER PROVIDES a detailed description of CoBL DT which has been implemented based on design parameters presented in the previous chapter. CoBL DT is a prototype that introduces a digital environment for supporting the early building design process.

The description begins with the environment in which CoBL DT is implemented followed by an introduction to the user interface. This interface is then divided into four areas consisting of command input, draw, organization and configuration/feedback areas and described in detail. In the last section the use of design knowledge in CoBL DT is described and illustrated.

6.1 Implementation Environment

In creating this prototype, there were considerations and constraints to be accounted for. First there is a need for an easy-to-learn programming environment that supports a quick implement-and-test cycle. The limited resources available for the research project make it important to have a tool that is affordable and widely accessible, including ample support with possibly a large community of users. It is also preferable to acquire a developing environment flexible enough to be integrated into possibly a diverse range of computer environments. A prototype represents only a part of a tool set for supporting a multi-stage design process.

The use of the Python computer language satisfies all the considerations given above for the implementation of the prototype. Python is an interpreted, interactive, object-oriented programming/scripting language. It was invented in 1989, but version 1.0 was publicly released in 1991. The release used in this thesis is version 2.1 which is currently available on many brands of UNIX systems including Linux. It is also available on MacOS, MS-DOS, Windows 9x/NT/2000, OS/2 and PalmOS platforms. Its portability has also been extended with Jython, an interpreter written in Java [Hilf and Cimafranca, 2003; Brueck and Tanner, 2001; Python Software Foundation, 1991].

The user-interface is created using wxPython, a GUI toolkit for the Python programming language which allows programmers to build graphical user interfaces for the python language. It is implemented as a Python extension module (native code) that wraps the popular wxWindows cross platform GUI library, which is written in C++.

Python, wxPython (and wxWindows) are Open Source tools which means that they are free for anyone to use or modify (as required for their particular need) [Dunn, 2001]. These systems are cross-platform tools which means that the same program created with these tools will run on multiple platforms without modification. These tools are actively being developed and supported both by their creators (those in charge of maintenance) and a large community of users [Python Software Foundation, 1991].

6.2 The User Interface

CoBL DT presents a main window that is divided into areas or mini-windows in which different activities take place as shown in Figure 6.1 and as explained below.

- The first area presents pull-down menus and toolbars of commands that the designer can use to control the creation and manipulation of design items as shown at "A". It provides a location for most of the commands used in CoBL DT.
- The second area presents a drawing window shown at "B" where the designer creates and manipulates the design items. Only one view (top or plan view) of the design item is currently provided and the creation of design items is limited only to rectilinear objects. The design items are captured automatically by the system and presented in the organization area where the designer can either review or rearrange them with regard to their relationship to each other.

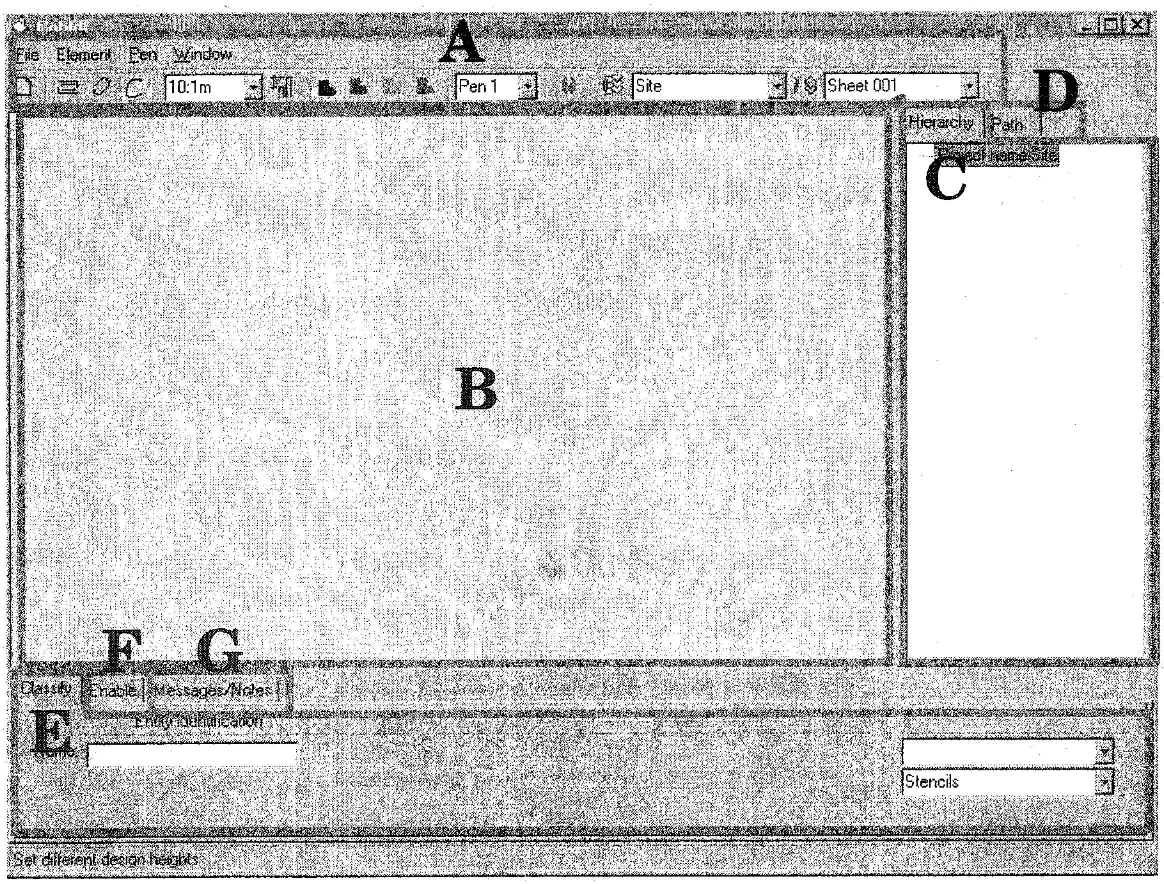

Figure 6.1: CoBL DT program main window

- The third area presents two windows for capturing design items at "C" or capturing design solutions at "D". Captured data is presented in hierarchical formats for the designer to use in organizing design items.

- The fourth area presents 3 windows. The first two provide the designer with an interface for configuring design items at "E" and "F". The third window at "G" is divided into two regions. The first is used by CoBL DT to provide text-based feedback to the designer about internal operations or consequences of certain interactions. The second region is provided for the designer to make personal notes.

6.2.1 Commands (A)

This area of the user-interface provides a set of commands for creating and manipulating design items. The use of a menu, although partially provided, is avoided as much as possible as specified in section 4.3. Instead, a toolbar and floating windows are provided for entering commands. The toolbar provides an intuitive access that involves a single click and its location can be easily memorized because it is unique and visible. Floating windows are used to provide context sensitive commands at the location where these commands are needed (current position of the cursor). They are normally hidden but become visible when requested by the designer. All commands are made visible and available depending on the location of the main window where the request is made or whether there is a design item present. In addition, the type of design item present at the location where the request is made also influences the type of commands available.

Using these commands the designer creates/manipulates corporeal and incorporeal design items. The following provides an illustration of each tool in this area of the user-interface as well as a short description of its use.

Figure 6.2: Open

Open (Figure 6.2) command allows the designer to open a window for creating design items. This window shows the top (plan) view of the design space.

Figure 6.3:
Corporeal A

Create Corporeal A (Figure 6.3) command allows the designer to create a corporeal by designating all the sides as shown in Figure 6.7.

Figure 6.4:
Corporeal B

Create Corporeal B (Figure 6.4) command allows the designer to create a corporeal in a free-form manner as shown in Figures 6.5 and 6.6. Figure 6.5 shows the use of a diagonal line to specify the top-left corner and the bottom-right corner of the corporeal while Figure 6.6 shows the creation of the top and side edges.

Figure 6.5:
Diagonal corners

Figure 6.6:
Top and side edges

Figure 6.7:
Top and side edges

When menu button *Create Corporeal B* (Figure 6.4) is selected, the designer is able to specify the corners of the rectilinear corporeal in a free-form manner as shown in Figures 6.5 and 6.6. Compare this to the selection of menu button *Create Corporeal A* (Figure 6.3) which allows the designer to specify the corporeal item as shown in Figure 6.7. All methods however, create similar rectilinear corporeal items as shown in Figure 6.8.

Figure 6.8: Corporeal design types

Figure 6.9:
Incorporeal

Create Incorporeal (Figure 6.9) command allows the designer to create incorporeal design items.

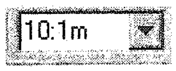

Figure 6.10: Design scale

Design Scale (Figure 6.10) command allows the designer to choose an appropriate scale for the design session. Selecting a scale changes how CoBL DT performs calculations without changing the designer's view of the design item.

Figure 6.11: Default Heights

Default heights (Figure 6.11) command allows the designer to set the default heights for the building level as shown in Figure 6.12. Here the designer can specify the heights for the building levels, the top of the door openings and the bottom of other wall openings such as the window. The height of all wall openings are assumed to be the same, equivalent to the door height. While the bottom of all openings are assumed to be the same except for the door.

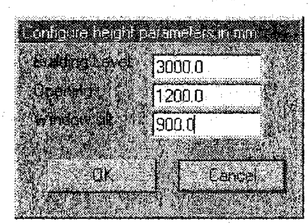

Figure 6.12: Setting default heights

Figure 6.13: Draw Colors

Draw Colors (Figure 6.13) command allows the designer to choose from four different colors namely black, blue, green and red.

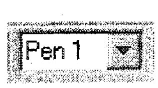

Figure 6.14: Pen Size

Pen Size (Figure 6.14) command allows the designer to choose the size of pen marks to use. There are up to 10 different sizes to choose from and when a size is selected from the drop-down window, it becomes the active choice.

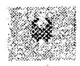

Figure 6.15: Trace Sheet

Trace Sheet (Figure 6.15) command presents the designer with the possibility to view sheets from any building level. The designer can then trace these items onto the current sheet or simply reference them as needed.

Figure 6.16: New Level

New Level (Figure 6.16) command allows the designer to create new building levels. The new level is automatically created, added to the collection of building levels in the design solution and made the current level (see selection list *Current Level* Figure 6.17). A new sheet is also created automatically along with the new building level (see menu button *New Sheet* Figure 6.18). A building level cannot be renamed.

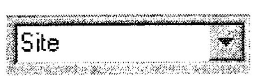

Figure 6.17: Current Level

The menu button *Current Level* (Figure 6.17) shows the current or active building level and allows the designer to select any other available building level to make active. It uses a list interface to collect all levels that are created in the design process but shows only the current working level at any time.

Figure 6.18: New Sheet

The menu button *New Sheet* (Figure 6.18) command allows the designer to create new electronic sheets or layers in which design items can be organized. Sheets are numbered consecutively and attached to the current level. Unlike building levels, sheets can be renamed by the designer at any time.

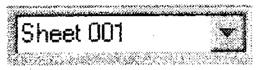

Figure 6.19: Current Sheet

The selection list *Current Sheet* (Figure 6.19) shows the current or active sheet being used and allows the designer to select any sheet in the current level to display its contents. The use of sheets is synchronized with the use of building levels. Items in each building level can be organized in more than one sheet so for each active level, clicking on this menu will only show those sheets that are associated with it. Selecting a sheet will show the items on that sheet only. To show sheets on other levels at the same time, the menu button *Trace Sheet*, Figure 6.15 can be used.

6.2.2 Draw (B)

The tools described in section 6.2.1 are used in this area of the user-interface to create design items. The organization of design items is provided using "Sheets" and "Levels" as shown in Figures 6.20 to 6.22. A Sheet provides a means for the designer to group design items while a Level allows the designer to group Sheets. Sheets are not part of the building and can be created without regard for their sequence but Levels represent the different building stories in the design and their sequence is significant. The designer can use from 1 to 999 Sheets and from 1 to 999 Levels. The following describes the major functions associated with this area of the user-interface.

Initially when the system is started, a building level is created automatically along with the first sheet for organizing design items on that building level. Figures 6.20 to 6.22 show the relationships between building Levels and sheets. More sheets can be added in any building Level as needed throughout the design. Each time a new building Level is inserted by the designer, CoBL DT automatically adds the first sheet for it as shown in Figure 6.22. The first Level, Building Level 000, is a special one because this is the Level where the designer must provide the site information. CoBL DT labels this Level accordingly "Site", as shown in Figure 6.23. The rest of this chapter will refer to this

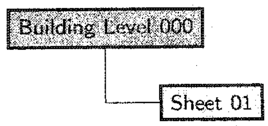

Figure 6.20: First Level

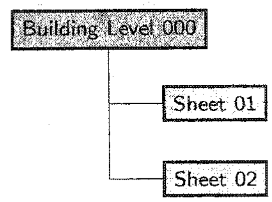

Figure 6.21: Adding one sheet

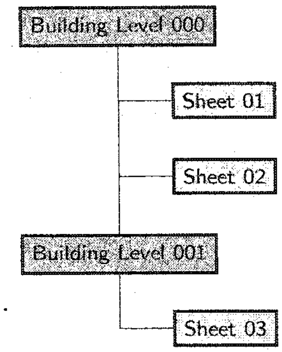

Figure 6.22: Adding one Level

Level as "Site". CoBL DT thus provides a reference to design items on the first Level

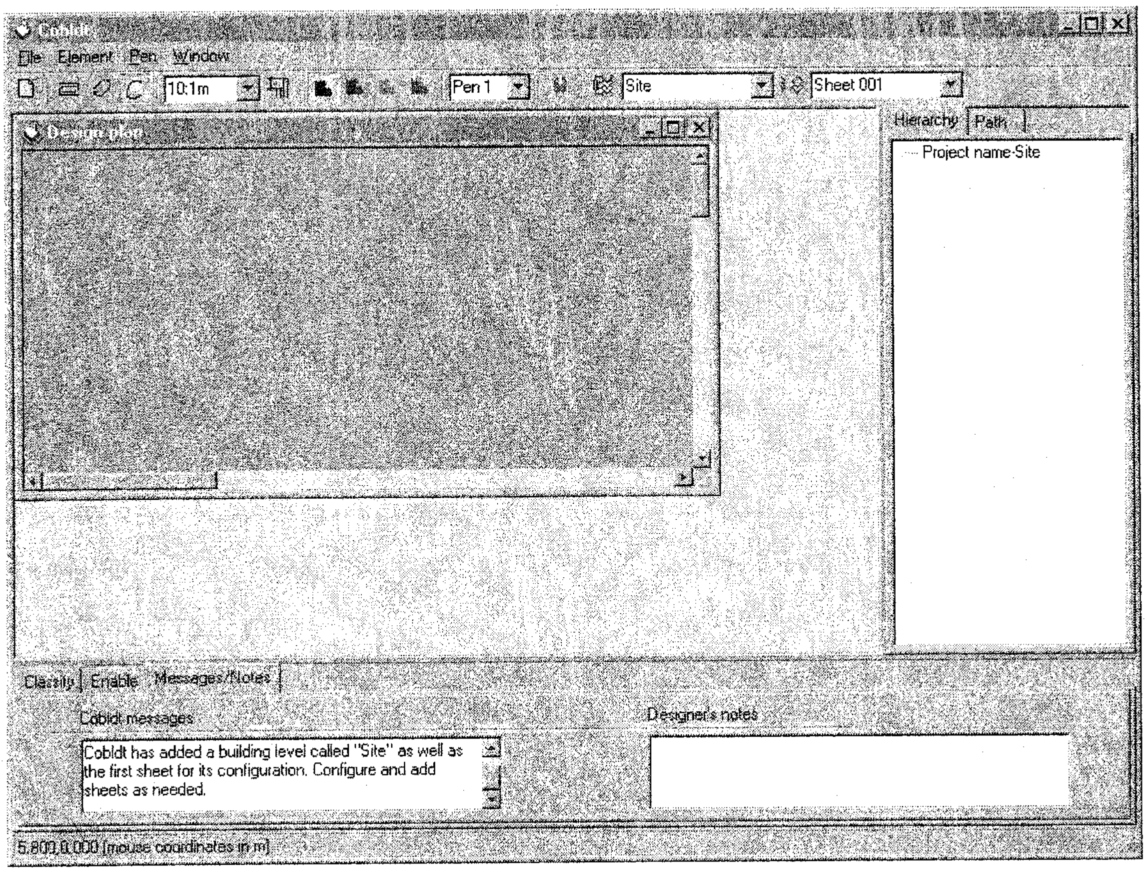

Figure 6.23: New window at the beginning of a design session

so that the designer can constantly refer to their size and location as this influences all other items in the design process greatly. This first Level is automatically created with the first sheet at the beginning of the design session. Figure 6.24 shows three design items on the Site Level. These items are drawn in dashed lines because the Site Level is not the current Level. All items drawn in the Site Level are dashed once the designer is no longer in the Site Level. This is a way for the designer to constantly refer to these items as they provide the basis for drawing all other design items. The designer can create

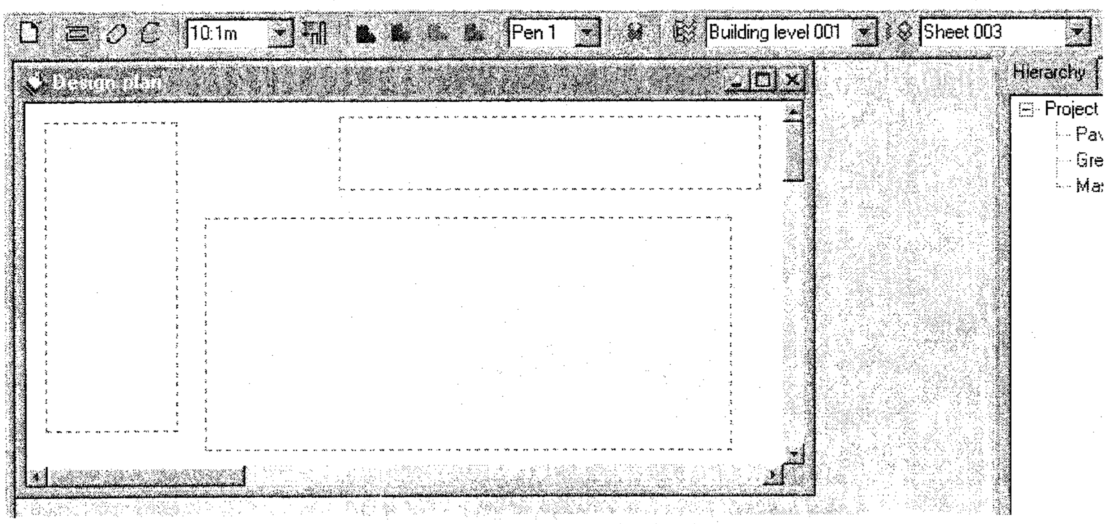

Figure 6.24: Showing three design items in "Site" Level while in another Level

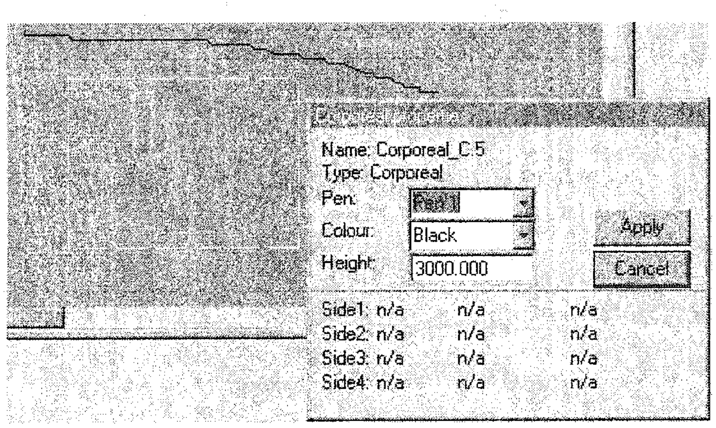

Figure 6.25: Changing corporeal properties

corporeal or incorporeal design items. When first created, corporeal items are drawn in long-dashed outlines to show that they currently do not play any specific role in the design process. In addition to the ability to move, copy or delete design items, their properties, such as line thickness, color and height, can be changed in the pop-up window shown in Figure 6.25. In this window, information about the "Name", "Type" and sides of the item is shown. The name of a design item is unique in the design solution while the type refers to the classification to which it belongs for example "floor slab". Side1 to Side4 in Figure 6.25 provides information on the state and treatment of the boundaries for the item depending on if the item is classified or not. In this case the item is not classified. The three parameters specify for each side, its status (open or closed), its length and its material.

6.2.3 Organization (C, D)

This part of CoBL DT provides tools that organize the designer's data in a way that provides an overview and gives access to the design solutions. Organization of data is broken down into two types. The first is the organization of design items which show the relationships between all items in each design solution (see section 5.5.3). The second is the organization of the solutions (see section 5.5.4). The design items created by the designer are automatically captured and the windows automatically updated requiring minimal assistance from the designer. The following sections describe these windows in more detail.

6.2.3.1 Design Decomposition Hierarchy

At the beginning of a design session, when CoBL DT is initiated, the head of the hierarchy tree is created using the project or site name. This is the parent of all items to be created for this design session as shown in Figure 6.26 at the top of list "C". The system sees all design items as either belonging to Building Levels or to classified corporeal items. When an item is created and is not confined within another corporeal item, the system adds it to the current Building Level window. Window "C" in Figure 6.26 shows more corporeal items added to the "Site" item tree. Although design items have been created on two

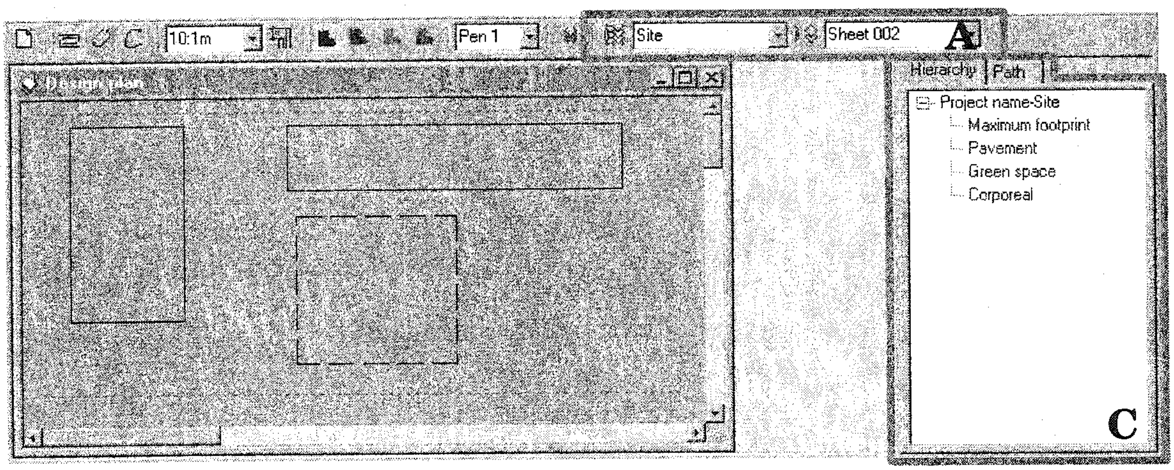

Figure 6.26: Representing design items

sheets, "Sheet 001" and "Sheet 002" (see Figure 6.26 "A"), they have all been added to the "Site" item tree without showing the different sheets they belong to in the tree. Sheets are for the designer's benefit and are not part of the "Site" item.

When a new Building Level is added, a new branch is created. When items are created within the boundaries of another item, they are automatically recognized as part of the prior item and are added accordingly in the tree as shown in Figure 6.27. The bathroom

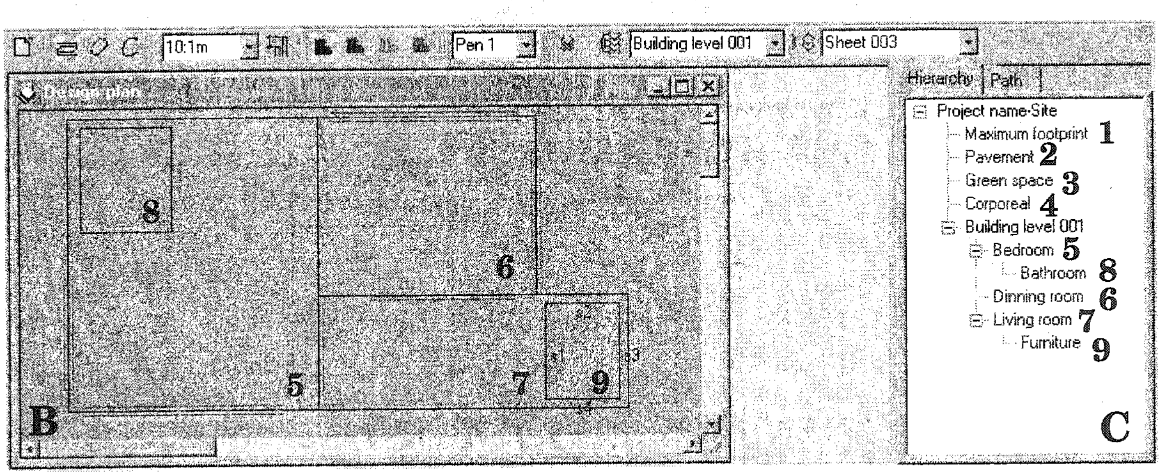

Figure 6.27: Further representation of design items

at "B8" is added to the bedroom at "B5", the relationship of which is automatically reflected in the tree at "C8". Commands available (in a floating window) when tree items are selected using the right button mouse include the ability to show the name of the sheet to which the selected item belongs and an option to copy this item to another sheet. In addition, other menu options provided are context-sensitive. When the selected item is in the current sheet (the Sheet that is presently visible in draw window "B"), a menu window as shown Figure 6.28 is provided, in this case for design item 9 "Furniture" on current Sheet003. The additional commands make it possible to rename, view, edit the selected item's properties or delete the item from the design solution. If the item

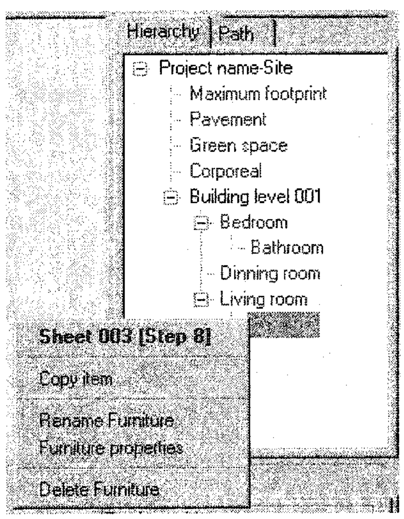

Figure 6.28: Current Sheet menu

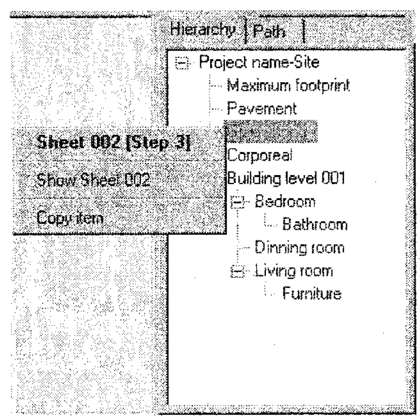

Figure 6.29: Other Sheet menu

is not located on the current Sheet, for example item 3 "Green space" on Sheet002, no additional option is provided (as shown in Figure 6.29).

6.2.3.2 Design Solutions

When design items are classified, they play a role in the design solution and CoBL DT captures them as a step in the path towards the solution. Ordinary corporeal items that are not classified are not captured in the solution path. Window "C" in Figure 6.30 shows the captured solution path represented by items in draw window "B". Solution paths are captured in the sequence in which design items are created and labelled as "Step *N*"

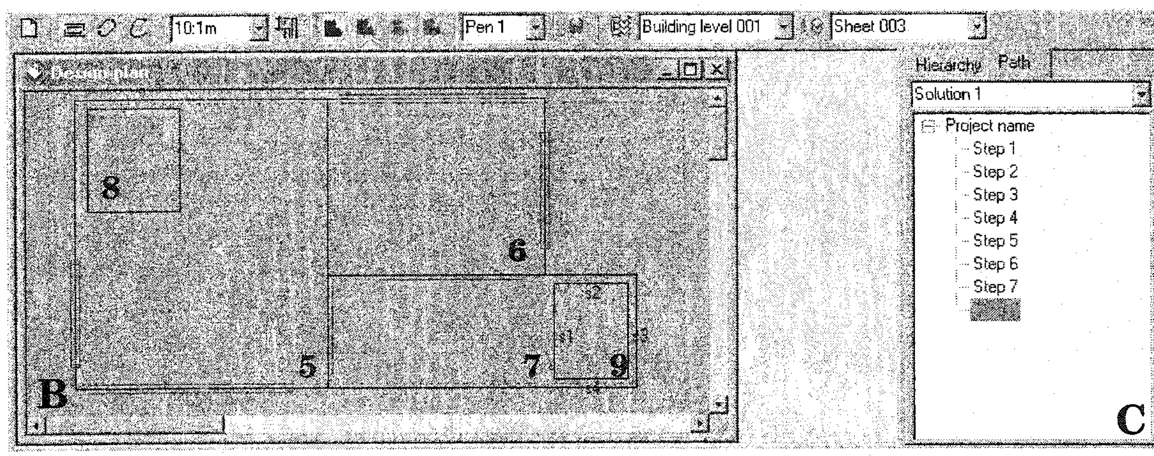

Figure 6.30: First solution path

where $N = 1,2,3\ldots$

New solutions paths are created when the designer presses the right mouse button on a step in the path window as shown in Figure 6.31. This makes the command for "New Path" visible for selection. If this command is selected, the system replicates the classified

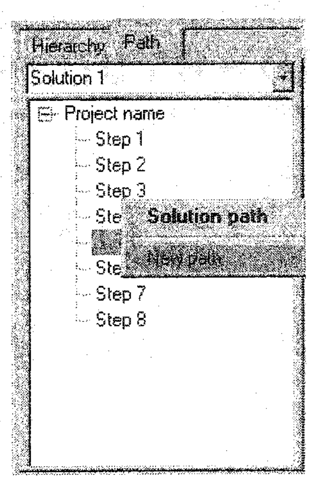

Figure 6.31: Creating a new solution path at "Step 5"

corporeal items in the current path (i.e. duplicates the items up to and including the selected item), labels the previous solution path and begins accepting new classified items for the new path as shown in Figure 6.32. The solutions are numbered and labelled as "Solution 1" using a similar process as the steps described above. Note that the steps in

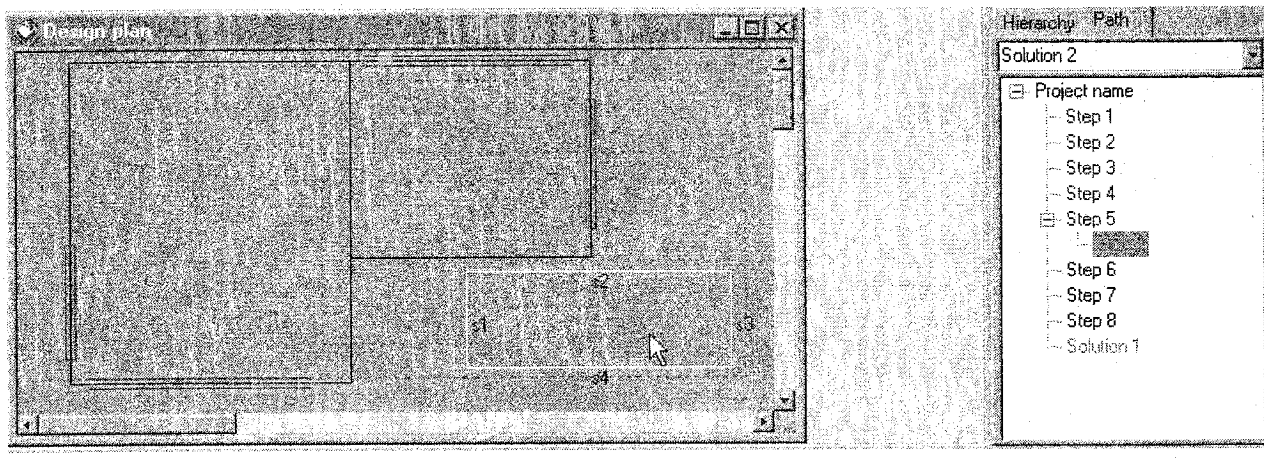

Figure 6.32: Second solution path

the path represent each classified item which does not include "C4", a corporeal item (see Figure 6.27). "C4" is an unclassified corporeal created in the "Site" Level in "Sheet 001". It would have been captured in "Step 4", if classified, in the sequence it was created.

When solution paths are collected, it becomes possible for the designer to browse through them which makes it possible to either present, evaluate or merge parts of the different design options. COBL DT provides a drop-down window at the upper part of Figure 6.33 "D" where the designer can select the preferred solution path. Selecting a path in this

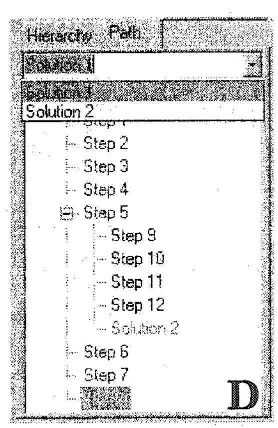

Figure 6.33: Browsing through solution paths

list will make that path the current one and the designer can either continue designing or continue browsing other solution paths.

6.2.4 Configuration/Feedback (E, F, G)

The windows presented in this section make it possible for the designer to identify the type of design items being created so that appropriate roles (see section 2.3.2) can be provided for participation in the design process. There are three windows. The first window enables the designer to specify the design types through a process called classification. This transforms an ordinary corporeal into a building component such as a "Living Room". The second window allows the designer to provide additional properties to classified items such as specifying a door between two spaces. The third window provides feedback of any consequences of the designers actions or requests in the design process. It also provides an interface for the designer to make personal notes. The following sections describes these windows in detail.

6.2.4.1 Classification of Design Item Types (E)

Corporeal and incorporeal design types are generic design items because they have not been recognized (see section 2.3.2) yet by the system. As a result they have not been assigned any roles and so do not provide any appropriate support in the design process. The designer, however is allowed to specify their role in the design through a classification process. The designer classifies design types by selecting from a list as shown in Figure 6.34 at "E2". CoBL DT tries to follow a systematic procedure when presenting items for classification. When a new design session is started, the designer is presented with the first Building Level (Building Level 000. See Figures 6.20 to 6.22) which is labelled "Site". The system expects the designer to create at least one corporeal item called "Maximum Footprint" on this Level. If this is not so, the system cannot provide additional building types such as "Bedrooms" for classification. The choice of classification types available depends on the types present in the CoBL DT as shown in Figure 6.34 at "E2".

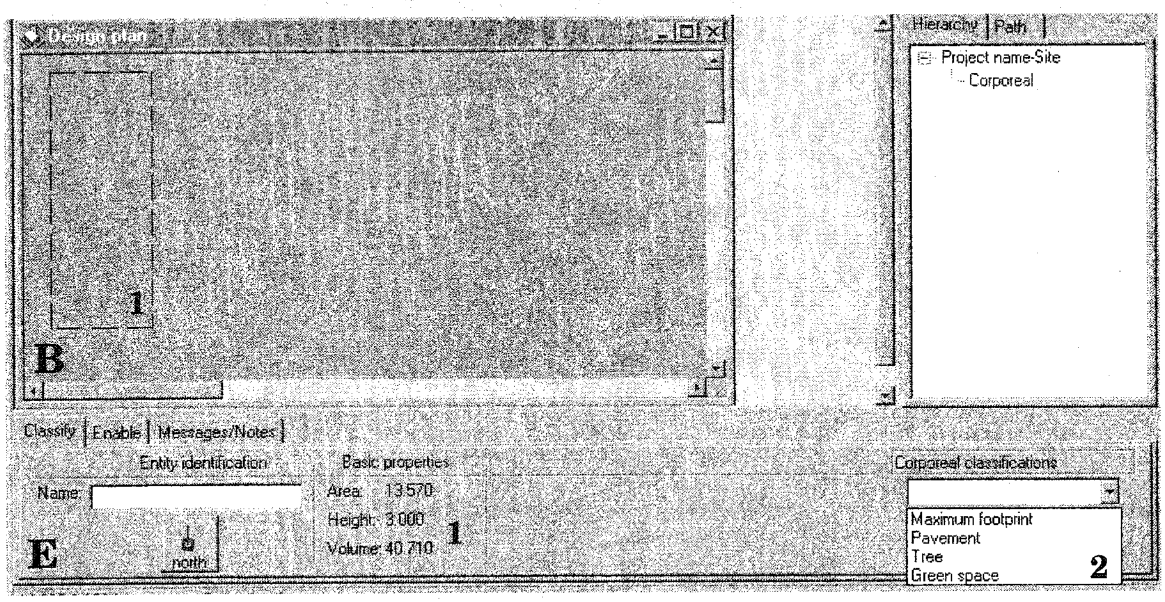

Figure 6.34: Preparing to classify a corporeal design item

When corporeal items are created, they are shown in long-dashed lines such as "B1" in Figure 6.34. This depicts their unclassified state. The system uses height information, provided by the designer using menu button 6.11, to calculate the volume. Such values are presented to the designer as shown in Figure 6.34 "E1".

When a corporeal design item is classified, the system represents it in solid lines as shown in Figure 6.35 "B1". It also labels the sides providing the designer with convenient references for "configuring" them. Configuration of classified corporeal design items can then be accessed in the classify window as shown in Figure 6.35 "E". The system presents the identification for the classified corporeal at "E1" (if the corporeal is under the mouse cursor) and the orientation of the design space (North Direction) at "E2". Basic properties inherited from unclassified corporeal design items is shown at "E3". Access to the boundary properties of the corporeal item is provided for selection and editing at "E4". By default, the boundary selected is "Side1" as shown in the "Name:" parameter. The other choices are "Side2", "Side3" and "Side4". Selecting a side allows the designer to change its status using the "Status:" list (default is "closed", with option of "open") and changing the material using the "Material:" list (default is "wood"). For each side shown

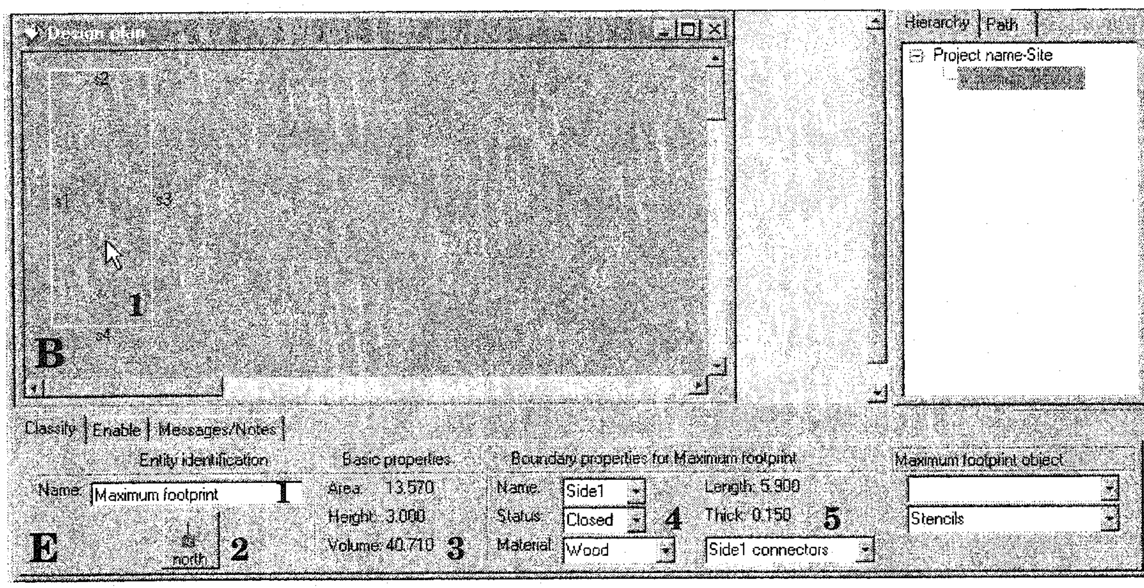

Figure 6.35: Classified corporeal design item

for configuration, length and thickness properties are displayed at "E5". The length can be adjusted when it becomes connected as illustrated later in Figure 6.41. Also when each boundary is selected, the list "Side1 connectors" can be accessed at "E5" to update the connectors.

6.2.4.2 Connectors (F)

Connectors are used to form relationships between corporeal types (see section 5.5.2). They can be added to the boundary of classified corporeal types through the same classification process as described in the last section but in such case, CoBL DT pays specially attention to the location of the connector in relation to the boundary that the connector is being installed in. Figure 6.36 shows the beginning procedure for adding a connector to a classified corporeal design item. The designer creates a new corporeal item near the side of the classified corporeal where the connector is to be installed as shown in Figure 6.36 at "B1", and selects "Door" for its classification as shown in Figure 6.37. The system creates a special icon that designates the location of a door item on the side as shown in Figure 6.38 at "B1". The interface automatically allows the designer to configure the

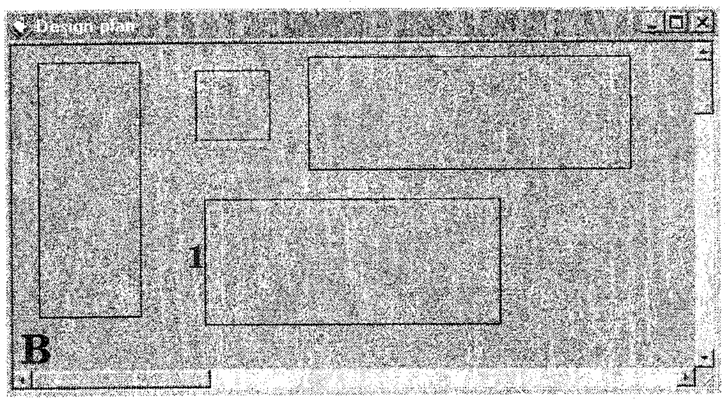

Figure 6.36: Creating a connector

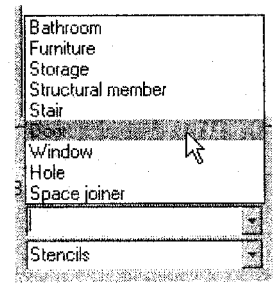

Figure 6.37: Classifying a connector

new connector, if desired as shown at the bottom of Figure 6.38 at "F".

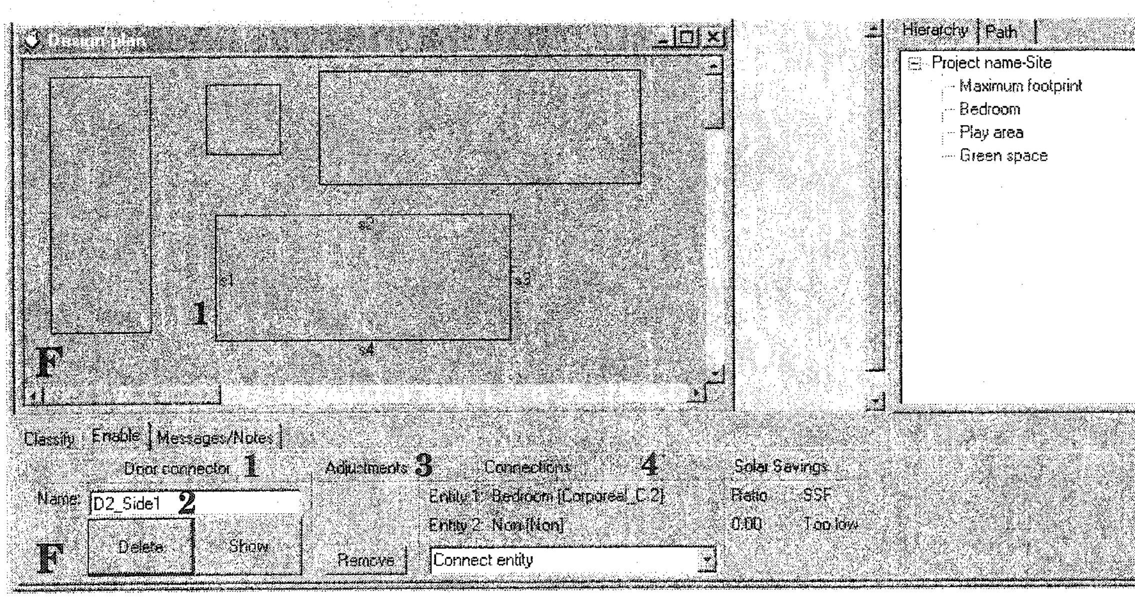

Figure 6.38: Using connectors

The system provides information about the type of connector at "F1" and its unique name in the design at "F2". The designer can remove the connector using the "Delete" button or find the connector in the design, assuming there are many others already existing, using the "Show" button which highlights the connectors in the draw window. The area for

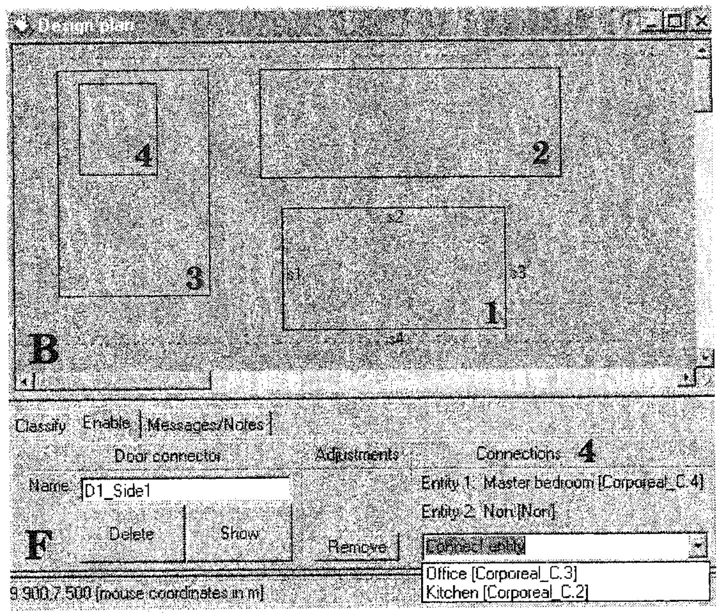

Figure 6.39: Possible connections

"Adjustments" at "F3" allows the designer to adjust the size of the connected side as well as to eliminate the connection provided by this connector. The area for "Connections" at "F4" provides information on active connections between spaces.

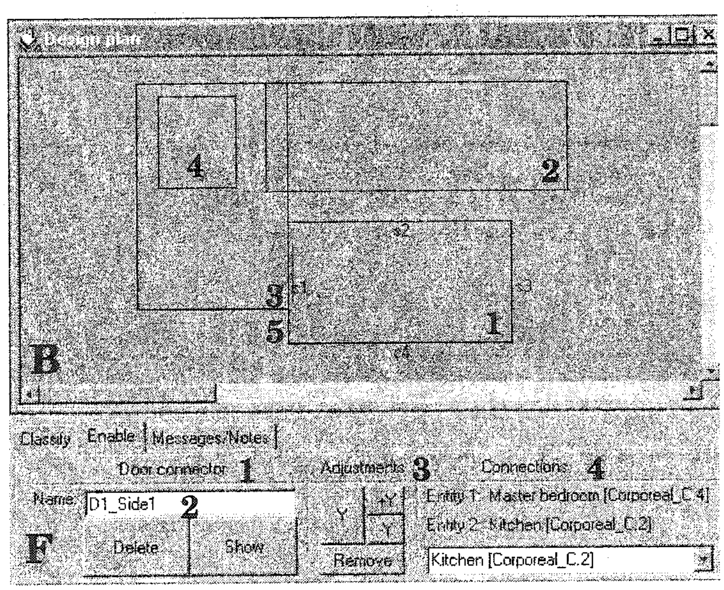

Figure 6.40: Connected design items

To establish a connection, the designer must first select another design item from the list as shown in Figure 6.39 at "F4". Figure 6.40 shows that, in window "B", corporeal design item "B2" "Office" and "B3" "Kitchen" are in the same hierarchy Level as "B1" "Master bedroom" which is the owner of connector type "Door" designated at "F1". Design item "B4" which is inside "B3" is not listed because it cannot be directly connected to the current item "B1", except through its parent "B3".

Connections enable two design items to be aware of each other but in addition they also make it possible for the designer to configure their boundaries with some accuracy in positioning. Figure 6.39 "B" shows that the designer can create spaces in approximate positions without worrying about accuracy; however, when connections are made the boundary of the item to be connected is modified to flush with that of the connector. Figure 6.40 at "B" shows the three spaces (Master Bedroom "B1", Office "B2" and Kitchen "B3") where a connection is made between "Master Bedroom" (owner of the connector) and the "Kitchen" with the boundaries flushed. Note that the Kitchen moves with the space inside it. Using the buttons in Figure 6.40 "F3", the size of "B3" is adjusted as shown in Figure 6.41 "B5".

Figure 6.41 shows a second connector (Window) added to the "Master Bedroom" and used to connect the design item "Office" at "B2". Note that in addition to modifying the position of items being connected, boundaries can be adjusted relative to the connecting item as shown in Figure 6.41 at "B5" (compare with Figure 6.40 at "B5"). This is accomplished through messages passed from the connector's boundary to the boundary of the design item to be connected. For example, when the "Master Bedroom" is being connected to the "Office" through the "window", the message to initiate a connection is first received by the window with an identification signature of the "Office". This message is forwarded to the "Side2" boundary of the "Master Bedroom" which is requested to send the appropriate location and/or size to the "Office". The "Office" uses this information to move/adjust itself accordingly. Both the "Mater Bedroom" and the "Office" then maintain constant communications regarding the location and size of the "Window" through their common boundary.

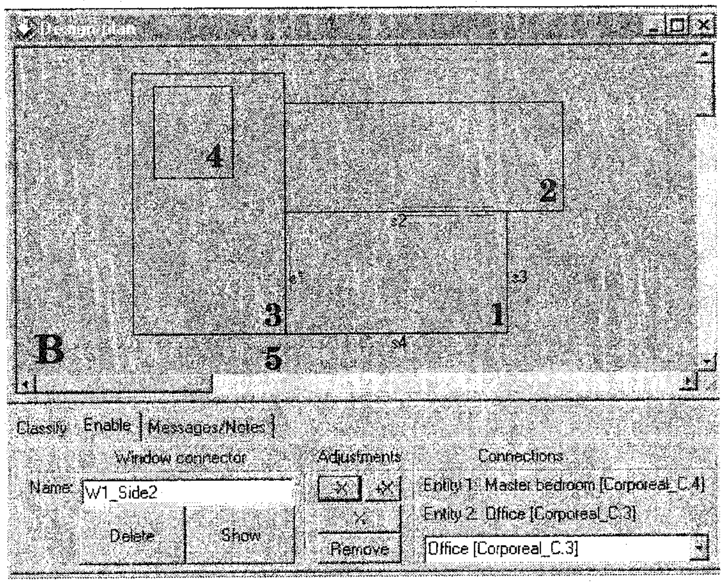

Figure 6.41: Adjusting connected items

Corporeal design items that have not been classified cannot contain other classified items because the system does not know their role yet, so the classification list cannot be determined for them. Design items can be classified long after they have been created. Likewise, it is possible to create connectors and return later to designate a connection with an adjacent space. Figure 6.42 shows the properties of a selected classified design item "Living room". Selecting a side (1 in this case) makes it possible for the designer

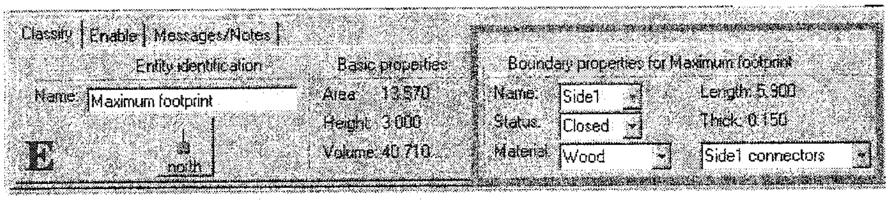

Figure 6.42: Accessing connectors

to access any of the connectors for that side from the "Side1 connectors" drop-down list. Selecting a connector here (door) will present Figure 6.43 where the designer can assign, delete or adjust connections.

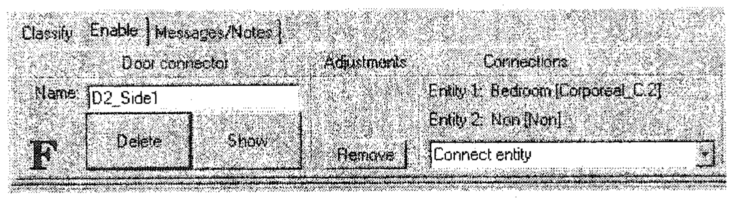

Figure 6.43: Connector properties

6.2.4.3 Notes/Feedback (G)

This window presents the designer with information regarding consequences of design actions or errors from the design process. An example of feedback provided by the system occurs when a classification fails. Figure 6.44 shows a corporeal to be classified at "B1" while Figure 6.45 shows a failed attempt to classify it as a stairwell. This failure is shown by the dash line at "B1". This failure occurred because the space provided by the unclassified design item is not enough for the proposed role of the item if it were to be classified. See section 5.4.1.1.

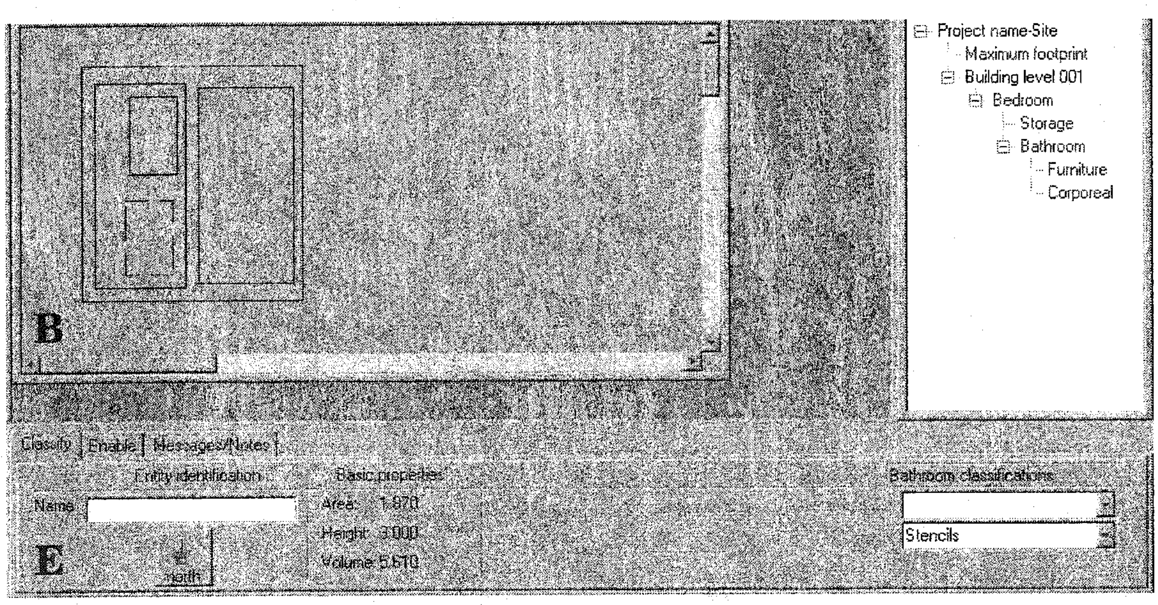

Figure 6.44: Corporeal to classify

When a classification failure occurs the system automatically changes the view from "E"

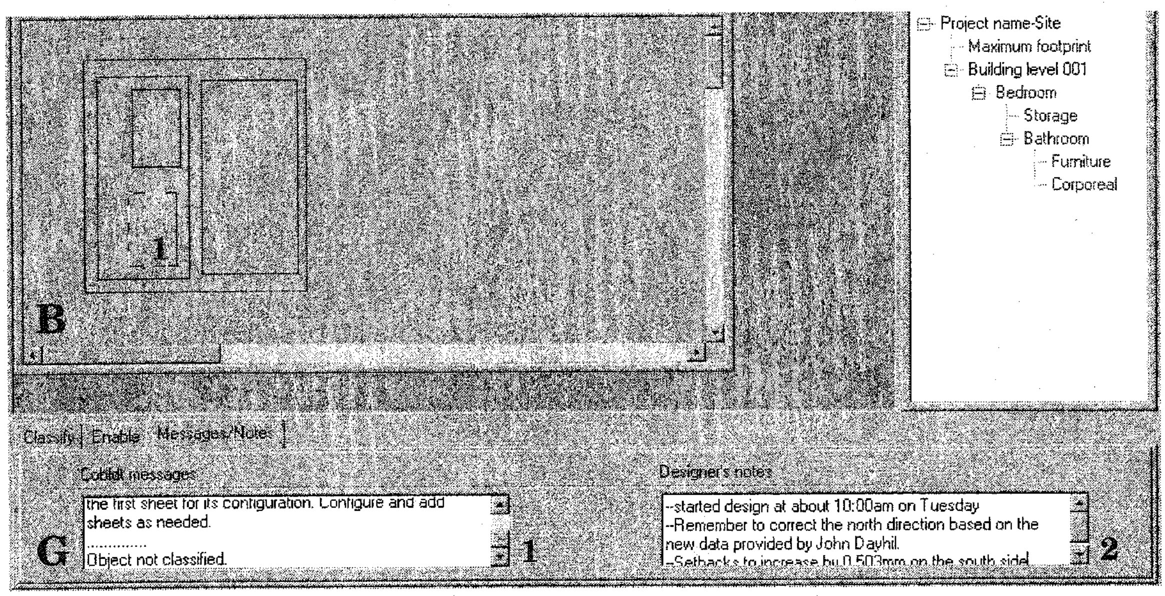

Figure 6.45: Feedback on failed classification

in Figure 6.44 to "G" in Figure 6.45 so as to draw the designer's attention to the problem. The error is reported at "G1". Subsequent messages are also added to this window so that all messages in design sessions can be reviewed. A second user-interface is provided for the designer to make personal notes as shown at "G2".

6.3 Application of Knowledge

CoBL DT provides assistance in situations where knowledge may be applied as discussed in 5.4. This section describes how this knowledge is implemented. It begins with a description of the structure of the central attribute/variable storage file which CoBL DT relies on for the facts it uses in making decisions. The implementation is then presented in two categories based on the formulations in section 5.4 namely, checking requirements and calculating properties.

6.3.1 Central Attribute/Variable Storage

CoBL DT considers (or can be set to consider) the designer's decisions in the background in order to provide assistance such as that needed to apply some engineering knowledge or rules-of-thumb. The attributes and variables used in making this possible is provided in a format that uses the four data fields of length, width, tolerance and height. Tolerance describes clearances around spaces such as that necessary for opening doors around a car. All fields require single values except tolerance which is a list of six numbers in the form of: "length, number_0, value_0, width, number_1, value_1". Reading these numbers the system will apply a tolerance of "value_0" to the total number ("number_0") of sides along the "length" of the design item and also applies a tolerance of "value_1" to the total number ("number_1") of sides along the "width" of the design item. For example Figure 6.46 shows a space with sides "A" to "D". The tolerance required to accommodate the space can be provided in the statement [length, 2, 150, width, 1, 100]. This states

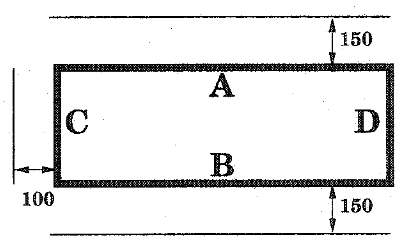

Figure 6.46: Illustrating tolerance determination

that along the length of the space, the two sides ("A" and "B") must have a minimum buffer of 150 while along the width, at least one side ("C" or "D") must have a buffer of 100. Table 6.1 shows the complete set of required fields for determining a standard space for a double bed in metric (millimeter) units.

length	:	1905
width	:	990
tolerance	:	[length, 1, 700, width, 2, 0]
height	:	0

Table 6.1: Fields for determining space for a double bed

Additional attributes may be added for components that need descriptions necessary for computing volumes such as in the case of checking and calculating stairwells. Table 6.2 shows such data where additional attributes with specific labels are provided to describe different aspects of a stairwell.

```
length         :  2533
width          :  915
tolerance      :  [ length, 2, 0, width, 2, 0 ]
height         :  2032
---additional information necessary for stair design---
minRiser       :  125
minThread      :  250
minAccessWidth :  1118
```

Table 6.2: Additional fields for building a stair

6.3.2 Checking Requirements

The designer uses building (or architectural) standards in the design process in addition to owner or designer's preferred standards. These predetermined items can be included in CoBL DT which are then monitored automatically to maintain a consistent or compatible set of design decisions. The following provides two examples of how such requirements can be checked.

6.3.2.1 Spatial Conformance

CoBL DT makes it possible for designers to create design items without worrying about precision or specific sizes but still be able to maintain standard sizes for all items created. When classification occurs, spaces must be able to perform their roles efficiently. This is partly achieved through spatial conformance as explained in section 5.4.1.1. The following illustrates how such conformity is maintained using garage design items as an example.

The architectural graphics standards provides the average dimensions of a car which are collected in CoBL DT as shown in Table 6.3 [Neufert, 1991]. These values are then

```
length    :  4750
width     :  1800
tolerance :  [ length, 2, 500, width, 2, 500 ]
height    :  450
```

Table 6.3: Standard car data

manipulated by a car utility design object which puts together the information in order to compare with a space for conformity. It returns a true or false answer to inform on the status of the comparison. Figure 6.47 shows two garage items drawn at different sizes. The first garage at "B1" has the size necessary to fit at least one standard car so

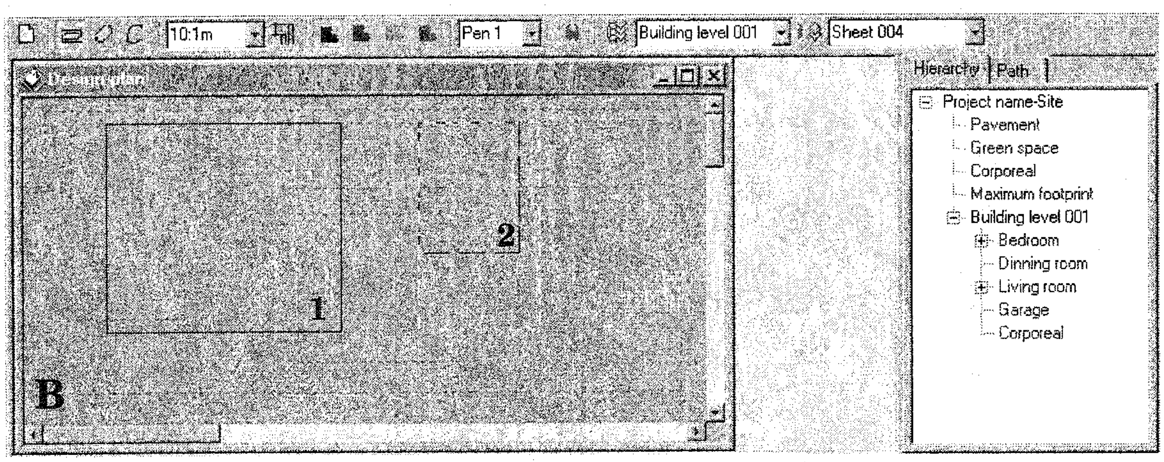

Figure 6.47: Checking spatial conformance

when it is classified, the system accepts it. The classification of the second design item at "B2" however, is rejected as a garage design item. When its dimensions are checked for conformance, it is found to be too small to fit a standard car. The system rejects it (i.e. does not classify it) but provides a visual feedback to the designer by temporarily drawing a white outline showing an acceptable size for a garage item.

6.3.2.2 Using Stencils

Stencils allow the designer to visually and interactively use standard design items, such as those from the architectural graphics standards, to design spaces. As opposed to merely returning a true or false value for a space to be classified, the car utility object displays a graphical representation for a car based on the dimensions given in Table 6.3. The stencil appears when and where the designer requests a "Standard car" stencil as shown in Figure 6.48 at "E1". The utility creates a graphic symbol which is attached to the

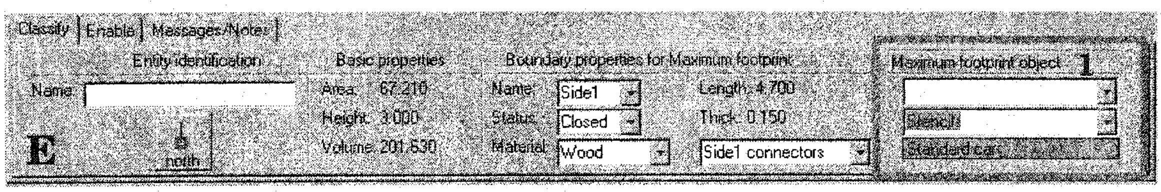

Figure 6.48: Choosing a stencil

mouse pointer, allowing the designer to place it interactively. More than one instance can be created and placed in this manner, making it possible to measure the required space for more than one vehicle. Figure 6.49 shows the use of car stencils to create a double-car garage.

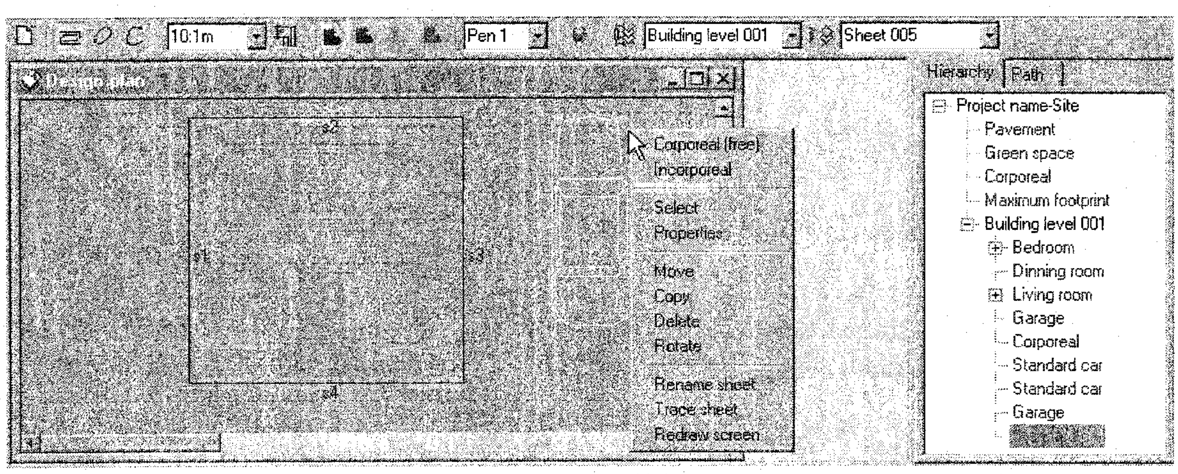

Figure 6.49: Using car stencils to create a double-car garage

6.3.3 Checking Properties

CoBL DT provides specific objects to check design properties of corporeal design items. Such checks usually affect or involve more than one design item. For example CoBL DT has an object for calculating the solar savings factor (SSF) of spaces. This calculation is performed each time a corporeal design item is classified. The overall solar savings factor is shown to the designer to help determine how efficient the space design is based on engineering design principles for heating and cooling of spaces. From Equation 5.1 and Table 5.2 in section 5.4.2, the following attributes are required. From Table 5.2,

```
---area of solar glazing as ratio of floor area---
ratio_low       :  0.25
ratio_high      :  0.50
---approximate SSF values with no night insulation---
SSF_no_I_low    :  0
SSF_no_I_high   :  0
---approximate SSF values with night insulation---
SSF_I_low       :  54
SSF_I_high      :  72
```

Table 6.4: Values for calculating SSF

"ratio_low" and "ratio_high" are labels identifying values for the lower and higher ratios of solar glazing area to floor area. "SSF_no_I_low" and "SSF_no_I_high" give approximate low and high SSF values when there is no night insulation whereas "SSF_I_low" and "SSF_I_low" correspond to the case with night insulation. These values are then applied to Equation 5.1.

Solar Savings Factor

CoBL DT can assist in the calculation of properties like the solar savings factor as discussed in section 5.4.2. The data for the calculation of this property is provided in Table 5.2 which can be applied using Equation 5.1. CoBL DT provides an SSF utility object that is created when the first space in a design session is classified. This object automatically obtains data (see Table 6.4) for the default region set in CoBL DT when it was installed.

The SSF utility object then monitors the orientation of design items on the site with respect to the north direction, the size of created spaces (floor size), window sizes and locations. This is accomplished by querying the CobldtNorth object for the true north direction which is set by the designer by clicking on the icon shown in Figure 6.50 at "E1".

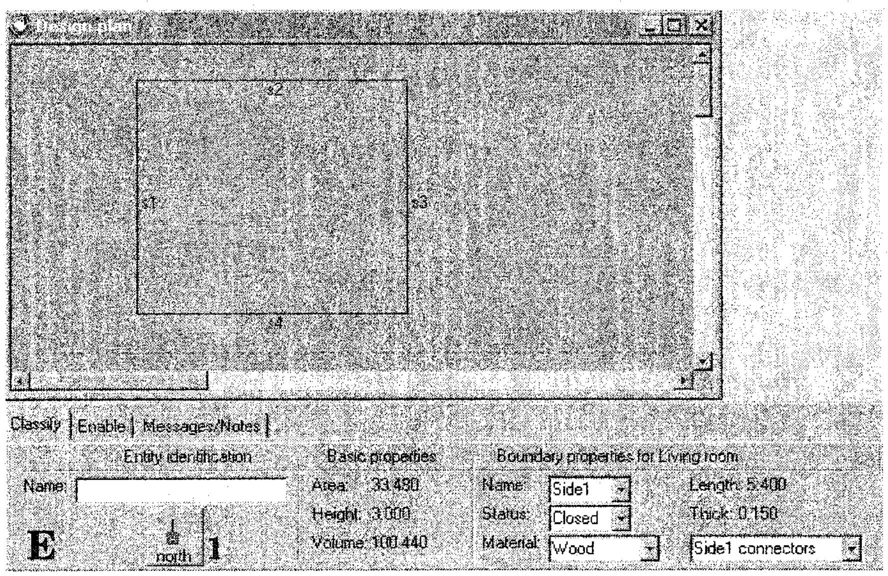

Figure 6.50: Design session showing North orientation

The size of floors and windows is obtained by querying classified objects. The collected

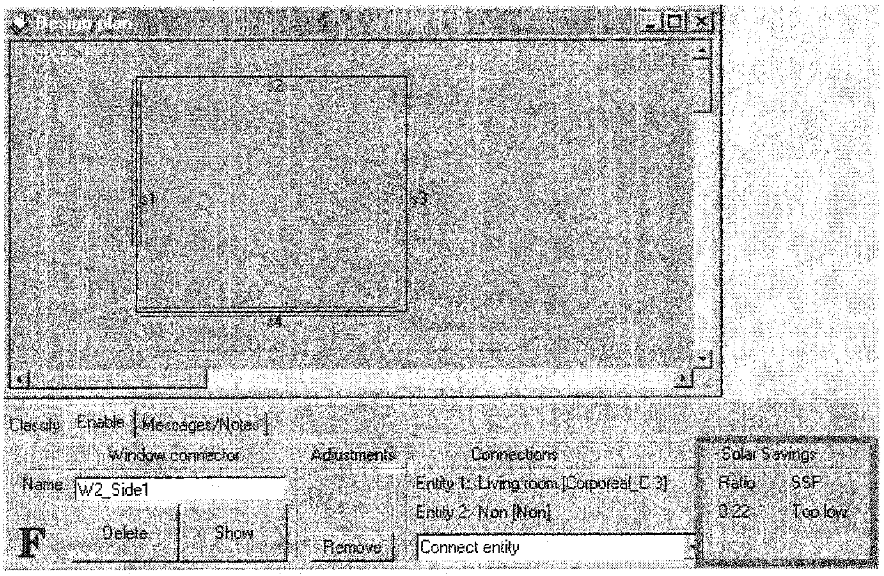

Figure 6.51: Inadequate solar savings

data is then used every time a new classification occurs to obtain the SSF ratio, compare it with the recommended values for the default region and then provides the designer with advice on whether or not the solar saving is acceptable. This keeps the designer informed on the consequences of spatial sizes/orientation and window sizes/location.

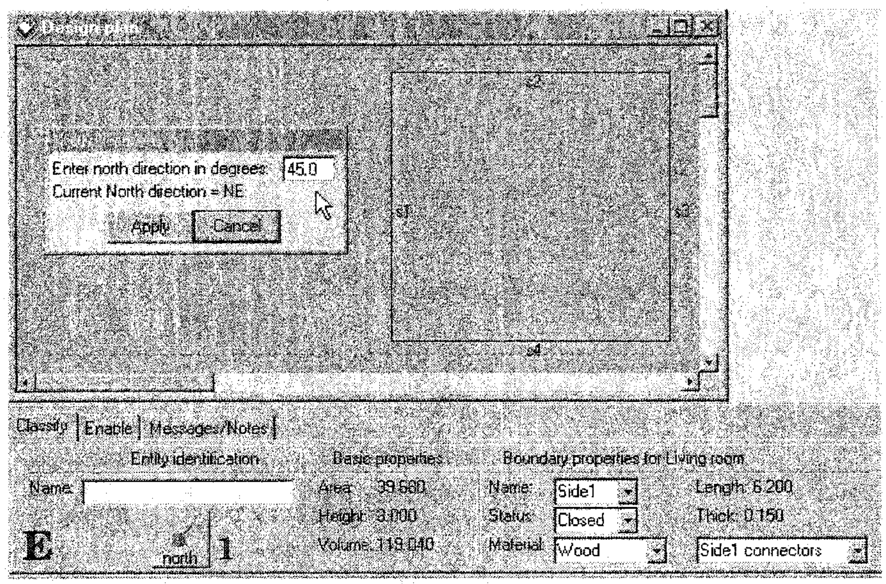

Figure 6.52: Changing orientation for favorable SSF calculation

Figure 6.50 shows the classification of a space with the North direction going upwards. When windows are added on both the south and west facades, COBL DT evaluates the SSF as insufficient to make substantial savings in artificial energy use (from the local utilities). This is shown in Figure 6.51. Acceptable values of SSF would be between the high and low values provided in Table 5.2 for any given region, which is 0.25 and 0.50 in this case for Edmonton, Alberta.

If the north direction is changed to 45^o in Figure 6.52 at "A" and a new, identical space is classified, the space is now being heated by the sun through the two facades with two windows. The combined glazing areas make it possible to admit enough solar heating to make a difference in the comfort Level of the space. Figure 6.53 shows that the SSF value is within the acceptable range.

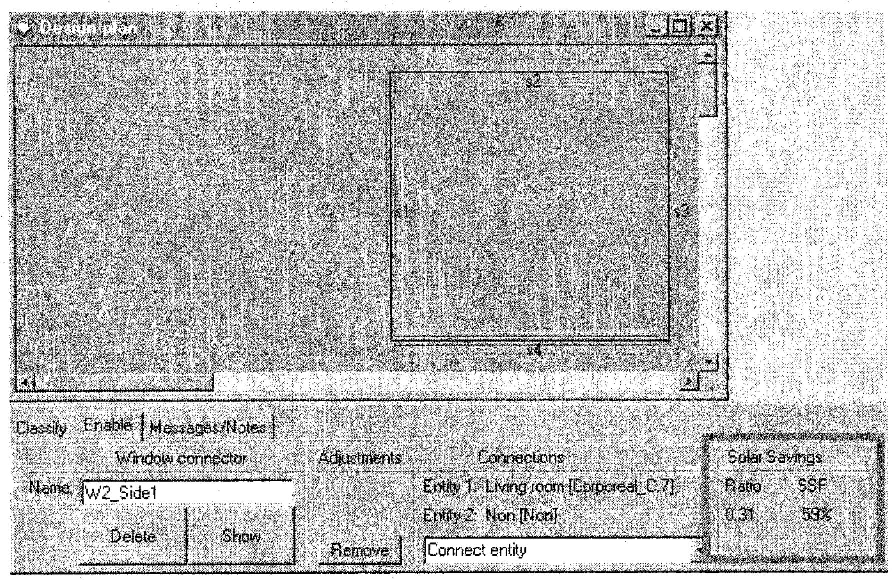

Figure 6.53: Adequate solar savings

6.4 Summary

This chapter describes the implementation of COBL DT, a system to assist designers in the early building design process. First the user interface is presented, showing how the design items created and manipulated by the designer are managed by the system. The interface description includes the description of available commands in COBL DT. It describes the use of corporeal and incorporeal object types to draw design items. It describes the capture of these object types in hierarchies and solution paths to show an overview and capture the design alternatives explored in the design. It describes the use of classifications to identify design items created by the designer and to assign them roles in order to enhance the support they provide. Details of how these design items are used in the design process is provided including how design notes and feedback are supported by the system. The support for the application of knowledge is described with examples. The performance of this system is validated in the next chapter against the requirements established in section 2.3.

Chapter 7

CoBL DT: Validation

AFTER HAVING DESCRIBED the design and implementation of a prototype in Chapter 5 and Chapter 6, CoBL DT, this chapter presents it in use during the early design process as a way to determine if it provides the necessary support for the early design process as discussed in this thesis. In order to provide a base for comparison, the design session described in this chapter follows the same presentation format as the sample design session in section 2.1. However the comments in each step of the session compares the two sessions to note any differences. In the end a summary of the advantages and disadvantages of using CoBL DT is provided.

The illustrations provided follow the same format used in the previous chapter where the command, draw, hierarchy, path, classify, enable and notes/feedback windows were referenced with "A", "B", "C", "D", "E", "F" and "G" consecutively.

7.1 Design Session with CoBL DT

The following problem (identical to session at section 2.1) is provided to the designer who is to create a solution using CoBL DT.

A residential house is to be located on a plot of land (specified in the site plan) to accommodate a family of five. Requirements include: 3 bedrooms, 2 bathrooms, living room, family room (preferably on the second level), kitchen, storage areas, outside green (or play) area, a garage for two cars, office with an area for woodwork. The office is to have a separate entrance.

7.1.1 Step 01

DESCRIPTION

At this point the designer simply needs to start CoBL DT.

COMMENT

Manual: The designer extracts requirements from the design brief (by reading and making notes) and lists them in a way that can be referenced easily.

CoBL DT: Common spaces and requirements for residential design need not be extracted from the brief. These items are programmed into CoBL DT.

7.1.2 Step 02

DESCRIPTION

Designer adjusts the north direction in Figure 7.1 at "E1".

COMMENT

Manual: The designer draws the north direction in a prominent location on the design workspace and coordinates the consequences of the orientation in the design project. The designer draws adjacent buildings and locates the proposed site entry point.

CoBL DT: All objects become aware of the site orientation and will use that knowledge during the design session, for example each space is now able to calculate its solar savings factor as described in section 6.3.3. Assistance for drawing adjacent buildings and locating the site entrance is not provided yet. The method through which support will be provided is already in place i.e. through classification, however the design items ("Site Entrance" and "Adjacent Building") have not been implemented yet. If implemented, the "Adjacent site" object can be classified as a normal design item while the "Site entry" can be classified as a connector.

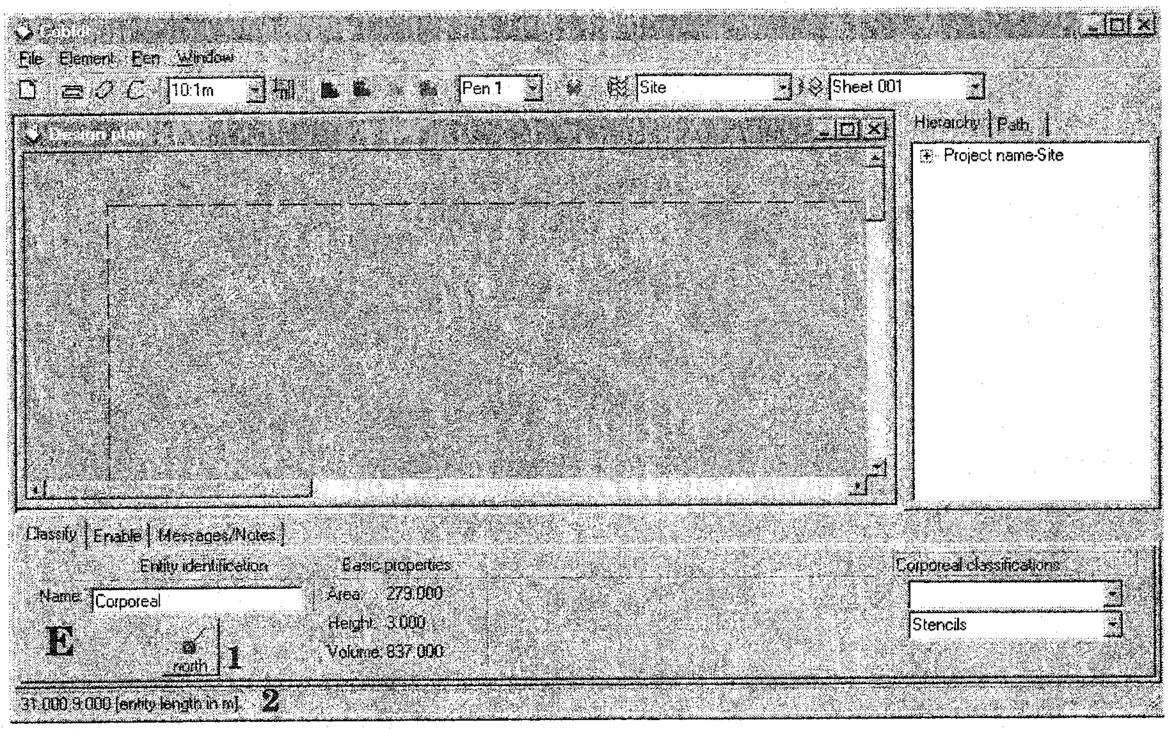

Figure 7.1: (Step 02) Adjusting site orientation

7.1.3 Step 03

DESCRIPTION

Designer uses a car stencil to offset the footprint at "B1A" to leave enough space for a driveway. Setbacks are marked off at various locations ("B1B") in order to be able to draw and classify the "Footprint" at "B1". The coordinate display shown in Figure 7.1 at "E2" is used to guide the drawing of the footprint.

COMMENT

The two sessions differ in the way the footprint was extracted because the designer had to

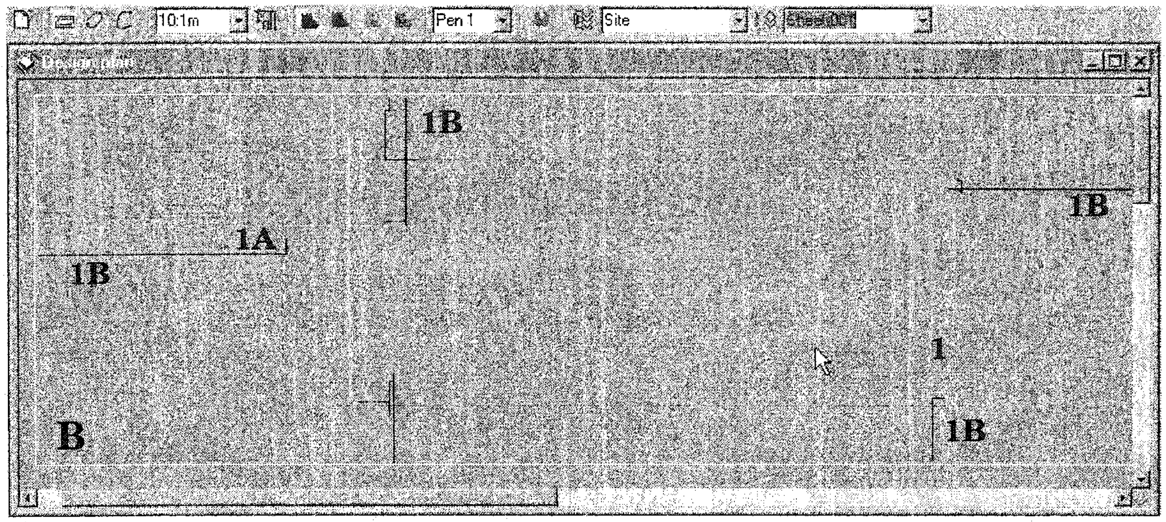

Figure 7.2: (Step 03) Extracting the maximum footprint

seek and use the Architectural Graphic Standards in extracting the footprint while using CoBL DT made the standard available right on the system as shown in Figure 7.2.

7.1.4 Step 04

DESCRIPTION

The designer draws all the spaces in the first level as shown in Figure 7.3 and includes a Garage at "B2" and Storage spaces at "B3" and "B4". A new space is drawn at "BX" and an attempt was made to classify it. The attempt to classify (into a staircase) does not succeed because the space is too small. CoBL DT informs the designer of the failed classification using white dashed lines at "BX" that show the minimum size possible for a space that can contain the straight flight of stairs. The designer deletes the first flight of stairs and replaces it with a bigger space that satisfies the minimum spatial requirements for a staircase. This is accepted by CoBL DT as shown in Figure 7.4 "B5". The designer finds this solution to be unsatisfactory and decides to explore an alternate solution. To do this, the designer first determines which item to branch from (i.e. point of departure from the current solution). Figure 7.4 shows the designer identifying an item in the tree.

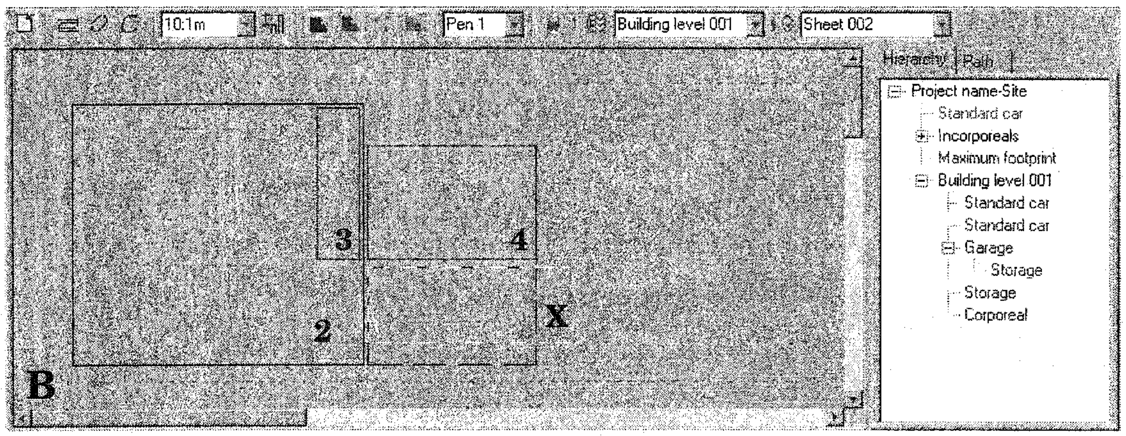

Figure 7.3: (Step 04) Failed classification

Moving the mouse over an item at "B4" highlights it in the hierarchy tree at "C4".

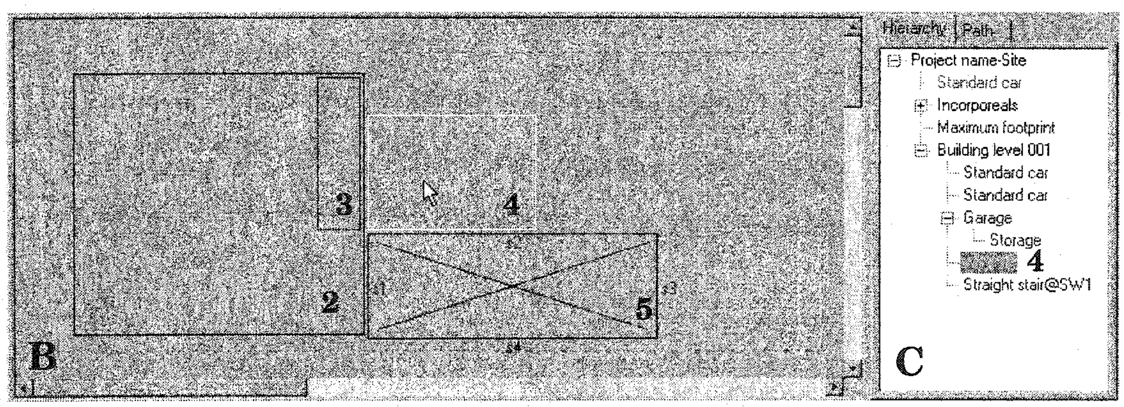

Figure 7.4: (Step 04) Identifying corporeal design item in hierarchy

Once the item is identified in the tree, the designer right clicks on it to identify its position in the solution path (i.e. the step it represents) as shown in Figure 7.5 at "C1". The Path window (Figure 7.6 at "D") is made visible by clicking at "D1". The appropriate step is then selected with the right mouse button to change to solution 2 as shown in Figure 7.6 at "D1".

COMMENT

Manual: The designer starts to draw at a smaller scale than was used for the site plan. The designer references prior drawn design items (stencils), such as cars, to be able to

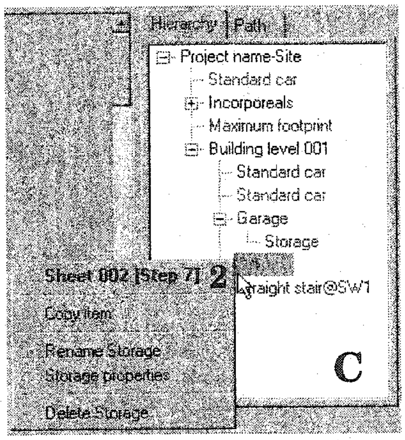

Figure 7.5: (Step 04) Corporeal design item's path number

create the garage space. The designer creates a straight flight of stairs. Then in order to coordinate the garage and staircase locations, begins the configuration of the upper floor. This helps in the making of a decision regarding the size and location of the staircase. When considering the progress of the design session, the designer monitors the list of requirements and eliminates spaces and requirements that have already been considered.

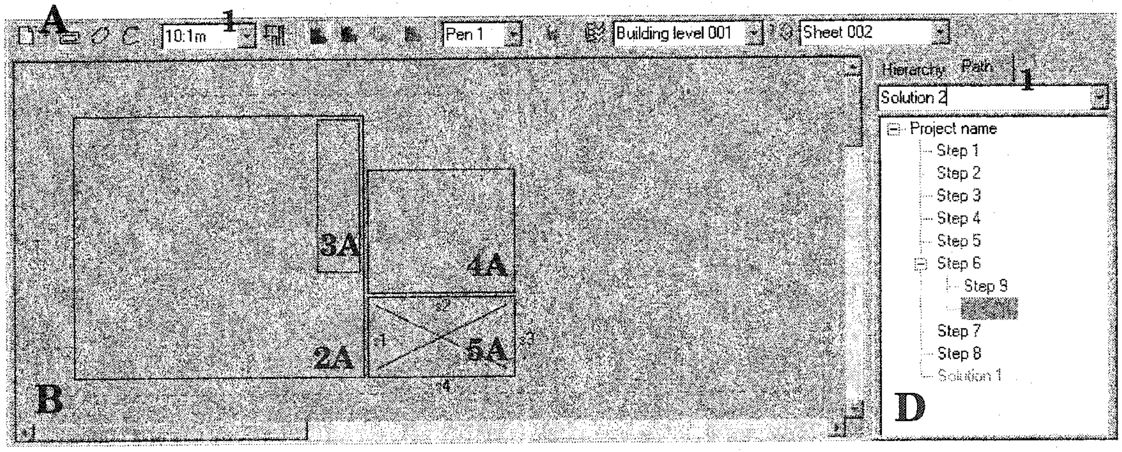

Figure 7.6: (Step 04) The first level—solution 2

<u>CoBL DT:</u> Scales can be changed by using the list box shown in Figure 7.6 at "A1". However, the choice of scales implemented so far does not change the zoom factor of

the drawing interface. Stencils are provided by the system and eliminate the need for the designer to draw/manage them. The size of items are checked when they are created to make sure that they will provide an adequate space for the role they are supposed to perform. The designer automatically knows if the staircase is appropriate or not. This information makes it possible for the designer to decide whether to move on with this option or to seek an alternate solution. The list of spaces created in the design session is automatically managed by the system, including showing their relationships with each other using the hierarchy window in Figure 7.7 at "C".

7.1.5 Step 05

DESCRIPTION

The designer changes to the second level, traces the "Living room" at "B6" and copies the "Circular stair" from the previous sheet to "B7" as shown in Figure 7.7. Then draws the "Kitchen" ("B10") and the "Foyer" ("B12") which is drawn inside the "Dining room" at "B11". The "Play area" is drawn at "B8" which contains a "Bathroom" at "B9".

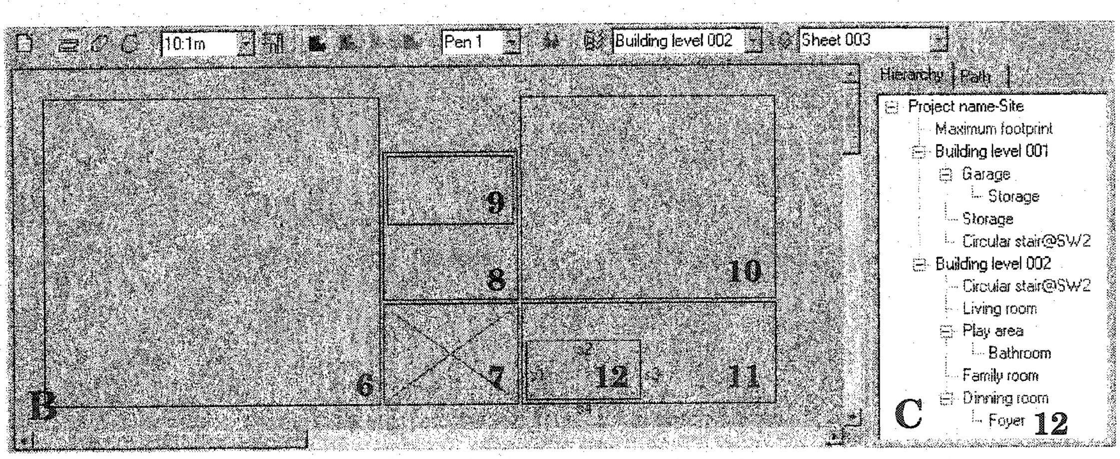

Figure 7.7: (Step 05) The second level—solution 2

COMMENT

Manual: The designer overlays the previous sheet with the new sheet in order to refer to the first level spaces while creating the second level spaces. All new items created must be traced. The designer reviews the design by putting together the different sheets that make up the design solutions.

<u>CoBL DT:</u> The tool gives the designer the option of tracing or copying items from previous sheets. Items are reproduced exactly when copied including the doors, windows and other included spaces. During tracing however, new items are drawn while referencing previously drawn ones. Copying is achieved through right-clicking on the item to be copied in the hierarchy window ("C"). To trace an item in a previous sheet the designer right-clicks the mouse button to access the "Trace sheet" menu as shown in Figure 7.8 at "X1". Selecting this menu opens the window at "X2" which allows the selection of the building level and the sheet containing the item to be traced. On this window the command "Trace sheet" shows the objects in the selected sheet so that the designer can refer to them. "Clear sheet" clears the current referenced sheet. "Clear all" clears all referenced sheets. "Done" ends the tracing command. Figure 7.8 shows the "Living room" "B6"

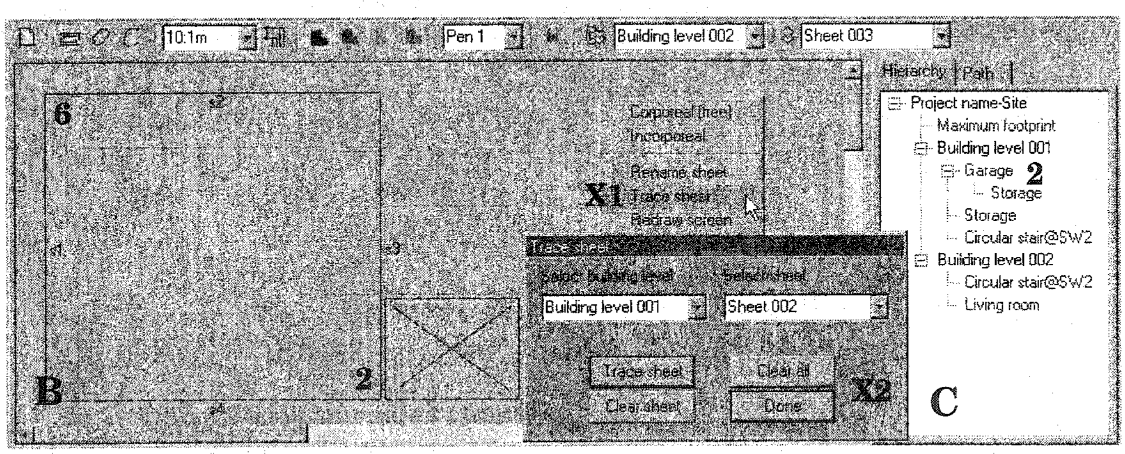

Figure 7.8: (Step 05) Tracing items in other sheets

being created in reference to the "Garage" "B2".

7.1.6 Step 06

DESCRIPTION

The designer is not satisfied with the location of the "Foyer" in Figure 7.7 at "B12" and

"C12". At first it is just moved to the "Living room" shown in Figure 7.9 at "B12" and "C12". On second thoughts however, the designer decides to keep it in the dining space as an alternate design. The "Foyer" is returned to the "Dining room" and the designer

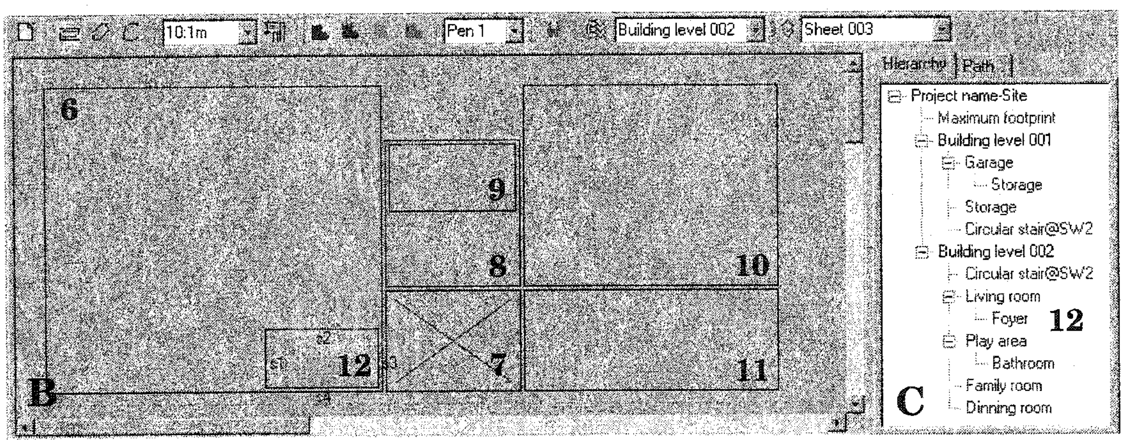

Figure 7.9: (Step 06) Moving "Foyer"

changes path to create "Solution 3" as shown in Figure 7.10 at "D1" the beginning of which is indicated by "Step 18". A new "Foyer" is then created within the "Living room" space at "B12A".

COMMENT

Manual: Design alternatives are carefully captured by the designer through the use of drawing sheets and posting the transparent sheets on the walls as references.

CoBL DT: The designer reviews the design by browsing the solutions using the solution path window "D" shown in Figure 7.10. The solution path makes it easy to save different alternate designs and to combine different parts to form better solutions.

7.1.7 Step 07

DESCRIPTION

The designer goes back to the first level to finish the configuration of the "Office" at "B14", which includes providing a separate entry at "B13" and two additional spaces as

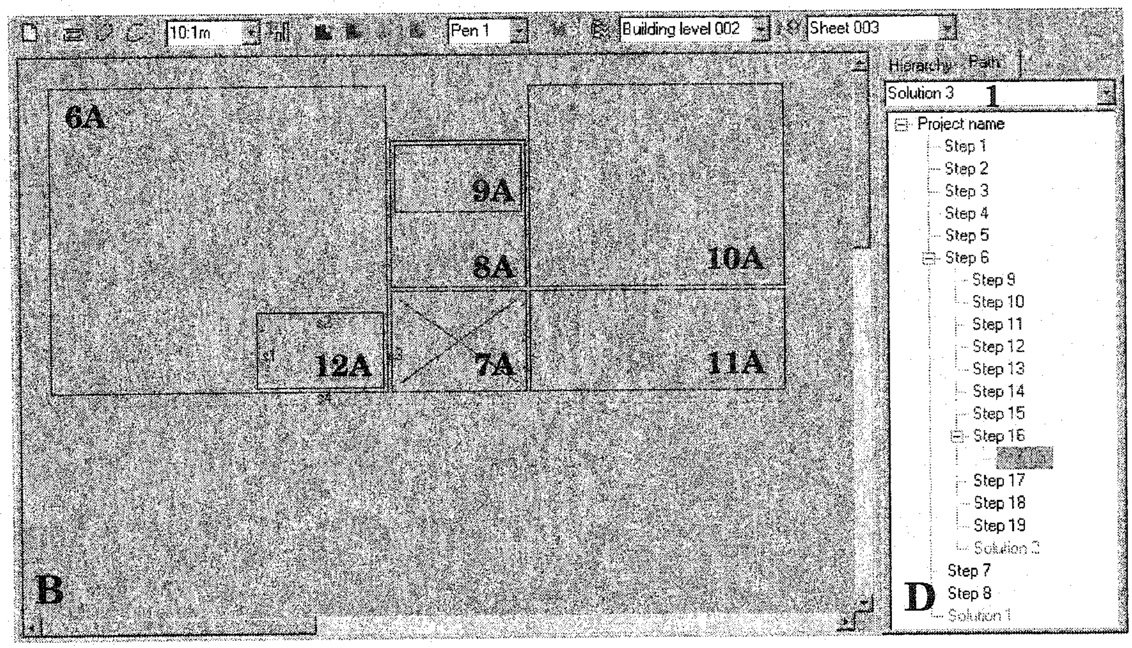

Figure 7.10: (Step 06) Alternate Solution 3

shown in Figure 7.11 at "B15" and "B16" which should be the "Library" and "Woodwork" spaces but are classified as "Storage" spaces.

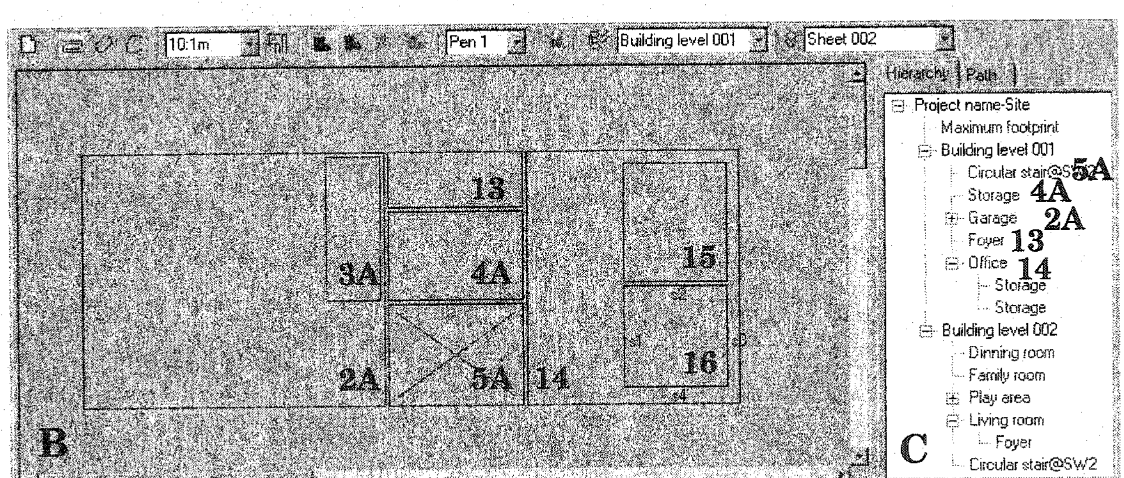

Figure 7.11: (Step 07) Finishing first level configurations

COMMENT

Manual: The designer creates spaces by drawing or labeling as required.

CoBL DT: At this point "Library" and "Woodwork" spaces cannot be created using CoBL DT because they have not been implemented yet. Instead two "Storage" spaces are created in "B15" and "B16".

7.1.8 Step 08

DESCRIPTION

The designer creates a third level and copies two spaces to this level from the second level. However, the design session ends with no additional work. Figure 7.12 shows the finished first level. It also shows that the designer has connected the spaces using windows and doors.

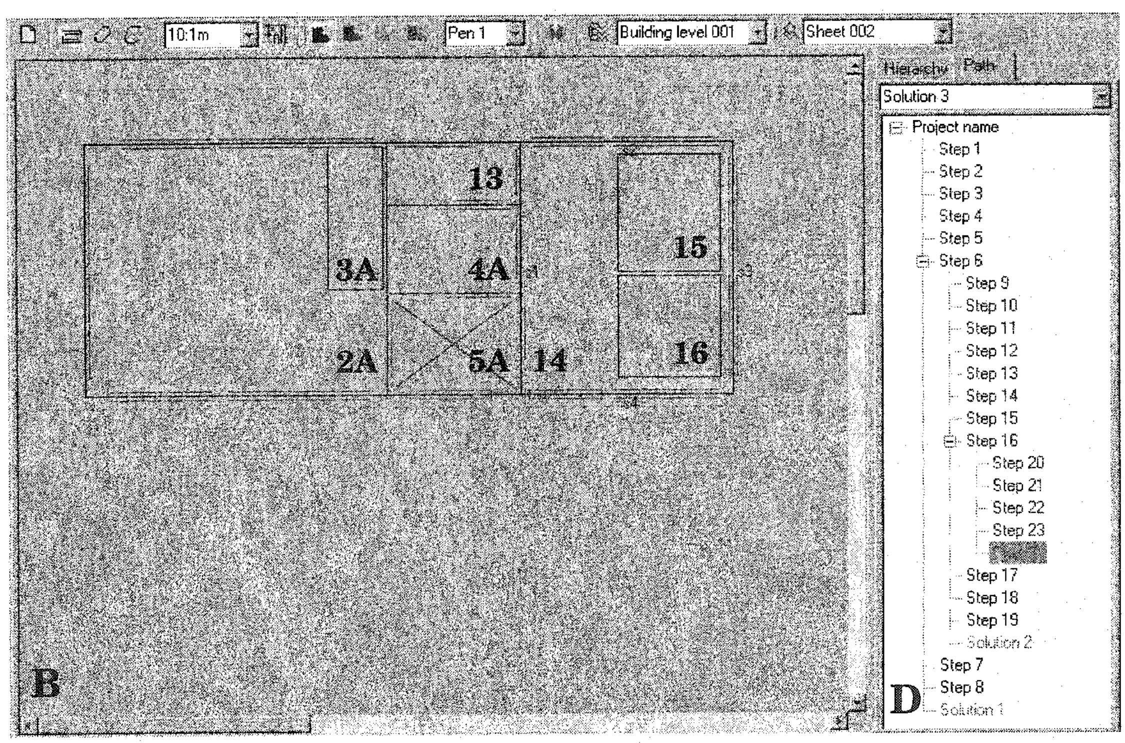

Figure 7.12: (Step 08)Finished first level

COMMENT

Manual: At this step the designer begins to redraw the second level with more accuracy.

More work is completed using additional steps before the designer arrives at a stage similar to the current one in which CoBL DT is being used.

CoBL DT: Redrawing of the design is not necessary because the corporeal design items automatically use appropriate dimensions (they could not be classified otherwise). The designer only adds windows and doors to the design (connects the spaces) in order to complete the solution configuration.

Figure 7.13 shows the finished second level.

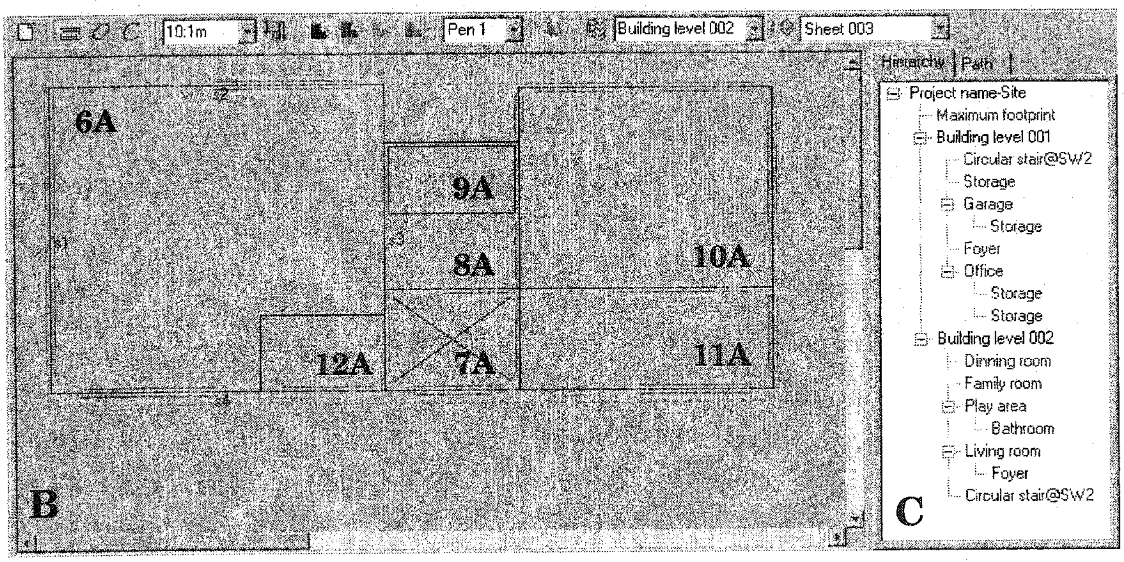

Figure 7.13: (Step 08) Finished second level

Figure 7.14 shows the hierarchical decomposition view of the accepted solution.

7.2 Summary

The use of CoBL DT in the design process has provided some advantages over the manual process although there are parts that need to be improved. The following is a list of advantages and disadvantages.

CoBL DT provides the following advantages over the manual process.

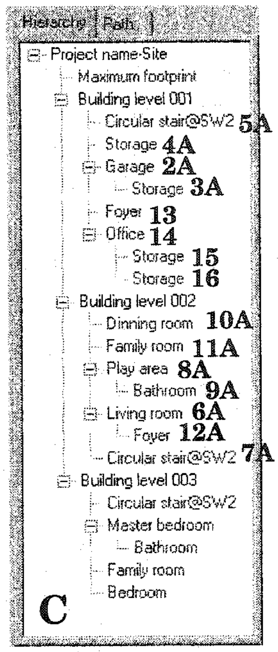

Figure 7.14: (Step 08) Hierarchical decomposition view

- The designer is told about consequences of decisions earlier on, which helps to conceive a well rounded solution. For example information (and assistance in creation) is provided regarding the minimum required size of spaces and how much passive solar energy is being gained in a space (see section 6.3.3).
- The designer does not spend time drawing items, instead more time is directed at solving early design issues such as consideration of spatial relationships. For example the walls and doors for the spaces are not drawn, the system only requiring the location of the space and the position of the opening (door).
- CoBL DT separates the constituent parts of a design project to expose how they come together as a solution. It shows the relationship between the different parts of a solution as a hierarchical decomposition as shown in Figure 7.14. This gives the designer an overview of the design process as well as a mental picture of the direction in which it is going.
- The system automatically documents the design process because the steps followed by the designer, shown in Figure 7.12 at "D", are automatically saved as shown in

section 5.3 . This makes it possible for the designer to backtrack to any point in the exploration for design solutions. The ability to document the designer's work through the collection of solution paths makes it possible to browse the design making it possible to either study the work process or to merge different parts to form a single best solution for the design.

- There is no need to redraw the final solution in order for the solution to be clear and well understood.
- It facilitates the use of engineering knowledge earlier on in the design process. Most designers find it difficult to stop and make calculations especially engineering calculations that are involved. The system makes it possible to automate such knowledge in a way that is easier to use in the early design process.
- It reduces error in the interpretation of design decisions. For example including a living room space in the design will not only show its label in the hierarchy view but its properties can be accessed as well. Any specific relationships being maintained with other items in the design will not be accidentally missed as the object is responsible for maintaining such a relationship.

CoBL DT provides reduced functionality in the following areas although more development should remove such limitations.

- The designer is limited to rectangular shapes. This is a problem in early design. The manual design process does not limit the designer in the variety of ways to draw design items or the type of design items to create. CoBL DT by contrast requires the designer to draw a corporeal design item which has to be a closed and rectangular figure.
- Customized spaces can be easily and quickly added to the design solution after the process has started as this requires only a new label in the manual process. For example Figure 7.11 shows the use of storage spaces in the office at "B15" and "B16". These items should have been "Workshop" and "Library" objects but they were not available in the current implementation of CoBL DT.

The comparison of the two design sessions show that the method of support suggested in this research in the form of the prototype CoBL DT is effective in supporting the early building design process. It reduces the amount of time spent on repetitive and physically demanding work such as the need to draw walls and precisely locate openings while allowing the designer more time to think about design. CoBL DT asssists the designer in coordinating information between sheets in a way that provides a superior overview and access to all parts of the design through the use of hierarchies. All solutions are managed in a way that makes it easy for the designer to explore the design in an in depth manner. Although not demonstrated here, CoBL DT adds the application of knowledge which not only brings a capability not available in the manual session without much effort, it introduces an interactive environment that is important in an early building design process.

Chapter 8

Summary and Conclusions

OBSERVING WITH SOME concern that computers are currently prevalent only in the later stages of design, yet conspicuously absent in the earlier stages when foundations for decisions in the later stages are created, this research project started with a few goals:

- The need to understand and present the early design process in such a way as to identify the parts that can benefit from computer support.
- To specify the type of support that can provide appropriate assistance for such a process, bearing in mind that the early design process is a very unique, personal and unstructured activity.
- To develop a prototype system in accordance with specifications to demonstrate one way of providing assistance.

This chapter summarizes the work accomplished and the contributions achieved in this research project. The last section outlines future work that could bring these contributions to full effect.

8.1 Summary

The design of buildings, especially in the early stages, is a demanding process that is made more complex by the unique nature of each new project and the need to resolve collaboration efforts between professionals of different specialties.

8.1.1 Requirements for Early Building Design Support

To provide a basic understanding of the early building design process, a sample traditional design session is first described. Analysis of this session shows important characteristics such as the manner in which spaces are created and manipulated, the identification of spaces created through labelling and the need for the designer to think about spaces and spatial relationships using simple forms, lines and marks on transparent sheets.

This sample session however presents a single designer's process and therefore a limited view from which to identify potential characteristics for appropriate computer support. To broaden this view a review of published research works on the early design process is carried out. Key issues regarding the nature of the process are identified and although many correlate with those introduced in the sample design session, the review provides additional knowledge of the process to establish preliminary requirements for design support. These requirements include *interface issues*, *design item recognition*, *collection of alternatives*, *design overview* and *knowledge integration*, all of which provide a guide to the functionalities needed when supporting the early design process.

A comprehensive review of currently available computer-based support for early design is carried out, using six categories of systems: 2D, 3D, integrated, virtual reality, generative and integrated design exploration. Based on this review, current support is found incapable of satisfying the requirements outlined above.

8.1.2 Specifications for Computer-Based Support

According to the review of literature and existing computer-based systems, there are no specific descriptions of how requirements can be satisfied in a digital environment. To obtain this information a protocol study, especially focused on the designer's workspace, is performed. Focus on the designer's workspace is important because the thesis is investigating assistance to designers in executing (organizing and managing) the early design process as opposed to support for design thinking. Eight designers ranging from recent master of architecture graduates with a minimum of 3 years experience to expert designers with over 20 years experience in the practice of architecture are included in the study. Each designer is subjected to a 1 hour session, 15 minutes interview and 45 minutes observation of design process. Each session is analyzed to extract the characteristics of the early building design process that need to be supported in a computer environment. These include the need for the designer to use *transparent sheets* to minimize drawing effort, *labels* to recognize spaces, *shades and renditions* for emphasis, *top-down* and *bottom-up* design sequences and *multiple drawing views* to enhance design perception as discussed in section 4.1.3.

Based on this analysis, eight specifications are extracted to guide the design and implementation of the computer-based prototype. These are discussed in detail in section 4.2 and summarized as follows:

- *Requirements repository* which specifies support for a means of capturing and managing design requirements.
- *Application of requirements* which specifies the need to constrain to certain parameters in the design process.
- *Multiple levels of abstraction* which specifies that design items should be viewed in different ways such as showing spaces with their boundaries (walls) or without.
- *Solution management* which specifies the need to collect design solutions being investigated by the designer and allow the exploration of several alternatives.
- *Element interaction* which specifies the need for design items to react with each

other (to determine how to connect them, for example) or with the designer (to determine feedback).

- *Automatic Feedback* which specifies that feedback should be provided not only for errors but also to serve as reminders or reports on the state of the design.
- *Design Overview* which specifies the need to capture relationships between design items using hierarchies.
- *Design Liability* which specifies that the designer must be allowed full control of the process including ability to impose their personal style of design using the supporting tool.

Due to limitations in time and resources, the following subset of specifications is selected for implementation in the prototype: **Application of Requirements**, **Solution Management** and **Design Overview**. However for the successful consideration of these specifications, it is also necessary to partially consider others which include *Multiple Levels of Abstraction*, *Element Interaction* and *Automatic Feedback*. *Requirements Repository* and *Design liability* are left for future work.

8.1.3 Implementation

Based on the requirements and the specifications including the features discovered during the protocol study/analysis, an object-oriented prototype system called CoBL DT is designed using the unified modeling language (UML) for documenting the process. CoBL DT provides support for the early building design process in two main ways. First through a carefully designed **user interface** that provides support through assisting the designer in the creation and management of design information. Secondly through the **application of knowledge** that supports the designer's interactions and widens the scope of available expertise utilized in completing the early building design session.

The interface is made up of three parts which include the **drawing**, **classifying** and **collecting** of design items and solutions. The drawing part makes it possible for the designer to draw abstract and concrete design items typical in early building design sessions (see section 3.1). The classifying part makes it possible for the designer to designate the spaces and other design items being created so that the prototype can accurately determine their role in the design. Once roles are determined, the items are provided with basic knowledge, which gives them the ability to interact intelligently with each other and with the designer. Feedback of the consequences of the designer's actions is meaningful and support is provided in the coordination of the features and characteristics of the design under development. This greatly improves interpretations of early design ideas and compatibility of design issues.

The collection part simplifies the complexity of early building design by showing the relationships between design items so that an overview of the work being done is maintained at all times. This makes quick access to both individual design items and solutions possible as well as alternate solutions. This organization is automatically provided by CoBL DT thereby greatly reducing the management effort expended by the designer, whose concentration can then be directed on design thinking and resolution issues.

CoBL DT assists the designer's process by providing design items that use knowledge in two ways. First, based on their roles, design items have to determine if typical building standards or the owner and designer's requirements have been satisfied. This determination is usually carried out by each design item, for example the "garage" design item must establish that the space provided for it is enough for the number of cars to be accommodated. Secondly, specialized design items are available that coordinate knowledge that is beyond typical or common design requirements utilized in the early building design sessions. These are usually knowledge beyond the common designer's expertise but which provide greater understanding of the design issues allowing the making of better informed design decisions. A typical example of this is the calculation of the solar savings factor which, if considered during early design, can greatly reduce the cost of heating and/or cooling of the building throughout its life cycle.

8.1.4 Assessment of Support

The prototype is finally tested in a design session that solves an identical design problem as that addressed in the sample traditional session described in section 2.1. The test shows that using CoBL DT facilitates the management of design items as well as reduces the number of steps used in the design process. The designer is thus able to maintain an overview of all available design solutions. This makes it possible for the designer to explore alternatives easily and to combine or browse among different alternatives to create better overall solutions than would be possible using only a traditional (manual) approach.

8.2 Contributions

This research project has generated five main contributions in the following areas: requirements for conceptual building support, digital environment support features, specifications for developing conceptual design support, proof-of-concept prototype and validation of the prototype. The following sections describe each contribution.

8.2.1 Requirements for Conceptual Building Design Support

This research describes the early building design process as it is currently understood based on analysis and literature review. An analysis of a traditional design session reveals what designers typically need during the design process to successfully complete it. These requirements are discussed in detail in section 2.3 and include:

- Appropriate Interface
- Ability to recognize the role of design items
- Exploration of design alternatives
- Overview of the design space
- Application of knowledge in the design process

A detailed description of the main issues affecting the early design process and the definition of the basic requirements for supporting this process with computers is presented.

These requirements are then used as the basis to evaluate existing computer-based tools to determine the current state of support for design with computers. The tools evaluated are divided into six categories of 2D, 3D, Integrated, Virtual Reality, Generative and Interactive Design Exploration systems. 2D systems do not provide adequate support but were evaluated to better understand the problems of interface issues. 3D systems build upon the 2D systems by adding the use of knowledge to support the design process. Interface issues however prevent this from practical applications in the early design process. Integrated systems build upon the 3D systems by providing some recognition in the design items so that construction information can be extracted and synchronized as details and changes are made. This however is further down the design process and does not occur in the early phases.

Virtual Reality systems create new solutions which properly address interface issues although this is currently difficult due to access of available equipment. Generative systems provide new solutions that encourage the exploration of design ideas however the exploration is not controlled in the way that is required in the early design process. The Interactive Design Exploration systems satisfied more requirements than the other tools providing support in interface issues, recognition of roles and knowledge integration but limited support is provided in design overview, interface issues and solution path

8.2.2 Protocol Study of the Early Stages of Building Design

This research project contributes to a better understanding of the early stages of design through a protocol study. The main findings which are described in sections 4.1.2 and 4.1.3 include the need for designers to operate in four sequential activities namely, Design Brief, Site Preparation, Building Space and Building Elements. Specific events are performed in these activities which are used by the designer to introduce or manipulate design items such as the following:

- Drawing alternate sketches
- Backtracking to a previous point in the design process
- Grouping of spaces
- Labeling of spaces
- Use of transparent sheets
- Use of shades or renditions
- Use of elevations, sections and 3D drawings
- Use of simple forms in the representation of spaces
- Use of top-down/bottom-up approaches in the development of design items.

The events provide examples of support that can be used in the activities to direct the resolution of design issues. These are necessary parts of the design process which will need to be correctly supported in a digital environment.

8.2.3 Specifications for Developing Conceptual Design Support

The computer environment is a unique and powerful one in which to support a relatively unknown process in ways that are not yet recognized in practice. There is a need to provide specifications to guide the creation of tools that provide appropriate support.

Results from the protocol study were analyzed in order to draw a set of specifications to guide the design and implementation of a software system for supporting the early building design process. These specifications are an attempt to match the characteristics of the early building design process with the capabilities of computers in order to address the most important parts of the process with the most effective features in the computer. These specifications are presented in detail in section 4.2 and include the following:

- Requirement Repository
- Application of Design Requirements

- Multiple Levels of Abstraction
- Solution Management
- Element Interaction
- Automatic Feedback
- Design Overview
- Design Liability.

Some of these specifications namely Application of Design Requirements, Solution Management and Design Overview are used to determine the nature of support to be provided in a computer-based system.

8.2.4 Proof-of-Concept Prototype

A proof-of-concept prototype (CoBL DT) is developed to provide a solution to a subset of the specifications as identified above. Using the subset of specifications, an approach is proposed that consists of the creation and classification of design items as shown in section 5.3. This provides a means of identifying digital design items in a way that makes it possible to establish their role in the design process. This prototype can satisfy the requirements that have been presented above and in section 2.3 as necessary for supporting the early building design process.

CoBL DT provides support for the requirements in four main capability areas namely Draw, Recognize, Apply Knowledge and Collect Solution. The designer is able to draw abstract and concrete design items using the draw capability which addresses the requirement for interface issues. The designer can classify items which addresses the design item recognition requirement. CoBL DT allows the use of knowledge in checking and maintaining the integrity of design requirements as well as providing the designer with access to expert knowledge for making more informed decisions which address the knowledge integration requirement. CoBL DT automatically collects solutions through relationships among individual design items in a hierarchical tree and the collection of solutions along with alternatives in solution paths. This addresses the collection of alternatives and design decomposition requirements.

8.2.5 Validation of Prototype

CoBL DT has been used in a design session similar to which was undertaken using traditional (manual) methods. The results show that CoBL DT:

- Reduces the drawing effort used by the designer
- Maintains an overview of the design space for the designer
- Collects the solutions and any alternates being explored by the designer
- Helps the designer to apply expert knowledge that would be difficult or tedious otherwise.

The contribution here is that using CoBL DT reduces the number of steps as compared to the manual process, provides a richer solution (readily available alternate solutions that are absent in the manual method) and a clear presentation of the main solution which can be easily interpreted and transferred to the next stages of the building design process.

8.3 Future Work

Additional work is needed to achieve full support for the early design stages. This is presented in the following two sections. The first section concentrates on research issues whereas the second one presents avenues of development that will improve the capability of the prototype CoBL DT.

8.3.1 Need for Additional Research

Additional issues have been identified that need further research so that better support for early building design can be provided.

- **Requirements Repository**. (See section 4.2). It is important for all design requirements to be available within the system because they can be used by the design items in reasoning about their roles. More research is needed to determine an appropriate way of entering these requirements. This is because there are different types of requirements. Some are available through a list of textual items (space names) but others are either graphical items (site drawings) or provided as rules (for example, the need for one space to be a certain distance from another specific space).
- **Multiple Levels of Abstraction**. (See section 4.2). Items used in early building design must be perceived differently at different times in the overall design process and also by different designers. For example architects like to see simple spaces (sometimes depicted in a circular fashion) at the very early beginning of the design session but more detailed spaces at the later stage. Engineers like to see structural items and loads as opposed to spaces and forms by architects. Further research is needed to determine how few design items (preferably using corporeal and incorporeal design items) can present these different views that will address the needs of the diverse designers involved in the early design process.
- **Element Interaction**. (See section 4.2). Design items need to interact with each other and with the designer in real time. For example the interaction needed between two items in order to coordinate their proximity so that certain rules regarding their relationship are maintained throughout the design process. Such interactions help the designer in keeping design decisions consistent as well as accessible. Further research here is needed to analyze the different decisions the designer makes and how these decisions are carried out in relation to the creation and manipulation of design items. This will provide a range of characteristics to be acknowledged between any two design items. Based on this characteristic a check can be made to make sure that compatibility and conformance issues are satisfied or maintained.
- **Automatic Feedback**. (See section 4.2). Feedback on design decisions occur in different ways for different design items. Some are just information or reminders

while others are errors that may need to be addressed. More research is needed to determine the type of errors that can be generated by design items. Then, a means of addressing these errors should be described to limit any distraction.

- **Design Liability**. (See section 4.2). The designer must be in full control of the decisions made at the early design stage. There is a need to keep the tool flexible and to be able to log activities and commands so that they can be reviewed.
- **Application of on-the-fly Requirements**. The design process is dynamic and therefore rules are created, removed or changed frequently. There is a need to assess the different types of knowledge that are required by the different types of designers. This will help in determining a way to introduce rules-of-thumb in an interactive way appropriate for each type of designer. For example while designing a bathroom a designer might decide to keep an item or a part of the bathroom space out of the line of sight from another part of the design, for privacy issues. It should be possible to create, remove or adjust such rules during the design process.

8.3.2 Improving CoBL DT

CoBL DT is a prototype that supports all the requirements established in this thesis. However, it cannot be used currently in an actual design session because current capabilities are not sufficiently robust and user friendly. As a prototype, the main focus is on demonstrating the feasibility of a solution to some of the issues being discussed in the thesis. To enable CoBL DT in actual practice, bearing in mind that not all specifications have been addressed fully, the following must be accomplished.

- **Saving Design Data.** CoBL DT is part of a set of tools for the design process. Data created with it should be transferable to other existing tools later in the process. Existing standards could be used for saving design data such as the use of object-oriented databases and Industry Foundation Classes (IFC) [IAI International Council, 2004; Froese, 1996; ISO TC184/SC4 committee, 2004].

- **Merging External Solutions.** CoBL DT allows the designer to merge work from different solution paths however it is conceivable that a merge from different design sessions saved in different files would be desirable. A way of importing solution paths from other design sessions and selectively merging items should be implemented.
- **Complexity in Corporeal Forms.** CoBL DT is limited to rectilinear corporeal design items. More complex shapes often used in the early design process should be supported as well.

Chapter 9

References

Akin O, *Psychology of Architectural Design* (Pion Ltd., London UK, 1986).

Akin O, **Aygen Z**, **Chang T**, **Chien S**, **Choi B**, **Donia M**, **Fenves S**, **Flemming U**, **Garrett J**, **Gomez N**, **Kiliöte H**, **Rivard H**, **Sen R**, **Snyder J**, **Tsai W**, **Woodbury R** and **Zhang Y** (1998) A Software Environment to Support Early Phases in Building Design. In: *International Journal of Design Computing, vol. 1*.

Atman C J, **Chimka J R**, **Bursic K M** and **Nachtmann H L** (1999) A Comparison of Freshman and Senior Engineering Processes. In: *Design Studies, vol. 20*(2): pp. 131–152.

auto*des*sys (2003) form*Z: The 3D form synthesizer. Tech. rep., auto.des.sys, Inc. URL `http://www.formz.com/web_site_2000/frames_pages/products.htm`.

Baya V and **Leifer L J**, Understanding Information Management in Conceptual Design. In: N Cross, H Christiaans and K Dorst, eds., *Analysing Design Activity*, chap. 7 (John Wiley & Sons, 1996), 151–168.

Bédard C (1988) Computers in building design. In: *Canadian Journal of Civil Engineering, vol. 15*: pp. 674–675.

Bédard C and **Gowri K** (1990) Automating building design process with KBES. In: *ASCE Journal of Computing in Civil Engineering, vol. 4*(2): pp. 69–83.

Bédard C, **Gowri K** and **Fazio P** (1991) Integration in the building industry. In: *Journal of Computing in Civil Engineering, vol. 5*(5): pp. 333–335.

Bentley (2003) MicroStation Triforma. Tech. rep., Bentley Systems Incorporated. URL `http://www2.bentley.com/products`.

Bermudez J and **King K**, Media Interaction & Design Process: Establishing a knowledge base. In: T Seebohm and S van Wyk, eds., *ACADIA '98. Digital Design Studios: Do computers make a difference?*, ACADIA (Quebec City, Canada, 1998), 6–25.

Bhavnani S K, **Garrett, Jr J H** and **Shaw D S**, Leading Indicators of CAD Experience. In: U Flemming and S van Wyk, eds., *CAAD Futures '93. Proceedings of the Fifth International Conference on Computer-Aided Architectural Design Futures*, CAAD Futures (Pittsburgh, USA, 1993), 313–334.

Björk B C (1989) Basic Structure of a Proposed Building Product Model. In: *Computer-aided design, vol. 21*(2): pp. 71–78.

Björk B C (1999) Information technology in construction: Domain definition and research issues. In: *International Journal of Computer Integrated Design and Construction, vol. 1*(1): pp. 3–16.

Brucker B (2002) Building Composer: criteria-based facility design. Tech. rep., US Army Corps of Engineers. Construction Engineering Research Laboratory. URL `http://www.cecer.army.mil/EARUpdate`.

Brueck D and **Tanner S**, *Python 2.1 Bible* (Hungry Minds, Inc., 2001).

Casakin H and **Goldschmidt G** (1999) Expertise and Use of Visual Anology: Implications for design education. In: *Design Studies, vol. 20*(2): pp. 153–175.

Chien S, A Pilot Study of Using Generative Systems in Architectural Design Process. In: *CAADRIA 2000, Proceedings of the Fifth Conference on Computer Aided Architectural Design Research in Asia* (Singapore, 2000), 135–144.

Chien S, **Magd D**, **Snyder J** and **Tsa W J**, SG-CLIPS: A System to Support the Automatic Generation of Designs from Grammers. In: T Sasada, S Yamaguchi, M Morozumi, A Kaga and R Homma, eds., *CAADRIA '98. Proceedings of The Third Conference on Computer Aided Architectural Design Research in Asia*, CAADRIA (Osaka University, Osaka, Japan, 1998), 445–454.

Cross N, ed., *The Automated Architect* (Pion Ltd., London, UK, 1977).

Cross N (1998) Natural intelligence in design. In: *Design Studies, vol. 20*: pp. 25–39.

Cross N, **Christiaans H** and **Dorst K**, Introduction: The Delft Protocols Workshop. In: N Cross, H Christiaans and K Dorst, eds., *Analysing Design Activity* (John Wiley & Sons, 1996), 1–16. Introduction.

Donath D and **Regenbrecht H**, VRAD (Virtual Reality Aided Design) in the early phases of the architectural design process. In: M Tan and R Teh, eds., *The Global Design Studio. Proceedings of the Sixth International Conference on Computer-Aided Architectural Design Futures*, CAAD Futures (Singapore, 1995), 313–322. URL http://www.caadfutures.arch.tue.nl/proceedings_95.htm.

Donath D and **Regenbrecht H** (1999) Using Immersive Virtual Reality Systems for Spatial Design In Architecture. In: *Proceedings AVOCAAD'99, Brussels*: pp. 308–318. URL http://infar.architektur.uni-weimar.de/infar/deu/forschung/pu%blic.

Dörner D (1999) Approaching Design Thinking Research. In: *Design Studies, vol. 20*(5): pp. 407–415a.

Dorst K and **Dijkhuis J**, Comparing Paradigms for Describing Design Activity. In: N Cross, H Christiaans and K Dorst, eds., *Analysing Design Activity*, chap. 12 (John Wiley & Sons, 1996), 253–270.

Dorta T and **Lalande P**, The Impact of Virtual Reality on the Design Process. In: T Seebohm and S van Wyk, eds., *ACADIA '98. Digital Design Studios: Do computers make a difference?*, ACADIA (Quebec City, Canada, 1998), 138–163.

Dunn R (2001) wxPython. URL `http://www.wxpython.org`, accessed - January 2004.

Eastman C M, ed., *Spatial Synthesis in Computer Aided Design* (Halstead press, 1975).

Eastman C M (1992) A Data Model Analysis of Modularity and Extensibility in Building Databases. In: *Building and Environment, vol. 27*(2): pp. 135–148.

Eisentraut R (1999) Styles of problem solving and their influence on the design process. In: *Design Studies, vol. 20*(5): pp. 431–437.

Ekholm A, A Systemic Approach to Building Modelling - analysis of some object-oriented building product models. In: *CIB W78 Workshop: Computer Integrated Construction* (Esbo, Finland, 1994), 193–205.

Ekholm A and **Fridqvist S**, Object-Oriented CAAD: Design object structure and models for buildings, user organisation and site. In: *CIB W78 Workshop: Modeling of Buildings Through Their Life-Cycle* (Stanford University, California, USA, 1995), 193–205.

Ekholm A and **Fridqvist S**, Modeling of User Organisations, Buildings and Spaces for the Design Process. In: *CIB W78 Workshop: Construction on the Information Highway* (Bled, Slovenia, 1996), 193–205.

Emkin L Z, Misuse of Computers by Structural Engineers: A clear and present danger. In: E T Miresco, ed., *1st International Conference on New Information Technologies For Decision Making in Civil Engineering* (École de technolgie supérieure, Université du Québec, Montréal, Canada, 1998), 1–10. Keynote lecture.

Fazio P (1990) A case study in integrated building design. In: *Canadian Journal for Civil Engineering, vol. 17*: pp. 636–642.

Flemming U (1990) Knowledge Representation and acquisition in the LOOS System. In: *Building and Environment, vol. 25*(3): pp. 209–219.

Fricke G (1999) Successful approaches in dealing with different precise design problems. In: *Design Studies, vol. 20*(5): pp. 417–429.

Froese T, STEP and the Building Construction Core Model. In: *Computing in Civil Engineering, ASCE, Proceedings of the Third Congress* (Anaheim, California, June 17–19, 1996), 445–451.

Gardner B M, The Grid Sketcher: An AutoCAD based tool for conceptual design processes. In: T Seebohm and S van Wyk, eds., *ACADIA '98. Digital Design Studios: Do computers make a difference?*, ACADIA (Quebec City, Canada, 1998), 222–237.

Gero J S and **McNeil T** (1998) An Approach to the Analysis of Design Protocols. In: *Design Studies, vol. 19*(1): pp. 21–61.

Goel V, *Sketches of thought* (The MIT Press, Cambridge MA, 1995).

Goldschmidt G (1991) The Dialectics of Sketching. In: *Design Studies, vol. 4*: pp. 123–143.

Goldschmidt G (1994) On visual design thinking: The vis kids of architecture. In: *Design Studies, vol. 15*(2): pp. 158–174.

Graphisoft (2003) Graphisoft ArchiCAD. Tech. rep., Graphisoft R&D Rt. URL `http://www.graphisoft.com/`.

Gross M D (1996) The Electronic Cocktail Napkin - computer support for working with diagrams. In: *Design Studies, vol. 17*(1): pp. 53–69.

Gross M D and **Do E Y** (1996) The Electronic Cocktail Napkin Project. URL `http://wallstreet.colorado.edu/Napkin/`, accessed - December 1998.

Günther J and **Ehrlenspiel K** (1999) Comparing designers from practice and designers with systematic design education. In: *Design Studies, vol. 20*(5): pp. 439–451.

Guthrie P, ed., *The Architect's Portable Handbook* (McGraw-Hill, inc., 1995).

Haptic Technologies, Inc (1999) Technology for your fingertips^{TM}. URL http://www.haptech.com.

Hernan C and **Goldschmidt G** (1999) Expertise and the use of visual analogy: Implications for design education. In: *Design Studies, vol. 20*(2): pp. 153–175.

Heylighen A, **Neuckermans H** and **Bouwen J E** (1999) Walking a thin line — between passive knowledge and active Knowing of components and concepts in architectural design. In: *Design Studies, vol. 20*(2): pp. 211–235.

Hilf B and **Cimafranca D** (2003) The Camel and the Snake or "Cheat the Prophet", Open Source Development with Perl, Python and DB2. Tech. rep., IBM Corporation.

Holtzblatt K and **Jones S**, Contextual Inquiry: A participatory technique for system design. In: D Schuler and A Namioka, eds., *Participatory Design: Principles and practice*, chap. 9 (Lawrence Earlbaum, Hillsdale NJ, 1993), 180–193.

Howard H C, **Levitt R E**, **Paulson B C**, **Pohl J G** and **Tatum C B** (1989) Computer Integration: Reducing Fragmentation in AEC Industry. In: *Journal of Computing in Civil Engineering, vol. 3*(1): pp. 18–21.

IAI International Council (2004) International Alliance for Interoperability. Tech. rep., IAI International Council. URL http://www.iai-international.org/iai_international/.

Intergraph (2003) SmartSketch. Tech. rep., Adept Scientific Inc. URL http://www.adeptscience.co.uk/products/mathsim/smartsketch/ov%erview.html.

ISO TC184/SC4 committee (2004) Standard for the Exchange of Product Model Data. Tech. rep., ISO TC184/SC4 committee. URL http://cic.vtt.fi/links/step.html.

Khemlani L (2002a) 3D Modeling, Rendering and Animation with form*Z. In: *CADENCE, vol. 82*. URL http://www.cadenceweb.com/newsletter/aec/0902_2.html.

Khemlani L (2002b) Alternate CAD Solutions: Microsoft Visio. In: *CADENCE, vol. 67*. URL http://www.cadenceweb.com/newsletter/aec/0102_2.html.

Khemlani L (2002c) Interoperability in Microsoft Visio: The BLIS Project. In: *CADENCE, vol. 68*. URL http://www.cadenceweb.com/newsletter/aec/0202_1.html.

Khemlani L (2002d) MicroStation V8. In: *CADENCE*. URL http://www.cadenceweb.com/2002/0702/fr0702.html.

Khemlani L (2003a) Advanced 3D Design and Visualization with Autodesk VIZ 4. In: *CADENCE, vol. 81*. URL http://www.cadenceweb.com/newsletter/aec/0902_1.html.

Khemlani L (2003b) AEC Tech News: Building Information Modeling Gains Momentum. In: *CADENCE, vol. 90*. URL http://www.cadenceweb.com/newsletter/aec/0103_2.html.

Khemlani L (2003c) Architectural Studio 3. In: *CADENCE*. URL http://www.cadenceweb.com/2003/0503/pr0503_archstudio.html.

Khemlani L (2003d) AutoCAD 2004. In: *CADENCE*. URL http://www.cadenceweb.com/2003/0603/fr0603_autocad.html.

Khemlani L (2003e) AutoCAD LT 2004. In: *CADENCE*. URL http://www.cadenceweb.com/newsletter/aec/0603_2.html.

Khemlani L (2003f) form*Z 4.0. In: *CADENCE, vol. 101*. URL http://www.cadenceweb.com/newsletter/aec/0703_1.html.

Kurmann D (1996) Sculptor, a tool for intuitive spatial modeling. URL http://caad.arch.ethz.ch/~kurmann/sculptor, accessed last - April 27 2003.

Kurmann D, Sculptor – How To Design Space. In: T Sasada, S Yamaguchi, M Morozumi, A Kaga and R Homma, eds., *CAADRIA '98. Proceedings of The Third Conference on*

Computer Aided Architectural Design Research in Asia, CAADRIA (Osaka University, Osaka, Japan, 1998), 317–325.

Kurmann D and **Engeli M**, Modeling Virtual Space In Architecture. In: M Green, K Fairchild and M Zyda, eds., *VRST '96. (Virtual Reality Software And Technology)* (Hongkong, 1996), 77–82.

Larsson N and **Pope S** (1999) Presentation on C-2000 Program and Related Initiatives. Tech. rep., Buildings Group, CETC, Natural Resources Canada, Ottawa.

Laurel B, ed., *The Art of Human-Computer Interface Design* (Addison-Wesley, 1990).

Leclercq P, Interpretative Tool for Architectural Sketches. In: J S Gero and B Tversky, eds., *1st International Roundtable Conference on Visual and Spatial Reasoning in Design: computational and cognitive approaches* (1999), 193–205. URL `http://www.arch.usyd.edu.au/kcdc/books/VR99/Lecler.html`.

Lipson H and **Shpitalni M** (2000) Conceptual Design and Analysis by Sketching. In: *Artificial Intelligence for Engineering Design, Analysis and Manufacturing, vol. 14*(5): pp. 391–401.

Long B (2003) Dip into SketchUp for easy 3D. In: *Creativepro.com*. URL `http://www.creativepro.com/story/review/18589.html`.

Mariën H (2003) DESI-III: 2D CAD System. Tech. rep., Personal project. URL `http://users.telenet.be/desi-iii/index.html`.

Marx A, A proposal for alternative methods for teaching digital design. In: T Seebohm and S van Wyk, eds., *ACADIA '98. Digital Design Studios: Do computers make a difference?*, ACADIA (Quebec City, Canada, 1998), 58–73.

Mazijoglou M, **Scrivener S** and **Clark S**, Representing Design Workspace Activity. In: N Cross, H Christiaans and K Dorst, eds., *Analysing Design Activity*, chap. 18 (John Wiley & Sons, 1996), 389–415.

McGown A, **Green G** and **Rodgers P A** (1998) Visible Ideas: Information patterns of conceptual sketch activity. In: *Design Studies, vol. 19*(4): pp. 431–453.

Meniru K C and **Schmitz G**, Megastructure Housing: Getting ready for the 3rd millennium. In: O Ural, D Altinbilek and T Birgönül, eds., *XXIVth International Association for Housing Science*, vol. 3 (Ankara, Turkey, 1996), 959–970.

Mottle J (2003) Review of Autodesk VIZ 4. Tech. rep., CGarchitect.com: CG Portal for Industry Professionals. URL `http://go.cadwire.net/?951,2,1`.

Mustun A, **Pichler G**, **Dicanio G**, **Sweevelt J V**, **Osintsev E** and **Kobayashi Y** (2003) QCad: The easy to use CAD for Linux. Tech. rep., RobbinSoft.com. URL `http://www.qcad.org`.

Neufert E, *Architects' Data, second (international) English edition* (The Alden Press, Oxford, 1991).

Newman W M and **Lamming M G**, *Interactive System Design* (Addison-Wesley, 1995).

Novitski B J (1998) Special Report: An architectural awakening. In: *Computer Graphics World*: pp. 22–35.

Purcell A T and **Gero J S** (1998) Drawings and the design process. In: *Design Studies, vol. 19*(4): pp. 389–430.

Python Software Foundation (1991) Python. URL `http://www.python.org`, accessed - January 2004.

Quatrani T, *Visual Modeling with Rational Rose and UML* (Addison-Wesley, 1998).

Raskin J, *The Humane Interface, New Directions for Designing Interactive Systems* (Addison-Wesley, 2000).

Richter C, *Designing Flexible Object-Oriented Systems with UML* (Macmillan Technical Publishing, 2001).

Rivard H (1997) *A Building Design Representation for Conceptual Design and Case-Based Reasoning*. Ph.d. dessertation, Dept. of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburg, PA.

Rivard H (1999) A survey on the impact of Information Technology on the Canadian Architecture, Engineering and Construction Industry. In: *Electronic Journal of Information Technology in Construction, vol. 5*: pp. 37–56. URL `http://itcon.org/2000/3/`.

Rivard H, **Bédard C**, **Fazio P** and **Ha K H** (1995) Functional Analysis of the Preliminary Building Envelope Design Process. In: *Building and Environment, vol. 30*(3): pp. 391–401.

Rivard H and **Fenves S J** (2000) A Representation for Conceptual Design of Buildings. In: *Journal of Computing in Civil Engineering, vol. 14*(3): pp. 151–159.

Rivard H, **Fenves S J** and **Gomez N**, Case-Based Reasoning for Conceptual Design. In: E T Miresco, ed., *1st International Conference on New Information Technologies For Decision Making in Civil Engineering* (École de technolgie supérieure, Université du Québec, Montréal, Canada, 1998), 355–366.

Rodgers P A, **Green G** and **McGown A** (2000) Using Concept Sketches to Track Design Progress. In: *Design Studies, vol. 21*(5): pp. 451–464.

Rosenman M A and **Gero J S** (1995) Modelling Multiple Views of Design Objects in a Collaborative CAD Environment. In: *Computer-aided design, vol. 28*(3): pp. 193–205.

Rush R D, ed., *The Building Systems Integration Handbook* (The American Institute of Architects, 1986).

Schmitt G, **Engeli M**, **Kurmann D**, **Faltings B** and **Monnier S** (1996) Multi-agent Interaction In A Complex Virtual Design Environment. In: *AI Communications - European Journal on AI, vol. 9*(2): pp. 74–78.

Stein B and **Reynolds J S**, *Designing for Heating and Cooling*, chap. 5 (John Wiley & Sons, Inc., 1992), 8 edn., 183–280. Mechanical and Electrical Equipment for Buildings.

Sutherland I E, *Sketchpad, a man-made graphical communication system* (Garland Publishing, Inc, New York and London, 1980).

Verstijnen I M, **Hennessey J M**, **van Leeuwen C** and **Hamel R** (1998) Sketching and creative discovery. In: *Design Studies, vol. 19*(4): pp. 519–546.

Weinzapfel G and **Handel S**, *Image: Computer Assistant for Architectural Design*. Spatial Synthesis in Computer Aided Design (Halstead press, 1975).

Zeleznik R C, **Herndon K P** and **Hughes J F**, Sketch: An interface for sketching 3D scenes. In: H Rushmeier, ed., *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH (ACM Press, New Orleans, Louisiana, 1996), 163–170. URL `http://www.cs.brown.edu/people/bcz/sketch/`.

Appendix A

Study Report: C03

A.1 C03: Interview

Table A.1: C03 Transcription: Interview

LEGEND

[ ] – additions by researcher or reference to items in other tables for this designer

Time	Questions	Designer's Answers
Trained as an architect and an engineer with up to 10 years in professional practice.		
Start time = 2:04.00		
2:04.38	What medium do you feel comfortable designing in. Is it manual...?	Whenever I go to the computer... I never touch the computer myself, I have technicians or junior architects... [do the work or me] when i am going from conceptual to preliminary and then on to the working drawings
2:05.00	OK. So most of my questions will be on manual drawings. When you get	First thing... I don't work a lot on the project... [i.e.] drawings. The project is in my mind. Since there is a lot

continued on next page

continued from previous page

Time	Questions	Designer's Answers
	a project, what are the first things you begin to think about towards [doing the design].	of work that [has been] done that I have to think about. I don't know if [I do this for] a day or 3 days or two weeks. The project will be in my mind all the time. Everything that goes around will feed into the project... I love the light, I love the shadows, I love the colors... anything. If you do an institutional project, you get all the space requirement from the client... I have never worked on an institutional project where the client was very involved. But on residential projects... then you get the client very involved... the client [provides the source of the information, the ideas and direction for the project]. If there is no client then you have to forge your own understanding of the problem.
2:06.46		I try to get all the information... all the [codes] and by-laws [C03001RS]... you have to be very strict with municipal by-laws. I will do a brief check with the national building code. Although when you have done a lot of projects you know a lot of it by heart. You have to be aware of then although they are very simple. These are all the technical aspects and the budget aspects, all the needs in terms of spaces and wishes, anything I can get. On top of that I try to find a direction to the project... try to have an idea, an intent... something that gets it together. Some projects are very hard to do that, they are very... not boring but closed so that you always have room to get something special... get... with colors so that there is room for something new... optimizing partitions... I try to get into that.

continued on next page

continued from previous page

Time	Questions	Designer's Answers
		[These are often] most business and institutional projects, [there are perspectives, a few isometric but a lot of sketches. A few drawings with people on it for scale. A bit later... sections. OK but not very early also.
2:14.16	So let me get this right. Once you begin you start first with getting the requirements, then you start the work with shapes and forms. At this time are you viewing the whole project as one shape or are you looking at individual spaces.	You have a different approach... I start with the overall shape.
2:14.45	OK now you are working with the overall shape how do you begin to do like the perspectives.	Cause you have got ideas about how you go from one room to another so you make a perspective of what you see from the outside and you play with the different shapes... also taking pictures of the site you work from the environment... how you can fit what you doing with the [environment].
2:15.20	One more question. Do you have any form of boundaries, things that give you scale. I know you use people but in terms of the 2D world, do you use grids?	No. When you are used to working in a scale like 1:50 or 1:100, you know the scale
2:15.45	If you were... I know you have not used computers to design but i am sure you have heard a lot about what they can do and you have seen some simulations. If you were to say something that is very crucial or that you feel is missing from computers, what	Well, when you have got the mouse and you know that it is not like a pen... It would have to be like a pen [C03006RS]. And you draw... it has to be on the screen... directly.

continued on next page

continued from previous page

Time	Questions	Designer's Answers
	would it be?	
2:16.21		What you draw appears on the screen [C03007RS].
2:16.43		Maybe there is another way to extract stuff... I have got this shape and I draw this line and then I continue as I draw with tracing paper [I keep some parts and maybe I have other ideas but they] fade away but you still have them very faint [C03008RS]. I think it should be on the screen. I think that if you can say that this is a wall and this is a wall, then it gives you a perspective and then I have got these two walls, I continue to play with this perspective, I got this window here and it shows there... I just have never thought of that before. If you say this is 1:50, it should be easy to get other information about the environment, then you can get the perspective of the building and the street.
2:18.17		At the beginning... when you draw a line, it could be a wall, it could be nothing. It could be something I will use a bit later. In the computer it should be possible to change the definition of things [C03009RS].
2:18.43		Another thing in computers is to check solar lighting... it is good for checking the by-laws so you don't even have to check that... if you are too close to the boundary, it will check the number of windows and openings in the codes [C03010RS].

Appendix B

Use Case Models

This appendix presents the flow of events used in documenting each use case used for creating the prototype. The flow of events for a use case describe the events needed to accomplish the required behavior of the use case. It is written in terms of what the system should do and not how it should do it.

B.1 Flow of Events for the *Draw* Use Case

This use case makes it possible for the designer to draw or create design items in the system in a process that is user-friendly in the early design process.

B.1.1 Preconditions

This use case requires that initial (default) settings be provided for the following items: color, pen size, building level, sheet number.

B.1.2 Main Flow

This use case presents the designer with a choice of creating CORPOREAL or INCORPOREAL item with the corporeal item as the default (A-1). As the designer draws, the system records the points that describe the corporeal. The LIST CLASSIFICATIONS subflow of *Classify* and the INSERT ITEM subflow of *SolHierarchy* use cases are called. The use case provides the following functions, ENTITY HEIGHT, LINE COLOR, LINE WIDTH, VIEW BUILDING LEVEL, NEW BUILDING LEVEL, NEW SHEET, CHANGE SHEET NAME.

If the function selected is ENTITY HEIGHT, the S-1: *Entity Height* subflow is performed.

If the function selected is LINE COLOR, the S-2: *Line Color* subflow is performed.

If the function selected is LINE WIDTH, the S-3: *Line Width* subflow is performed.

If the function selected is VIEW BUILDING LEVEL, the S-4: *View Building Level* subflow is performed.

If the function selected is NEW BUILDING LEVEL, the S-5: *New Building Level* subflow is performed.

If the function selected is NEW SHEET, the S-6: *New Sheet* subflow is performed.

If the function selected is CHANGE SHEET NAME, the S-7: *Change Sheet Name* subflow is performed.

B.1.3 Subflows

S-1: Entity Height

An entry field is provided for the designer to enter the height of all spaces (corporeals) to be created. This setting remains effective until changed.

S-2: Line Color

A collection of toggle menus make it possible for the designer to choose between four colors to render the items being created.

S-3: Line Width

A selection box allows the designer to select from a number of preset line widths used in drawing the items being created.

S-4: View Building Level

The designer is allowed to choose a sheet in any building level including the current one to view, while work is being accomplished in a different one (as a reference).

S-5: New Building Level

Creates a new building level and makes it the current level. It causes the NEW SHEET sub-function to execute automatically so that each new building level is created with a sheet. It collects the created building levels and allows the designer to choose to view any of the existing levels.

S-6: New Sheet

Creates an electronic sheet and makes it the current sheet. Makes it possible for the designer to organize data within building levels. Sheets are consecutively numbered when created and only one sheet can be worked on at any time. Only the sheets created in a particular level are visible at any time.

S-7: Change Sheet Name

Changes the name of the sheet from the automatically generated one to that assigned by the designer (A-2).

B.1.4 Alternative Flows

A-1: If the designer chooses to draw an incorporeal item, the system allows the designer to draw an open shape and presents the coordinates of the first and last points to the designer.

A-2: If the name already exists, the assignment will fail.

B.2 Flow of Events for the *Classify* Use Case

This use case makes it possible for the system to recognize entities in the design.

B.2.1 Preconditions

This use case requires that initial (default) settings be provided for the following items: height of building level, height of doors, height of sill for windows (all windows are placed from height of the door to the sill height), North direction

B.2.2 Main Flow

This use case begins when an item is created in the *DRAW* use case. It prepares a set of possible classifications for the drawn item. When the designer selects a classification from the list, the selected list item compares it spatial needs to that existing in the drawn corporeal item (A-1), (A-2). If this comparison is satisfactory (A-3), the classification is accepted and the drawn item is updated to the new classification. The following sub-functions are provided by this use case, PREPARE CLASSIFICATIONS, APPLY CLASSIFICATION, ORIENTATION, LABEL BOUNDARY, BOUNDARY MATERIAL, BOUNDARY STATUS, REMOVE CONNECTION, SHOW CONNECTION, ADJUST CONNECTION

If the function selected is PREPARE CLASSIFICATIONS, the S-1: *Prepare Classifications* subflow is performed.

If the function selected is APPLY CLASSIFICATION, the S-2: *Apply Classification* subflow is performed.

If the function selected is LABEL BOUNDARY, the S-3: *Label Boundary* subflow is performed.

If the function selected is BOUNDARY MATERIAL, the S-4: *Boundary Material* subflow is performed.

If the function selected is BOUNDARY STATUS, the S-5: *Boundary Status* subflow is performed.

If the function selected is REMOVE CONNECTION, the S-6: *Remove Connection* subflow is performed.

If the function selected is SHOW CONNECTION, the S-7: *Show Connection* subflow is performed.

If the function selected is ADJUST CONNECTION, the S-8: *Adjust Connection* subflow is performed.

B.2.3 Subflows

S-1: Prepare Classifications

The parent of the drawn item is used to prepare its possible classifications. The site is the default parent so all items that are created with no parent are assumed to belong to the site. Every corporeal item has a list of other corporeal items it can contain so this function retrieves and formats this list for the designer to make a choice from. Other general list may be added depending on the parent for example, connection items will not be provided if the parent is a green space.

S-2: Apply Classifications

The identity of a corporeal item is updated to that of a known building component providing it with the characteristics and behavior of the building component. The LABEL BOUNDARY sub-function is executed.

S-3: Label Boundary

The boundary for all newly classified items are identified and labelled in the design for easy reference. The BOUNDARY STATUS sub-function is executed.

S-4: Boundary Material

Default material for all boundaries are provided to those with the setting of "close". It allows the designer to choose a limited set of materials to apply to boundaries.

S-5: Boundary Status

When items are first classified, the default boundary status is "close" however the designer can later choose to specify the boundary as "open". The BOUNDARY MATERIAL sub-function is executed.

S-6: Remove Connection

The connector is removed by deleting it as well as its unique id from the boundary where it existed. The same call is made to any connected spaces.

S-7: Show Connection

The connector is located and flashed in the design. The helps the designer in configuring connectors as there can be more than one window, for example, in a boundary.

S-8: Adjust Connection

When a connector is connecting two spaces, the designer can use this function to adjust the size of the two spaces to match each other. This is a way to quickly organize or "clean up" spaces without the overhead of precision in drawing necessary in the early design process.

B.2.4 Alternative Flows

A-1: If the classification selected is a connector (example a window) the system calculates the location of the corporeal item to determines the side and location in the parent corporeal item to install the connector. This installation is indicated with an icon that designates the type of connection. The boundary to which it is installed is assigned the connectors unique identification to provide access to it by the designer. The corporeal item is deleted from the design.

A-2: If the item created is an Incorporeal item, no spatial comparison is carried out. The classification is accepted.

A-3: If the space is not satisfactory, the system provide a feedback to the designer of the minimum size of space needed. The classification is rejected and the designer may reclassify the drawn item.

B.3 Flow of Events for the *Capture Entity* Use Case

This use case makes it possible to collect and present each entity along with its relationship to all other entity objects in the design.

B.3.1 Preconditions

This use case requires that entities have unique identifications and be able to store the parent's id as well as a list of all the children's id as well.

B.3.2 Main Flow

This use case begins when an entity item is created in the *Draw* use case. It collects the unique identity of the item and adds it to the hierarchy tree (A-1). It provides the following functions for managing the hierarchical tree: FIND ITEM, LABEL ITEM, DELETE ITEM, FIND CHILDREN, FIND FAMILY, MAKE CHILD, CHANGE PARENT, BUILD TREE, INSERT ITEM

If FIND ITEM is called, the S-1: *Find Item* subflow is performed.

If LABEL ITEM is called, the S-2: *Label Item* subflow is performed.

If DELETE ITEM is called, the S-3: *Delete Item* subflow is performed.

If FIND PARENT is called, the S-4: *Find Parent* subflow is performed.

If FIND CHILDREN is called, the S-5: *Find Children* subflow is performed.

If MAKE CHILD is called, the S-6: *Make Child* subflow is performed.

If CHANGE PARENT is called, the S-7: *Change Parent* subflow is performed.

If BUILD TREE is called, the S-8: *Build Tree* subflow is performed.

If INSERT ITEM is called, the S-9: *Insert Item* subflow is performed.

B.3.3 Subflows

S-1: Find Item

The unique id of the item is collected and used to find the item in the design. The use case ends.

S-2: Label Item

The unique id and the name of the item is collected. The id is used to substitute the old id in the hierarchical tree and is used by this use case to locate the item in the tree. The name replaces the old name in the tree as a label for the new item. The use case ends.

S-3: Delete Item

The unique id of the item is collected and used to find the item in the hierarchical tree. It is then removed from the tree (A-2 and A-3) and removed from the design. The REMOVE subflow of the Solution Path is called. The use case terminates.

S-4: Find Parent

The unique id of the item is collected and found in the tree where its parent's unique id is collected and returned. The use case ends.

S-5: Find Children

The unique id of the item is collected and identified in the tree. All children's ids that have it as a parent are also colleted and returned. The use case ends.

S-6: Make Child

The unique id of the item is collected along with the unique id of the parent. The parent is located in the tree and child is added as one of its children. The use case continues.

S-7: Change Parent

The unique ids of the new parent and child are collected. The child is located in the tree, removed as a child of the current parent and relocated to the new parent's tree.

S-8: Build Tree

The unique ids of all the items to be put in the tree is required. Using the ids, a list of all parents for the first level is collected from the design. Using each of these items, the use case performs the FIND CHILDREN subflow and builds the current items branch of the tree. Once a branch is complete, each child in the branch is querried for any children and

the process for finding and building the childs branch of the tree is performed until there are no more children for that branch. The second parent in the first level is processed the same way until the last parent is complete. S-9: Insert Item
The unique id of the item is collected and used to add the item in the hierarchy tree (A-4).

B.3.4 Alternative Flows

A-1: If the item is created inside an existing corporeal item, the MAKE CHILD subflow of this use case is performed. It adds the unique id of the new item as a child of the parent then it uses the unique id of the parent to add the new item to the hierarchical tree.
A-2: If the item has a parent, the FIND PARENT subflow is performed. The item is removed as a child of the parent then the child is removed from the tree. The use case terminates.
A-3: If the item has children, the FIND CHILDREN subflow of this use case is performed. All the children and their children if any, are removed from the tree along with the item. The use case terminates. A-4: If there is a parent, the identification is collected and the item is added as a child to the parent.

B.4 Flow of Events for the *Solution Path* Use Case

This use case makes it possible for the sequence of entity objects created during a design session to be saved in such a way that each solution investigated by the designer can be recalled at a later time. The path is represented as the sequence of objects that make up any design solution.

B.4.1 Preconditions

This use case requires that entity objects have unique identifications and also that they have a means of being designated as belonging to a specific solution path.

B.4.2 Main Flow

This use case is started when the designer classifies an entity object. The object's unique id is collected and added to a list that represents the path for the current design solution. The current solution path, which is a number, is assigned to the object. It provides the following commands for managing the solution path: NEW PATH, CHANGE PATH, NO BRANCH POINT, REPLICATE PATH, REMOVE

If NEW PATH is called, the S-1: *New Path* subflow is performed.

If CHANGE PATH is called, the S-2: *Change Path* subflow is performed.

If NO BRANCH POINT is called, the S-3: *No Branch Point* subflow is performed.

If REPLICATE PATH is called, the S-4: *Replicate Path* subflow is performed.

If REMOVE is called, the S-5: *Remove* subflow is performed.

B.4.3 Subflows

S-1: New Path

The system prepares to collect the new set of objects for the new path. All members of any previous path is removed. The number for labelling the new solution path is also incremented from its previous value and is used to create a label for the new solution path.

S-2: Change Path

The NO BRANCH POINT subflow of this use case is called to make sure it is possible to change the solution path (A-1). This subflow is described with reference to Figures refsola, refsolb, refsolc. Assuming the current solution path is shown in Figure refsola, and the designer requests for a change in path starting from Step [3a] in Figure refsolb, the *system first checks that the point for the new solution path is not a no-branch-point , then all entity items from that point to the begining of the solution path is replicated as shown in Figure refsolc. The original path, illustrated by Step [1a] to Step [4a], is saved with the current solution path label. The NEW PATH subflow of this function is performed. The replicated entity items, Step [1c] to Step [3c], are assigned and designated the new path label. The new path is then ready to recieve new items such as the entity item illustrated in Step [4c]. The use case continues.*

S-3: No Branch Point

The system checks and returns true if it is possible to create a new solution path from the current branching point (A-2). The system takes note of points in every solution path that cannot have a new path created from it. With reference to Figure refsolc, the following are points that cannot be branched from to create new solution paths. The first and last points in the path (Step [1a] and Step [4a]) and all removed entity objects in the path. The use case continues.

S-4: Remove

If this function is called, it collects the point in the tree for the item to be removed and saves it for use by the NO BRANCH POINT subflow in this use case. The object's position in the solution path is maintained. A visual mark telling the designer that the entity item occupying the position has been removed from the path, is provided. The use case terminates.

B.4.4 Alternative Flows

A-1: If it is not possible to change solution path, the use case stops and a message explains that a change in path cannot be started at the selected point.

A-2: If it is not possible, the point is noted for next time and a false flag is returned. The use case continues.