ABSTRACTION IN CONCEPTUAL MODEL DESIGN

By

Diana Kao

A Thesis

Submitted to the School of Graduate Studies

in Partial Fulfilment of the Requirements

for the Degree

Doctor of Philosophy

McMaster University

ABSTRACTION IN CONCEPTUAL MODEL DESIGN

DOCTOR OF PHILOSOPHY (1992) McMASTER UNIVERSITY

(Management Science / Systems) Hamilton, Ontario

TITLE: Abstraction in Conceptual Model Design

AUTHOR: Diana Kao, LL.B. (National Chen-Chi University, Taiwan)
M.B.A. (McMaster University)

SUPERVISOR: Dr. Norman P. Archer

NUMBER OF PAGES: x, 204

Abstract

The provision of software support for conceptual model design processes has been an important issue in Model Management Systems research. This study examines the issue by empirically studying design behaviour and the techniques used during design, with a focus on the use of abstraction in the design process.

A framework that explains the relationships between the use of abstraction and the output of the design process is proposed. We classify abstraction into vertical, horizontal and general abstraction techniques. We then propose that three dimensions of a design: the completeness of the design, higher level concepts in design, and the organization of the design, are affected by effective use of these abstractions. We also propose aids to support these three types of abstraction, and develop measures to evaluate their effectiveness.

A software prototype was developed to illustrate the implementation of the proposed abstraction aids. A pilot study was also conducted to test the effectiveness of these aids using different versions of the prototype.

Three hypotheses that test the effectiveness of the proposed abstraction aids were tested in an experimental study with treatments that included three design environments and two training methods. The three design environments were two which were supported by pencil and paper design, and one which was supported by a

software prototype. The two training methods used were based on our proposed abstraction aids. The results of the experiment indicate some significant differences in the performance of the participants in different treatment groups.

Acknowledgements

I wish to acknowledge and express my deepest gratitude to my supervisor, Dr. Norm Archer, for his valuable guidance and advice, as well as his patience and understanding throughout this study.

I would also like to acknowledge the helpful advice of the members of my supervisory committee, Dr. Ian Begg and Dr. Yufei Yuan.

Most of all, I would like to thank my husband, Wayne, for his love and encouragement. This dissertation would not exist without his support. During my years of study, he assumed the roles of both father and mother to a young family. Last, but not least, I want to thank my two children, Molly and Christine, for their love and understanding when I kept extra long working hours.

List of Tables Page

List of Figures **Page**

Chapter One

INTRODUCTION

The provision of an interactive problem solving environment that supports the creative and intuitive aspects of decision making has been an important issue in Decision Support Systems (DSS) research for some time [Alter, 1980; Sprague & Carlson, 1982; Weber, 1986; Turban, 1988]. However, only very limited advances have been made in this field [Elam & Mead, 1987, 1990]. A similar phenomenon has been observed in Model Management Systems (MMS) research, which is a subset of DSS research that concentrates on the creation, implementation and use of models in DSS environment. Although model management systems should have as a major objective the provision of support for model formulation, little research has been done to develop effective techniques for problem conceptualization and creative model design [Hwang, 1985; Pracht & Courtney, 1988; Pracht, 1990].

The lack of support for the problem conceptualization process, also known as the **upstream design** process, may be attributed to the intangibility and uniqueness of such processes. The process is intangible because it is often performed in the model designer's mind in an iterative fashion before a tangible form of the design is expressed. This makes understanding and providing support for these processes extremely difficult. The characteristic of uniqueness results

from the fact that design is a cognitive process, which is highly dependent on individual differences. The process followed in creating a design is unique to the individual designer because of personality factors, the designer's previous design experience, and problem domain knowledge. Hence, the intangibility and uniqueness of design processes have been major obstructions in the study of such processes and thus have hindered the development of more effective general design support tools.

The design process can be made more tangible by building a conceptual model of the application problem. We therefore believe that, by providing appropriate support for constructing conceptual models we are, in essence, supporting the problem solving process and/or the upstream model design process. The arguments are as follows. First, conceptual models of a problem, that often use entities and relationships to portray the objects of the problem and the relations among these objects, may be used as qualitative aids to analyze a problem by manipulating the structure and the content of related conceptual models, thus examining the problem at the desired level of abstraction. Second, although analytical models are often used for solving a variety of problems in a decision support system-supported environment, overlaying every analytical model there is a conceptual model that maps the problem situation into that model. Therefore, the model designer, either as an intermediary staff member or as the ultimate decision maker, will benefit from being able to conceptualize his/her views of the problem

more effectively and efficiently before the form of the analytical model takes form, by using a conceptual model of the problem.

However, the conceptual model design process is known to be influenced by both the designer's cognitive process [Simon, 1978; Best, 1986] and the problem solving techniques chosen by the designer [Newell & Simon, 1972]. This influence is often reflected in the procedure that a designer adopts to simplify the problem [Jaques, 1978; Guindon et al., 1987a]. We have found, through a thorough review of relevant studies in design, that although abstraction was suggested as the most common and powerful tool to support the problem simplification process [Ossher, 1987], little work was devoted to understanding and measuring its role in conceptual model design. Also missing from the existing literature is a framework that explains the relationship between the use of abstraction and its effects on the designs which are created. This has become the major motivation behind this study.

Our study begins with a literature review that includes studies in four areas. First, studies in MMS are reviewed. Ideally, in a DSS-supported environment, the conceptual model of a problem is designed through an MMS. Therefore, an MMS provides an ideal environment for implementing techniques that support conceptual model design. Our review is focused on the capabilities of current MMSs in supporting conceptual modelling. Second, we review different types of models and the upstream model design process. The intention is to examine the nature of

conceptual modelling further and to identify its strengths in supporting creative problem solving and model design. Design has been considered as a cognitive process that is influenced by individual characteristics. Therefore, some of the relevant studies of the impact of individual characteristics on model design are summarized. Third, techniques from published studies that support the conceptual model design process are reviewed and synthesized. The reviews in these three areas help us to establish the crucial link between the design process and the use of abstraction. Last, we review the concept of abstraction, its effects on model design, and techniques that support the use of abstraction.

The literature review forms the basis for our empirical studies of the model design process. The focus of our empirical studies is on how individuals apply the technique of "abstraction" and how to support the use of this technique. This will help us to identify relationships between the use of abstraction and the design output, to be used in the development of a theoretical framework that explains this relationship.

The trend of current DSS and MMS research is toward developing systems that enhance creativity during problem solving and decision making [Elam & Mead, 1990]. Therefore, it is of great importance to be able to implement some of the abstraction supporting techniques in a software supported design environment. We have devoted a great deal of effort both in studying the ways to implement these techniques, and in measuring their effectiveness.

The goals of this study are (1) to develop a better understanding of upstream design processes, (2) to study how abstraction is used in the design process, (3) to develop a theoretical framework to explain the relationships of abstraction to conceptual model design, (4) to develop quantitative measurements that reflect the use of abstraction in design, (5) to develop a software prototype to demonstrate the implementation of techniques supporting the use of abstraction in model design, and (6) to conduct an experimental study which evaluates the effectiveness of the proposed abstraction support in certain design environments.

The thesis is organized as follows. Chapter 2 reviews the relevant studies, leading to the research questions and goals. Research goals provide a more detailed description of how we will approach the problems, thus serving as a bridge linking the literature review and our research in the chapters thereafter. The pre-prototype empirical studies of design behaviour are presented in Chapter 3. The studies are oriented toward observing and analyzing model designers' behaviour in the modelling process, with an emphasis on the use of abstraction. We adopt the technique of protocol analysis by tape-recording the "thinking aloud" design process. Based on the findings of our empirical studies and the related literature review, we propose a theoretical framework that illustrates the relationship between the use of abstraction and design. Also incorporated in the framework are our recommended abstraction aids and a measurement of their use.

A software prototype was developed to partially implement the proposed abstraction aids as well as to facilitate the study of the effectiveness of these aids in a computer-supported design environment. The prototype development and testing procedures are described in Chapter 4. The effectiveness of the proposed abstraction aids are tested in both the paper and pencil design environment and the prototype-supported environment. The experimental design and the test results are presented in Chapter 5. Chapter 6 includes a final summary of findings and conclusions, and suggests directions for future research.

Chapter Two

REVIEW OF RELATED STUDIES

2.1 Model Management Systems (MMS)

2.1.1 Definition and Functions of MMS

The term "model management system" was first defined by Will [1975], as a software system that is capable of establishing and maintaining model banks that contain a variety of operational models. Elam and Konsynski [1987] stated that an MMS is a software system that can a) identify the tasks required to build and use models in a decision support environment, and b) provide a high level of software support for performing these tasks. Applegate et al. [1986] defined a model management system as a software system that provides for the creation, storage, manipulation and access of models. In general, the key concept of MMSs calls for the capture of both the problem domain-related knowledge and model structure-related knowledge, as well as the storage of this knowledge in a processible form in order to support more flexible decision making [Henderson, 1987].

Elam and Konsynski [1987] organized the functions of MMSs into three groups in terms of model formulation, analysis and results interpretation. By their definition, model formulation includes formulating new decision models, exploring

ideas and analyzing issues, as well as choosing existing models from a model base. These activities, except those of choosing from existing models, are generally recognized as upstream design activities, which will be discussed in more detail in section 2.2. Their [Elam & Konsynski, 1987] classification provides a conceptual base in terms of what functions a model management system should have in order to enhance the analytical and problem solving ability of a decision support system. In terms of physical implementation, two major functional areas were proposed by Applegate et al. [1986]: model storage functions, and model manipulation functions. Model storage functions include model representation, physical model storage and logical model storage. Model manipulation functions include model instantiation, model selection and model synthesis. These functional requirements have become the foundation for proposing MMS frameworks.

Many of the MMS functions mentioned above can be supported by using knowledge representation concepts and techniques developed in artificial intelligence (AI). The logic-based approach, semantic networks and frame-based approach are three techniques that have been adopted to support MMS prototypes. It has been suggested that a combination of a semantic inheritance network and frame representation would provide the best knowledge representation for an MMS [Dolk & Konsynski, 1984; Applegate et al., 1986].

2.1.2 Model Management System Applications

Although research in MMS is relatively new, a growing number of MMS prototypes have appeared in recent literature. Most of these prototypes were developed to demonstrate a particular design and implementation framework and basically are still at the research stage. A few well-developed and commercially available systems such as IFPS (Interactive Financial Planning System) and SPSSX (a statistical data analysis software package) were also classified by Dolk and Konsynski [1984] as primitive MMSs because they are equipped with a few MMS functions.

Management science models are the most commonly used analytical tools in supporting complicated decisions. In this subsection, we will discuss MMS examples chosen from the management science application area, with emphasis on their ability to support model design.

The traditional approach of implementing management science models in a decision support environment is to provide users with a set of available models. However, formulation of the problem and selection of the model are given very little support and are basically left to the users [Binbasioglu & Jarke, 1986; Parlar, 1989]. Current efforts by model management system researchers in this aspect are directed towards supporting model design and model selection. The majority of the prototypes have adopted techniques from artificial intelligence to build a knowledge base concerning the problem domain and the model structure, by using FORTRAN,

PROLOG or other higher level languages. However, we observe a wide spectrum of variation in terms of what design activities are supported and how they are supported in these prototypes. To demonstrate this phenomenon more clearly, we classified a few of the published MMS prototypes into the following three categories with respect to their capability in supporting problem conceptualization and model design.

Category 1 : Model Selection Tools

Systems in this category usually support model selection in a specific domain. One example is EXPIM [Parlar, 1989], which is an expert system that assists users in selecting models from a model base containing more than thirty inventory models. Systems of this type are helpful to users in classifying problems into certain pre-specified categories and in mapping the problems into pre-designed models. For example, EXPIM can help the user to decide whether a basic EOQ model or a quantity discount model is more suitable for the problem on hand. This type of system may help users to conceptualize the problem to some extent during the question-answer process, but support for conceptual modelling is very limited, for two reasons: a) the pre-specified classification of models has imposed certain constraints on the problem analysis process, and b) the domain is too narrow to incorporate many dimensions of the problem (e.g., EXPIM does not consider the human aspect of inventory problems). Many other existing software packages such

as IFPS and SPSSX also belong to this category [Naylor & Schauland, 1976; Fuerst & Martin, 1984; Hahn, 1985; Gale, 1986].

Category 2: Semi-Automatic Model Design Tools

Systems in this class can be exemplified by the linear programming building tool developed by Murphy and Stohr [1986]. Their prototype had the ability to construct linear programming models from user-identified decision variables and constraints. Hence, the models were not pre-designed. This type of system separates the knowledge of problem domains (i.e., the application areas such as finance, marketing, etc.) from that of the structure of the model. Therefore, they are domain independent. Like systems in category 1, this type of system provides very little support for conceptualizing the problem. For example, it does not support the identification of the decision variables based on the problem statement when the initial conceptual model is built by the user. Consequently, this type of system is most useful in supporting experienced model designers in the construction of complicated models. Excelerator [Whitten & Bentley, 1986], a CASE (Computer Aided System Engineering) tool that automatically generates data flow diagrams and corresponding documentation for system analysts once the entities and activities of a problem are identified, is another example in this category.

Research in the simulation field is known for its efforts in supporting simulation model design [McRoberts et al., 1985; Hill & Roberts, 1987]. A

number of simulation software packages belong in this category since they have the ability to generate graphical models and associated simulation programs automatically once the entities and relationships of the process are defined. The modern graphical interfaces for the SIMAN [Pegden, 1986], SIMSCRIPT [1990] and SLAM [Pritsker, 1986] simulation languages are good examples of such design tools.

A recent trend is to provide users with a general software development environment that has built-in functions for supporting development. For example, KEE (Knowledge Engineering Environment) [Intellicorp, 1986] integrates frames and rule-based languages into a single unified representation facility that allows the use of object-oriented programming. KEE also enables users to build behavioral models of the domains by specifying the behaviour of the domain objects.

Systems with graphical interfaces such as SIMAN, SIMSCRIPT, SLAM, and KEE provide support for conceptual model building to a certain extent, by allowing users to map the objects of a real problem to model objects more easily. However, simulation packages are specialized to simulation, and systems like KEE stress the support of the design and building of expert systems. Therefore, they are limited to certain domains and lack generality.

Category 3: Fully-automated Model Design Tools

Systems in the this category have demonstrated an ability to interpret problem statements semantically, to translate the statements into decision variables and constraints, and then to formulate the corresponding model. One example is a prototype developed by Binbasioglu and Jarke [1986], which completely automates the model building process for linear programs. Not many prototypes have been developed in this category because of the complexity involved. Systems in this category are also domain-limited, and will probably always be because of their enormous complexity.

The above classifications indicated that MMS software in the first category did not support conceptual model design directly, and software or prototypes in the second and third categories focused mainly on domain-related issues of model conceptualization. Contrary to their approaches, we want to address the general properties that are fundamental to all conceptual modelling approaches regardless of the problem domain or the design approach used. The importance of studying these general properties in the context of MMS is elaborated further in the following subsection.

2.1.3 MMS and Model Design

Our brief review of research in MMS has revealed that the majority of research studies have centred around issues in model selection, model representation, model structure and model manipulation [Blanning, 1986; Dolk, 1986; Dolk & Konsynski, 1984]. These issues do not address directly how to support problem conceptualization in the model building process, leaving a gap in current MMS research. This finding is consistent with Elam & Mead's [1990] suggestion that there is a need for a creativity-enhancing DSS environment in which creativity-relevant skills are supported. This gap provides us with the major motivation behind the current study, with an intention to narrow the gap, based on a study of the behaviour of designers during model design.

A key characteristic of MMS [Dolk and Konsynski, 1984; Fedorowicz & Williams, 1986] is that it should serve as a bridge linking the decision maker's problem environment with the appropriate models by building a knowledge base of that environment. The implied assumption is that there are existing models in the model base that may be suitable for solving all the problems that might be encountered in that environment. However, this is not always the case. Turban and Watkins [1986] relaxed this assumption and emphasized that a knowledge-based MMS would provide the heuristic and judgmental elements of model building to the decision maker, thus supporting the model design process. Although most MMS applications do emphasize support for model building in a decision environment, the

support is mainly for choosing among existing models or formulating new models using a specific technique in a specific domain. There are only a few studies discussing how to support problem conceptualization before a model is built or selected [Young, 1983; Pracht, 1990; Elam & Mead, 1990]. However, these studies were not undertaken in the context of model design.

Our research interest does not focus on supporting the choice of already existing models, or formulating models from existing modules. Instead, we want to extend Turban and Watkins' [1986] view a step further by proposing the development of certain tools to support the upstream design phase of model design. Two major activities involved in upstream design are problem conceptualization and strategy formulation [Elam & Konsynski, 1987]. Problem conceptualization is the process of analyzing a problem and identifying the key issues of that problem. Strategy formulation refers to the selection of the problem solving method to be used in conceptualization [Bonczek et al., 1981]. Support for these activities may be used in conjunction with MMSs belonging to all of the three categories classified previously, either as frontend systems or simply as functions available through the user interfaces of existing MMSs. The development of such support tools calls for further study of the general properties associated with problem conceptualization/upstream design.

Summary

In this section, we have briefly reviewed MMSs and have identified a major gap in current MMSs which fail to provide general support for the problem conceptualization stage of model design. Bridging this gap requires the development of support tools for the conceptual modelling process. The understanding of the general properties associated with conceptual model design is an essential first step toward the development of such tools. This has led to the motivation for the first goal of this study which is to develop further understanding of upstream design through empirical studies.

In the next section, we will summarize some of the basic concepts and issues of conceptual model design that are relevant to this study. The review will lead to the development of a framework that provides the theoretical foundation for the empirical studies we have undertaken.

2.2 Model Design

Issues related to model design are discussed in this section. First, in subsection 2.2.1, we present a taxonomy of models. The emphasis is on conceptual models and their important role in supporting the model designer's problem conceptualization process. Subsection 2.2.2. reviews the general model design process summarized from previous studies. Techniques supporting the model design process are outlined in subsection 2.2.3. The impact of individual

characteristics, especially the designer's prior design experience and domain knowledge, on model design is discussed in subsection 2.2.4. Support for model design is discussed in subsection 2.2.5.

2.2.1 Types of Models

Models are symbolic descriptions of systems. The descriptions can be expressed either conceptually (qualitatively) or mathematically (quantitatively). For example, a conceptual model could be a descriptive model of office activities, or a data flow diagram describing how information flows in an organization; whereas a quantitative model could be a mathematical model, such as a linear program used to find the best mix for animal feed. Conceptual models tend to be more descriptive and are usually used for the demonstration and analysis of the systems modeled. Often a conceptual model represents the model designer's semantic interpretation of the problem, and precedes the construction of any other form of model [Wilson, 1984], whereas quantitative models are usually used to characterize systems by means of mathematical expressions or algorithms.

A simulation model is a special type of model that has the characteristics of both conceptual and quantitative models [Mihram, 1972]. Basically, simulation models have a set of variables and procedures to model time-dependent events or processes and use the procedures to manipulate or to assign values to the variables successively [Vemuri, 1978]. Simulation models are more general than

mathematical models in the sense that they are executed by a set of algorithmic programs rather than by using mathematical operations. Simulation models can be represented by simulation graphs and languages. Simulation graphs usually serve as the conceptual models which demonstrate how the problem is conceptualized and how the system or process is abstracted. For example, concepts relevant in the problem are represented symbolically as graphical entities or objects; these concepts are then related to each other by relationships [Sowa, 1984]. The simulation language is then used to transform the graphic images into computer-executable procedures and programs.

All conceptual models, simulation models and mathematical models are forms of symbolic models. In general, symbolic models are represented by written expressions. These written expressions could be in the form of natural languages, diagrams, pictorial images, programming codes or mathematical formulas, all of which enable the models to be implemented on computers. Traditionally, most such implementations have emphasized the use of simulation and mathematical models simply because computers are designed for symbol manipulation. However, there is an increasing interest in using computers to aid both the design and the manipulation of conceptual models [Archer & Cui, 1992]. A few factors have contributed to this change. First, there are many social, environmental and economic problems and phenomena that can not be modeled by formulas, and these problems are as important as problems that can be formalized [Wilson, 1984].

Second, the increasing capability of digital computers in processing power, improved human-computer interfaces, and the advancement in software development have made the design and manipulation of conceptual models easier. Third, overlying every simulation or mathematical model, there is a conceptual model that maps the problem situation into that particular model [Wilson, 1984]. This conceptual model may or may not be explicitly shown in the modelling process, but it usually exists as a mental image in the model designer's mind [Greeno, 1974].

Supporting the conceptualizing process has received little attention in prior research on decision support or model management systems (Elam & Mead, 1990). We believe that a model management system with the capability to assist model designers to conceptualize their views of the problem more effectively and more efficiently will undoubtedly benefit the model design process. The lack of understanding of upstream design behaviour leads to a lack of theory and fundamental principles in providing support for conceptual design. This problem has motivated us to examine the upstream model design process further. However, before we start any empirical study, it is imperative to obtain a fundamental understanding of the processes and characteristics of model design. To this end, a review of model design processes as well as factors that affect model design is included in the next subsection.

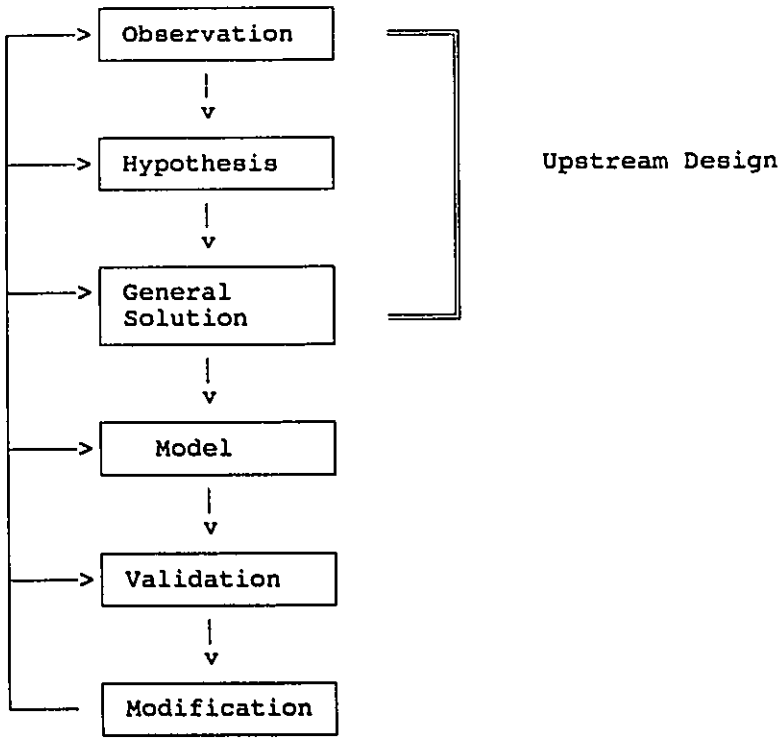
2.2.2 General Characteristics of The Model Design Process

Numerous studies have been conducted to study the design process [Newell & Simon, 1972; Akin, 1979; Broadbent, 1979; Malhotra et al., 1980; Guindon et al., 1987a, 1987b]. Many general characteristics of the design process were observed. We have conducted a literature review and synthesized from the existing literature three major characteristics of model design processes. These general characteristics are discussed in this subsection.

Characteristic 1 **Model design is an iterative process.**

Model design is the process of building a model to represent a problem situation in order to solve the problem. Vemuri [1978] has suggested a general framework for model building as shown in Figure 2.1. The process starts with observation of the problem to be solved. From observation, a hypothesis of the behaviour or characteristics of that problem is made, and a general solution to the problem can be attempted. This general solution usually widens the scope of the problem or relaxes the problem constraints. A preliminary design of the model can then be generated by imposing some constraints on the general case. Once a model is built, inferences can be drawn from the model. The model and the inferences drawn are then compared with observations for validation. If the comparison appears to be unsatisfactory, modification to the hypothesis is made and the design cycle is repeated, as an iterative process.

Figure 2.1 The Iterative Model Design Process
(Adapted from Vemuri, 1978 and Malhotra et. al., 1980)



For example, to design a conceptual model of a city parking garage, a designer would either observe the activities in a similar establishment, or make assumptions of what and how activities should be carried out in such an establishment, before a preliminary design is generated. The preliminary design is then validated either by comparing it to an existing parking garage, or by consulting with an expert. Modifications of the preliminary design are sometimes triggered by new ideas of the designer or by unsatisfactory comparisons at the validation stage.

As indicated in Figure 2.1, the activities that lead to the initial general solution are considered as upstream design. At this stage, the complexity of the design problem is first being dealt with, and initial ideas of the design problem are generated. As we have mentioned earlier, we plan to study designer behaviour empirically at the upstream design stage. Particularly, we are interested in examining what techniques are used in this stage to reduce the complexity of the problem, and how iterations are carried out.

Characteristic 2 **Model design involves constant generation of sub-design tasks and constant moving among such tasks.**

In general, a design process consists of many sub-design tasks. These tasks mainly involve formalizing and refining the design goals into many functional requirements of the problem which can be matched by the various properties of the design [Akin, 1979; Broadbent, 1979; Carroll et al., 1980; Malhotra et al., 1980]. Using the parking garage design process as an example, one design goal could be to describe the layout of the garage. The functional requirements associated with this design goal may include the location of the entrance door, the position of the toll booth, and so on. The resulting design, therefore, should have properties that satisfy these functional requirements.

Malhotra et al. [1980] observed that there are three phases involved in every sub-design task. The first phase is **goal elaboration** (analysis) consisting of the

statements and discussions of design goals. The second phase is **design generation** (synthesis). In this phase, the designer tries to develop a design that will meet the functional requirements of the problem based on the previous analyses. Usually the goal elaboration and design generation processes are interwoven and are performed iteratively. The third phase is **design evaluation** (evaluation). The evaluation process involves comparison of the design with the specified functional requirements. Design evaluation is sometimes performed within the design generation process. That is, tentative designs are constantly being generated and being evaluated at the same time. Hence, each sub-design task is also performed in an iterative fashion.

Akin [1979] studied the design processes of a group of architecture students who were asked to design a single dwelling on a lot. It was observed that a unique aspect of design behaviour is the constant generation of new design tasks by defining new design task goals and related task constraints. Akin [1979] also pointed out that not only is it unnecessary to complete one design task before starting another, but that not every design task is performed following the rigid steps of analysis-synthesis-evaluation. For example, one of the design tasks was to determine the location and size of the entrance door. This task could be carried out simultaneously with many other design tasks such as gathering information on the potential occupant and deciding on the material to be used for the entrance door. In addition, a tentative solution (e.g., a standard wooden door) is likely to be

attempted before all the relevant factors (e.g., the preferences of the occupant) are considered. Therefore, in some cases the analysis phase is actually performed, not as the first step in approaching a new design task, but rather is performed later in conjunction with other design tasks.

A similar phenomenon was observed by Guindon et al. [1987a, 1987b] in a study of the software design process. Subjects were asked to design a software program to control a lift. Their findings indicated that designers can work at many different tasks at the same time. In addition, subjects tended to move back and forth among different tasks. The tasks involved subproblems from different levels of abstraction and different levels of detail in the solution. For example, they observed that the designers could be making decisions to choose among alternative solutions (an abstraction level), then move on to make an assumption about the problem (a different abstraction level).

Characteristic 3 **The model design process is influenced by the designer's design schema. This often determines the techniques to be used during design.**

The concept of a design schema is very important in understanding the modelling process. A schema is a higher-order knowledge structure that governs the designer's behaviour in a particular domain or activity. A design schema is not tied to any specific problem domain; instead, it consists of abstract knowledge about

the structure of a particular design and the processes involved in the generation of that design [Jeffries et al., 1981]. The design schema is developed as a result of design experience. A designer with no prior experience in a particular problem domain will approach a problem with only general problem-solving strategies. As more and more experience accumulates these general strategies, with the addition of domain-specific concepts, tactics, and evaluative criteria, are transformed into a specialized schema known as the design schema. But whenever a designer's specialized schema is inadequate to solve a problem, more general strategies will be used. This process explains why design has been considered to be a generalizable and domain-independent form of problem solving activity [Newell & Simon, 1972; Thomas & Carroll, 1979].

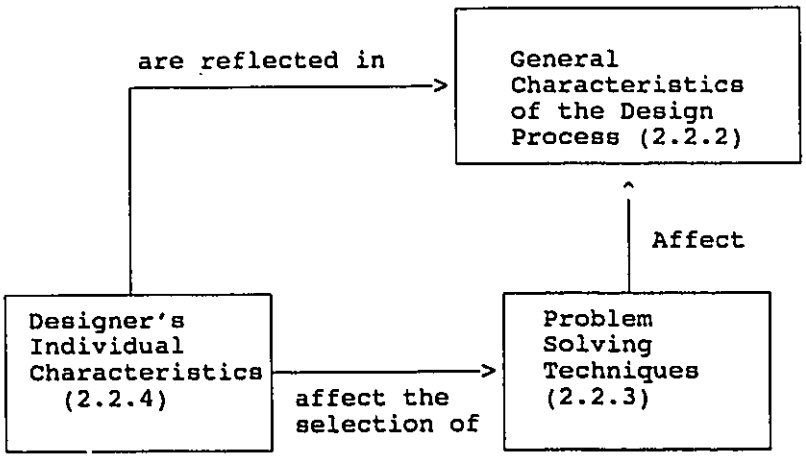
The goal of model design is to solve a problem by creating a model, using some problem solving strategy. If the model designer is experienced in the problem domain and certain design techniques, the design schema may exist and will guide the designer toward a more efficient and effective design path. However, not all model designers are experienced. Hence, they will need support to assist them to form a more effective design schema, by using general problem solving strategies. We believe it is possible to develop a general design support tool for both experienced and inexperienced designers because of the generalizable and domain-independent nature of the conceptual model design process.

Summary

Three main characteristics of the design process were discussed in this subsection. Characteristics 1 and 2 are consequences of the nature of upstream design. A designer often does not know what he/she is going to do next or how the design is going to evolve at this stage because design at this level is at its most abstract [Spillers, 1989]. The iterations, constant generation of sub-design tasks and constant movement among these tasks show that designers are constantly exploring the problem and trying to reach a particular design goal. The design schema, discussed in characteristic 3, usually develops through a designer's previous design experience and problem domain knowledge.

In the next two subsections, we will review two important factors that we consider to have partially contributed to the general characteristics of the design process summarized in this subsection. The two factors are the techniques generally useful for design, and the impact of a designer's individual characteristics on design behaviour. The relationships among the general characteristics of the design process and these two factors are shown in Figure 2.2.

Figure 2.2 Major Factors Affecting General Characteristics of the Design Process



The general problem solving techniques mentioned in the diagram above are reviewed in subsection 2.2.3, followed by a review of individual designer differences in subsection 2.2.4. A more detailed review of the studies in these areas should provide us further understanding of the dynamics among them, and help us to identify the general properties associated with the conceptual model design process.

2.2.3 General Techniques Useful for Problem Solving

As mentioned previously, design has been considered as a general form of problem solving (see under Characteristic 3). Therefore, many problem solving strategies are also useful for model design. Problem solving strategies have been studied extensively in the discipline of cognitive psychology [Best, 1986]. We have compiled a list of commonly used strategies from published literature [Newell &

Simon, 1972; Rasmussen, 1983, 1986; Eberts et al., 1988]. The understanding of these strategies will facilitate our empirical study of design behaviour and provide insight in developing design support software. Some of the strategies are described briefly below:

(a) APPLICATION OF EXAMPLES

When solving a new problem, people tend to use an example of an already solved problem to determine how to solve the new one. Or, people may recognize a new problem to be similar to a problem which has been recently solved, thus using the solved problem as an example [Anderson et al, 1981].

(b) ANALOGIES

Problem solving by analogy is similar to solving problem by example. In using analogies, a problem solver may use techniques learned from a familiar domain to solve a problem in an unfamiliar domain [Carroll et al., 1980].

(c) MENTAL SIMULATION

People tend to store a procedure that can be used to reconstruct the problem facts, rather than the solution of a problem [Rasmussen, 1983].

(d) MEANS-END ANALYSIS

Means-end analysis refers to the perception of differences between the current situation and the final solution. The analysis will suggest some sort of action to reduce the discrepancy, i.e., identify actions to bring the current state closer to the desired final state [Newell and Simon, 1972].

(e) SUBGOAL ANALYSIS

This heuristic approach suggests the finding of intermediary goals between the current state and the final state. The size of the problem space is usually reduced by subgoal analysis [Ebert et al., 1988].

(f) BACKWARD SEARCH

The backward search approach refers to working backwards from the goal state to find the initial state. This technique will be useful if the problems have a uniquely specified goal or many plausible initial states [Newell and Simon, 1972].

(g) SYSTEMATIC SEARCH

These are enumeration procedures for finding an optimal solution from a large number of feasible solutions, by systematically eliminating the least desirable solutions [Hillier & Lieberman, 1980]. Examples are branch and bound, and dynamic programming.

Among the above general strategies, application of examples, analogies, and mental simulation appear to be most applicable to upstream conceptual design, because a clear statement of the final solution is not required before these techniques can be applied to solve the problem. As mentioned in our earlier summary of the upstream design process, the design at this stage is at its most abstract, and the designer often does not know how the design is going to evolve. The application of examples and

analogies will be shown in Chapter 4 to support the use of abstraction in our software prototype. In the following subsections, we discuss how model design is affected by a designer's individual characteristics, and propose research into how to compensate for some of these different characteristics.

2.2.4 Individual Characteristics and Model Design

Individual differences are believed by some researchers to have a profound impact on cognitive activities such as problem solving, learning and decision making [Simon, 1978; Carrier & Jonassen, 1987]. Relationships between cognitive style and the implementation of information systems as well as the use of models have been extensively studied by MIS/MS researchers [Mason & Mitroff, 1973; Zmud, 1979; Taylor & Benbasat, 1980; Keen & Bronsema, 1981; Huber, 1983; Ramaprasad & Mitroff, 1984; Ramaprasad, 1987]. Many recent studies focus on how to support individual differences through adaptive user-interface design [Rich,1979; Benyon et al, 1987; Murray, 1987; Benyon & Murray, 1988]. However, comparatively little research has been done in how model design is affected by individual differences [Ramaprasad & Mitroff, 1984] and the development of software design principles to accommodate these differences [Pracht, 1990].

In general, individual differences can be grouped into categories such as aptitude variables, cognitive style and personality variables [Carrier & Jonassen, 1987]. Aptitude variables, usually measured by standardized tests, include the

intelligence level and the achievements of individuals. Cognitive style includes numerous items such as field independence/dependence, memory, cognitive complexity and so on. Examples of personality variables are intrinsic/extrinsic, introversion/extraversion, risk-taking/risk-adverse, etc. An important individual characteristic that also influences the design process is the designer's prior knowledge, which is often affected by his/her education, training and experience. It was pointed out by Greeno [1978] that cognitive style and prior knowledge appeared to have the most influence on design activities. These two characteristics are discussed in more detail below.

(1) Cognitive Style

Cognitive style usually accounts for the differences in how people prefer to interrelate ideas, and the sequence in which they prefer to gather and analyze information [Keen & Bronsema, 1981]. These activities are also related to how people approach conceptual model design.

There have been numerous studies of human cognition and its effects on problem solving and design activities [Greeno, 1974; Young, 1983; Murphy & Mitchell, 1986; Weber, 1986; Pracht & Courtney, 1988; Pracht, 1990]. We have compiled a list of cognitive attributes affecting design. Again, the intention is to understand these attributes in order to facilitate further studies in design behaviour and in the development of a software prototype to support design. The list is presented in Table 2.1.

Table 2.1 Cognitive Attributes Affecting Design

Cognitive Attribute	Description
Mental Ability in Problem Conceptualization	Individuals with higher levels of mental capacity in dealing with abstraction tend to perform better in conceptualizing a problem. I.e., they are better at changing from the concrete to the abstract mode of thought and work. This ability often leads to new ways of abstracting a problem, hence results in a more general or more creative design (Jaques, 1978).
Ability to Deal with Complexity	Human problem solvers use their short term memory and previous experience in dealing with complexity. Limited short-term memory capacity and prior knowledge cause limitations on human ability to deal with complexity. (Rasmussen, 1983; Sandford, 1985)
Limited Short Term Memory	Problem-solving sometimes involves the retrieval of information from long-term memory. The degree of success of problem-solving is sometimes affected by how well the problem-solver/designer can retrieve information from long-term memory (Murphy & Mitchell, 1986).
Minimization of Short-term Memory Load	Problem solvers tend to stick with simple, familiar and defensible strategies, hence sometimes fail to consider more complex situations. This behaviour occurs because problem solvers want to minimize the load on their short-term memory (Murphy & Mitchell, 1986).
Inflexible Representation in Design	People tend to fixate on one form of representation even when multiple forms are available during problem solving/design (Murphy & Mitchell, 1986).
Presentation-Dependent Design	Problem-solver's interpretation of the problem is dependent on how the problem is presented (Murphy & Mitchell, 1986).
Dual Coding System	Human cognitive behaviour is mediated by both an image system dealing with perceptual information and a verbal system dealing with linguistic information (Paivio, 1986). Human problem solvers sometimes use imaginal processes, or just draw pictures to represent the important relations in a problem (Greeno, 1974). Pictures can also serve as external memory aids for checking inconsistency in design (Weber, 1986).

Two of the cognitive attributes listed in Table 2.1 are of particular interest to this study, namely, mental ability in problem conceptualization, and the ability to deal with complexity. Conceptual model design is basically a form of problem conceptualization, which in turn is a form of dealing with the complexity of the problem by applying problem solving techniques to simplify, or abstract, the problem. These two cognitive attributes are most likely to benefit from proper use of abstraction. By providing effective support for abstraction, we may be able to improve the designer's mental capacity to deal with problems at high levels of abstraction, and/or to expand the designer's ability to deal with complexity.

(2) Prior Knowledge

There are two types of prior knowledge that are important in model design. One is the designer's prior knowledge of the application domain. The other is the designer's prior knowledge of design. In dealing with a particular design problem, a person could be an experienced designer who has no knowledge of the problem domain. On the other hand, a person could be a domain expert, but have no previous design experience.

It is sometimes difficult to separate domain knowledge and design knowledge in our discussion of the effects of prior knowledge. For example, previous studies in design behaviour that considered the effects of designers' prior knowledge can be

divided into two groups. Studies in one group tend to focus on design activities in a particular problem domain. Examples can be found in studies of software design [Jeffries et al., 1980; Adelson, 1981, 1984; Guindon et al., 1987a, 1987b; Batra & Davis, 1989], architectural design [Akin, 1979] and mechanical engineering design [Ullman et al., 1989; Hwang & Ullman, 1990]. These studies were conducted with the intention of understanding the design activities in that particular domain so that support could be developed for related design activities. Studies in this group tend to focus on observing the effects of prior design experience, but often fail to separate the designer's prior problem domain knowledge from prior design experience.

Studies in the second group are often led by cognitive psychologists who are interested in studying design behaviour in general [Chase & Simon, 1973; Greeno, 1974; Engle & Bukstel, 1978; Larkin, 1983; Anderson, 1981, 1985]. Their studies vary across many design disciplines and different problem domains. It is generally agreed that there are extensive interactions between a designer's design experience and domain knowledge during problem solving, and that the interaction is too complex to be fully understood at the present time [Anderson, 1985]. Our empirical study in Chapter 3 is designed in such a way that the impact of these two dimensions will be observed separately.

Newsome and Spillers [1989] summarized three characteristics of expert designers from various studies (see Table 2.2). They did not differentiate between design experience and domain experience. Nevertheless, these characteristics provide

us with a general understanding of the effects of prior knowledge in conceptual model design, and give us a starting point for our pre-prototype empirical study, which will be discussed in chapter 4.

Table 2.2 Characteristics of Expert Designers' Design Process

Characteristics	Description
A breadth approach to design problems	Expert designers tend to approach a problem in a breadth rather than a depth manner. I.e., experts tend to break a problem into a complete list of sub-problems before working through any sub-problem to its lowest level.
Ability to use abstract representations	Expert designers have a tendency to utilize relatively abstract representations or patterns in solving a problem while novices tend to concentrate on more concrete features of a problem. I.e., expert designers have better ability in categorizing problems into standard types(abstractions) based on underlying domain principles (Batra & Davis, 1989).
Expanded memory of problem-solution information	Expert designers appear to have the ability to process larger chunks of information relevant to a problem solution.

Note that all three characteristics listed in Table 2.2 are either direct or indirect consequences of effective abstraction. For example, a breadth approach taken by the expert designer is a result of horizontal abstraction (the identification of many problem facets at a particular level of detail), whereas the ability to use highly abstracted representations results from vertical abstraction (the description of several levels of detail along a particular problem facet). Both characteristics are direct consequences of abstraction. Good ability to process larger chunks of information (memory expansion) often results in a better organized design. By better organization we mean that the design ideas are retained better, hence the ideas are linked better,

because fewer pieces of information are forgotten or missing. The better organization of design ideas will also benefit from effective horizontal and vertical abstraction, which tend to reduce the complexity of the problem.

Summary

In this subsection, major characteristics of design behaviour under the influence of cognitive attributes and prior knowledge were summarized (Table 2.1 and Table 2.2). We have argued that the three characteristics of expert designer behaviour are at least partly due to their ability to use abstraction techniques more effectively. We have also identified, among cognitive attributes that affect design behaviour, two attributes (mental ability in problem conceptualization and ability to deal with complexity) that could be enhanced by effective abstraction support. In the following subsection, we will examine some of the design support proposed by researchers, and discuss the relevancy of those proposed tools with respect to supporting the use of abstraction.

2.2.5 Support for Model Design

It was pointed out by Guindon et al. [1987a] that, although upstream design has the most impact on the direction and success of a software development project, design activities are least understood by researchers and are not supported well by current tools and methodologies. Spillers and Newsome [1989] also pointed out that

there is a lack of theory and fundamental principles in providing supports for conceptual design. Furthermore, Akin [1985] stated that neither manual design tools nor computer-aided design tools were used widely, because the dissimilarity between design aids and the designer's needs is so great that it is often very difficult for the designer to use these design aids effectively. This problem may be attributed to the inability of current design support tools to accommodate the designer's cognitive attributes and individual characteristics. In subsection 2.2.2 and 2.2.3, we have summarized the general characteristics of the design process, and the effects of individual characteristics on design. In this subsection, we will discuss the support of upstream design in the context of our previous discussion.

Selected studies in supporting design are reviewed briefly first. In his review paper on research in design, Cross [1986] noted that some studies had focused on support for problem exploration and analysis (i.e., upstream design). Support is provided by assisting designers to a) identify all the factors that have to be taken into account, and b) establish the relationships among these factors in order to facilitate the identification of sub-problems. These two aspects are equivalent to the concepts of entities and relationships in conceptual modelling that we mentioned in subsection 2.2.1. Basically, this type of support dealt with the management of the ideas generated and data collected by designers during the design process. One example is Jones' [1963] method of systematic design, which attempted to leave the designer's mind as free as possible for generating random, creative ideas by providing systematic

references to maintain data, information and the requirements of the problem, for the use of designers. The emphasis is to provide a mechanism to organize ideas and information.

Young [1983] and Weber [1986] have suggested qualitative decision aids in the context of Decision Support Systems. Young [1983] proposed a theoretical framework for a system which supports decision makers in structuring problems. The key elements of his proposed system are a number of qualitative data bases which provide expert experience and knowledge in solving similar problems. The main objective is to help users to determine general parameters of the problem; these parameters are in turn used as guidelines to refine problem solving strategies. The development of this type of support tends to be time consuming and difficult because it requires the construction of extensive domain specific knowledge bases. In addition, a large amount of computing power is essential to ensure quick response time in order to cope with the speed of human thinking. Therefore, developing systems of this type is sometimes impractical for small organizations because of financial and manpower constraints. Nevertheless, Young's [1983] theoretical framework may point to a future direction for such support systems.

Weber [1986] suggested specific tools such as a scratch-pad for recording words and images that come to the user's mind; clustering techniques for organizing ideas into groups; graphics capability for sketching ideas; and the capability of combining and recombining problem elements to facilitate problem solving.

However, implementation issues were not discussed in detail.

Malhotra et al. [1980] have also proposed a set of aids to support the design process. These aids can be used as (a) an interface to show designers a number of possible designs, (b) unstructured domain-independent aids to assist designers in memory search (e.g., a list of random words), (c) complementary aids to support designers in areas where their knowledge is weak, and (d) structuring aids that lead designers to consider various aspects of the design and to ensure completeness. These aids were proposed in accordance with human cognitive attributes that affect the design process and are extremely useful for our study.

To stimulate creativity during problem solving, Elam and Mead [1990], in their laboratory experiment setting, used a DSS environment ods/CONSULTANT that included features such as qualitative thinking aids for encouraging idea generation, brainstorming, and quantitative thinking aids for prioritizing ideas. Of particular interest to this study is that ods/CONSULTANT also included utilities to support categorization of ideas along user-selected dimensions, which corresponds to our earlier discussion of horizontal and vertical abstraction in subsection 2.2.4.

In Table 2.3, aids suggested in the literature that support the process of conceptual design are grouped and labelled. These aids, listed in the right hand side column in Table 2.3, are classed according to our earlier review of general characteristics of the design process (subsection 2.2.2), cognitive attributes that affect design, and characteristics of expert designers (subsection 2.2.4). The general

Table 2.3 Suggested Aids For Supporting Conceptual Design

Design Characteristics	Supporting Aids
General Characteristics of Design Process	
Iterative Process	Information Aid -iteration paths -data collected during iteration -simultaneous display of information -checklists
Frequent Generating and Moving among Design Tasks	Information Aid -subtasks identified -subproblems solved -relations of subproblems to the overall design
Design Schema and Design Strategies	Knowledge Aid -expert's design experience -domain expert's domain knowledge Problem Solving Strategy Aid -problem solving strategies -possible design paths
Cognitive Attributes Affecting the Design Process	
Mental Ability in Problem Conceptualization	Abstraction Aid -suggestion of possible levels of abstraction -assist designers in identifying level of abstraction being worked at -keeping track of movement among levels of abstraction
Limited Ability in Dealing with Complexity	Abstraction Aid -information on complexity management techniques and examples
Characteristics of Expert Designers	
Breadth Approach to Design Problems	Abstraction Aid -suggestion of possible facets of the design problem Complementary Aid -support designers in their weaker areas
Ability in Using Abstract Representation	Abstraction Aid -support classification of details into abstract concepts
Expanded Memory of Problem-Solution Information	Information Aid -keep track of subproblem solutions

characteristics of the design process are listed in the top section of the left column; the cognitive attributes affecting design and the characteristics of expert designers are shown in the middle and lower sections of the same column respectively.

Two types of aids that appear to be most relevant in supporting upstream design have been identified in Table 2.3. The first type includes tools that are used mainly for recording the data collected by the designers, and ideas that come to the designer's mind. We labelled this type of tool **'Information Aid'**. Examples are list of subtasks identified, list of ideas generated, check list of tasks to complete, and so on. The second type of aid includes tools mainly for supporting designers in dealing with the complexity of the problem, such as assisting in identification of selected dimensions of the problem, supporting classification of design ideas, suggesting problem solving techniques, and so on. We have labelled this type of tool **'Abstraction Aid'**. Note that although many of the aids listed in Table 2.3 were recommended in the literature as possible tools for supporting the general design process, not all of them were actually implemented.

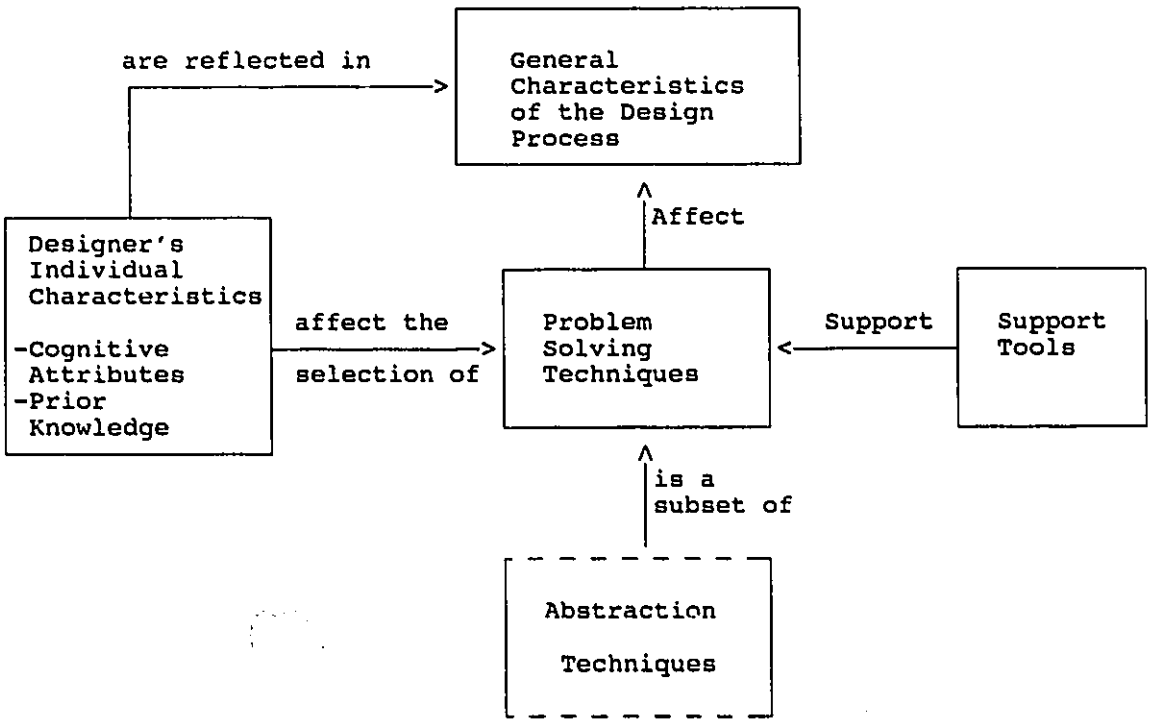
As we previously indicated in figure 2.2, in order to explain the general characteristics of upstream design, it is essential to understand design behaviour with respect to two aspects: the cognitive attributes affecting the design process, and expert designers' design behaviour. We have established in Table 2.3 that abstraction aids would provide a common support for designers in terms of the cognitive attributes affecting the design process and the characteristics of expert

designers. For example, abstraction aids would enhance an individual's cognitive ability to deal with complexity by assisting the designer in identifying the level of abstraction at which to work, in which case the problem conceptualization process is less complicated because only a subset of the entities and relationships of the design problem is dealt with at a time. Another example is that the abstraction aids would assist in broadening design dimensions by suggesting possible facets of a design problem. Of all the techniques needed to support upstream design, abstraction appears to be the most crucial, but the least understood because very little research has been done in this area. Therefore, we see a great need to understand better the role of abstraction in the upstream design process, and to use this understanding to develop abstraction aids for general design. Nevertheless, the development of such aids should not rely exclusively on the general problem solving techniques outlined earlier in subsection 2.2.3, but should also be based on the techniques specific to abstraction. This view is incorporated into Figure 2.3, which is an extension of Figure 2.2.

Several crucial links are made in Figure 2.3. First, there appears to be a distinct relationship between a designer's individual characteristics and the manner in which a design is generated. We believe that the designer's cognitive attributes and prior knowledge (design knowledge and/or domain knowledge) affect the way in which designs are developed (see Tables 2.1 and 2.2). This in turn explains to a great extent the general design characteristics identified in subsection 2.2.2. Second,

we also suspect that these characteristics can be supported by proper abstraction aids (see Table 2.3). However, it is unclear at this point how the use of abstraction and the support of abstraction can become an integral part of this framework. Therefore, the abstraction techniques are shown in a dashed outline, indicating an area that will be elaborated in the following section.

Figure 2.3 Abstraction Support for Design



Summary

In this section, selected general design aids suggested in the literature were reviewed. We established in Table 2.3 that abstraction aids are crucial to the support of upstream design. In Figure 2.3, we presented a framework that integrates abstraction techniques into the upstream design process. We also argued that there is a lack of understanding of the role of abstraction and its support in upstream design. This leads to our discussion of the concepts and techniques of abstraction in the next section.

2.3 Abstraction

The discussion in this section is organized as follows. The concept and the characteristics of abstraction are introduced in subsection 2.3.1. In subsection 2.3.2, abstraction is discussed in the context of model design. Techniques useful for abstracting are summarized in subsection 2.3.3.

2.3.1 Concepts and Characteristics of Abstraction

The concept of abstraction was introduced by Bartlett [1932] in studying the properties of the human cognitive system. He observed that people do not remember precise details of their experience, but rather form a summary representation, called an "abstract schema", of their experience. Therefore, abstraction generally involves the recoding of information in a reduced or condensed form by leaving the details

unspecified [Posner & Rogers, 1978; Zeigler & Rada, 1984].

Studies related to abstraction can be found in many disciplines. Rosch [1978] studied the effect of abstraction on human learning and memory, with emphasis on categorization and typicality; Jaques et al. [1978] studied the levels of abstraction in logic and mental activity (see Table 2.1 in section 2.2.3); Anderson [1981, 1985] discussed the use of abstraction as a major technique in problem solving; the use of abstraction in computer software used for design is frequently observed [Jeffries et al., 1981; Guindon et al., 1987a, 1987b]; engineers use the concept of abstraction to analyze task complexity to improve human-machine interaction [Rasmussen, 1983, 1986]; and process abstraction has been extensively used in building simulation models [Zeigler & Rada, 1984; McRoberts et al., 1985; Bond & Soetarman, 1988; Fishwick, 1988].

The objective of creating abstractions of a problem is usually to reduce the complexity and dependency of that problem [Whittlesea, 1983; Fishwick, 1988]. The complexity of a problem is implicitly defined by the number of objects in the problem and the values associated with those objects. The more objects there are in a problem, the more complex the problem. Dependency is usually indicated by relationships (e.g., cause-effect relation between objects, co-occurrence of object values) among the objects. More complicated relationships among the objects imply a more complex problem. In general, abstractions are created by reducing the number of objects and their associated values as well as by simplifying the

relationships between these objects. As a result of the abstracting process, fewer chunks of information need to be processed by a person's cognitive system when dealing with abstraction, because the number of objects in an abstraction of a problem is usually fewer than that in the original problem. Therefore, abstraction is considered to be one of the most powerful tools for handling complexity [Ossher, 1987].

Conceptual modelling is also a process dealing with the complexity of a design problem, by identifying the objects (concepts or design ideas) and relationships among the objects of the design problem. Hence, the process of conceptual model design is similar to the abstracting process described above. Using the city parking garage design problem as an example, by choosing to deal with the location aspect of the problem only, a designer basically has abstracted the problem in such a way that only those concepts and relationships that are relevant to the selection of a location are dealt with. It is very difficult, if not impossible, to specify the contents of various abstractions of a problem. If we define problem dimensions to be the different facets of a problem, then a **horizontal abstraction** of the problem may include many problem facets at a particular level of detail. A **vertical abstraction** of the problem may involve several levels of detail along a particular problem dimension. Using the design of the city parking garage as an example, the horizontal abstractions may include problem facets such as location of the garage, layout of the garage, and the environmental aspect. Vertical abstractions

may include details of a particular facet; e.g., the layout of the garage may include details such as number of levels, location of the toll booths, and so on.

As one would expect, horizontal abstractions generally deal with the breadth of a problem, whereas vertical abstractions deal with the depth of a problem. A problem abstraction usually includes both horizontal and vertical abstractions to some extent. Because the number of possible combinations of horizontal and vertical abstractions is usually very large, this leads to a very large number of potential abstractions for a problem. Hence, the content of the various problem abstractions are very difficult to specify. However, this characteristic allows the designer a great deal of flexibility if abstraction can be applied freely during the design process.

A key factor related to the number of potential abstractions is the effect of individual characteristics outlined in subsection 2.2.3. The process of creating an abstraction, i.e., the process of reducing complexity and dependency, may be based on any set of criteria selected by an individual [Jaques et al., 1978]. The selection is affected by a combination of factors such as how the problem is perceived by the individual; the individual's prior knowledge and cognitive style; as well as the individual's mental capacity in dealing with abstraction.

In addition to the difficulty in determining the content of various abstractions of a problem, it is also difficult to classify abstraction levels. For every problem, there could be a spectrum of abstractions ranging from a form that is extremely abstract to a form that is concrete. In general, a higher level of abstraction means

dealing with the concepts of the problem, thus moving away from the details. A lower level of abstraction tends to involve more details, hence often implies dealing with the problem at either the operational level or procedural level [Deogun & Nakhforoush, 1984]. This general classification of abstraction is applicable for abstracting both static systems and dynamic processes. For example, a hierarchical data flow diagram as used in information systems design usually includes a set of abstraction models describing a system from the conceptual level to the procedural level [De Marco, 1979]. Simulation models are known for using abstractions to represent various levels of detail of dynamic processes [Torn, 1981; Bond & Soetarman, 1988].

There may be numerous levels of abstraction for each problem. However, it is not necessary to deal with just one level of abstraction at a time during the problem solving process. A problem may be approached simultaneously at several levels of abstraction if desired. In fact, constant generation of design tasks combined with flexible use of analysis-synthesis-evaluation during the design process (see subsection 2.2.2) is an example of working with several levels of abstraction at the same time.

In summary, abstraction is a process that can be used to control the complexity of a problem by including the appropriate level of detail, and abstraction can be applied to a problem horizontally or vertically. The four characteristics of abstraction during problem solving are a) it is difficult to determine the content of

various abstractions of a problem; b) it is difficult to classify abstraction into various levels in terms of the details involved; c) the use of abstraction is influenced by individual characteristics; and d) people tend to work with several abstractions of a problem simultaneously.

Little research to date has focused on studying how abstraction is applied in the design process, or on the effects of such application. However, because of its power and flexibility in handling complexity, abstraction is a technique frequently used by model designers. In addition, how it is used is an influential factor which affects the model design. We will discuss abstraction in the context of model design in the following subsection.

2.3.2 Abstraction and Model Design

In the modelling process, a model designer tends to extract relevant information or data from the problem situation to form a conceptual model of the problem to represent his/her comprehension of the problem. This process is often performed through some form of abstraction [Piaget, 1974; Norman, 1981; Rasmussen, 1983; Ramaprasad & Mitroff, 1984]. The generation of conceptual models allows model designers to simplify the problem, so that they can (1) focus on certain problem facets, (2) deal with problems at a desired level of complexity, and (3) think about the problem rather than being occupied by the details.

A study by Jeffries et al., [1980] showed that a more experienced designer may form a conceptual model based on his/her existing design schema. However, for a novice designer who does not have prior knowledge in design, generating a more abstract conceptual model is usually the intuitive first step taken to reduce the complexity of the problem. Ramaprasad and Mitroff [1984] presented a theoretical argument that the unique way of applying abstraction in the model design process is not only affected by the designer's previous design experience, but it also corresponds to the designer's personal type. However, there is a lack of empirical study explaining in what way the application of abstraction is related to these differences.

It was mentioned in section 2.3.1 that many levels of abstraction can be identified in the design process. One example is the general conceptual-operational-procedural classification [Deogun & Nakhforoush, 1984]. Another example is given by Guindon et al. [1987b], who identified three levels of abstraction: level of the design process, level of criteria to evaluate alternative solutions, and the level of solution to the problem. In studying the design process, this classification is very useful for analyzing why certain design tasks are generated. However, during the design process, a model designer may not organize the problem based on these levels of abstraction. In fact, a designer probably is not aware that he/she is working on the criteria-evaluation level or on the solution-finding level. Instead, he/she may have different perspectives on how to abstract a problem. For example, a system

analyst and a programmer might have different views on the same application software problem, thus abstracting the problem differently.

We have emphasized in section 2.2.4 that a designer's cognitive style and prior knowledge have strong influences on how people structure the problem. We suspect that the impact of these individual characteristics on the use of abstraction in model design can be studied by observing (1) the techniques used in abstracting a problem, which may be observed in terms of the general design techniques outlined in subsection 2.2.3, or the abstraction technique to be discussed in 2.3.3; (2) the information chosen by the designer to be included in the abstractions; (3) the number of abstraction levels identified; and (4) the ways designers move among various levels of abstractions. These observations are made in our pre-prototype studies, discussed in Chapter 3.

To facilitate our empirical study, we review a few techniques that are useful in problem abstraction, which are summarized in the following subsection.

2.3.3 Techniques Useful for Abstraction

There are various techniques for creating model abstractions. Ossher [1987] compiled a list of complexity management techniques that are highly relevant to our study. These techniques may be adopted as fundamental tools to deal with problem complexity or for developing other abstraction aids. Some of the techniques are briefly introduced below:

(a) CLUSTERING

Clustering involves grouping entities that have some important common properties into a cluster, associating these properties with the cluster itself, and then regarding the cluster as an entity whenever possible.

(b) HIGHLIGHTING

Highlighting is intended to draw the reader's attention to a small number of important facts. It is useful in showing the important aspects of a complex system.

(c) ANALOGY AND DEVIATION

Analogy is the use of rules developed for a familiar situation on an unfamiliar problem. Deviation specifies the difference between the known problem and the unknown problem. The combination of analogy and deviation is often extremely effective.

(d) APPROXIMATION

Approximation describes a real situation by leaving out some of the details. It is sometimes adequate to know the approximation of actual reality when the reality is too complex.

(e) CONVERGENT APPROXIMATION

A complex situation can be explained in a sequence of approximations. The initial approximations in the sequence are gross, but easy to understand. The later approximations are more accurate and difficult to understand, but

understanding is facilitated by exposure to the previous approximations.

(f) LOCALIZATION OF INFORMATION

Localization of information implies that closely related information should be physically close together in a region of space. In addition, within such a region, there should be no unrelated information. The purpose is again to leave out the unnecessary details.

(g) USE OF DIAGRAMS

Most human beings visualize structure graphically. The use of graphics such as various forms of conceptual graphs [Sowa, 1984] usually enhance understanding of structural concepts and relationships.

The abstracting techniques listed above are not only useful for the abstracting process, but also can be used to support the general problem solving techniques outlined in subsection 2.2.3. For example, analogy appears to be a useful technique for both abstraction and for general design. Ossher [1987] suggested that the use of these techniques can be enhanced if they are incorporated properly into computer interface design. Based on the material we have discussed, Table 2.4 contains a summary of the three dimensions of abstraction, abstraction aids associated with these three dimensions, and potential useful methods to implement these aids. More discussion on abstraction and its support is presented in the pre-prototye studies in Chapter 3.

Table 2.4 Types of Abstraction and Suggested Support

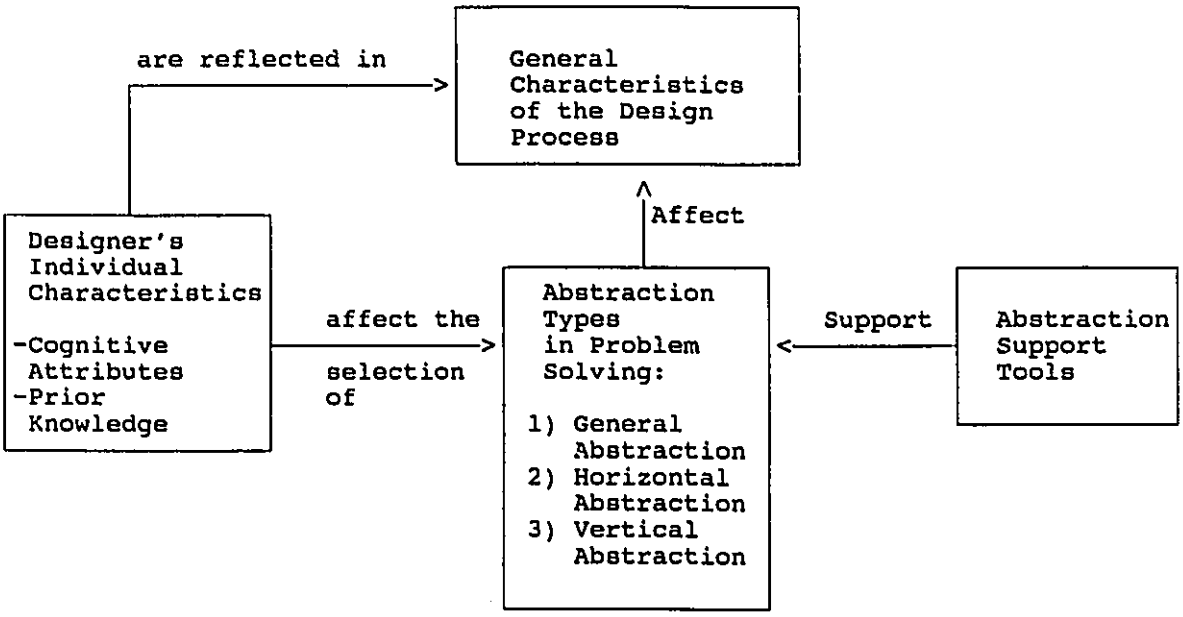
Abstraction Type	Support	Techniques
General Abstraction	-list of abstracting techniques -possible design paths -support movement among levels of abstraction	Examples [Anderson, 1981] Analogy [Carroll et al., 1980] Reference [Jones, 1963]
Horizontal Abstraction	-suggestion of horizontal dimensions	Examples Analogy
Vertical Abstraction	-suggestion of possible levels of abstraction -assist designers in identifying which level of abstraction is being worked at -support classification of details into abstract concepts	Examples Analogy Reference Clustering [Ossher, 1987]

Summary

In this section, abstraction concepts and techniques were reviewed. We have identified three general types of abstraction and their respective supporting techniques, as summarized in Table 2.4. These findings are now further incorporated into the framework, shown earlier in Figure 2.3, to explain further the areas related to abstraction. In the revised diagram in Figure 2.4 below, we have identified three abstraction types as a subset of problem solving techniques. These types can be implemented by abstraction support techniques summarized in Table 2.4. The general characteristics of the design process of any individual designer should be affected by the problem solving/abstraction technique chosen, or how the uses of these techniques are supported. Therefore, we believe that there will be a direct relationship between the use of abstraction and the general characteristics of the final design. However,

this relationship has never been studied. This explains our emphasis on the study of abstraction and its support. The theoretical framework shown in Figure 2.4 will guide our empirical studies of model design towards verifying the existence of certain relationships between the use of abstraction and the general characteristics of conceptual model design, and explaining what those relationships are.

Figure 2.4 Conceptual Design Framework Incorporating Abstraction Techniques



We will conclude our literature review in this chapter by summarizing the research questions identified and the research goals of this study in the following section.

2.4 Research Questions and Research Goals

In this section, we first summarize the research problems that we have identified. The research goals are derived directly from the problems. The intention is to address these issues and to further our understanding in these areas.

First, the research problems are outlined as follows:

(1) **Although MMS research emphasizes the need for support for model design, a review of the literature has indicated that there is little understanding of how to provide proper support for building models at the conceptual level.**

(2) **The lack of understanding of upstream design behaviour contributes to the lack of theory and fundamental principles in providing support for conceptual design.**

(3) **Previous research has often failed to separate domain knowledge and design experience in studying designer behaviour.**

(4) **It appears that abstraction is one of the most powerful tools used by model designers to manage problem complexity. Studies in a number of application areas have identified extensive use of abstraction. However,**

abstraction was often discussed as a by product associated with observations of other phenomena, rather than the issue being studied. No study appears to have been conducted to record the use of abstraction or to measure the effectiveness of abstraction in improving the quality of design.

We have established our research goals to address some of the issues relevant to the above listed problems. Our goals are outlined as follows:

(1) To develop further understanding of upstream design through empirical studies.

(2) To study how abstraction is used during the design process. The findings will bring insights to the development of strategies for supporting the use of abstraction.

(3) To develop a theoretical framework to explain the relationship of abstraction to conceptual model design.

(4) To develop quantitative measurements that reflect the use of abstraction in design.

(5) **To develop a software prototype to demonstrate the implementation of techniques supporting the use of abstraction in model design.**

(6) **To conduct an experimental study which evaluates the effectiveness of the proposed abstraction support in certain design environments.**

These goals will lead to the development of a framework that addresses the relationship between abstraction and the general characteristics of conceptual design. In the following chapters, we will present three major efforts toward fulfilling these research goals. The first effort is to conduct a pre-prototype empirical study to observe upstream conceptual design behaviour and the use of abstraction. Based on the findings of these empirical studies, we will develop methods to support abstraction as well as measurements of abstraction and to evaluate its contribution to the final design. The pre-prototype empirical studies are presented in Chapter 3. The second effort is the development and testing of a software prototype, which is documented in Chapter 4. The third effort involves the testing of the effectiveness of the abstraction support methods. Two different support methods and three different design environments are included in an experimental study. The procedures and the results of the experimental study are presented in Chapter 5. Chapter 6 concludes the study.

Chapter Three

PRE-PROTOTYPE EMPIRICAL STUDIES

This chapter discusses our efforts towards achieving the first four research goals stated in section 2.4. The goals are 1) to develop further understanding of upstream design, 2) to study how abstraction is used during the design process, 3) to develop a framework to explain the relationships of abstraction to conceptual design, and 4) to develop measurements representing the use of abstraction. The conceptual design framework shown in Figure 2.4 was used to guide the planning of the empirical studies, so that the findings of the studies would further our understanding of some of the relationships in that framework.

Three phases of protocol studies are included in this chapter. These studies are referred to as pre-prototype empirical studies, because they were conducted before the development of a software prototype to support design. These studies are briefly introduced in Table 3.1, with their respective objectives.

As outlined in Table 3.1, Phase I was conducted as a preliminary study with two objectives in mind. The first objective was to gain experience in conducting protocol analysis. The second objective was to observe the general characteristics of design behaviour as well as to identify problems associated with protocol studies of

this nature. The observations from phase I studies were used as input in the design of the second phase studies. Phase II studies were conducted in such a way that the effects of both previous design experience and the presence of software support could be studied. The phase III study was aimed at observing design processes of domain experts. The design problem chosen for all protocol studies was the design of a high school counselling office. Therefore, the domain experts we studied were counsellors from various high schools. The protocol data and analysis of all three phases were used later, in conjunction with theories from the literature, in the development of a design support prototype.

Table 3.1 Objectives of the Pre-Prototype Empirical Studies

	Phase	Objectives
I	Preliminary studies	1.To obtain experience in conducting protocol analysis 2.To observe the design process in general
II	Design behaviour of non-domain experts	1.To study the effects of previous design experience and the presence of software support on design processes 2.To observe how abstraction is applied by non-domain experts.
III	Design behaviour of domain experts	1.To study design behaviour of domain experts 2.To observe how abstraction is applied by domain experts 3.To compare design patterns of domain experts and non-domain experts

Protocol analysis was adopted for all three phases of the studies used to observe model design behaviour. Protocol analysis [Byrne, 1983; Evans, 1988] is the collection of data on the activities of the subjects while they are performing tasks, by requiring them to think aloud (talk to a recording machine). At the same time, their behaviour can be observed either directly by the experimenter or by making a video recording. The purpose of protocol analysis is to observe the thinking and behaviour of subjects relating to their problem solving processes. According to Akin [1979], protocol analysis is especially suitable for the purpose of studying design processes. First, it provides a context in which design can be explored in its entirety; i.e., in an aggregated form. Second, no a priori experimental hypotheses about the design tasks are essential. Third, it can provide rich data in the form of verbal protocols as well as reaction times and/or visual attention cues.

The findings of all the pre-prototype studies are presented in sections 3.1, 3.2 and 3.3 respectively. From these findings, further strategies were developed to support the use of abstraction, and measurements suggested for recording/evaluating the use of abstraction. The recommended abstraction aids and abstraction measurements are discussed in section 3.4 and section 3.5 respectively.

3.1 Phase I - Preliminary Study

Objectives

This preliminary study allowed us to obtain experience in conducting protocol analysis and to observe design behaviour in general.

In this exercise, participants were required to incorporate a set of decision criteria, such as 'accept the job if salary is higher than 40,000 dollars', into the structure of a decision table. They were asked to choose a job offer from a set of fictitious job offers, using the decision table they had designed (see Appendix I).

A decision table design problem was chosen because of its structure, which allows explicit design of abstraction hierarchies. The purpose was to observe general design behaviour, and how abstraction was applied in the design process. The career-decision problem was stated in such a way that it could be represented either by hierarchies of simple decision tables or by one very complex decision table.

Subjects

Three business doctoral students were randomly selected to participate in the preliminary studies. All participants were unfamiliar with the decision table technique.

Design Tasks

The subjects were explicitly asked to perform two tasks. The tasks were (1) designing the decision table, and then (2) making a career decision using the decision

table. We refer to the first task as the design task and the second task as the application problem-solving task.

Procedure

The study was conducted with each individual separately. Upon entering the room, the participant was given a description of the structure of a decision table and an example of how a decision table could be used. After the participant finished reading the material, he/she would start the design process using pencil and paper. Participants were also asked to verbalize their thinking process while they worked. This was tape-recorded for subsequent protocol analysis. No time limit was imposed on the participants.

Observations

It was observed that all three subjects were reluctant to use the decision table as a tool for solving the career-decision problem. They tended to approach the problem using a more familiar method, such as the decision tree method, and then to transform their design into decision tables. In addition, they referred to the decision tree rather than the decision table in the process of making a career decision.

As a result, we gathered only limited information on how they would perform the design task when a pre-specified tool (the decision table) was given. However, we could clearly observe the use of abstraction in their design processes through the various ways by which they decomposed the career decision problem into subproblems.

Following is a brief summary of the observations from phase I:

(1) In a situation when a person acts as both the model designer and problem solver, he/she may be unwilling to use a particular method (or model) with which he/she is unfamiliar. This is because his/her goal is to solve the application problem (i.e., to make a career decision based on the given criteria) instead of designing a model which could be used in solving the application problem later;

(2) All participating subjects were sophisticated in using various kinds of scientific methods. Thus, it was natural for them to use the decision tree method (or any other method) to design a model for solving the given problem;

(3) The use of abstraction was observed. Levels of abstraction could be identified through the hierarchies of sub-decisions formulated by the subjects;

(4) An open-ended design problem (i.e., a problem with no particular solution) would be more suitable for our purpose. In that case, subjects would tend to focus more on the design task than on finding a solution to the application problem;

(5) A study similar to this one would be useful for studying how designers would react when they are restricted to certain design methods; this is a common problem that exists in all current computer-aided design environments.

A key observation from this study is that people are reluctant to use unfamiliar methods for problem solving. For designers of conceptual models, the problem solving process is the model design process. In other words, for them, the problems are solved when the models are designed. As we have mentioned in the previous chapter, model design is a process that requires an ability to manage the problem's complexity. The implication of this observation is that, although there are various ways of managing complexity (e.g., aggregation, clustering, abstraction, systematic design, etc.), if the methodology does not agree with the preference or training of the designer, he/she will not be likely to make the best use of that method. Furthermore, this observation makes the use of abstraction very appealing because there is no underlying procedure imposed on its generalized use. In other words, model designers may benefit from the use of abstraction no matter how they abstract the problem. Hence, abstraction seems to allow more freedom in terms of accommodating a designer's individual preferences during the design process.

In summary, we learned the following in the Phase I study:

(a) The design task (i.e., design of the decision table) and the application problem-solving task (i.e., choose a particular job offer) should be separated;

(b) No design tool or design method should be imposed on the subjects;

(c) Abstraction in some form seems to be a technique that will accommodate a designer's individual preferences;

(d) An open-ended design problem should be used to encourage creativity and imagination in conducting further studies into abstraction; and

(e) For comparative evaluation of the use of abstraction in different situations, relevant measures must be developed.

These findings were later incorporated into the planning of phase II studies, which are presented in the following section.

3.2 Phase II - Design Behaviour of Non-Domain Experts

Objective

The phase II studies were conducted to observe actual design behaviour. In chapter 2, we identified three factors that are relevant to our study of design processes. The factors are 1) previous design experience of the subjects, 2) the subjects' domain knowledge of the application problem (see prior knowledge in subsection 2.2.3), and 3) the availability and usefulness of software support during design. The objective of the phase II study was to develop a better understanding of the effects of previous design experience and the presence of software support. Preliminary measurements of abstraction were made. In addition, a graphical method was used to display the thinking process of subjects as they moved among different abstractions.

Experimental Design

We adopted the two-way classification analysis of variance design to observe the behavioral difference in design under the influence of previous design experience, and availability of software support. The experimental design is shown in Table 3.2. This design was actually quasi-experimental because subjects were not randomly assigned to the four treatment groups. Subject selection is dicussed in more detail in the following subsection.

Table 3.2 Experimental Design of Phase II Protocol Studies

Design Experience	Software Support: NO	Software Support: YES
NO	Group 1	Group 2
YES	Group 3	Group 4

Group 1: Subjects with neither experience in design nor experience with computer-supported model design tools.

Group 2: Subjects with no experience in design, but with some experience in using a computer-supported model design tool. Software support was provided to this group.

Group 3: Subjects with extensive design experience, but without experience in computer-supported design tools, and,

Group 4: Subjects with extensive design experience and good knowledge in using computer-supported model design tool(s). Software support was provided to this group.

This experimental design allowed us to :

a) observe the design behaviour of both experienced and inexperienced designers;

b) observe the design behaviour of designers with and without software support;

c) study protocols gathered from users of an existing design package (group 2 and group 4);

d) gather data on the use of abstraction in design; and

e) gather information for the development of a software prototype that would support abstraction in the design process.

Note that the domain knowledge factor was not included in this phase of the study. Only non-domain experts were used, in order to reduce the influence of prior domain knowledge (domain experts were studied in phase III).

Subjects

There were three subjects in each treatment group. Subjects in group 1 were selected from various backgrounds, including an accounting clerk and two college students: one majored in German, and the other majored in mathematics. Subjects in group 2 were MBA students who did not have previous design experience, but had received some training in using Excelerator (a CASE (Computer Assisted System Engineering) tool). Subjects in group 3 included a software designer, a system

designer and a civil engineer. Subjects in group 4 were professional system analysts who had used Excelerator extensively to support their daily design tasks.

Design Tasks

Modifications to the selection of the type of design problem to be used in the study were made in phase II, based on the findings of phase I. a) The subjects were asked to design a high school counselling office (see Appendix II.C.); b) The subjects were asked only to perform design tasks; c) No design method was imposed on the subjects who were not supported by the software; and d) Individuals who were not familiar with scientific problem solving methods and/or with design methodologies were included in the study.

A counselling office is perhaps one of the most important departments in a high school. It provides academic, career, personal and health counselling to the students. Academic counselling usually includes course selection/scheduling, advice for post secondary education, job skill training, and so on. Career decision information and job placement are provided to students through this office. Personal problems, ranging from dealing with stress from the family to facing peer pressure, are often dealt with on a one to one basis between the student and the counsellor. Health issues such as drugs and alcohol are discussed in special sessions regularly. Students with special needs are also referred to the counselling office. Therefore, every high school student will usually have some contact with the counselling office.

The selection of the high school counselling office design problem was based on three criteria. First, it is a general and open-ended problem that is familiar to many individuals with different backgrounds. Second, the problem is not too complicated to distort our focus on the analysis of the use of abstraction. Third, the implicit hierarchical structure of the problem (for example, skill training and job placement can be clustered as a dimension related to career counselling), facilitates our analysis of how abstraction is applied.

Procedures

All potential subjects were screened by a preliminary questionnaire (see Appendix II.A.) from which we obtained information on their prior knowledge in design and the problem domain area, as well as their attitudes toward computer use in general. Subjects were assigned to the appropriate treatment groups based on the result of the preliminary questionnaire.

All studies were conducted on an individual basis. Each individual's design activities were observed, and protocols were recoded and analyzed. Treatment groups 1 and 3 conducted their design using pencil and paper. Upon entering the room, the subject was given the design problem sheet (see Appendix II.C.), and was asked to write down his/her design on the paper. The subjects were also told that their verbalized thinking processes would be recorded for further analysis. In general, subjects in groups 1 and 3 used approximately forty-five minutes to conduct the design.

Treatment groups 2 and 4 were given both pencil and paper, along with access to Excelerator which was installed on a personal computer. The same procedure was used for the pencil and paper design groups. Subjects in these two groups averaged sixty minutes to complete their designs.

Software Selection

Excelerator was selected for this study for the following reasons: a) It is a standard CASE tool which supports graphics and the documentation of system analysis and design; b) It belongs to the second category of model management systems according to our classification (i.e., as a semi-automatic design tool, it supports model building; however it emphasizes more the documentation than the design); and c) It is widely used in industry and is readily available commercially. Most importantly, because Excelerator does not provide full support for the conceptual model design processes, by observing subjects using Excelerator to perform design tasks, we were able to identify some of the constraints imposed by a current commercially used design tool.

Data

As pointed out in the summary of findings of the Phase I study, more specific measurements for observing the use of abstraction and design behaviour were required. Therefore, in addition to the qualitative data extracted from the protocols, we also collected quantitative data that are related to the use of abstraction. These data, summarized in Table 3.3, were grouped into four categories:

Table 3.3 List of Dependent Variables of Phase II Study

Data	Description
(1) Problem Facets Identified	
1	Total Number of Problem Facets
(2) Number of Ideas Identified	
2	Total Number of Ideas
3	Number of Ideas at High Level of Abstraction
4	Number of Ideas at Middle Level of Abstraction
5	Number of Ideas at Low Level of Abstraction
(3) Movement Among Levels of Abstraction	
6	Number of Times Moving from a High Level Idea to a High Level Idea
7	Number of Times Moving from a High Level Idea to a Middle Level Idea
8	Number of Times Moving from a High Level Idea to a Low Level Idea
9	Number of Times Moving from a Middle Level Idea to a High Level Idea
10	Number of Times Moving from a Middle Level Idea to a Middle Level Idea
11	Number of Times Moving from a Middle Level Idea to a Low Level Idea
12	Number of Times Moving from a Low Level Idea to a High Level Idea
13	Number of Times Moving from a Low Level Idea to a Low Level Idea
(4) Design Task Generation	
14	Total Number of Tasks Generated
15	Number of Tasks Completed
16	Number of Tasks Not Completed
17	Task Completed Then Go to a New Task
18	Task Completed Then Go to an Old Task
19	Task Uncompleted Then Go to a New Task
20	Task Uncompleted Then Go to an Old Task
21	Total Number of Activities

(1) Problem Facets Identified

The number of problem facets identified in the final design was counted. By our definition, problem facets refer to the different dimensions of a design problem, which is equivalent to horizontal abstraction as defined in Table 2.4. Examples of problem facets are office functions, personnel, physical layout, location of the office, and so on.

(2) Number of Ideas Identified

The ideas identified by the subjects in each facet were further grouped into high, middle and low levels of abstraction based on the details involved (see Appendix III). This classification is similar to the vertical abstraction discussed in Table 2.4. The total number of ideas was also tabulated.

(3) Movement Among Levels of Abstraction

The movement among different levels of abstraction was captured by tabulating the number of times each subject moved from one idea to another idea. The ideas could be in the same or different levels of abstraction. Two examples of our graphical analysis of the movements among different abstraction levels are illustrated in Figure 3.1.

(4) Design Task Generation

The quantitative aspect of design behaviour was measured by counting i) the total number of design tasks generated, ii) the total number of design tasks completed, iii) the number of times the subject moved to a new/old task after completing the task on hand, and iv) the number of times the subject moved to a new/old task before completing the task on hand.

The qualitative observations from the protocol studies are presented in subsection 3.2.1, followed by the exploratory analyses of the quantitative data in subsection 3.2.2.

Figure 3.1a Graphical Analysis of Movements Among Abstraction Levels
Phase II - Example 1

This is an example of the analytical top-down design approach. We can observe that there is relative little interaction among different levels of abstraction.
Legend: ( ) indicates design tasks identified

⟶ indicates transition from one task to the next tasks
numbers on the arcs indicate the sequence of movement

– – – indicates boundaries between levels of abstraction

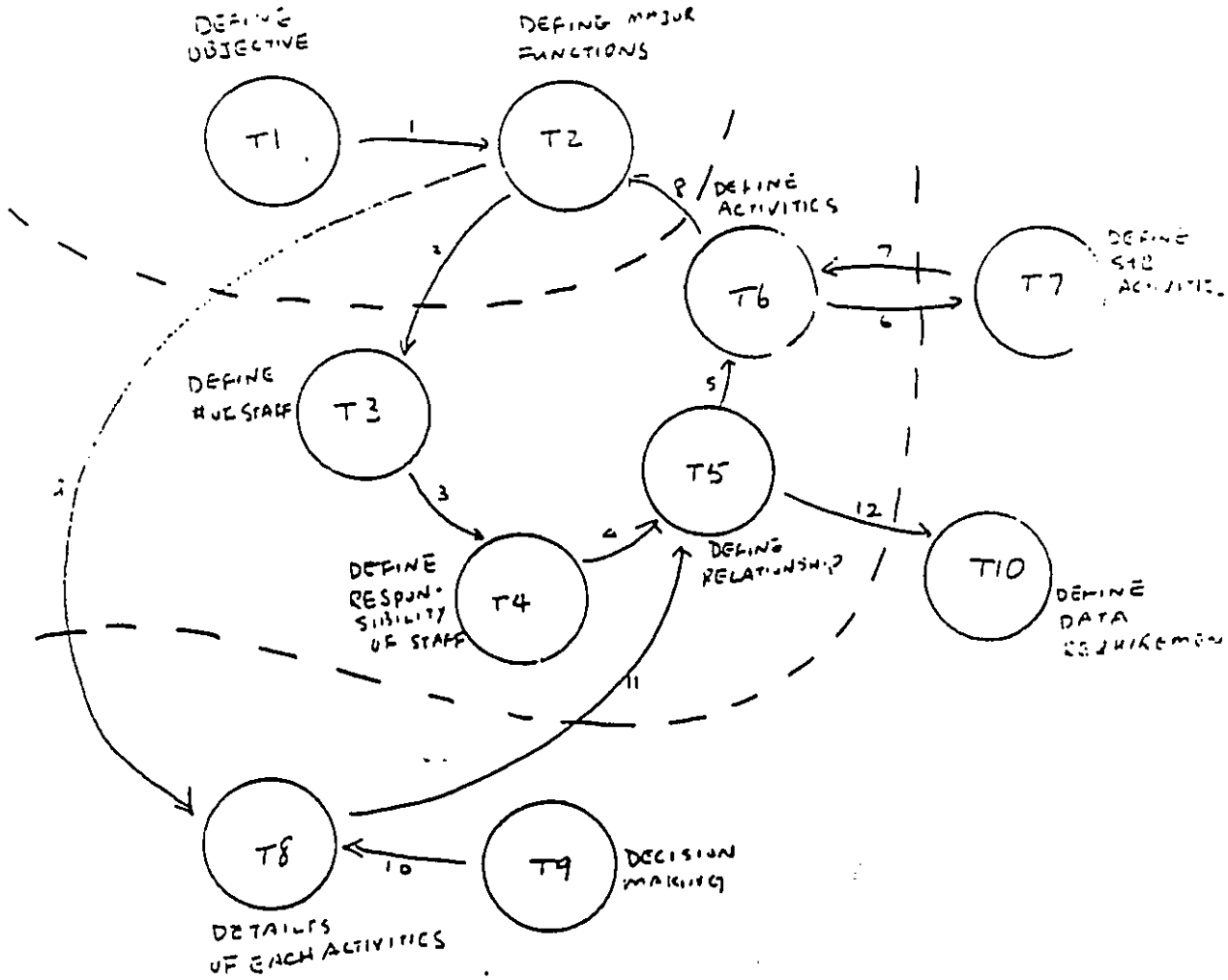


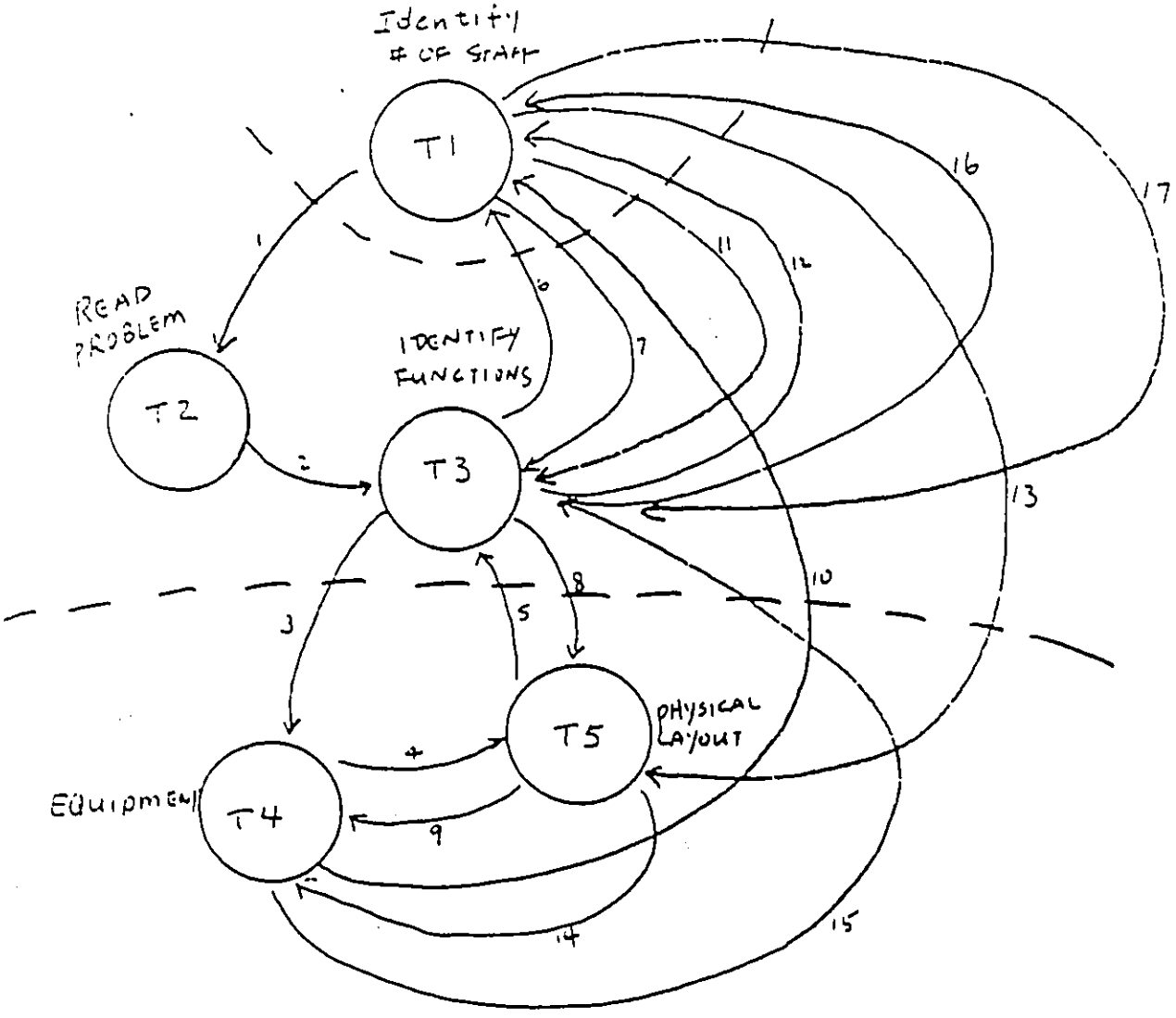
Figure 3.1b Graphical Analysis of Movements Among Abstraction Levels
Phase II - Example 2

This is an example of the heuristic bottom-up design approach. We can observe that the designer did not differentiate between different levels of abstraction. The designer constantly moved from one level of abstraction to another. The majority of the design was focused on the lower abstraction level.

Legend: ( ) indicates design tasks identified

——→ indicates transition from one task to the next tasks
numbers on the arcs indicate the sequence of movement

– – – indicates boundaries between levels of abstraction



3.2.1 Qualitative Analysis

The qualitative analyses are summarized into three groups and discussed in this subsection. The three groups are (1) general observations applicable to both the pencil and paper design groups and the Excelerator users, (2) observations summarized from subjects using Excelerator, and (3) suggestions of the subjects for desirable features in design support software.

(1) General Observations

Observation 1

The protocols demonstrated that new design tasks were being generated constantly and that the subjects moved from one design task to another constantly, which conforms with the first general characteristic of upstream design (see section 2.2).

This phenomenon is demonstrated by an excerpt taken from one of the protocols collected in phase II ([ ] indicates editorial clarification). The time index (in seconds) is added to indicate the time spent on each line. For example, it took this particular subject 6 seconds to make the statement and complete the activity indicated in line 1. These protocols are also analyzed line by line in terms of design task, in Table 3.4. It can be seen that in the process of completing task #1, the subject also generated tasks #2 and #3. However, task #3 (Line 4,5,6) appeared to be irrelevant and was discarded. Task #2 (Line 2,3,8) turned out to be useful and

was used as a constraint for reaching a solution for task #1.

Time Index

00 Line 1: So, first, just guess the number of people that are gonna work in the office.

06 Line 2: For a high school, I'll have to guess how large the high school is.

15 Line 3: Say, 3,000 people, students.

20 Line 4: It's an ordinary secondary high school, so there will be a graduating class of, say 25% of 3,000.

30 Line 5: O.K. Say, there will be.... that's too large.

45 Line 6: Oh well, say there are 500 [students] graduating each year,...

50 Line 7: Career planning....

70 Line 8: O.K., estimate that half of the [students] at some time in the year need counselling. So, you are down to 1,500 students [who] will need counselling.

100 Line 9: And say, one person can handle 300 [students]. You'll need a staff of 5 counsellors,... 5 counsellors and 2 secretaries. Sounds good.

120 Line10: So there is a staff of 7.

Table 3.4 An Example of Constant Generation of Design Tasks

Design Task	Line #	Description
1	1	guess the number of people working in the office
2	2	guess how large the school is
2	3	tentative solution for task #2 (i.e., 3000)
3	4	decide the size of graduating class and corresponding constraint
3	5	discard the constraint set for design task #3 (i.e., that's too large)
3	6	tentative solution for task #3 (i.e., 500)
1	8	new constraint for task #1 (i.e., 1500 kids need counselling) based on the solution of task #2
1	9	tentative solution for task #1 (i.e., 5 counsellors)
1	10	solution for task #1 (i.e., 7 people work in the office)

Observation 2

In the same protocol segment, we can also identify several levels of problem solving-related abstraction. For example, design task #1 is at the level of finding a solution to the problem, whereas task #3 is at the level of evaluating the criteria chosen for reaching the solution. Hence, the protocols demonstrated that the designer kept moving from one level of abstraction to another. These observations are similar to previous findings by Guindon et al. [1987a,1978b]. However, our observations are based on both experienced and inexperienced designers as compared to their study, in which they used only experienced designers.

Observation 3

In terms of problem domain-related abstractions, the designers also demonstrated frequent movement among ideas at different levels of abstraction. Two graphical examples of this movement were shown in Figure 3.1. We can observe that the subjects not only moved from one facet to another facet at the same abstraction level, but they also moved from one idea to another idea belonging to different abstraction levels. These movements were performed without previous planning, thus reflecting the natural thinking process of these subjects.

Observation 4

It was observed that how the designer perceives the application problem determines the focus of the design. For example, one subject focused on the physical layout of the office whereas another subject focused on the personnel aspect

of the problem. These perceptions sometimes limited a designer's ability in looking at the design problem from a broader view. The implication in developing software support is that some support should be provided for broadening the possible design directions of the problem.

In this segment, four observations that were common to both the pencil and paper designers and the Excelerator users were summarized. These observations conformed with the general characteristics of upstream design that we reviewed in section 2.2. In the following segment, we will concentrate on observations that are unique to the Excelerator users.

(2) Observations Summarized from Excelerator Users

Observation 1

All subjects were concerned about the presentation of their design. It appeared that the graphics capability of the supporting tool was very important.

Observation 2

Of the six subjects interviewed, four subjects used pencil and paper for initial design before using Excelerator. Only two subjects designed directly on the screen. Design behaviour of these two groups, i.e., all of the subjects using the software, is summarized respectively below.

(a) Subjects who used pencil and paper first, then transformed their design to the computer:

i) Subjects seemed to spend a relatively small portion of the exercise time on design tasks. Most of the time was spent inputting the design to the computer.

ii) Once the design was created on paper, subjects rarely changed their paper design. In addition, little change or modification was made on the screen.

(b) Subjects who started design directly on screen:

i) These subjects designed interactively with the computer. They tended to start with one idea and let more ideas develop as the design became more concrete and complete.

ii) The design approach was dominated by the structure of the design tool provided. For example, Excelerator implicitly imposes a top-down approach to the users.

iii) The subjects tended to complete one problem facet first (e.g., the functions of the career counselling office). They usually identified other facets after they had worked deeper into the details of another particular facet. Therefore, their development of vertical hierarchies dominated the development of horizontal dimensions. This phenomenon may be attributed both to the lack of domain knowledge and to the structure of the Excelerator interface.

(3) Desirable Design Tool Features Suggested by Excelerator Users:

Following features were suggested by the Excelerator users:

i) Easy manipulation of the symbols and characters on the screen.

ii) Good support for making changes (due to the nature of design activities which involve constantly changing ideas).

iii) Support for note-taking and keeping track of ideas.

vi) Freehand drawing.

v) A good selection of drawing alternatives (e.g. types of graphs and symbols).

3.2.2 Exploratory Analysis of Protocol Data

The data collected from each group were compiled and analyzed (see Appendix V). First, the protocols were coded by the experimenter at the end of each interview. After interviews of all four treatment groups were completed, the coded data were reviewed again to ensure that the coding was consistent across all groups. During the coding process, several individuals, other than the participants, were consulted for opinion in the assignment of the generated ideas to the different levels of abstraction, to minimize potential bias.

We performed a two way ANOVA (Analysis of Variance) of each data item listed in Table 3.3. The results are summarized in Appendix V.E., and discussed below.

(1) Both design experience and the availability of software support appear to have some effect on design behaviour and how abstraction is applied.

(2) Experienced designers seemed to be able to examine the problem in more detail and to keep a global view of the problem at the same time. This was demonstrated by more frequent movement between middle and low level abstraction ideas. However, the consequence is that they sometimes had to forego an incomplete task on hand by either referring to an old task or by starting a new task in order to keep that global view.

(3) The impact of software support is mainly shown through how subjects move among levels of abstraction. For example, Excelerator has a hierarchical structure. Subjects using Excelerator tended to move from higher levels to lower levels of abstraction. They also tended to complete the task on hand at the lowest abstraction level before they moved to another problem facet at higher levels of abstraction.

(4) Both experienced designers and Excelerator users tended to refer to ideas generated earlier in their designs.

(5) We found that relatively few data items that we measured were significant in our exploratory studies. For example, data collected on the movement among different levels of abstraction, and the design tasks generated, did not differ significantly among the four groups. These findings helped us to concentrate

only on the more relevant data items when we developed the abstraction measures in section 3.5.

Summary

The observations and analyses presented in the last two subsections have provided us with crucial information that can be incorporated into the design of software for design support. **First, we learned that the structure of software might have a constraining effect on design style.** Therefore, software should be designed in such a way that it does not impose a particular design style on users, unless they are inexperienced and need help. **Second, we observed that previous design experience does affect how abstraction is applied during the conceptual design process.** This behaviour may have contributed to the design characteristics that Newsome and Spillers [1989] observed with experienced designers (see Table 2.2). Therefore, if proper aids are provided for supporting the various ways of applying abstraction, it is possible that we can enhance the performance of less experienced designers so that they can perform more like expert designers.

In the next section, we will study the design behaviour of domain experts.

3.3 Phase III - Design Behaviour of Domain Experts

Objectives

It was mentioned in section 2.3 that a designer's prior knowledge in the problem domain area often affects the way a problem is abstracted. In the Phase II studies, we examined non-domain experts only, and one key observation was that previous design experience may affect how abstraction was applied during the design process. In the Phase III studies, we interviewed domain experts, i.e., high school counsellors, in the design problem we used as the basis for this phase of the study. Our goals, as outlined in Table 3.1, are 1) To observe design behaviour of domain experts, 2) To study how domain experts use abstraction, and 3) To compare design behaviour of domain experts with that of non-domain experts.

Subjects

Eight high school counsellors from four high schools consented to participate in this study. All participants had teaching and counselling experience, which ranged from five to twenty years.

Procedure

The phase III studies were similar to the phase II studies, except that only pencil and paper studies were conducted. Counsellors, interviewed individually in a private room, were asked to design a high school counselling office. During the design process, they were asked to speak aloud their thinking processes, which were tape-recorded. They were also given a pad of paper to jot down their design ideas if

they wished. The design sessions ran approximately 30 minutes on average. Only one counsellor wrote down a few preliminary ideas before discussing the proposed design. All others felt more comfortable just talking about their suggested designs.

The protocols were analyzed qualitatively. The observations were summarized into two groups: general observations and problem domain-related observations. First, we present the general observations derived from an analysis of the protocol data.

(1) General Observations

Observation 1

All domain experts started their design by stating the factors (facets) that they considered to have the most significant effect on the success of counselling office operations. For example, the location of the counselling office, its physical layout, and personnel, were the three most often mentioned facets.

Observation 2

Domain experts seemed to be able to keep an arbitrary boundary between the major problem facets, as they usually categorized particular activities under particular facets. For example, the qualifications of the staff working in the counselling office were often categorized under the personnel facet. During design, each facet was described only to the level of detail where the boundary could continue to be identified. Once the boundary became vague, the activities involved

were deemed by domain experts to be too trivial to be considered at the design stage.

Observation 3

Domain experts tended to emphasize the concepts and the principles of the counselling office during the design stage. The underlying concepts and principles were often used to justify why a particular design was suggested. Therefore, their designs tended to be more cohesive.

Observation 4

Less detail was described by the domain experts than observed in our previous study of non-domain experts. Activities were often presented in groups or categories. Examples were given instead of detailed descriptions of the activities. Domain experts demonstrated a good ability to recall facets that they had previously mentioned. They also were able to make good connections between issues currently being discussed and facets previously mentioned. It seemed that their domain experience enabled them to trace their design ideas better, as a result of an existing design schema. They appeared to exhibit an expanded memory for problem-solution information. This was outlined in Table 2.2 as one of the attributes of experienced designers. The consequence is that these domain experts were able to organize their design ideas with few hanging ideas (ideas that are not followed up or related to other ideas).

The design behaviour of domain experts indicated a fundamental difference from our observations of the non-domain expert designers in the Phase II studies. For example, non-domain experts tended to identify many different problem facets, but were unable to highlight the more important ones. However, domain experts were able to use horizontal abstraction more effectively to focus their design at certain dimensions (facets) in the initial stage of the design process. Consequently, domain experts were more likely to adopt a breadth design approach rather than a depth approach because they tended to forego details and concentrate more on the higher levels of abstraction.

Observation 2 indicated that the domain experts were able to cluster their ideas better than the non-domain experts. The non-domain experts often listed many detailed ideas or activities, but failed to group those activities into meaningful facets. This was an indication that the domain experts were able to apply vertical abstraction more effectively.

In contrast to observation 3 that domain experts often emphasized concepts and principles in their designs, the non-domain experts rarely mentioned highly abstract concepts such as objectives, goals and principles. The focus of the non-domain expert designers was mainly on the activities and functions of the counselling office. Therefore, it appeared that domain experts were more capable of dealing with the design problem at a higher level of abstraction. Also, they were more capable of using highly abstracted concepts to represent their designs. Similarly, it

can be stated that the domain experts had greater ability to deal with abstraction than the non-domain experts in this particular problem domain.

Observation 4 indicated that domain experts were able to organize their design ideas in such a way that they could account for most of the problem facets and design ideas generated. Compared to the domain experts, non-expert designs tended to be scattered across various problem facets. On many occasions, the non-experts had difficulty keeping a global view of their designs.

Table 3.5 summarizes and compares the design behaviour of the domain experts and the non-domain experts described in section 3.2. Also included in the right hand side column of Table 3.5 are our comments on how abstraction was used differently in the design process by the domain experts, based on our Table 2.4 summary of the different types of abstraction.

(2) Problem Domain-Related Observations

Observation 1

All domain experts emphasized the importance of location and physical layout of the office. In fact, most of the detailed descriptions concerned these facets. On the other hand, only one non-domain expert mentioned that the location of the office should be considered at the design stage. Although most non-domain experts were aware of the physical aspect of design, they discussed much less detail on this facet.

Table 3.5 Comparison of general design behaviour of domain experts and non-domain experts

Domain Experts	Non-domain Experts	Comment
Able to identify and state which facets are most crucial to this design problem	Unclear which facets are more important	Domain experts applied horizontal abstraction more effectively
Demonstrate good ability in clustering ideas into high level problem facets	Often list many ideas, But failed to group these ideas into meaningful problem facets	Domain experts applied vertical abstraction more effectively
Tended to emphasize the high level concepts and principles of the design problem and use them to guide the entire design	Lack of goals and objectives to guide the design	Domain experts generated more cohesive designs
Their domain knowledge appeared to have been internally organized into a design schema	Lack of design schema	Domain experts were able to organize their design ideas with fewer hanging ideas

Observation 2

Many more details of personnel issues were discussed by the domain experts. Examples include counsellor personality, attitude toward students, competence, available time, and gender issues. However, the non-experts only touched on personnel issues in general. This indicated that expert designs tended to be people-driven, while the non-expert design tended to be activity-driven.

Observation 3

Much detail of the physical layout was discussed by the domain experts. Examples were lighting, office colour, carpeting, sound-proofing, windows, glass panels, browsing areas, computer areas, round tables, meeting rooms and so on. These details evolved from their experience in working in different environments. It was stressed by many experts that the physical layout and the physical conditions of the office often dictated how the counselling office could be operated and whether the operation would be successful.

Observation 4

Computer software supporting students in making career decisions was strongly emphasized by all the domain experts, although most of the experts thought the software packages available to the students were not used as much as they could be. Computer support in the operation of the counselling office also was discussed by most of the non-experts. However, their emphasis was on supporting the functions of the office staff. Student access to computer databases was rarely mentioned by non-experts. The computer support aspect seemed to be approached from different viewpoints by the experts and non-experts. In addition, domain experts seemed to design the problem from the viewpoint of a service-provider, whereas the non-domain experts approached the problem from the viewpoint of a service-user. This is not too surprising in retrospect, because this reflected the past experience of both classes of subjects.

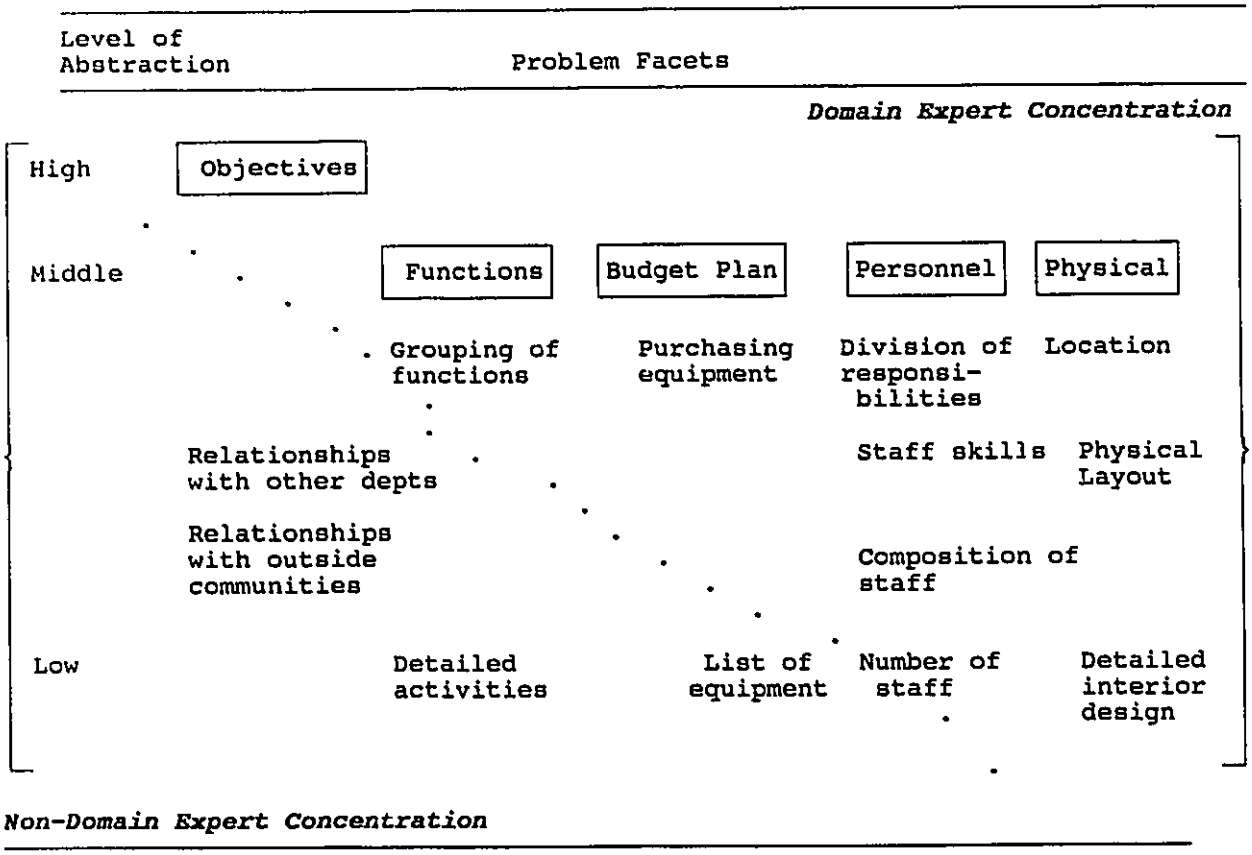
Observation 5

The designs developed by all the subjects can be arbitrarily defined to include four high level abstraction facets: objectives and principles, budget planning, personnel aspects, and physical aspects. Each of these high level abstraction facets will have several corresponding middle level abstraction facets, which will in turn have several corresponding low level abstraction facets. A general structure summarized from the protocols of both Phase II and Phase III studies is shown in Figure 3.2.

The design pattern of domain experts and that of non-domain experts can be differentiated by a diagonal line, shown by the dotted line in Figure 3.2, from the top-left corner to the lower-right corner, thus dividing the problem structure space into two areas. The domain experts tended to concentrate their design on facets in the top-right triangular area, which includes mostly the high and middle level abstraction facets. All domain experts stated that the four high level abstraction facets were important aspects to be considered in the design stage. A fair amount of design on the middle level of abstraction was also discussed by domain experts. Only facets of extreme importance to them were elaborated in detail. The domain experts seemed to be able to maintain a better view of the boundaries between facets, and also to keep a better global view of the problem. On the other hand, the non-domain experts tended to concentrate on the facets in the lower left triangular

area. Many activities were suggested in their design, without a global view of how those activities could fit into the entire design. The middle abstraction level facets often were discussed in general without either being extended in detail or being related to the higher level abstraction facets.

Figure 3.2 Design Shown in Terms of Problem Facets and Levels of Abstraction



Summary

The study of domain expert design behaviour was presented in this section. Design behaviour comparisons between domain experts and the non-domain experts (Table 3.5) indicated that domain experts appeared to have applied abstraction techniques more effectively than non-domain experts. Problem domain-related observations also indicated that domain knowledge (which appears to be associated with an internalized schema) helps to guide the design process, as shown in Figure 3.2.

Another interesting finding is that it appeared that previous design experience did not have as much influence on the design schema as domain knowledge. In other words, only domain experts demonstrated a superior ability in all three aspects of abstraction, while experienced designers did not. Therefore, we contend that it is logical to establish a set of standards based on the domain experts' design patterns, and to use these standards to develop software support for the use of abstraction and to evaluate its effectiveness.

In the next section, we will propose a strategy which addresses how to support the use of abstraction, based on the findings in our protocol studies. We also derived from these findings certain measurements that can be used to quantify the use of abstraction and to evaluate how effectively abstraction has been applied by examining the quality of a design. The proposed measurements are discussed in section 3.5.

3.4 Abstraction Support Aids

The protocol analyses confirmed our belief that there exists a cause-effect relationship between the use of abstraction and the general characteristics of conceptual design. In addition, it is not only possible to study the use of abstraction in the design process, but through an understanding of this process we may be able to develop methods to support the use of abstraction.

A comparison of the design approaches between the domain experts and non-domain experts indicated that their approaches differ in three major dimensions. The three dimensions are the **completeness** of the design, the designer's ability to deal with **high level abstraction concepts** and the **organization of design ideas.** Completeness of the design is measured by the designer's ability to identify possible problem facets. Usually the more experienced designers and the domain experts tended to generate more problem facets by applying horizontal abstraction during their design. Hence, their designs tended to be more complete. Similarly, more high level abstraction concepts or design ideas were generated by these individuals. As for the organization of ideas, experienced designers and domain experts tended to follow up their ideas better, resulting in designs with better organization. These three dimensions have been repeatedly identified in our reviews of design behaviour and in our protocol studies (see Table 3.5). We therefore argue that these three dimensions are good representations of design aspects that are the results of the use of abstraction, and measurements on these dimensions can be useful in evaluating the

quality of the design with respect to these three dimensions. Our goal is therefore to support abstraction by developing methods that will enhance the performance of designers with respect to these three dimensions, no matter what their level of design or domain experience.

We summarized three abstraction types and their respective support techniques in Table 2.4. We maintain that these abstraction support techniques are relevant to all three dimensions outlined above. More specifically, the completeness of the design will be supported by horizontal abstraction aids, the generation of high level abstraction concepts will be enhanced by vertical abstraction aids, and the organization of design ideas will be facilitated by general abstraction support. Note that each abstraction aid is particularly useful for a certain dimension, but it is not exclusive to the support of only that particular dimension. In Table 3.6, we have summarized our recommended abstraction support, based on the research we have done. These abstraction aids are incorporated into our design support software prototype described in Chapter 4. The effectiveness of these aids are tested in an experimental study in Chapter 5.

Table 3.6 Recommended Abstraction Support

Dimension of Design	Abstraction Aid
Completeness	Horizontal abstraction aid -suggestion of horizontal facets
High Level Abstraction Concepts	Vertical abstraction aid - suggestion of possible level of abstraction - assist designers in identifying which level of abstraction is being worked at - support clustering of details into abstract concepts
Organization of Design Ideas	General Abstraction Aid - possible design paths - support movement among levels of abstraction

3.5 Abstraction Measures

Measurements developed to quantitatively study the use of abstraction are described in this subsection. These measurements were selected to meet three criteria. The first criterion is that they ought to represent the results of using abstraction during the design process. For example, generating highly abstracted design ideas is a direct consequence of applying a vertical abstraction technique. Therefore, the number of highly abstracted ideas generated is a representation of the result of an abstraction activity. The second criterion is that the measurements should be related to the three dimensions that we have identified in Table 3.6. I.e., the measurements should be relevant to the completeness of the design, high level abstraction of the design, and the organization of the design. The third criterion is

that the measurements should be useful in evaluating whether the quality of design differs when the recommended support is available to the designer.

Three measurements that are most relevant to these dimensions are as follows: a) to measure completeness, we tabulate the total number of problem facets identified in the design content, b) the number of high level abstraction ideas generated in the design is used to represent the high level abstraction dimension, and c) the total number of ideas being followed up is tabulated from the design content and used to measure the organization of the design. Higher values of a particular measurement indicate a better quality in that particular design dimension. These measurements are summarized in Table 3.7.

Table 3.7 Measurements of Abstraction Use

Dimension of Completed Design	Measurements
Completeness	Total number of problem facets identified
High Level Abstraction	Number of ideas generated at high level of abstraction
Organization	Number of design ideas being followed up

The open nature of any design problem disallows setting an absolute value for any of the measurements established above. For example, we can not claim a design with five problem facets is complete, or that a design with four problem facets as incomplete. All measurements are relative. Using the design pattern of domain experts as a general standard, a design with more problem facets identified than in

another design is considered to be superior with respect to its completeness. The same argument can be applied to the other two dimensions, high level abstraction and organization, that are being measured.

In validating our measures, we note that Straub and Carlson [1989] pointed out that the validation of a developed instrument needs to be examined from three aspects. The three aspects are content validity, construct validity, and the reliability of the measures. Content validity questions whether the proposed measures were drawn from all possible measures of the properties under investigation [Straub & Carlson, 1989]. Our proposed measures relied only partially on the results of the exploratory statistical analysis of the Phase II study, in which twenty one measures were tested and only one measure was kept, but more heavily on our observations of the design performance of domain experts. As indicated by Flynn et al. [Flynn et al., 1990], content validity can be improved through empirical studies that lead to evolving knowledge, and we believe our studies have moved substantially in this direction. Additional measures may be developed as more studies in the use of abstraction in design processes are conducted in the future.

Construct validity challenges whether the measures show stability across methodologies [Straub & Carlson, 1989]. We were unable, due to the nature of this study, to conduct repeated tests on the same subjects, to evaluate the influence of different design environments on the same subject. In the remaining of the studies, we did use subjects with similar backgrounds to compare design behaviour.

Nevertheless, no significant difference in the total number of problem facets generated by the pencil and paper designers and the software users shown in the exploratory statistical analysis of our Phase II study, may be an indication of construct validity of the measure.

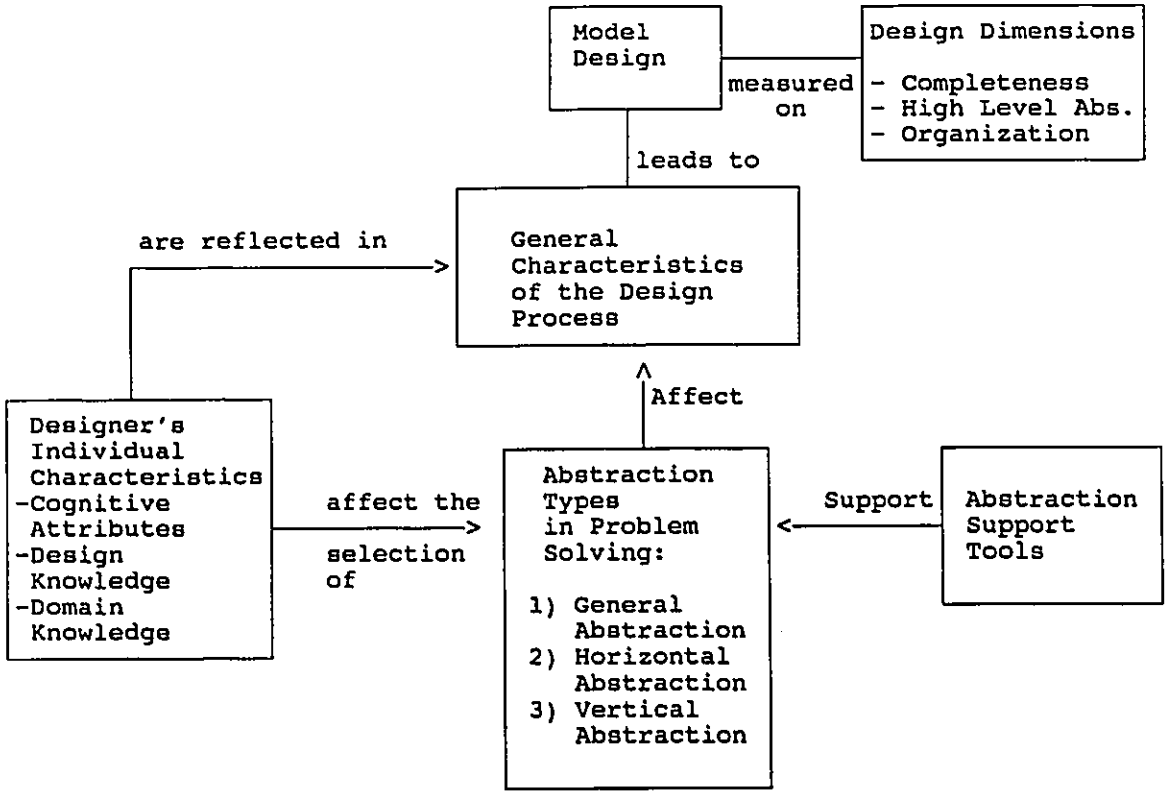
Reliability deals with the accuracy/consistency of the responses [Straub & Carlson, 1989]. The open-ended nature of our design problem, as well as the measures we proposed to capture the spontaneous thinking processes of the designers, make reliability a less relevant issue.

Summary

In this chapter, we presented the procedures and the findings of the pre-prototype empirical studies. We discussed in this chapter various abstraction aids, and measurements to measure the effectiveness of proposed aids. We have incorporated these findings into a theoretical framework that demonstrates the relationship between the use of abstraction and its effect on the upstream design process, and the output of the design process. This theoretical framework is an extension of Figure 2.4, and is shown in Figure 3.3.

Figure 3.3 indicates that each designer may take different abstracting approaches in conceptual model design. The more experienced designers or domain experts will tend to apply abstraction techniques more effectively, resulting in designs that are superior in their completeness, the presence of high level abstraction concepts, and in better organization.

Figure 3.3 Theoretical Framework of Abstraction and Upstream Design



At this point, we have accomplished the first and the second research goals by conducting the pre-prototype studies of the upstream design process and how abstraction is used in that process. We have also achieved the third and the fourth goals of this study. I.e., the development of a theoretical framework that demonstrates the relationship between the use of abstraction and the conceptual design process, and measures that reflect the use of abstraction in design.

We believe that user application of abstraction techniques can be enhanced by abstraction support that facilitates a designer's ability to deal with horizontal, vertical and general abstraction. In Chapter 4, we will describe the procedure of developing and testing a software prototype that incorporates our recommended abstraction aids.

Chapter Four

SOFTWARE PROTOTYPE FOR DESIGN SUPPORT

A software prototype that supports conceptual model design, with an emphasis on supporting the use of abstraction, was developed to fullfill our fifth research goal. The purposes of developing this prototype were twofold. The first purpose was to implement the recommended methods (see section 2.4) in a computer-supported design environment. The second purpose was to facilitate the evaluation of the effectiveness of these methods in a software-supported design environment, based on the theoretical framework that we developed in the previous chapter (see Figure 3.3). The prototype development and testing processes are documented in section 4.1. Specifically, the principles for developing the prototype, the basic structure of the prototype, and the four different versions of the prototype are described in subsections 4.1.1, 4.1.2 and 4.1.3 respectively. The prototype testing procedure is documented in section 4.2.1. During testing, a pilot study was conducted to observe the design behaviour of prototype users. Observations from this study are discussed in section 4.2.2. A sample data set was collected in the pilot study for exploratory analysis.

4.1 Prototype Development

4.1.1 Prototype Design Principles

As stated above, the goals of developing this prototype were to incorporate the recommended abstraction aids into a computer-supported design environment and to study the effectiveness of such aids. It was essential that the design environment supported by the prototype be as close as possible to a pencil-and-paper design environment for two reasons. First, it would be obstructive to include unnecessary features that may distort our observation of the use of abstraction. Second, by providing users with a natural design environment, with as little intervention built into the design environment as possible, we can make observations of the prototype users' design behaviour that is comparable to those of the pencil and paper designers who are supported with similar but non-computer-supported aids. Therefore, the fundamental principle guiding the development of this prototype was to maintain a balance between providing the users with as much abstraction support as possible and at the same time to simulate an unconstraining paper and pencil design environment.

Adhering to the above fundamental principles, as well as using input from the comments and observations made in the pre-prototype empirical studies, the prototype was developed around a set of basic guidelines, which are summarized in Table 4.1 with the corresponding observations and comments (shown in the right-hand side column in the table) that were described in chapter 3. Indicated in the

middle column of the table are how each guideline can be useful in supporting the three abstraction types that we identified in Table 2.4.

Table 4.1 Prototype Design Guideline

Design Guideline	Abstraction Types/ Techniques	Comments/Observations
(1) Window application	Vertical abstraction aid/ Clustering	Phase II: 3(i)- easy manipulation 3(ii)- support for making changes
(2) Blank screen/ reference number	Horizontal abstraction/ General abstraction/ Reference	Basic principle: - similar to pencil and paper environment Phase II: - support for keeping track of ideas
(3) Minimum intervention		Basic principle: - similar to pencil and paper environment Phase II: (b)(ii)- to provide maximum flexibility
(4) User friendly		Basic principle
(5) Screen usage	General abstraction/ Highlighting	Basic principle
(6) Use of examples/ analogy	General abstraction, Horizontal abstraction, Vertical abstraction/ examples, analogy	

Following is a detailed description of each guideline:

(1) **The prototype was developed as a windowed application.**

The prototype runs under Microsoft Windows software so that all the basic window manipulation and text editing features are readily available to the users. Observations from our empirical studies indicated that designers tend to move back and forth between abstraction levels, as well as to constantly manipulate the design by adding, deleting and moving parts of it. KnowledgePro(Windows), a software package that provides a window applications development environment, was chosen for developing the prototype. The prototype users had access to built-in text editing functions. The tiered-window structure supported users in clustering their ideas and summarizing into higher level concepts, and it assisted users to expand high level concepts into more detail. The window feature therefore provided mainly vertical abstraction support.

(2) **Blank screens were provided to the users for generating their designs. Numeric values were inserted in the design screens for easy reference.**

Blank screens tend to mimic a pencil-and-paper design environment. Numeric values were inserted so that it was easier for users to relate their design ideas when they moved from window to window. Referring to Table 2.4, this is an application of the reference technique that supports both vertical abstraction and movements among levels of abstraction. The numbering system is especially useful

for supporting the organization of the design ideas generated.

(3) **Minimum intervention was used for maximum design flexibility.**

Only an implicit structure is provided to facilitate the use of abstraction. By suggesting to the users various ways of organizing their ideas, we indirectly suggested different ways of applying abstraction. However, users made their own decisions as to which structure to use and whether or not to use any proposed structure.

(4) **The prototype should be user-friendly.**

User friendliness was achieved by (a) keeping the structure of the prototype simple and self-explanatory, and (b) simplifying the required key strokes (other than typing in the text for the actual design) as much as possible. For example, advancing from one screen to the next screen in the prototype only required clicking the mouse at a [Continue] box at the bottom of the screen. To make the prototype self-explanatory, all the instruction windows were displayed simultaneously with their associated design windows, to explain alternative actions that the users could choose at that point. In addition, two demonstrations with examples were developed to show the users the structure of the prototype and to train them in its use.

(5) **Make the most efficient use of the limited screen space but at the same time restrict the amount of information presented on the screen.**

In a paper-and-pencil design environment, a designer can generally have all of the design ideas easily accessible to him/her at any stage of the design. However, when design is performed on the screen, the users are limited by the screen size, so only limited information can be displayed at one time. The strategy used to reduce this problem was to use different windows for different stages of the design. Only the currently used window and the relevant instruction window(s) were displayed in full. The remaining windows were inactivated and only shown partially. This is an application of the highlighting method, to assist users in concentrating on the immediate design tasks.

(6) **Use of Examples and Analogies**

Examples were built into the prototype training session. An example similar to the design problem (i.e., the design of the high school counselling office) was chosen so that users could draw analogies from the examples. Effective use of examples and analogy is a technique reviewed earlier in Chapter 2. In addition, different ways of organizing the ideas (general abstraction support), possible problem facets (horizontal support) and top-down/bottom-up design strategies (vertical abstraction) were illustrated in the examples.

4.1.2 Basic Structure of the Prototype

The prototype was designed with an implicit tree structure. Initially, the users were provided with a preliminary design window (see Figure 4.1). Ideas related to the design problem were entered into the preliminary design window by the users. They were then given the option to branch from the preliminary design window to either a summarization window (moving up in abstraction level) or a detail-elaboration window (moving down in abstraction level). Two levels of summarization and two levels of detail elaboration were built into this prototype. The five-tiered structure is shown in Figure 4.1. A partial screen display is shown in Figure 4.2. The user could use any or all of these windows, as desired.

The tiered structure provides an implicit abstraction mechanism to facilitate the user's design process. Giving users the freedom to choose when and how to branch imposed minimum intervention on their natural thinking patterns. This structure was designed to accommodate design approaches such as top-down versus bottom-up as well as broad versus deep approaches. In other words the preliminary design window, in which all the users start, can be perceived by them as either at the bottom, the top or the middle level of their design. If a user chose not to branch at all, he/she could stay in the preliminary design window and complete the entire design in that area.

Figure 4.1 Basic Window Structure of the Prototype

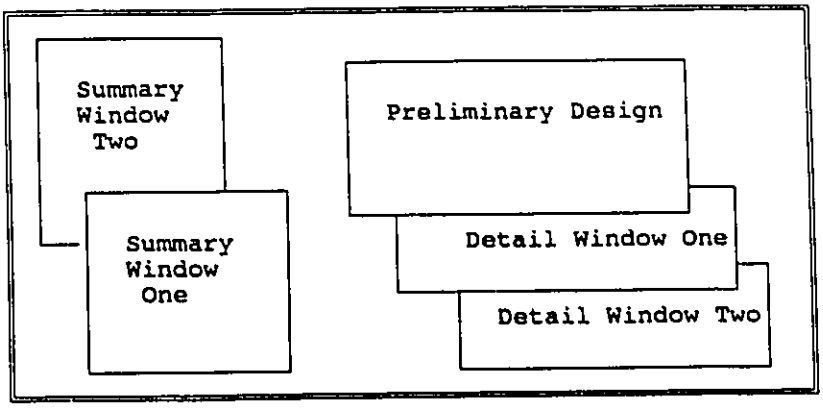
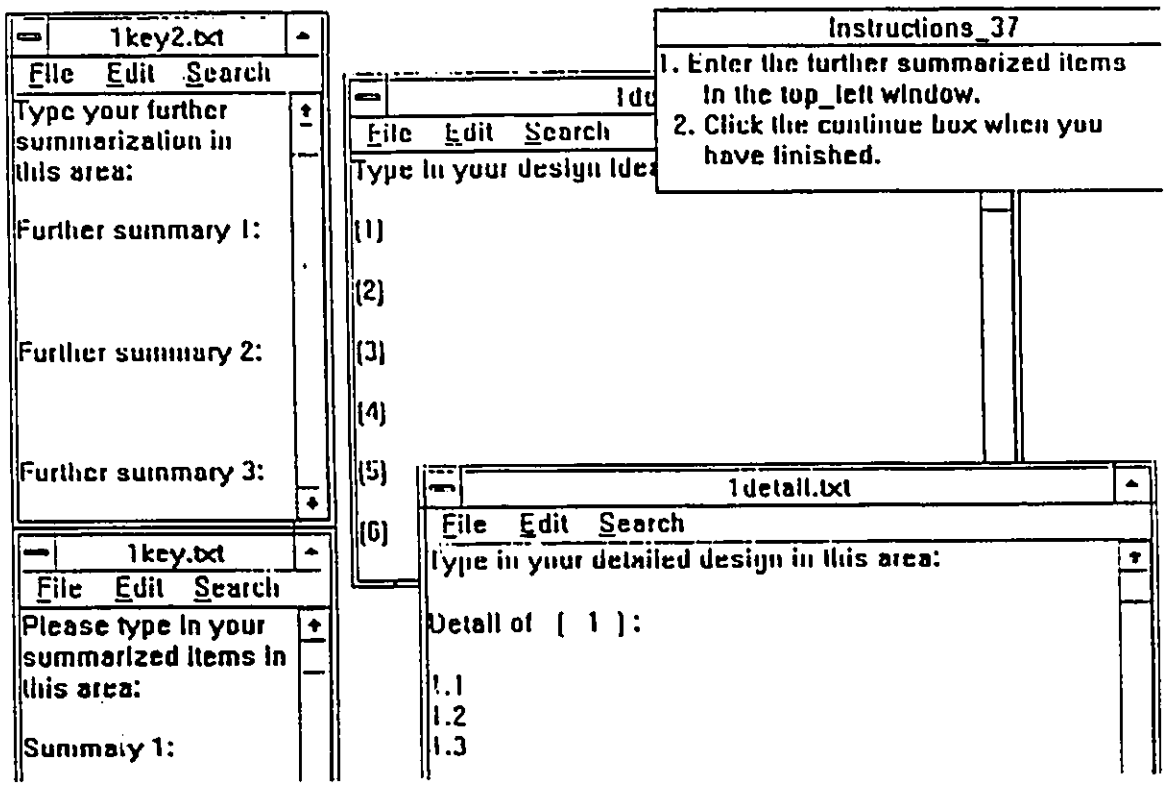


Figure 4.2 Partial Screen Display of the Design Windows



The prototype included four modules. The first two modules were the demonstration/training modules, with built-in examples. The third module provided a basic design environment where the users performed the actual design exercise. Both demonstration modules and the actual design exercise module had the same five-tiered structure shown in Figure 4.1. The fourth module was developed using hypertext techniques, to provide an additional window to display theoretical information on design techniques and more detailed explanation of the problem domain. The four modules are summarized in Table 4.2 below.

Table 4.2 Modules of the Prototype

Module	Name	Description
Module 1	Demo One	An example of designing an information booth for a shopping mall
Module 2	Demo Two	An example of designing an introductory computer course for first year university students
Module 3	Design	A shell for inputting actual design
Module 4	Infobox	An information window displaying additional information on design techniques and problem domains

The purposes of the demonstration modules were threefold. The first objective was to use them as training tools so that potential users could learn the structure of the prototype and to have some hands-on experience with the software before the actual design exercise. The second objective was to provide the users with examples illustrating the general format of a conceptual design when they used

this software. The third objective was to use the demonstration modules as vehicles to communicate a pre-determined amount of information to the users. The example chosen for the first demonstration module was the design of an information booth in a shopping mall. The nature of this example is similar to that of the actual design problem, in which the users are asked to design a high school counselling office. This is a direct application of the abstraction technique of analogy. I.e., users are provided with a similar scenario with sample solutions, so that they could draw analogies from it. The example used in the second demonstration module was the design of an introductory computer course for first year college students. The top down approach was adopted for the first example and the bottom up approach was adopted for the second example, to demonstrate both approaches to the users. Further explanations of these examples are included in the following sections.

4.1.3 Four Versions of the Prototype

Four different versions of the prototype were created during the development process. Table 4.3 includes a summary of the modules of each version of the prototype. Versions 1, 2 and 4 all include two demonstration modules and the design module, while version 3 includes two demonstration modules, the design module and the information module.

Table 4.3 Modules of the Four Versions of the Prototype

Version #	Demo One	Demo Two	Design Module
1	Limited training exercise	Limited training exercise	Basic 5-tiered design environment
2	Extended training exercise	Extended training exercise	Basic 5-tiered design environment
3	Extended training exercise	Extended training exercise	Basic 5-tiered design environment + Information Module
4	Comprehensive training exercise	Comprehensive training exercise	Basic 5-tiered design environment

As shown in Table 4.3, the demonstration modules in each version are characterized as limited, extended or comprehensive training exercises, based on their content. In a limited training exercise, as shown in Figure 4.1, only one or two design ideas were shown in each design window. The intent was to reveal as little information as possible both in terms of how to structure a conceptual design and what might be suitable design ideas.

More detailed design ideas were included in the extended training exercises. However, the design technique used was not explicitly communicated to the users in the examples of the extended training exercises. The purpose was to observe whether the users were capable of learning, from the more detailed examples, to

Figure 4.1 Design Ideas Shown in the Limited Training Exercise of Demo One

Window	Design Ideas Displayed
Preliminary Design Window	1. In this design, I will discuss the layout of an information booth and the services to be provided by an information booth. 2. Maybe I should also decide how many information booths there should be in a shopping mall.
Detailed Design Window - One	1.1 The layout of an information booth should adopt an open concept. That is, shoppers can approach the booth from any direction.
Detailed Design Window - Two	Further detail from 1.1: - A kiosk with big display signs that can be seen from all directions. - Bright colors to attract shoppers' attention.
Summary Window	1. - Layout - Services - Number of booths

generate more design ideas as well as to adopt the implicit design techniques applied in the examples. Our prediction was that users who were trained with the extended training exercises would draw analogies from the examples and thus design in a manner that approached more closely that of the domain experts than the users supported only by the limited training exercises (see hypotheses 1, 2 and 3 in section 4.3). The content of an extended training exercise of Demo One is shown in Figure 4.2.

Figure 4.2 **Design Ideas Shown in the Extended Training Exercise of Demo One**

Window	Design Ideas Displayed
Preliminary Design Window	1. It is important to know what are the objectives and goals of an information booth before we start the design. 2. For example, the objective is to provide directions and other services to the shoppers in a friendly manner. 3. Also, it would be helpful to find out more background information. For example, knowing the size of the shopping mall will be helpful when we design the information booth. 4. Since we don't have a lot of background information, let's make a few assumptions before we start the actual design: - The shopping mall is relatively small so that one information booth will be enough. - Only conceptual design is involved in this design, no physical design is involved.
Detailed Design Window - One	1.1 Sometimes the objectives are clearly identified by the shopping mall administrators. Sometimes the objectives are set by the designers. 2.1 Services to be provided might include: - directions - shopping cart rental - lottery tickets sales 2.2 Personnel - friendly personality - part-time with flexible hours - both male and female staff 2.3 Location - a cental location near the entrance - high traffic location - spacious surroundings 2.4 Appearance
Detailed Design Window - Two	Further detail from 2.4: - Bright colors and big signs usually help in attraction shoppers' attention
Summary Window	1. Determine the goals and objectives of the information booth.

As discussed in the previous section, an information module was developed, for use in conjunction with the extended training exercises. In the information module, various design techniques and abstraction techniques as well as the problem domain (i.e., the high school counselling office) were described. Users could retrieve the information at any time during their design exercise. Examples of the information module screen displays are included in appendix VI. The information module was designed with the expectation that users would be able to use the theoretical design information, compare it to the extended examples in the extended training exercises, and then apply the techniques in their own design. However, it was found in our pilot study that the information box was not an effective design support tool. One reason was that the abstract theoretical descriptions of design techniques were difficult for the users to comprehend and apply, in the absence of a real application. This is in accord with the findings on student learning and understanding by Needham and Begg [1991]. The second reason was that the retrieval of information and the operation of the information box is different from that of other design windows in the prototype. The additional learning effort required appeared to become a barrier for the users to use the information box effectively. More discussion of this issue is provided in section 4.2, in which the results of the pilot study are presented.

Prototype version #4 with the comprehensive training exercises was developed after the pilot study showed that version #3 was not effective. Although

the information module did not seem to be successful, we believed that the additional design information still could be useful to the users if the information were provided in a different format. Therefore, we developed version #4 with comprehensive training exercises that combined both the extended training exercises and the information module. I.e, the design techniques used in the demos were explicitly explained to the users. For instance, as shown in Figure 4.3, it was indicated to the users that the top-down approach was used in the design for Demo One and the bottom-up approach was used for Demo Two. By incorporating the design information into the training exercises, we managed to eliminate the hypertext information box, thus eliminating the need for the users to learn two different window manipulation techniques in a short training session.

The intent of version #4 was to observe whether the design performance of the users was affected by the explicit hints of design techniques incorporated in the comprehensive training exercises. Our hypothesis was that users who were provided with the comprehensive training exercises would benefit from the explicit application of design techniques, and thus would develop designs that were superior in the three abstraction dimensions, relative to those developed by users who were only provided with limited training exercises.

Figure 4.3 Design Ideas Shown in the Comprehensive Training Exercise of Demo One

Window	Design Ideas Displayed
Preliminary Design Window	1. A top-down approach is used for this example. First, the objectives and goals of an information booth are identified. Then some of the major areas related to this design problem are listed. The list will be used to guide further detailed design. 2. Generally, I would say that the objective of an information booth is to provide directional information and to provide other services to the shoppers in a friendly manner. 3. Some of the important issues surrounding the design of an information booth may include: - location - staff - services 4. Sometimes it is necessary to make a few assumptions before starting the actual design: - The shopping mall is relatively small so that one information booth is enough - Only conceptual design is involved in this design, no physical design is involved.
Detailed Design Window - One	1.1 Sometimes the objectives are clearly identified by the shopping mall administrators. Sometimes the objectives are set by the designers. In this case, the objectives are determined based on personal experience and common sense. 2.1 Services to be provided might include: - directions - shopping cart rental - lottery tickets sales 3.1 The booth should be located in a cental location inside the mall. 3.2 Staff - friendly personality - part-time with flexible hours - both male and female staff

Figure 4.3 (continued)

Detailed Design Window - Two	Further detail from 2.1: - A map of the mall would be helpful for the staff to explain the location of the stores and various facilities. - Big signs to attract shoppers and to indicate the services provided. Further detail from 3.1: - A location that has high traffic flow. - Maybe the booth should also be close to other public facilities such as public phones and washrooms. Further detail from 3.2: - Staff should be properly trained so that they can perform their duties efficiently.
Summary Window	1. A top-down design approach is used. 2. The major issues are location, staff and services.

4.2 Prototype Testing and the Pilot Study

The prototype was informally tested and revised several times. User input was used to modify the structure of the software, the material in the instructions and in the examples, the colours used, the sizes of the windows, and the screen layout. After the prototypes were well debugged, versions one, two and three were formally tested in a pilot study. Sample data were also collected for exploratory statistical analyses. Prototype version four is a modified version of prototype version three and was used in the actual experiment as discussed in Chapter 5. The procedures used in the pilot study and the major observations from the studies are presented in the following sections.

4.2.1 Test Procedures

Twelve subjects, including six second year business students and six second year engineering students, were observed in the pilot study. All subjects were non-domain experts. All business students had completed at least one introductory level computer course. They are therefore familiar with the basic operations of a personal computer. All engineering students had completed at least one CAD (Computer Aided Design) course, thus had some experience in designing in a software supported environment.

Subjects were randomly assigned to use prototype version one, two or three. Four subjects, including two with previous design experience and two without such experience, were assigned to each prototype. Sessions were conducted in a private room where only the administrator and the participant were present. It was ensured that there was no communication regarding this study among the subjects before they were tested.

Prior to the training session, the project was described briefly to each subject, who was asked to sign a consent form. It was also stated in the consent form that the participants would be rewarded upon completion of the design task. The procedures for the one hour to one and half hour exercise were:

Step 1: A brief introduction of the use of Windows (5 minutes)

Step 2: Training session - Demo One (5-10 minutes)

Step 3: Training session - Demo Two (5-10 minutes)

Step 4: Discussion. Subject's opinion of the software was recorded (5-10 minutes)

Step 5: Demonstrate the hypertext module. This step applied only when prototype version 3 was used (3-5 minutes)

Step 6: Actual design exercise (25-45 minutes)

4.2.2 Observations and Pilot Studies

Several key observations were made in the pilot study. The observations are classified into three categories. The first category contains the subject's opinions of the prototype software, and were recorded immediately after the training sessions, before the design exercise. In the second category are observations derived from design performance of the subjects. The third category includes the exploratory statistical analysis.

Category 1: Observations from the Training Sessions

Questions concerning the structure and the design of the prototype software were asked after the subject viewed both of the demonstration modules. Following is a summary of the responses to some of the questions asked.

(1) Is the software user friendly?

In general, all subjects responded that the software was easy to use and the instructions were clear and easy to follow.

(2) Should more demonstrations be shown before the actual exercise?

All except one subject thought that they understood the structure well enough to proceed to the actual exercise after viewing the two demonstrations.

(3) Are there too many windows on the screen during the training exercises?

A major concern during the development of the prototype was to display as much information on the screen as possible, but at the same time not to confuse the users with too many windows. The majority of the subjects thought that the number of windows on the screen at time was not too many or that they caused confusion. When they were asked if they would prefer having fewer and bigger windows by reducing some of the windows to icons, only two of the twelve subjects thought that this tradeoff might be worthwhile. In general, it appeared that novice Windows users would prefer overlapping but visible windows to hidden windows that would require additional effort to retrieve. From these observations, we were convinced that a good balance in the number of windows and the sizes of the windows on the screen had been achieved.

(4) How was the hypertext information window perceived?

The general consensus of the subjects who were shown the hypertext information windows was that there were too many windows and too much manipulation was required. They also found that the theoretical information provided in the hypertext information window was not as useful as the

examples provided in the demonstration modules. As a result, only one subject referred to the information window even briefly during the actual design exercise. One subject chose not to use the information window at all and two subjects browsed through the information windows before they started the actual design, but did not use the information during their design exercises.

Based on these observations, it was decided that the hypertext information window was not as useful as expected and a different method should be used to provide the users with information on design techniques. As a result, version four of the prototype was developed. In version four, the theoretical design information was incorporated into the examples in the two demonstration modules. Therefore, both the number of windows on the screen and the required manipulation were reduced compared to that of version three (hypertext). Moreover, since version four explicitly related the design technique information to the examples in the training exercises, this allowed observations of whether the users were able to apply theoretical design techniques more effectively than the users who were not provided with this information.

Category 2: Observations of the actual design exercise

By analyzing the contents of the actual design produced by the subjects, two key observations were made, as discussed below.

Observation 1 ***The use of abstraction technique by designers can be enhanced by support built into the design environment.***

We discovered that the designs produced by the non-domain experts in our phase II studies and the subjects who used the prototype with the limited training exercises were very similar. In both cases, the design tended to remain at the middle and lower abstraction levels. The completeness of the design was also very similar in terms of number of problem facets listed. However, the subjects who had the prototype support appeared to have better control on staying at the same level of abstraction during the design process. This type of efficiency was observed through the subject's ability to work at one abstraction level as long as needed to complete work at that level before moving to a different abstraction level. In other words, subjects tended to generate a relatively complete list of design ideas at a particular window before moving to a different window. This preliminary observation indicated some evidence that the prototype encouraged the use of abstraction.

Observation 2 ***The design performance of users may differ significantly depending on the type of examples provided to them during the training process.***

There is strong evidence that the content of the resulting design differed significantly between the group provided with the limited training exercises and the group provided with the extended training exercises. In general, the group given the extended training exercises generated designs that consisted of more highly abstracted ideas. In addition, ideas at high abstraction levels were usually well elaborated, and ideas in the low abstraction levels were clustered into summary points more consistently. This kind of structure consistency did not occur in the previous pencil and paper design exercise of the non-domain experts, nor did it occur in designs by the group provided with limited training exercises. A similar kind of design feature was observed only in our previous interviews with domain experts (i.e., the high school counsellors).

In summary, the pilot study showed evidence that (a) the prototype software was easy to use and well debugged, (b) the prototype supported the use of abstraction, and (c) useful design technique knowledge can be provided to the users through training exercises, with positive effects. Based on these observations, an exploratory statistical study, presented in section 4.3, was conducted to statistically examine the effects of the prototype's support on design behaviour.

4.3 Exploratory Statistical Studies Using Prototype Versions 1,2 and 4

The objective of this experiment was to test the effects of a single factor with three levels, i.e., versions one, two and four of the prototype, on conceptual design performance. In chapter four, we established that three dimensions of design can be enhanced by abstraction aids: (1) **Completeness** - more problem facets are identified in the design, (2) **High level of abstraction** - more ideas at higher levels of abstraction are generated in the design, and (3) **Organization** - more ideas are followed up in the design. These three criteria were used to evaluate the design performance of the subjects in our experiment. Measurements on these dimensions are summarized in Table 4.8. The hypotheses, experimental design and the results are presented in this section.

Hypotheses

It was predicted that the users supported by the prototype versions with the extended and comprehensive training exercises would perform better, i.e., approach more closely to the performance of the domain expert, on all three dimensions than users supported by the prototype version with the limited examples. The hypotheses are stated as follows.

Hypothesis 1 **The completeness of the conceptual model design by the users while supported by the prototype with extended and comprehensive training exercises, will be better than by the users who are supported by the prototype version with limited training exercises.**

Hypothesis 2 **Users supported by the prototype versions with extended and comprehensive training exercises will generate more ideas at high abstraction level in their conceptual model design than the users who are supported by the prototype with limited training exercises.**

Hypothesis 3 **The organization of the conceptual model design by the users while supported by the prototype with extended and comprehensive training exercises, will be better than by users who are supported by the prototype version with limited training exercises.**

Experimental Design

A one-factor design was adopted in our experiment, to test the effectiveness of abstraction aids in three treatment groups: prototype versions with limited, extended and comprehensive training exercises (prototype versions 1, 2 and 4) respectively. Based on the pilot test, we expected that eight subjects per treatment would be sufficient to arrive at conclusions on those hypotheses.

Data collected from each of the three treatment groups was used to test the hypotheses. The Tukey test of paired comparisons was used to measure the significance of differences among the three treatments groups for each measurement.

Subjects

All participants in this experiment were non-domain experts, i.e., people who have not worked in a high school counselling office. Subjects were solicited on a voluntary basis from second year business and engineering students at the University of Windsor. Participating engineering students were required to have taken two design-related courses, with one of these course being a Computer-Aided Design (CAD) course. All participating business students had taken at least one business computing course. Eight subjects, four with previous design experience and four without previous design experience, were randomly assigned to the three treatment groups to ensure that the groups were comparable on both design experience and minimum computer experience.

Procedure

The same procedure used in the pilot study was followed. After being assigned to a treatment group, the participant was shown the two demos appropriate to that treatment, to learn how to use the prototype software. Participants performed the design exercise after the training session.

Exploratory Data Analysis

Data were collected, for exploratory statistical analysis by one way analysis of variance, from the prototype versions with the limited training exercises (treatment 1), the extended training exercises (treatment 2) and the comprehensive training exercises (treatment 3).

Hypothesis 1 predicted that participants under treatment 2 and treatment 3 would generate a more complete design than participants under treatment 1. The means of the total number of facets identified by the three groups were 3.25, 4 and 6 respectively. The F-test ($F(2,21)=8.31$, $p<0.05$) indicated a significant difference. The Tukey pairwise comparisons indicated that participants in treatment 3 generated more facets than participants in treatment 1 as well as participants in treatment 2. However, treatment 2 did not differ significantly from treatment 1 in terms of completeness.

Our results also showed that there was no significant difference between the three treatment groups in terms of the total number of ideas generated, with means of 17.75 ideas, 14 ideas and 18.75 ideas for treatments 1, 2 and 3 respectively. However, further testing showed that the distribution of these ideas at the high abstraction level was different. The F test showed that the number of ideas generated at the high abstraction level differed significantly across the three treatment groups ($F(2,21)=6.24$, $p<0.05$). The Tukey method further indicated that the group mean of treatment 3 (2.25 ideas) was significantly higher than that of treatment 1 (0.25 ideas) at the 0.05 level. This result indicated that the participants in treatment group 3 generated significantly more ideas at the high abstraction level than participants in treatment 1. In other words, we have shown that participants supported by the comprehensive training exercises developed their designs at a higher abstraction level than participants supported by the limited training exercises.

However, we did not have sufficient evidence to state that participants supported by the extended training exercises designed at a higher abstraction level than participants supported by the limited training exercises.

Hypothesis 3 predicted that subjects in treatment 2 and treatment 3 would generate designs with a better organization, by following up more design ideas. The number of ideas being followed up ($F(2,21)=2.51$, $p>0.05$) did not indicate a significant difference among the three treatments in this preliminary data analysis. A larger sample data set would allow us to further examine this hypothesis.

This exploratory statistical analysis led us to believe that the prototype version with the extended training exercises (treatment 2) did not differ significantly from either the version with limited training exercises or the version with comprehensive training exercises. Therefore, the prototype with the extended examples was excluded from our final testing of the effectiveness of the recommended abstraction aids. The complete experimental study thus only used the prototype versions with the limited training exercises and the comprehensive training exercises, along with two pencil and paper design environments, as presented in Chapter 5.

Summary

In this chapter, we documented the development and the testing processes of a software prototype that was designed to facilitate the use of abstraction in the conceptual design process. The development of this prototype was based on the theoretical framework that we proposed earlier. The recommended abstraction aids such as examples, analogy, provision of referencing, highlighting and clustering were incorporated into the prototype. More specifically, examples that would help the users to generalize (a form of analogy) were used to demonstrate the use of horizontal abstraction. Clustering technique was demonstrated to support the use of vertical abstraction. Referencing technique was used to support the use of general abstraction. Therefore, we have accomplished our fifth research goal.

Four different versions of prototype were developed. A version with a hypertext information module appeared to be unsuitable, and thus was excluded from further studies. Three versions of the prototype were used to collect data for an exploratory analysis. The three versions were: version #1 with limited training exercises, version #2 with extended training exercises, and version #4 with comprehensive training exercises. Measurements were collected from the design generated by a sample of prototype users, to test the three proposed dimensions of design that were affected by the use of abstraction. The three dimensions are the completeness of the design, the designer's ability to design at higher levels of abstraction, and the organization of the design.

The exploratory data analysis indicated the following. In terms of completeness and the ability to design at higher level of abstraction, the participants using the prototype with comprehensive training exercises generated significantly more problem facets and highly abstracted ideas then the participants supported by the prototype with limited training exercises. The results were inconclusive with respect to the organization of the design, since none of the three groups showed a significant difference in performance in terms of the number of ideas being followed up. Based on these results, the prototype version with the extended training exercises was also excluded from future studies.

The studies in this chapter demonstrated that it is feasible to develop abstraction aids and incorporate these aids into a software-supported design environment. We have also substantiated the procedure to validate our proposed framework using the measurements developed in the previous chapter. Therefore, we have achieved our fourth research goal; i.e., to develop a software prototype to demonstrate the implementation of techniques supporting the use of abstraction in model design.

In the following chapter, a complete experimental study that includes both a pencil and paper design environment and the software supported design environment is conducted to make measurements based on the proposed framework.

Chapter Five

EXPERIMENTAL STUDIES

The studies in this chapter were directed toward achieving our sixth research goal, i.e., to conduct an experimental study which evaluates the effectiveness of the proposed abstraction aids in certain design environments with respect to the three dimensions of design (completeness, ability to design at high abstraction level and organization) identified in our framework.

There are two distinct aspects of our recommended abstraction aids. One is the tiered structure that supports the organization of the design ideas generated. The second is the provision of the training exercises that illustrate problem facets and highly abstract concepts. The pilot studies in chapter 4 indicated that there were significant differences between the designs of participants supported by the prototype with limited training exercises and that of participants supported by the prototype with comprehensive training exercises, with respect to the completeness of the design and the ability to design at higher levels of abstraction. However, there was no significant difference between the results in terms of better design organization. We suspect that the reason that no significant difference was observed was that all participants were influenced by the structure that we provided in the prototype. This

is consistent with the findings of our pre-prototype studies of the Excelerator users. To further examine this phenomenon, we conducted the experiment in three different settings: 1) a pencil and paper design environment with no abstraction aid for structuring the design (i.e., the five-tiered design structure was not provided), 2) a pencil and paper design environment supported by our recommended abstraction support (i.e., the five-tiered design areas were provided on the paper), and 3) the five-tiered design environment supported by our prototype. These settings allowed us to examine a) the effect of the proposed tiered structure on the organization of design in the paper-and-pencil design environment and the software-supported design environment, and b) the effects of the limited training exercises and the comprehensive training exercises on designs created in all three design environments.

5.1 Experimental Design

The objective of this experiment was to test the effects of the recommended abstraction aids on the three dimensions of design, with respect to two different prototype versions (limited training exercises and comprehensive training exercises), in three different design environments. The following hypotheses were tested in our experiment.

Hypotheses

Hypothesis 4 (Completeness)

The completeness of conceptual model design by the participants while supported by the comprehensive training exercises will be better (more problem facets identified) than by the participants supported by the limited training exercises.

Hypothesis 5 (Higher level of Abstraction)

The participants provided with the comprehensive training exercises will design at a higher abstraction level by generating more ideas at high abstraction level in their conceptual model design than the participants supported by the limited training exercises.

Hypothesis 6 (Organization)

The structure of conceptual model design by the participants provided with a tiered structure will be better (more ideas being followed up) than that of the participants provided with no structure.

Hypothesis 4 focused on the dimension of completeness. According to our framework, it was expected that the participants supported by our abstraction aids with comprehensive training exercises would perform better, i.e., approach the

performance of the domain experts, by generating more problem facets in this dimension than the participants supported only by the limited training exercises.

The fifth hypothesis focused on ability to design at a higher level of abstraction. We expected that participants supported by the comprehensive training exercises would show superior performance by generating more higher level ideas than the participants supported by the limited training exercises, mainly by drawing analogies from the comprehensive examples.

Hypothesis six focused on the organization of the design. We believed that the organization of a design is affected by the tiered design environment that we provided to the designer. I.e., the tiered design windows and the reference numbers provided as part of the design environment will enhance the ability of designers to follow up their ideas, by being able to refer to their previous design ideas easily, and thus leaving fewer hanging ideas.

Experimental Design

We adopted a three by two analysis of variance design for this experiment. The experimental design is shown in Table 5.1. Each group included eight subjects. In this experiment, Factor A represents the design environment and Factor B represents the abstraction aids that we provided. Factor B actually represents two different levels of training: the limited training exercises, and the comprehensive training exercises incorporating our abstraction aids.

Table 5.1 Experimental Design for the Effectiveness of the Abstraction Aids in the Conceptual Modelling Process

Factor A (Design Environment)	Factor B (Training Level): Limited Training Exercises	Factor B (Training Level): Comprehensive Training Exercises
Paper and Pencil (no structure)	group 1	group 2
Pencil and Paper (structured)	group 3	group 4
Software Prototype	group 5	group 5

Measurements

Three dimensions of design, i.e., completeness, high abstraction concepts and organization of the design were measured. Measurements established in Table 3.7 were used, reproduced below for easy reference.

Table 3.7 (Duplicated) Measurements of Abstraction Use

Dimension of Design	Measurements
Completeness	Total number of problem facets identified
High Level Abstraction	Number of ideas generated at high levels of abstraction
Organization	Number of design ideas being followed up

Subjects

There were eight subjects in each treatment group. All participants in this experiment were non-domain experts. Subjects were solicited on a voluntary basis from second year business and engineering classes at the University of Windsor. Participating engineering students were required to have taken two design-related courses, with one of these course being a Computer-Aided Design (CAD) course. Participating business students should have taken at least one business computing course. Equal numbers of subjects with and without previous design experience were randomly assigned to each of the six treatment groups to ensure that the groups were comparable in both design experience and minimum computer experience. Subjects were rewarded for their participation in this study.

Procedure

A similar procedure was used for the paper and pencil design setting supported by the tiered structure. Instead of blank paper, each participant in groups 3 and 4 was given sheets with pre-marked areas for the preliminary, detail, further detail, summary and further summary design. The same reference numbers used in the software prototype were also noted on the design sheets (see Appendix VIII).

The procedure used in the pilot study was again used for the two prototype treatment groups (see subsection 4.2.1). That is, each participant in group 5 or group 6 was shown the two demos appropriate to that treatment, followed by the actual design exercise.

5.2 Data Analysis

Test of Hypothesis 4

To test hypothesis 4, a three by two analysis of variance was used to check the overall effect and the interaction effect, followed by Bonferroni pairwise comparisons to compare the differences among factor levels. The results of the statistical analysis which tested hypothesis 4 is reproduced in Table 5.2.

Table 5.2 Statistical Analysis of Problem Facets

(a) Mean Number of Problem Facets Generated

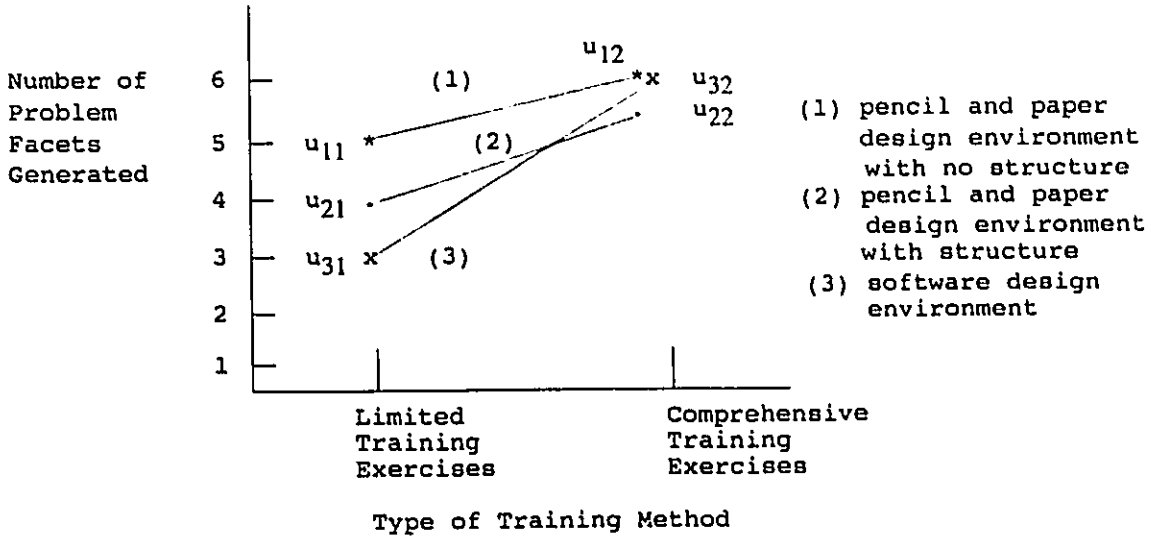
Factor A (Design Environment) i		Factor B (Training Level) j: Limited Training Exercises j=1	Factor B (Training Level) j: Comprehensive Training Exercises j=2
Pencil and Paper (no structure)	i=1	4.00 (SD=1.41)	5.25 (SD=1.28)
Pencil and Paper (structured)	i=2	5.00 (SD=1.06)	5.75 (SD= .88)
Software Prototype	i=3	3.00 (SD= .53)	5.75 (SD=1.48)

(b) ANOVA Table

Source of Variation	Sum of Squares	DF	Mean Square	F	Sig of F
Main Effects	38.750	3	12.917	9.602	.000
TRAINING	30.083	1	30.083	22.363	.000
ENVIRON	8.667	2	4.333	3.221	.050
2-Way Interactions	8.667	2	4.333	3.221	.050
TRAINING ENVIRON	8.667	2	4.333	3.221	.050
Explained	47.417	5	9.483	7.050	.000
Residual	56.500	42	1.345		
Total	103.917	47	2.211		

The ANOVA analysis showed that there was a significant interaction between the design environment and the training method used. The interaction is depicted in Figure 5.1.

Figure 5.1 Design Environment-Training Method Plot for Problem Facets Generated



To investigate the interaction effect, we examined the difference (denoted by D(k), k= 1,2,3 represents the three different design environments) of mean facets generated between the two different training methods:

$$D(1) = u_{12} - u_{11}$$
$$D(2) = u_{22} - u_{21}$$
$$D(3) = u_{32} - u_{31}$$

where u_{ij} denote the treatment means,
i=1,2,3, representing the three design environments;
j=1,2, representing the two training methods.

The Bonferroni 95% confidence intervals for the family of comparisons were:

$$-0.217 =< u_{12} - u_{11} =< 2.717$$
$$-0.717 =< u_{22} - u_{21} =< 2.217$$
$$1.283 =< u_{32} - u_{31} =< 4.217 \quad *$$

Based on these results, the following conclusions may be drawn with family confidence coefficient of 95 percent: (1) For both pencil and paper design environments, the number of problem facets generated by the participants provided by the two training methods did not differ significantly, and (2) For the prototype environment, participants provided with the comprehensive training exercises generated significantly more ideas then those participants provided with the limited training exercises (indicated by *).

The interaction effect indicated that the comprehensive training exercises was most effective in supporting the generation of problem facets in the prototype supported environment, as opposed to the two pencil and paper design environments. This effect was shown by a steeper slope of the line connecting the two treatment means of the prototype supported settings, as depicted in Figure 5.1. On the other hand, this phenomena could be explained in such a way that the generation of problem facets was hindered when only the limited training exercises were provided in the prototype setting. More study is warranted in this area.

Test of Hypothesis 5

A three by two analysis of variance was used to check the overall effect of both factors in testing hypothesis 5. The analysis of variance are summarized in Table 5.3.

Table 5.3 Statistical Analysis of High Level Ideas

(a) Mean Number of High Level Ideas Generated

Factor A (Design Environment) i		Factor B (Training Level) j: Limited Training Exercises j=1	Factor B (Training Level) j: Comprehensive Training Exercises J=2
Pencil and Paper (no structure)	i=1	0.125 (SD=0.35)	1.375 (SD=1.18)
Pencil and Paper (structured)	i=2	0.250 (SD=0.46)	1.500 (SD=0.92)
Software Prototype	i=3	0.125 (SD=0.35)	1.750 (SD=1.16)

(b) ANOVA Table

Source of Variation	Sum of Squares	DF	Mean Square	F	Sig of F
Main Effects	22.979	3	7.660	11.239	.000
TRAINING	22.687	1	22.687	33.288	.000
ENVIRON	.292	2	.146	.214	.808
2-Way Interactions	.375	2	.187	.275	.761
TRAINING ENVIRON	.375	2	.188	.275	.761
Explained	23.354	5	4.671	6.853	.000
Residual	28.625	42	.682		
Total	51.979	47	1.106		

The ANOVA result indicated that the interaction effect was not significant. Therefore, we proceed to examine the main effects. The example (i.e., the training methods) was significant ($F(1,42)=33.288$, $p<0.05$). The comparison of the

treatment level means between the groups that were given the comprehensive training exercises and the groups that were given the limited training exercises showed that the former groups had a higher average of high level ideas (i.e., u.2 > u.1 or 1.54 ideas > 0.17 ideas). Therefore, we conclude that hypothesis 5 is supported. I.e., the participants supported by the comprehensive training exercises were able to generate more higher level ideas then the participants supported by the limited training exercises.

Test of Hypothesis 6

The three by two analysis of variance, summarized in Table 5.4 indicated that the interaction effect was not significant. However, both the training methods provided and the environment provided to the participants had a significant effect on how effectively the design ideas were being followed up.

It can be concluded that (1) the comprehensive training exercises were significantly more effective in terms of supporting the participants in following up their design ideas than the limited training exercises, (2) the pencil and paper design environment supported by our structure differed significantly from the pencil and paper design environment not supported by our structure, in terms of supporting the organization of design ideas. However, further Tukey pairwise comparisons indicated that no significant difference was found between either of the pencil and paper design environments and the prototype design environment in terms of supporting the

organization of design ideas.

Table 5.4 Statistical Analysis of Ideas Being Followed-up

(a) Mean Number of Ideas Being Followed Up

Factor A (Design Environment) i		Factor B (Training Level) j: Limited Training Exercises j=1	Factor B (Training Level) j: Comprehensive Training Exercises j=2
Pencil and Paper (no structure)	i=1	7.875 (SD=7.14)	15.875 (SD=8.23)
Pencil and Paper (structured)	i=2	19.250 (SD=7.61)	22.750 (SD=6.84)
Software prototype	i=3	15.375 (SD=3.96)	18.750 (SD=5.94)

(b) ANOVA Table

Source of Variation	Sum of Squares	DF	Mean Square	F	Sig of F
Main Effects	965.313	3	321.771	7.031	.001
TRAINING	295.021	1	295.021	6.446	.015
ENVIRON	670.292	2	335.146	7.323	.002
2-Way Interactions	55.542	2	27.771	.607	.550
TRAINING ENVIRON	55.542	2	27.771	.607	.550
Explained	1020.854	5	204.171	4.461	.002
Residual	1922.125	42	45.765		
Total	2942.979	47	62.617		

For hypothesis 6 we can conclude that the organization of the design was enhanced by both the training exercises and the structure provided. Specifically, participants supported by our structure in the pencil and paper design environment were able to organize their design ideas significantly better than the participants supported by the pencil and paper design environment with no design structure support.

Summary

In this chapter, we measured the design process in an experimental setting by testing three hypotheses with respect to the three design dimensions that we identified in our framework.

In terms of the completeness of the design, we hypothesized (hypothesis 4) that the comprehensive training exercises provided in our abstraction aids would have a significant effect on the number of problem facets generated by the participants. Our findings indicated that both the training exercises and the structure that we provided had a significant effect on this dimension. The summary of treatment means indicated that there were increases in all three design environments in terms of the number of problem facets generated by the participants. However, the most significant increase was observed in the prototype design environment. We are hesitant to conclude that the provision of comprehensive training exercises was most effective in supporting participants to generate more problem facets, since the treatment mean of group 5 (i.e., the group supported by the limited training exercises in the prototype design environment) was low relative to the means of all other groups. More complete studies are needed to verify this result.

Hypothesis 5 tested the dimension of the participants' abilities to design at higher levels of abstraction. Our finding indicated that the participants supported by the comprehensive training exercises were able to generate more higher level ideas than the participants supported by the limited training exercises. Therefore,

hypothesis 5 was supported.

Hypothesis 6 measured the dimension of the organization of the design. We found that the organization of the design was enhanced by both the training exercises and the structure provided. Specifically, participants supported by our structure in the pencil and paper design environment were able to organize their design ideas significantly better than the participants supported by the pencil and paper design environment with no design structure support. Therefore, hypothesis 6 was partially supported. The organization of the prototype users' designs did not differ significantly from either of the pencil and paper design groups. A more detailed study may help to explore the possible reasons behind this result.

At this point, we have achieved all our research goals. In the next chapter, we will conclude this study by discussing the implications of our findings and propose future research related to this study.

Chapter Six

FINDINGS AND CONCLUSIONS

In Chapter 2 we pointed out that studies in many disciplines have identified extensive use of abstraction in design, but that no study has been conducted to our knowledge which examines how abstraction is used in design or to measure the impact of effective use of abstraction on the quality of design. This research was conducted to fill this gap by studying this fundamental aspect of conceptual model design. Our findings are summarized in this chapter, with respect to the six research goals outlined in section 2.4. We also state the conclusions of the study and discuss future research issues.

6.1 Findings

Our first and second goals were to study empirically upstream design behaviour and the use of abstraction in design processes. Three phases of protocol studies were conducted to achieve these goals. We studied the design processes of expert designers and non-expert designers (none were domain experts), as well as domain experts who were not design experts. We also observed the design processes of users of a commercial design software package. We found that design experience

did affect how abstraction was applied during conceptual design, which we believe is mainly due to an internalized schema, that a designer develops through design experience. Experienced designers tended to move frequently between design ideas at the middle level and the low level of abstraction, but still maintained a global view of their designs, unlike inexperienced designers who were not domain experts.

The difference in design between experienced and non-experienced designers was not substantial. However, design differences were observed when domain expert design outputs were studied. Their internalized design schema appeared to be better developed, based on their domain knowledge, resulting in designs that were superior to those of non-domain experts on three dimensions: the completeness of the design, the ability to design at higher levels of abstraction, and the organization of the design. In our literature review, we identified three types of abstraction: vertical, horizontal and general abstractions. We found that domain experts were able to effectively apply these abstraction techniques to enhance their designs on all three of these dimensions.

Based on our observations, we proposed a framework that exhibited the relationships between the use of abstraction and conceptual model design, stated as our third research goal. In our framework, we indicated that effective use of abstraction techniques would result in designs that are superior on the three design dimensions of the completeness of the design, the ability to design at higher levels of abstraction, and the organization of the design. We also proposed that proper aids would facilitate the effective use of certain abstraction techniques, thus enhancing

design performance. We developed measures of performance in the three design dimensions that could be applied by analyzing completed designs. This allowed us to achieve our fourth research goal.

Recommended abstraction aids were synthesized from published literature. These included: using examples, analogy, referencing, highlighting, and clustering. We also learned from our empirical studies that the structure of supporting software might have a constraining effect on design style. Using this phenomenon to our advantage, we recommended imposing a structure that would assist the organization of design ideas.

To fulfill our fifth research goal, we demonstrated the implementation of the proposed abstraction aids in a software prototype that supported conceptual model design. Several versions of the prototype were developed. Results of the pilot study suggested that a training demo consisting of comprehensive training exercises which incorporated descriptions of abstraction techniques had significantly enhanced designer ability to generate more problem facets and to design at higher levels of abstraction than they did with a training demo consisting of only limited training exercises. Both prototype versions were otherwise similar, and provided the users with a built-in tiered design environment.

To further test the effectiveness of the proposed abstraction aids in different design environment, and to exercise our proposed framework, we conducted an experimental study, stated as our sixth research goal. The effectiveness of two

training methods that included training exercises with different levels of detail were tested, in two pencil and paper design environments and with design environments supported by the software prototype. One pencil and paper design environment was supported by our proposed tiered structure while the other was not. Intriguing statistical results were obtained, as discussed below.

First, an interaction effect was shown between the design environments and the type of training exercises provided, with respect to the number of problem facets generated. We expected the relative changes in the number of problem facets would be similar across all three design environments, because all design environments were provided with the same examples. We suspect that the relatively high increase in the number of problem facets generated in the prototype design environment, causing the interaction effect, may be due to a hindering effect of the combination of limited training exercises and an unfamiliar computer-supported design environment. A future study of design processes that takes into account user-computer interactions is needed to examine this effect in detail.

Second, participants supported by the comprehensive training exercises were able to generate significantly more higher level ideas than the participants supported by the limited training exercises, in all design environments.

Third, the measure of design organization of the software prototype users was not significantly different from that of either of the two pencil and paper design environments. We had expected that both the prototype users and the pencil and

paper designers with the design structure we provided would outperform the pencil and paper designers without structural support. This result again suggested that it is critical to include a measure of user-computer interaction in future studies of software support for design purposes.

6.2 Conclusions

The fundamental issue addressed in this study is how abstraction is used in the conceptual model design process, and how its use can affect resulting design outputs. Our empirical observations indicated that different types of abstraction were used extensively in design processes, and that design quality could be enhanced by more effective use of abstraction. The major contributions of this study to knowledge are as follows.

First, we explored and established the relationships between the use of abstraction and the resulting conceptual designs. We found that effective use of horizontal, vertical and general abstraction has a positive impact on design completeness, design development at higher levels of abstraction, and design organization. We have also shown how to determine whether abstraction techniques were used effectively during the design process, by examining the design output. This helps to put on a more solid foundation the understanding of how abstraction can be used in conceptual model design. Our findings also have implications for software development, since they can be used to develop tools to evaluate design support

software for its effectiveness in supporting the three design dimensions we identified, or they can be used as guidelines when new design support software being developed.

The second contribution of this study is that we developed measurement techniques which can be used to evaluate the use of abstraction, so that abstraction needs no longer be regarded as an intangible concept. These measures can be applied to various problem solving situations, to assist in identifying areas in which the problem solver could improve his/her use of abstraction.

Finally, we demonstrated the usefulness of incorporating abstraction aids into conceptual model design support environments, in conjunction with other qualitative aids suggested in the DSS literature.

6.3 Future Research

In this study we attempted to explain, and measure the relationships between abstraction and conceptual model design. However, we were limited to the study of selected dimensions of this problem due to its complexity. To further our understanding of this topic, future research is needed in the following areas:

(1) We have included in our framework three individual characteristics: cognitive style, previous design experience and domain knowledge, that we considered to have a direct impact on design processes. Other individual characteristics, such as aptitude variables and personality styles, may also play important roles

in conceptual design. The development of a more comprehensive framework that includes other relevant variables would further our understanding of the use of abstraction in design.

(2) Cognitive style of our subjects was not measured in this study. We assumed that our suggested abstraction aids would be general enough to support designers with different cognitive styles. However, it would be very interesting and useful to study how people with different cognitive styles would use different abstraction techniques, and how their selection of techniques is reflected in the designs they create.

(3) Several fundamental abstraction techniques, such as examples, analogy, referencing, highlighting and clustering were suggested in this study, to support the three dimensions of design that were identified. A study of other abstraction techniques that could support these design dimensions would be useful for developing more comprehensive conceptual design support tools.

(4) This study did not deal with other complex issues related to software design and user-computer interaction. A more comprehensive empirical study that incorporates some of the implementation issues would help us to explore some of the currently unexplained phenomena that affect the use of abstraction in design.

In conclusion, the study presented in this dissertation was a first attempt to explain the relationships between the use of abstraction and conceptual model design through theoretical synthesis and empirical studies. This work provides a basis for future theoretical research that will extend the framework to include more individual variables and more abstraction techniques. Issues concerning the implementation of the abstraction techniques in design are of vital importance to MIS researchers, and more extensive efforts are needed to address these issues.

References

Adelson, B. "Problem Solving and the Development of Abstract Categories in Programing Languages," Memory and Cognition, Vol.9, No.4, 1981, pp.422-433.

Adelson, B. "when Novices Surpass Experts: The Difficulty of a Task May Increase With Expertise," Journal of Experimental Psychology: Learning, Memory and Cognition, Vol.10, No.3, 1984, pp.483-495.

Akin, Omer, "Issues in Design Support Systems for Architects," DSS Transaction, 1985.

Akin, Omer, "Exploration of the Design Process," Design Methods and Theories, Vol.13, No.3/4, 1979, pp.115-119.

Alter, S.L. Decision Support Systems, Reading, Addison-Wesley Publishing Company, 1980.

Anderson, J.R. Cognitive Psychology and Its Implications, New York: W.H.Freeman & Company, 1985.

Anderson, J.R. Cognitive Skills and Their Aquisition, W.H. Freeman & Company, Hillsdale, Lawrence Erlbaum, 1981.

Applegate, L.M., Konsynski, B.R. and Nunamaker, J.F. "Model Management Systems: Design for Decision Support," Decision Support Systems, Vol.2, 1986, pp.81-91.

Archer, N.P. and Cui, J. "An Object-Oriented Mode Base for Generic Model Management," ASAC Annual Conference, Quebec, Quabec, 1992.

Bartlett, F.C. Remembering, Cambridge University Press, 1932.

Batra, D. and Davis, J.G. "A Study of Conceptual Data Modeling in Database Design: Similarities and Differences Between Expert and Novice Designers," Proceedings of 10th International Conference on Information Systems, Jamie I. De Gross, John C. Henderson and Benn R. Konsynski (Editors). New York: ACM, 1989.

Benyon, D., Innocent, P. and Murray, D. "System Adaptivity and the Modeling of Stereotypes," In Proceedings Interact'87, Second IFIP Conference on Human-Computer Interaction, H.J. Bullinger and B. Shackel (Editors). Elsevier Science Publishers B.V., Armsterdam, 1987, pp.245-253.

Benyon, D. and Murray, D. "Experience with Adaptive Interfaces," The Computer Journal, Vol.31, No.5, 1988, pp.465-473.

Best, J.B., Cognitive Psychology, St.Paul, MN: West Publishing Co., 1986.

Binbasioglu, M. and Jarke, M. "Domain Specific DSS Tools for Knowledge-Based Model Building," Decision Support Systems, Vol.2, 1986, pp.213-223.

Blanning, R.W. "An Entity-Relationship Approach to Model Management," Decision Support Systems, Vol.2, 1986, pp.65-72.

Bonczek, R.H., Holsapple, C.W. and Whinston, A.B. "A Generalized Decision Support System Using Predicate Calculus and Network Data Base," Operations Research, Vol.29, No.2, 1981, pp.263-281.

Bond, A.H. and Soetarman, B. "Multiple Abstraction in Knowledge-Based Simulation," in Artificial Intelligence And Simulation: The Diversity of Applications, Troy Henson (Ed.), San Diego, CA: Society for Computers Simulation, 1988. pp.61-66.

Broadbent, G. "Design Theory," Design Methods and Theories, Vol.13, No.3/4, 1979, pp.103-107.

Byrne, R. "Protocol Analysis in Problem Solving," in Thinking and Reasoning, J.St.B.T. Evans (Ed.). Routledge & Kegan Paul Publishing, London, England, 1983.

Carrier, C.A. and Jonassen, D.H. "Adapting Courseware to Accommodate Individual Differences," in David H. Jonassen (Ed.) Instructional Designs for Microcomputer Courseware, Hillsdale, N.J.:Lawrence Erlbaum Assoc., pp.203-226. 1987.

Carroll, M., Thomas, J.C., and Malhotra, A., "Presentation and Representation in Design Problem-solving," British Journal of Psychology, Vol.71, 1980, pp.143-153.

Chase, W., and Simon, H. "Perception in Chess," Cognitive Psychology, No.4, 1973.

Cross, N. "Understanding Design: The Lessons of Design Methodology," Design Methods and Theories, Vol.20, No.2, 1986, pp.409-438.

De Marco, T. Structured Analysis And System Specification, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1979.

Deogun, J.S. and Nakhforoush, A. "A Conceptual Approach to Development of Decision Support Systems," Proceedings of the Seventeenth Annual Hawaii International Conference on System Sciences, Volume 1, James P. Fry, et al. (Editors), 1984.

Dolk, D.R. "Data as Models: An Approach to Implementing Model Management," Decision Support Systems, Vol.2, 1986, pp.73-80.

Dolk, D.R. and Konsynski, B.R. "Knowledge Representation for Model Management Systems," IEEE Transactions on Software Engineering, Vol. SE-10, No.6, November, 1984, pp.619-628.

Eberts, R.E., Nof, S.Y., Zimolong, B. and Salvendy, G. "Dynamic Process Control: Cognitive Requirements and Expert Systems," in Human-Computer Interaction, Salvendy G. (Ed.), Elsevier Science Publishers, B.V. Armsterdam, 1988.

Elam, J.J. and Konsynski, B.R. "Using Artificial Intelligence Techniques to Enhance the Capabilities of Model Management Systems," Decision Sciences, Vol.18, No.3, Summer 1987, pp.487-501.

Elam, J.J. and Mead M. "Can Software Influence Creativity?" Information Systems Research, Vol.1, No.1, March 1990.

Elam, J.J. and Mead M. "Designing for Creativity: Considerations for DSS Development," Information and Management, Vol.13, 1987, pp.215-222.

Engle, R.W., and Bukstel, L. "Memory Processes Among Bridge Players of Differing Expertise," American Journal of Psychology, Vol. 91, 1978, pp. 673-689.

Evans, J. St. B.T., "The Knowledge Elicitation Problem: A Psychological Perspective," Behaviour and Information Technology, Vol.7, No.2, pp.111-131.

Fedorowicz, J. and Williams, G.B. "Representing Modeling Knowledge in an Intelligent Decision Support System," Decision Support Systems, Vol.2, 1986, pp.3-14.

Fishwick, P.A. "The Role of Process Abstraction in Simulation," IEEE Transactions on Systems, Man, and Cybernetics, Vol.18, No.1, January/February, 1988, pp.18-39.

Flynn, B.B., Sakakibara, S. Schroeder, R.G., Bates, K.A. and Flynn, E.J. "Empirical Research Methods in Operations Management," Journal of Operations Management, Vol.9, No.2., April, 1990. pp.250-284.

Fuerst, W.L. and Martin, M.P. "Effective Design And Use of Computer Decision Models," MIS Quarterly, March, 1984, pp.17-26.

Gale, W.A. Artificial Intelligence & Statistics, Reading, Addison-Wesley Publishing Company, 1986.

Greeno, J.G. "Notes on Problem Solving Abilities," In W.K. Estes (Ed.), Handbook of Learning and Cognitive Process, Vol.5, pp.239-270, Hillsdale, N.J.:Lawrence Erlbaum Associates, 1978.

Greeno, J.G. "The Structure of Memory and the Process of Solving Problems," In R.L. Solso (Ed.), Contemporary Issues in Cognitive Psychology: The Loyola Symposium. New York: Wiley, 1974.

Guindon, R., Burtis, B. and Krasner, H. "A Model of Cognitive Processes in Software Design: An Analysis of Breakdowns in Early Design Activities by Individuals," MCC Technical Report Number STP-283-87, August 1987a.

Guindon, R., Krasner, H. and Burtis, B. "Cognitive Processes in Software Design: Activities in Early, Upstream Design," In Proceedings Interact'87, Second IFIP Conference on Human-Computer Interaction, H.J. Bullinger and B. Shackel (Editors). Elsevier Science Publishers B.V., Armsterdam, 1987b, pp.383-388.

Hahn, G.J. "More Intelligent Statistical Software and Statistical Expert Systems: Future Directions," American Statistical Association, Vol.39, No.1, Feb.1985, pp.1-8

Henderson, J.C. "Finding Synergy Between Decision Support Systems And Expert Systems Research," Decision Sciences, Vol.18, 1987, pp.333-349.

Hill, T.R. and Roberts S.D. "A Prototype Knowledge-based Simulation Support System," Simulation, Vol.48, No.4, April 1987, pp.152-161.

Hillier, F.S. and Lieberman, G.J. Introduction to Operations Research, Holden-Day, Inc., Oakland, California, 1980.

Huber, G.P. "Cognitive Style As a Basis for MIS and DSS Designs: Much Ado about Nothing," Management Science, Vol.29, No.5, May 1983, pp.567-579.

Hwang, S. "Automatic Model Building Systems: A Survey," DSS Transaction, 1985.

Hwang, T.S. and Ullman, D.G. "The Design Capture System: Capturing Back-Of-The-Envelope Sketches," International Conference on Engineering Design (ICED 90), Dubrovnik, Yugoslavia, August 1990.

Intellicorp, The SimKit System: Knowledge-Based Simulation Tools in KEE, Mountain View, CA: Intellicorp, 1986.

Jaques, E., Gibson, R.O. and Isaac, D.J. (editors) Levels of Abstraction in Logic and Human Action, Heinemann, London, 1978.

Jaques. E. "The system of Levels and Components," in Levels of Abstraction in Logic and Human Action, Jaques, E., Gibson, R.O. and Isaac, D.J. (editors). Heinemann, London, 1978.

Jeffries, R., Turner, A., Polson, P.G. and Atwood, M.E. "The Processes Involved in Designing software," in Cognitive Skills and Their Acquisition, J.R. Anderson (Ed.), W.H. Freeman & Company, Hillsdale, N.J., 1981.

Jones, J.C., "A Method of Systematic Design" in Jones, J.C. and Thornley, D. (eds), Conference on Design Methods, Pergamon Press, Oxford, U.K. 1963.

Keen, P.G.W. and Bronsema, G.S. "Cognitive Style Research: A Perspective for Integration," Proceedings of the Second International Conference on Information System, December 7-9, 1981, Cambridge, Ma. Ross, C.A.(Editor), pp. 21-52.

Larkin, J. "Problem Representation in Physics in Mental Models," In Mental Models, D.R. Gentner and Stevens (Editors), LEA, 1983.

Malhotra, A., Thomas, J.C., Carroll, J.M., and Miller, L.A. "Cognitive Processes in Design," International Journal of Man-Machine Studies, Vol.12, 1980, pp.110-140.

Mason, R.O. and Mitroff, I.I. "A Program for Research in Management," Management Science, Vol.19, No.5, 1973.

McRoberts, M., Fox, M. and Husain, N. "Generating Model Abstraction Scenarios in KBS," in AI, Graphics and Simulation, Birtwistle, Graham (Eds.). La Jolla, california: The Society for Computer Simulation, 1985. pp.29-33.

Mihram, G.A. "The Modeling Process," IEEE Transactions On Systems, Man, And Cybernetics, Vol.SMC-2, No.5, November 1972, pp.621-629.

Murphy, E.D. and Mitchell, C.M. "Cognitive Attributes: Implications for Display Design in Supervisory Control Systems," International Journal of Man-Machine Studies, Vol.25, 1986, pp.411-438.

Murphy, F. and Stohr, E. "An intelligent System for Formulating Linear Programs," Decision Support Systems, Vol.2, No.1, 1986, pp.39-48.

Murray, D.M., "Embedded User Models," In Proceedings Interact'87, Second IFIP Conference on Human-Computer Interaction, H.J. Bullinger and B. Shackel (Editors). Elsevier Science Publishers B.V., Armsterdam, 1987, pp.229-235.

Naylor, T.H. and Schauland H. "A Survey of Users of Corporate Planning Models," Management Science, Vol.22, September 1976, pp.927-937.

Needham, D.R. and Begg, I.M. "Problem-Oriented Training Promotes Spontaneous Analogical Transfer: Memory-Oriented Training Promotes Memory for Training," Memory and Cognition, Vol.19, No.6, 1991. pp.543-557.

Newell A. and Simon, H.A. Human Problem Solving, Prentice Hall, Englewood Cliffs. New Jersey, 1972.

Newsome, S.L. and Spillers, W.R. "Tools for Expert Designers: Supporting Conceptual Design," in Design Theory '88, S.L. Newsome, W.R. Spillers and S. Finger (Editors), Springer-Verlag New York, Inc. 1989. pp.49-55.

Norman, D.A. "Some Observations on Mental Models," in Mental Models, D. Gentner and A.L. Stevens (Editors). Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1981. pp.7-14.

Ossher, H.L. "A Mechanism for Specifying the Structure of Large, Layered Systems," in Research Directions in Object-Oriented Programming, Bruce Shriver and Peter Wegner (Eds.), The MIT Press, Cambridge, Massachusetts, 1987.

Paivio, A. Mental Representation: A Dual Coding Approach, New York: Oxford University Press, 1986.

Parlar, M. "EXPIM: A Knowledge-based Expert System for Production/Inventory Modeling," International Journal of Production Research, Vol.27, No.1, 1989, pp.101-118.

Pegden, C.D. Introduction to SIMAN, State College, PA:Systems Modeling Corp, 1986.

Piaget, J. Understanding Causality, Norton, New York, N.Y., 1974.

Posner, M.I. and Rogers, M.G.K. "Chronometric Analysis of Abstraction and Recognition," In W.K. Estes (Ed.), Handbook of Learning and Cognitive Process, Vol.5, pp.239-270, Hillsdale, N.J.:Lawrence Erlbaum Associates, 1978.

Pracht, W.E. and Courtney, J.F. "The Effects of an Interactive Graphics-Based DSS to Support Problem Structuring," Decision Sciences, Vol.19, 1988, pp.598-621.

Pracht, W.E. "Model Visualization: Graphical Support for DSS Problem Structuring and Knowledge Organization," Decision Support Systems, Vol.6, 1990, pp.13-27.

Pritsker, A.A.B. Introduction to Simulation and SLAMII, John Wiley & Sons, New York, N.Y. 1986.

Ramaprasad, A. "Cognitive Process as a Basis for MIS and DSS Decision," Management Science, Vol.33, No.2, February 1987, pp.139-148.

Ramaprasad, A. and Mitroff, I.I. "On formulating Strategic Problems," Academy of Management Review, Vol.9, No.4, 1984, pp.597-605.

Rasmussen, J. Information Processing and Human-Machine Interaction:An Approach to Cognitive Engineering, Elsevier Science Publishing Co., Inc. New York, 1986.

Rasmussen, J. "Skills, Rules, and Knowledge; Signals, Signs, and Symbols, and Other Distinctions in Human Performance Models," IEEE Transactions on Systems, Man and Cybernetics, Vol.SMC 13, No.3, May/June 1983.

Rich, E. "User Modeling via Stereotypes," Cognitive Science, Vol.3, 1979, pp.329-354.

Rosch, E. "Principles of Categorization," In E. Rosch and B.B. Lloyed (Eds.), Cognition and Categorization, Hillsdale, N.J.:Erlbaum, 1978.

Sandford, A.J. Cognition and Cognitive Psychology, New York: Basic Books, Inc., 1985.

SIMSCRIPT II.5 Reference Manual, La Jolla, California, CACI Products Company, 1990.

Simon, H.A. "Information Processing Theory of Human Problem Solving," In W. Chase (Editor). Handbook of Learning and Cognitive Process, Volume 5: Human Information Processing, Hillsdale, New Jersey: Lawrence Erlbaum, 1978.

Sowa, J.F., Conceptual Structures, Reading, MA: Addison-Wesley, 1984.

Sprague,Jr. R.H. and Carlson, E.D. Building Effective Decision Support Systems, Prentice Hall, Englewood Cliffs, New Jersey, 1982.

Spillers, W.R. and Newsome, S. "Design Theory: A Model for Conceptual Design," in Design Theory '88, S.L. Newsome, W.R. Spillers and S. Finger (Editors), Springer-Verlag New York, Inc., 1989. pp.198-212.

Straub, D.W. and Carlson, C.L. "Validating Instruments in MIS Research," MIS Quarterly, June, 1989. pp.147-166.

Taylor, R.N. and Benbasat, I. "A Critique of Cognitive Styles Theory and Research," Proceedings of the First International Conference on Systems, 1980.

Thomas, J.C. and Carroll, J.M. "The Psychological Study of Design," Design Studies, Vol.1, No.1, July 1979.

Torn, A.A. "Simulation Graphs: A General Tool for Modeling Simulation Designs," Simulation, December 1981, pp.187-194.

Turban, E. Decision Support and Expert Systems, Macmillan Publishing Company, New York, 1988.

Turban, E. and Watkins, P.R. "Integrating Expert Systems and Decision Support Systems," MIS Quarterly, June 1986, pp.121-136.

Ullman, D.G., Wood, S. and Craig, D. "The Importance of Drawing in the Mechanical Design Process," NSF Engineering Design Research Conference, Amherst, Mass., June, 1989.

Vemuri, V. Modeling Of complex Systems: An Introduction, Academic Press, New York, N.Y., 1978.

Weber, E.S. "Systems to think With: A Response to 'A Vision for Decision Support Systems'," Journal of Management Information Systems, Vol.2, No.4, Spring 1986, pp.85-97.

Whitten & Bentley, Using Excelerator For Systems Analysis and Design, Times Mirror/Mosby College Publishing, 1986.

Whittlesea, B.W.A. The Representation of Concepts: An Evaluation of the Abstractive and Episodic Perspectives, Doctoral thesis, McMaster University, Hamilton, Ontario, Canada. 1983.

Will, H.J. "Model Management Systems" In E. Rochla & N. Szypenski(Eds.), Information Systems and Organizational Structure. Berlin: W. De Grayter, 1975.

Wilson, B. Systems: Concepts, Methodologies, and Applications, John Wiley & Sons, New York, N.Y. 1984.

Young, L.F. "Right-Brained Decision Support Systems," Data Base, Summer 1983, pp.28-36.

Zeigler, B.P. and Rada, R. "Abstraction in Methodology: A Framework for Computer Support," Information Processing and Management, Vol.20, No.1-2, 1984, pp.63-79.

Zmud, R.W. "Individual Differences and MIS Success: A Review of the Empirical Literature," Management Science, Vol.25, No.10, October, 1979, pp. 966-979.

Appendix I Protocol Studies: Phase I - Design of Decision Table

A. Protocol Analysis of Design Process - Instruction
B. Description of Decision Tables
C. Problem Statement

A. Protocol Analysis of Design Process

A brief description of the purpose and structure of decision tables is provided in the following page. Once you are familiar with decision tables, you will be asked to read a problem and to construct a decision table to represent the decision rules in the problem.

The purpose of this practice is not to test whether you can find the best solution to the problem, instead, the objective is for us to learn the process of how you design a decision table which you think will represent the problem situation clearly. Therefore, it is very important for you to continue to speak to the dictation machine, explaining what comes to your mind, why you decide to do certain things, and what you think you can achieve in doing these things.

You may spend as much time as you want to understand the description of decision tables. When you are ready to tackle the problem, please let me know and I will provide you with the problem and activate the dictation machine to record your thinking process. During the process, you may go back to the description and examples for information.

Thank you for your time and cooperation.

B. Description of Decision Tables

A decision table is a compact representation of a procedure in which alternative courses of action are specified for various combinations of conditions. The table specifies what action (decision) should be taken for a given combination of conditions.

Basically, there are four principal parts in a decision table:

Condition stub	: description of the condition
Condition entries	: indicating whether the condition should or should not be met
Action stub	: description of the action to be taken
Action entries	: indicating which action is to be taken under the corresponding condition of the column.

The following is an example of a decision table constructed for an inventory control decision:

Statement 1: If inventory is less than or equal to the reorder point, then order a replenishment. Otherwise, don't order.

Two actions and one condition (the inventory level) are implied in the above statement:

Condition 1: Inventory is less than or equal to the reorder point
Action 1 : Order
Action 2 : Do not order

The decision table can be constructed as below:

Table 1

	Condition	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5
1	Inventory less than or equal to reorder point	yes	no			
2						
3						
	Action					
1	Order	x				
2	Do not order		x			

... continued

Table 1 should be read as following:
Rule 1: if condition 1 is met, then take action 1
Rule 2: if condition 1 is not met, then take action 2

Decision tables are also useful to represent hierarchical decision rules. For example, a hierarchy of decision tables could be used for the inventory control example if the problem statements are the following:

Statement 1: If inventory is less than or equal to the reorder point, then order a replenishment. Otherwise, don't order.

Statement 2: If the item is ordered from supplier and the item's ID number starts with 9, check table 1 for inventory control decisions. Otherwise, check with the production department.

A decision table which is at a higher level than table 1 can be constructed to represent statement 2:

Table 0

	Condition	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5
1	Item ordered from supplier	yes	no			
2	Item's ID number starts with 9	yes				
3						
	Action					
1	Inventory problem -use table 1	x				
2	Check with production department		x			

C. Problem Statement

Mr. A has just graduated from the university and is faced with a career decision. He could either get a job or pursue a higher degree. He will go back to school if the school which he applied offers him a scholarship or TA work. the amount of the scholarship is not an important decision factor. However, he estimates that he will need a minimum of $8,000 per year to cover tuition fees and living expenses. He thinks that he will definitely go back to school if the offer is from a co-op MBA program with or without scholarship. the criteria he uses to choose among job offers are the location of the firm, the size of the firm, the salary offered, the prospects for promotion and the job position. He has set up a few rules in making the choice. Ideally, the job is located in Ontario. However, he will go to a farther location (i.e., the job is located in other provinces) if the pay is over $45,000 per year. He prefers to work for a medium size firm (defined to have between 100 and 500 employees). The salary is negotiable, but 28,000 per year is the minimum he will accept. He will refuse the offer is the prospects for promotion is poor. He is most interested in positions such as staff analyst or line management.

Please construct a decision table to represent the decision rules.

Appendix II Protocol Studies: Phase II - Design of High School Counselling Office

A. Confidential Preliminary Questionnaire
B. Consent Form
C. Problem Statement

A. Confidential Preliminary Questionnaire

Model Design Project

Confidential Preliminary Questionnaire

1. Name (Print): _____

2. Sex : M _____ F _____

3. Age : 20-25 _____ 26-30 _____
31-35 _____ 36-40 _____
41-45 _____ 46-50 _____
51 and up _____

4. Education (Degree(s) and Field(s)): _____

5. Your experience in design-related work:

	Yes	No	Months of Experience	Computerized Design Tool(s) Used
Software design (including programming)	___	___	_____	_____
System design	___	___	_____	_____
Engineering design (e.g., machine design)	___	___	_____	_____
Architectural design	___	___	_____	_____
Other (please specify) _____			_____	_____

6. In the following questions, please circle the numbers that most closely represent your opinions:

	Strongly Disagree	Dis-agree	Neu-tral	Agree	Strongly Agree
1) Computers are easy to use.	1	2	3	4	5
2) Computers are useful in many ways.	1	2	3	4	5
3) I enjoy using computers.	1	2	3	4	5
4) I would encourage using computers whenever it is possible.	1	2	3	4	5

7. The following questions are related to computer support for design. Please circle the numbers which most closely represent your opinion.

	Strongly Disagree	Dis-agree	Neu-tral	Agree	Strongly Agree
1) Most design activities can be easily supported by computers.	1	2	3	4	5
2) Computers are more useful for structured design than for creative design.	1	2	3	4	5
3) Computerized design tools are useful in supporting my creative design thoughts.	1	2	3	4	5
4) Computerized design tools are useful in supporting my routine design tasks.	1	2	3	4	5
5) Computers are more useful when I am dealing with design problems from familiar domains than from unfamiliar domains.	1	2	3	4	5
6) The design process is better if it is carried out in a systematic fashion.	1	2	3	4	5
7) A systematic design approach imposes unnecessary restriction on creative design activities.	1	2	3	4	5

8. Your participation in this research will involve filling in an MBTI (Myers-Briggs Type Index) questionnaire and taking part in an empirical study session in which you will perform a design task and simultaneously describe verbally what you are doing. We will tape-record your comments while you perform the design tasks. You are assured total confidentiality. In addition, you may withdraw from the project at any time if you wish.

   It is estimated that it will take 15 minutes to complete the MBTI questionnaire and approximately 1 hour for the recording session. The recording session will be arranged at your convenience.

   If you are willing to participate, please sign and date the consent form on the following page.

   Thank you for your time and cooperation.

B. Consent Form

Model Design Project

Consent Form

I, _____ (print) agree to participate, if selected, in the Model Design Project led by Dr. N. Archer and Ms. Diana Kao, based on the following terms:

(1) The purpose of the study

I understand that the purpose of this research is to observe and study the cognitive processes of different individuals while they are performing symbolic model design functions.

(2) Completion of an MBTI questionnaire

I will complete an MBTI questionnaire. I understand that the result of my MBTI analysis will be released to me only, at my request.

(3) Participation in protocol analysis

I will participate in a protocol analysis in which I will solve a design problem either using pencil and paper or using a computer software package. I understand that my comments and other data recorded during the session will be analyzed.

(4) Confidentiality

I understand that my identity will not be revealed in connection with any data or research released as a result of this study.

(5) Right to withdraw

I understand that I may withdraw from this study at any time by informing Ms. Diana Kao verbally or in writing.

Signature _____ Date _____

I can be reached for arranging the recording session by phone at this

number: _____

C. Problem Statement

Please design a career counseling office in a high school. Your design should include the personnel and their functions, the activities of the office and the relationships of the office with other departments of the school or organizations outside the school.

Appendix III Problem Facets and Associated Level of Abstraction For the High School Student Counseling Office Design Exercise

A. Problem Facets Identified - Phase II
B. Activities Identified - Phase II

A. Problem Facets Identified - Phase II

Very High Level of Abstraction
- Objectives of the office
- School image
- Flexibility of the office to deal with problems

High Level of Abstraction
- Personnel
- Budget
- Planning for the sub-divisions of the office
- Location of the office

Middle level of Abstraction
- Size of the office (# of staff)
- Skills of the staff
- Division of responsibilities among staff
- External organizations
- Relations with other departments of the school
- Physical layout of the office
- Structure of the office

Low Level of Abstraction
- Detailed Activities
- Equipment (Hardware and software)

B. Activities Identified - Phase II

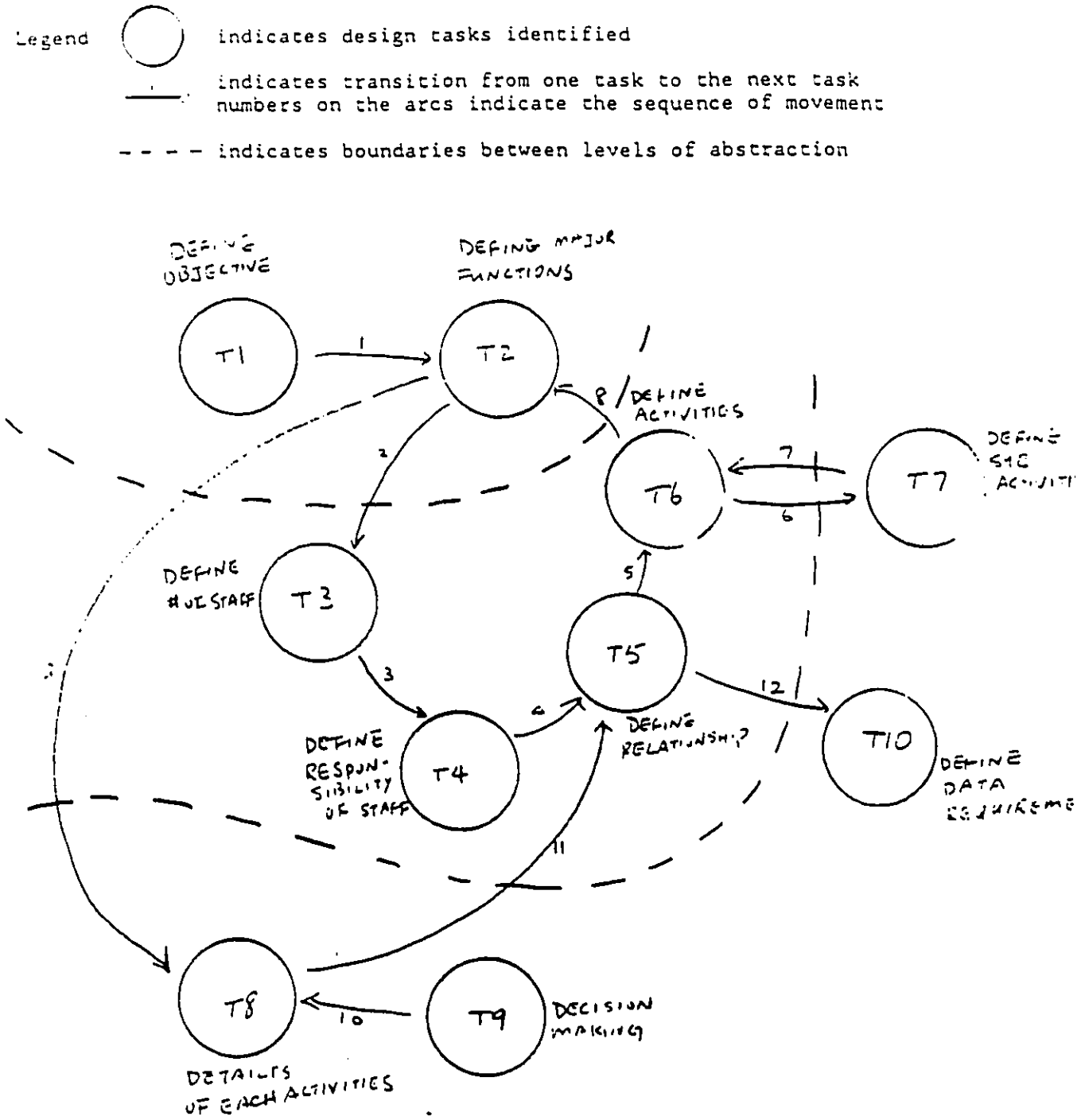
1. Help students to find jobs or further education
2. Deal with outside organizations (gather information)
3. Provide useful information to high school students
4. Understand students' needs (distribution, etc.)
5. Recommend good students to companies or universities
6. Provide feedback to the community and provide information of community to the students
7. Provide guidelines and directions
8. Provide special help to the less intelligent students
9. Become familiar with the activities of other departments
10. Record keeping of what jobs are available, prepare list of potential placement
11. Give talks to students, info session, etc.
12. Get temporary summer jobs and training programs for the students
13. Show students how to access career information
14. Discuss students' progress with teachers
15. Filing of material and references
16. Make appointments
17. Find out the strength and weakness of students
18. Manage transcription
19. Arrange, prepare and conduct interview session
20. Follow-up on interview sessions
21. Arrange outside companies and universities for talks
22. Review courses
23. Run and evaluate aptitude tests
24. Review application forms
25. Send report to principal
26. Improve high school courses
27. Make recommendations in terms of which school to go to
28. Review newspaper, trade journals
29. Planning for career day (budget, advertisement, people etc.)
30. Co-ordinate activities
31. Help students with the English and Math
32. Arrange field tours
33. Categorization of different jobs by disciplines
34. Suggestion on what subjects to take
35. Discuss financial aspect of the problem
36. Counseling on personal problems
37. Group students into groups of same interest
38. Broaden students search space, generate ideas for students
39. Develop departmental plan

Appendix IV Examples of Graphical Representation of Protocols

A. Phase II - Example 1
B. Phase II - Example 2

A. Phase II - Example 1

This is an example of the analytical top-down design approach. We can observe that there is relatively little interaction among different levels of abstraction.



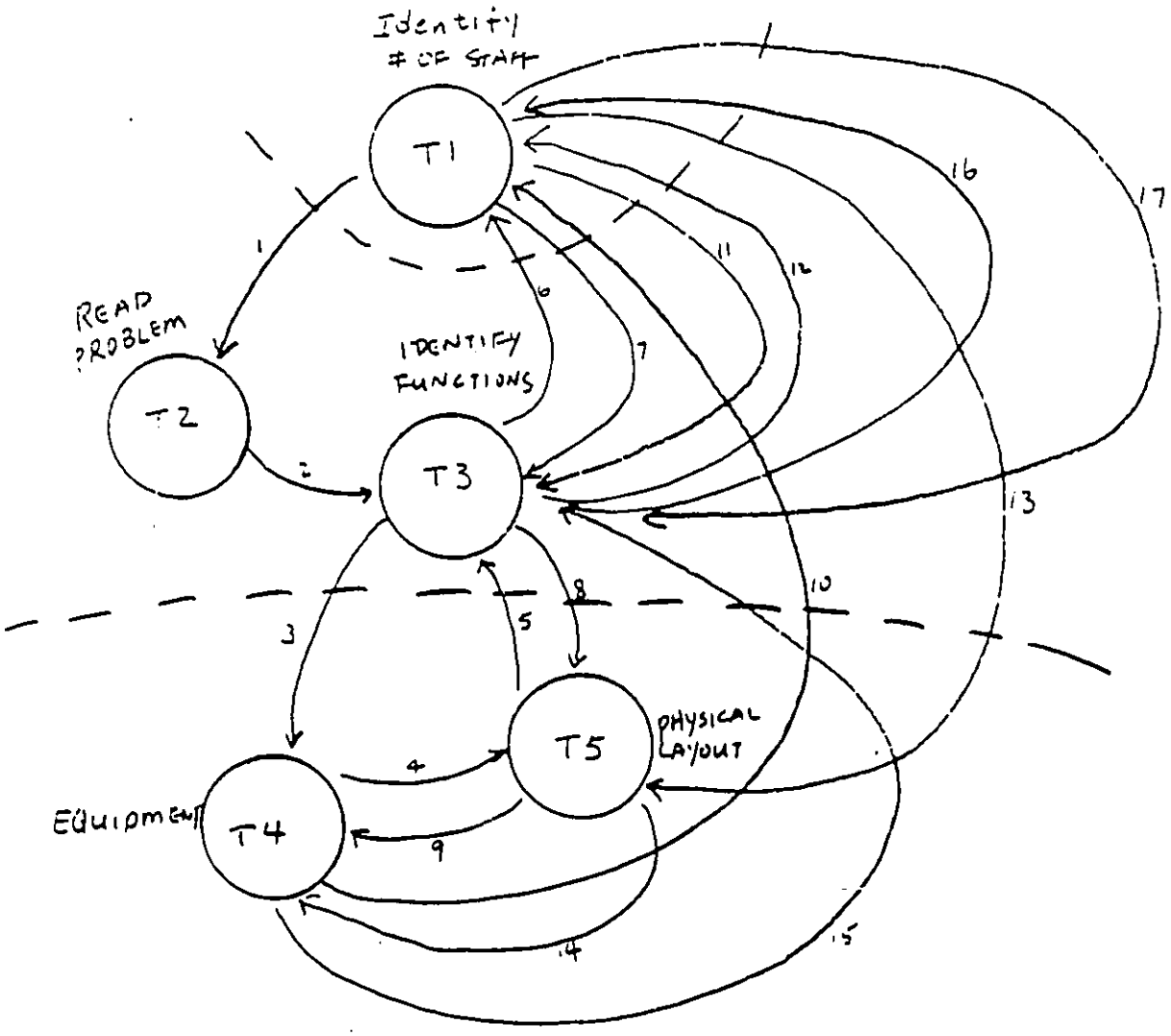
B. Phase II- Example 2

This is an example of the heuristic bottom-up design approach. We can observe that the designer did not differentiate between different levels of abstraction. In other words, the designer constantly moved from one level of abstraction to another. And the majority of the design was focused on the lower abstraction level.

Legend: ○ indicates design tasks identified

→ indicates transition from one task to the next task
numbers on the arcs indicate the sequence of movement

- - - - indicates boundaries between levels of abstraction



Appendix V Protocol Studies: Phase II - Quantitative Data

A. Data Collected from Group 1
B. Data Collected from Group 2
C. Data Collected from Group 3
D. Data Collected from Group 4
E. Statistical Analysis

A. Data Collected from Group 1

Factors	Subject #1	Subject #2	Subject #3
Design Experience	no	no	no
Software Support	no	no	no
Domain knowledge	no	no	no
MBTI	ENTJ	ESTP	INTJ
Data Collected from Protocols			
Number of facets identified	6	4	6
High level abstraction	1	1	1
Middle level abstraction	4	2	4
Low level abstraction	1	1	1
Number of movements from			
High - High	0	0	0
High - Middle	0	0	2
High - Low	1	1	0
Middle - High	1	0	2
Middle - Middle	5	0	1
Middle - Low	2	2	1
Low - High	0	1	0
Low - Middle	2	2	1
Low - Low	0	0	0
Number of activities identified	5	11	15
Total number of tasks generated	15	5	6
Completed	13	5	13
Incompleted	2	0	4
Task completed then go to			
New task	12	1	0
Old task	8	3	2
Task not completed then go to			
New task	1	3	4
Old task	1	0	0
Number of times review completed tasks	1	0	0
Number of times making cross reference	1	3	2

B. Data Collected from Group 2

Factors	Subject #1	Subject #2	Subject #3
Design Experience	no	no	no
Software Support	yes	yes	yes
Domain knowledge	no	no	no
MBTI	ESTJ	ESTP	ISFJ
Data Collected from Protocols			
Number of facets identified	5	6	3
High level abstraction	0	1	0
Middle level abstraction	4	3	2
Low level abstraction	1	2	1
Number of movements from			
High - High	0	0	0
High - Middle	0	1	0
High - Low	0	0	0
Middle - High	0	0	0
Middle - Middle	2	0	1
Middle - Low	1	3	1
Low - High	0	1	0
Low - Middle	1	1	0
Low - Low	0	1	3
Number of activities identified	6	8	6
Total number of tasks generated	6	6	3
Completed	5	6	3
Incompleted	1	0	0
Task completed then go to			
New task	5	4	2
Old task	0	0	0
Task not completed then go to			
New task	0	1	1
Old task	1	2	0
Number of times review completed tasks	0	0	3
Number of times making cross reference	0	0	0

C. Data Collected from Group 3

Factors	Subject #1	Subject #2	Subject #3
Design Experience	yes	yes	yes
Software Support	no	no	no
Domain knowledge	no	no	no
MBTI	ENTJ	ENTJ	ENTJ
Data Collected from Protocols			
Number of facets identified	6	7	11
High level abstraction	1	0	3
Middle level abstraction	4	5	6
Low level abstraction	1	2	2
Number of movements from			
High - High	0	0	0
High - Middle	1	0	2
High - Low	0	0	1
Middle - High	0	0	1
Middle - Middle	1	1	4
Middle - Low	5	7	5
Low - High	1	0	1
Low - Middle	5	6	4
Low - Low	2	2	1
Number of activities identified	8	8	9
Total number of tasks generated	14	5	11
Completed	10	5	9
Incompleted	3	0	2
Task completed then go to			
New task	7	1	4
Old task	4	3	4
Task not completed then go to			
New task	6	3	6
Old task	4	10	5
Number of times review completed tasks	6	0	5
Number of times making cross reference	10	5	2

D. Data Collected from Group 4

Factors	Subject #1	Subject #2	Subject #3
Design Experience	yes	yes	yes
Software Support	yes	yes	yes
Domain knowledge	no	no	no
MBTI	ENTJ	ENTJ	INTJ
Data Collected from Protocols			
Number of facets identified	9	7	4
High level abstraction	2	1	0
Middle level abstraction	5	4	2
Low level abstraction	2	2	2
Number of movements from			
High - High	0	0	0
High - Middle	1	1	0
High - Low	0	0	0
Middle - High	0	0	0
Middle - Middle	3	2	1
Middle - Low	2	4	5
Low - High	1	1	0
Low - Middle	0	4	7
Low - Low	4	8	4
Number of activities identified	8	13	16
Total number of tasks generated	8	7	5
Completed	8	7	5
Incompleted	0	0	0
Task completed then go to			
New task	5	2	0
Old task	2	3	4
Task not completed then go to			
New task	2	4	4
Old task	2	12	9
Number of times review completed tasks	1	3	1
Number of times making cross reference	1	2	4

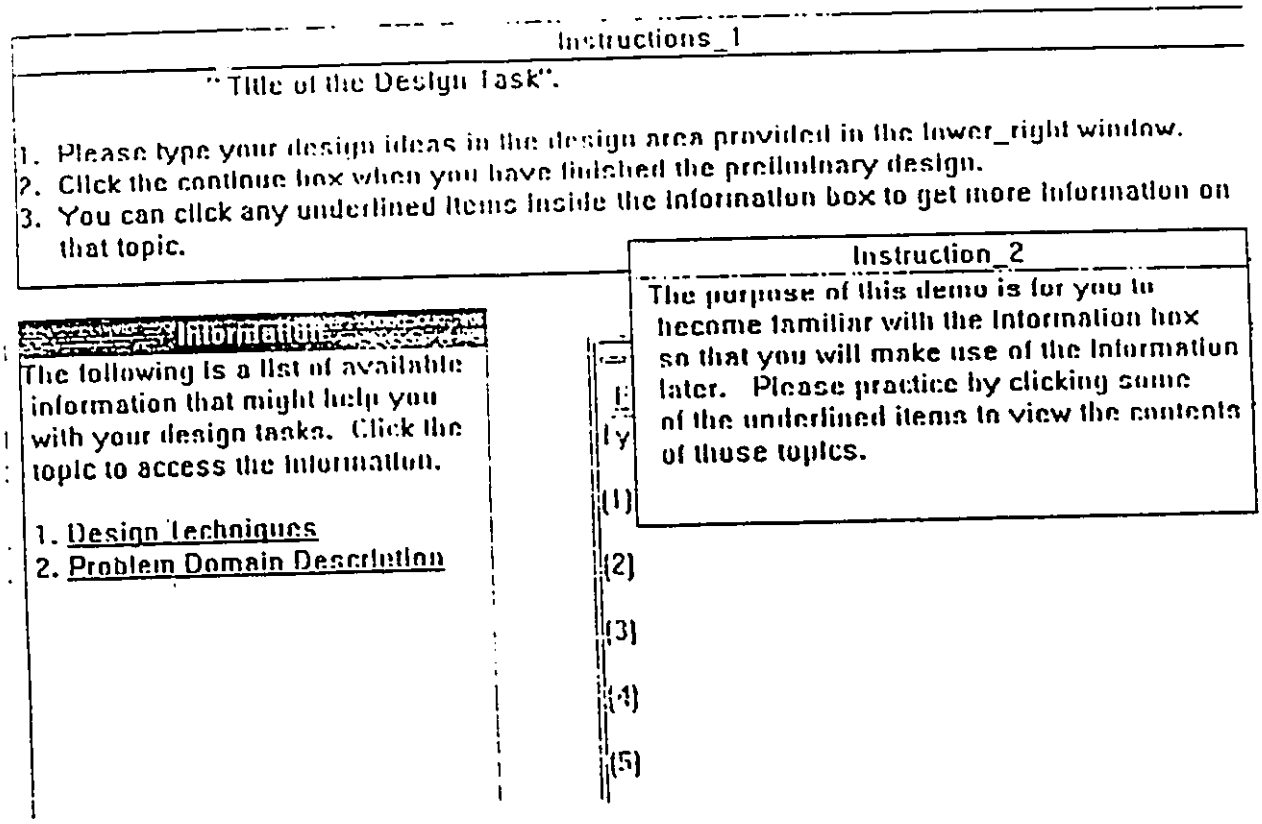
E. Statistical Data Analysis

The statistical results were summarized below:

Data Collected from Protocols	Design Experience	Software Support
Number of facets identified	F(1,11)=3.843,p<.10**	F(1,11)=0.706,p>.05
High level abstraction	F(1,11)=0.818,p>.05	F(1,11)=0.818,p>.05
Middle level abstraction	F(1,11)=2.882,p>.05	F(1,11)=1.471,p>.05
Low level abstraction	F(1,11)=6.250,p<.05*	F(1,10)=0.250,p>.05
Number of movements from		
High - High	no observation	
High - Middle	F(1,11)=0.444,p>.05	F(1,11)=0.444,p>.05
High - Low	F(1,11)=0.500,p>.05	F(1,11)=4.500,p<.10**
Middle - High	F(1,11)=1.000,p>.05	F(1,11)=4.000,p>.05
Middle - Middle	F(1,11)=0.250,p>.05	F(1,11)=0.250,p>.05
Middle - Low	F(1,11)=20.25,p<.05*	F(1,11)=2.250,p>.05
Low - High	F(1,11)=1.000,p>.05	F(1,11)=0.000,p>.05
Low - Middle	F(1,11)=8.595,p<.05*	F(1,11)=1.167,p>.05
Low - Low	F(1,11)=12.04,p<.05*	F(1,11)=9.375,p<.05*
Number of activities identified	F(1,11)=0.931,p>.05	F(1,11)=0.008,p>.05
Total number of tasks generated	F(1,11)=0.476,p>.05	F(1,11)=2.594,p>.05
Completed	F(1,11)=0.010,p>.05	F(1,11)=4.455,p<.10**
Incompleted	F(1,11)=0.200,p>.05	F(1,11)=5.000,p<.10**
Task completed then go to		
New task	F(1,11)=0.134,p>.05	F(1,11)=0.263,p>.05
Old task	F(1,11)=1.400,p>.05	F(1,11)=6.429,p<.05*
Task not completed then go to		
New task	F(1,11)=10.71,p<.05*	F(1,11)=5.762,p<.05*
Old task	F(1,11)=12.66,p<.05*	F(1,11)=0.316,p>.05
Number of times review completed tasks	F(1,11)=3.200,p>.05	F(1,11)=0.356,p>.05
Number of times making cross reference	F(1,11)=5.492,p<.05*	F(1,11)=4.339,p<.10**

** marginally significant at p<.10
* significant at p<.05

Appendix VI Screen Display of the Information Module

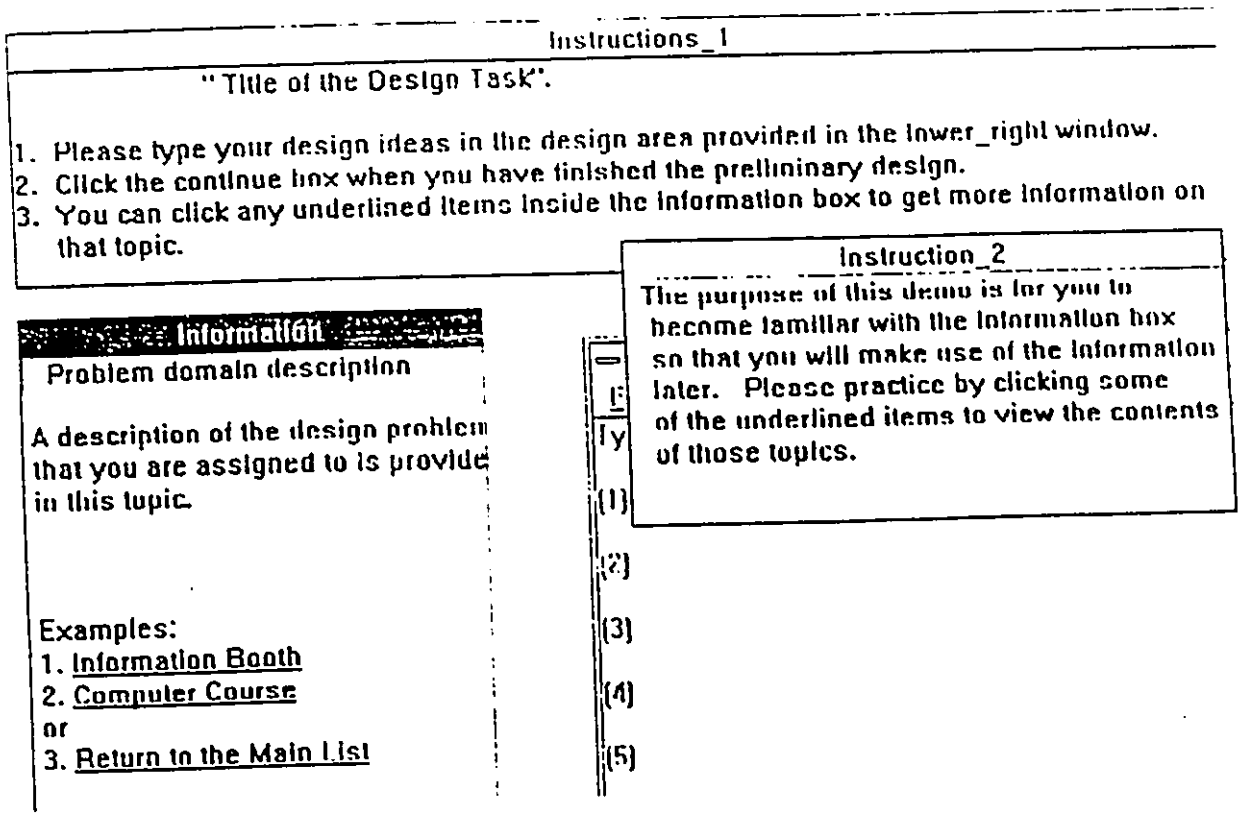


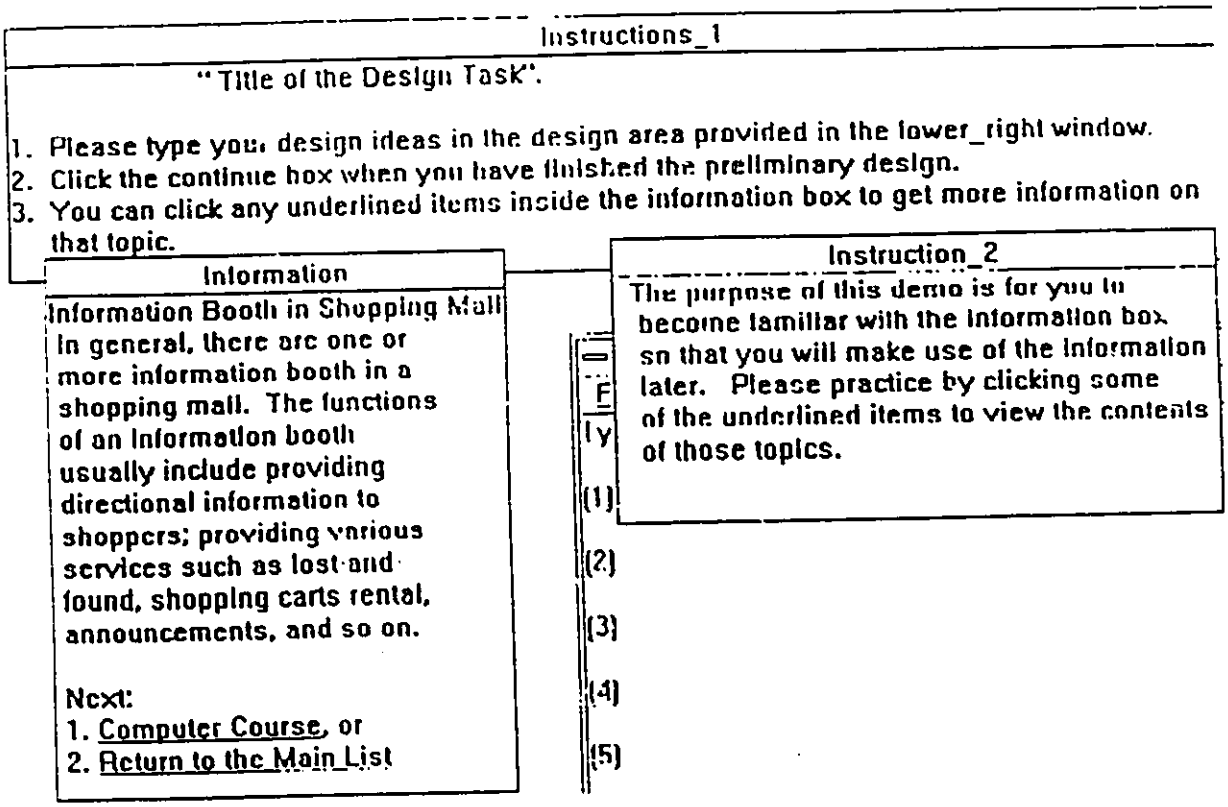
Note: If 1. Design Techniques is selected,

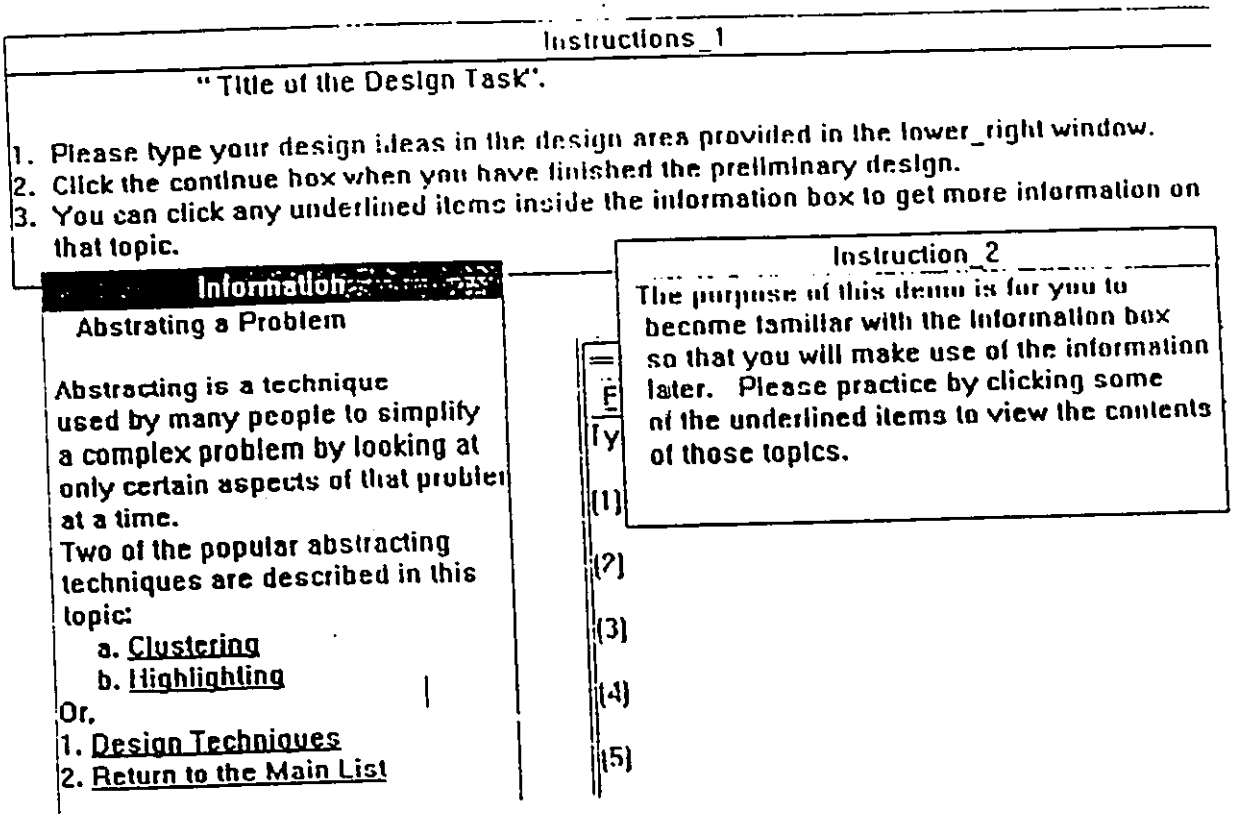
=> The information box titled Abstracting a Problem will display on the screen. The user than can choose to view a specific abstracting technique.

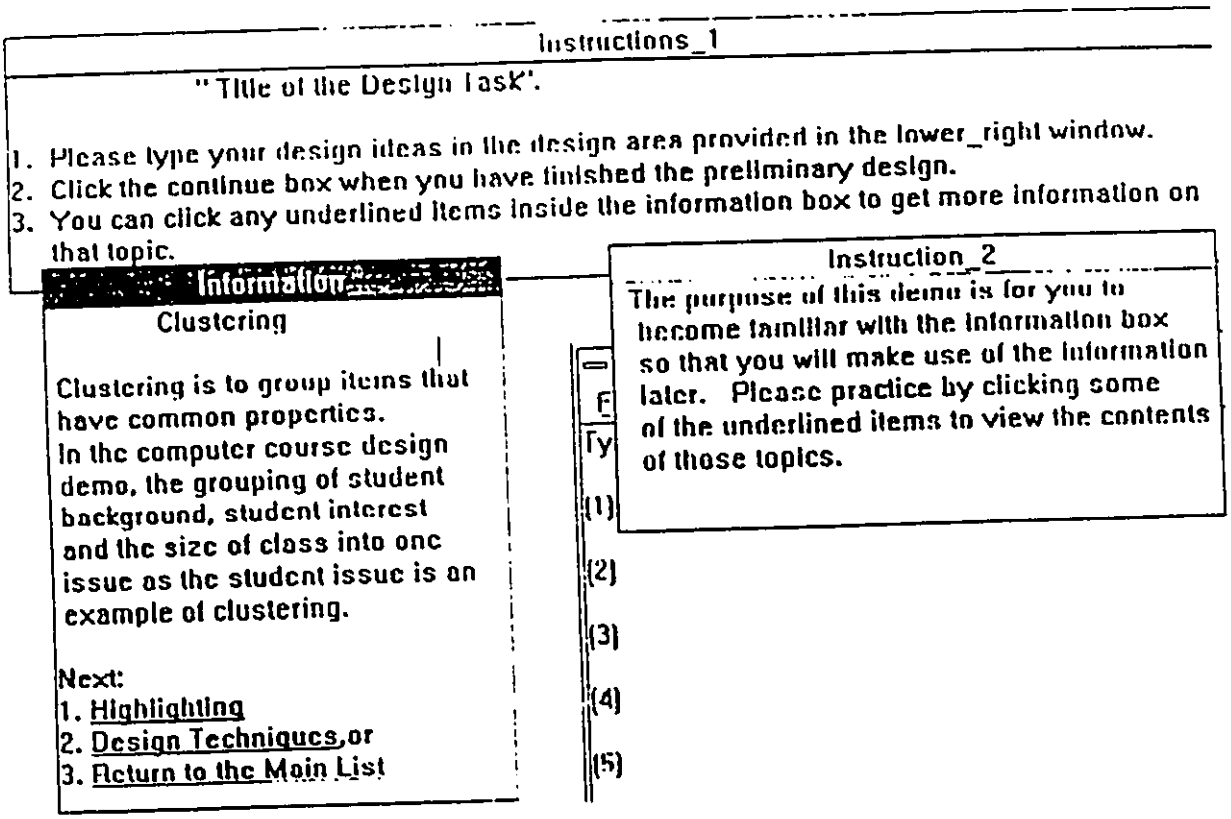
If 2. Problem Domain Description is selected,

=> The information box titled Problem Domain Description will display on the screen. The user than can choose to view a specific problem domain.


Instructions_1
"Title of the Design Task".
1. Please type your design ideas in the design area provided in the lower_right window.
2. Click the continue box when you have finished the preliminary design.
3. You can click any underlined items inside the information box to get more information on that topic.
Instruction_2
The purpose of this demo is for you to become familiar with the information box so that you will make use of the information later. Please practice by clicking some of the underlined items to view the contents of those topics.
Information
Problem domain description
A description of the design problem that you are assigned to is provide in this topic.
Examples:
1. Information Booth
2. Computer Course
or
3. Return to the Main List
(1)
(2)
(3)
(4)
(5)


Instructions_1
"Title of the Design Task".
1. Please type your design ideas in the design area provided in the lower_right window.
2. Click the continue box when you have finished the preliminary design.
3. You can click any underlined items inside the information box to get more information on that topic.
Information
Information Booth in Shopping Mall
In general, there are one or more information booth in a shopping mall. The functions of an information booth usually include providing directional information to shoppers; providing various services such as lost and found, shopping carts rental, announcements, and so on.
Next:
1. Computer Course, or
2. Return to the Main List
Instruction_2
The purpose of this demo is for you to become familiar with the information box so that you will make use of the information later. Please practice by clicking some of the underlined items to view the contents of those topics.
(1)
(2)
(3)
(4)
(5)


Instructions_1
"Title of the Design Task".
1. Please type your design ideas in the design area provided in the lower_right window.
2. Click the continue box when you have finished the preliminary design.
3. You can click any underlined items inside the information box to get more information on that topic.
Information
Abstrating a Problem
Abstracting is a technique used by many people to simplify a complex problem by looking at only certain aspects of that problem at a time.
Two of the popular abstracting techniques are described in this topic:
a. Clustering
b. Highlighting
Or,
1. Design Techniques
2. Return to the Main List
Instruction_2
The purpose of this demo is for you to become familiar with the information box so that you will make use of the information later. Please practice by clicking some of the underlined items to view the contents of those topics.
[1]
[2]
[3]
[4]
[5]


Instructions_1
"Title of the Design Task".
1. Please type your design ideas in the design area provided in the lower_right window.
2. Click the continue box when you have finished the preliminary design.
3. You can click any underlined items inside the information box to get more information on that topic.
Information
Clustering
Clustering is to group items that have common properties. In the computer course design demo, the grouping of student background, student interest and the size of class into one issue as the student issue is an example of clustering.
Next:
1. Highlighting
2. Design Techniques,or
3. Return to the Main List
Instruction_2
The purpose of this demo is for you to become familiar with the information box so that you will make use of the information later. Please practice by clicking some of the underlined items to view the contents of those topics.
F
Ty
(1)
(2)
(3)
(4)
(5)

Appendix VII Abstraction Aids Provided in the Pencil and Paper Design Environment, without Structural Support

Pencil and Paper Design Exercise

Procedure:

Part I You will view two conceptual design examples:

Example 1: Design an Introductory Computer Course for First Year College Students

Example 2: Design an Information Booth for a Shopping Mall.

Part II You will be asked to perform a design task using pencil and paper. The design problem will be revealed to you after you have viewed the examples.

Example One : Design an Introductory Computer Course for First Year College Students

Preliminary Design

1. For first year students, a computer course should include some concepts and some hands-on practice.

2. The text book to be used should not be too difficult. Books with many colors will make the course seem more interesting.

3. Number of assignments to be handed out during the term.

4. The exam format. For example, if there are hands-on exercises, should there be lab tests also?

Summary

1. Course Content:
   - basic concepts of hardware and software

2. Course Structure:
   - lecture
   - lab exercises

3. Textbook:
   - contents
   - suitability
   - availability

Further Summary

1. Overall, there are several aspects that need to be addressed in this problem:
   - course content
   - course structure
   - text books

Detailed Design

Detail of 1. in preliminary design

1.1 For example, concepts of hardware configuration of a personal computer.

1.2 Concepts of systems software and applications software.

1.3 hands-on exercises in the computer lab to practice some of the popular applications software such as Lotus1-2-3 and WordPerfect.

Example Two: Design an Information Booth for a Shopping Mall

Preliminary Design

1. In this design, I will discuss the layout of an information booth and the services to be provided by an information booth.

2. Maybe I should also decide how many information booths there should be in a shopping mall.

Detailed Design

1.1 The layout of an information booth should adopt an open concept. That is, shoppers can approach the booth from any direction.

Further Detailed Design:

Further detail from [1.1]:
- A kiosk with big display signs that can be seen from all directions.
- Bright colors to attract shoppers' attention.

Summary

1. - Layout
   - Services
   - Number of booths

Design Exercise

Your design task is to

" Design a High School Counselling Office "

Please use the provided sheets for your design. You will have 30 minutes to perform your design. Thank you.

Appendix VIII Abstraction Aids Provided in the Pencil and Paper Design Environment, with Structural Support

Pencil and Paper Design Exercise

Procedure:

Part I You will view two conceptual design examples.

Example 1: Design an Information Booth for a Shopping Mall.

Example 2: Design an Introductory Computer Course for First Year College Students

Part II You will be asked to perform a design task using pencil and paper. The design problem will be revealed to you after you have viewed the examples.

Example One : Design an Introductory Computer Course for First Year College Students

Preliminary Design

1. For first year students, a computer course should include some concepts and some hands-on practice.

2. The text book to be used should not be too difficult. Books with many colors will make the course seem more interesting.

3. Number of assignments to be handed out during the term.

4. The exam format. For example, if there are hands-on exercises, should there be lab tests also?

Summary

1. Course Content:
   - basic concepts of hardware and software

2. Course Structure:
   - lecture
   - lab exercises

3. Textbook:
   - contents
   - suitability
   - availability

Further Summary

1. Overall, there are several aspects that need to be addressed in this problem:
   - course content
   - course structure
   - text books

Detailed Design

Detail of 1. in preliminary design

1.1 For example, concepts of hardware configuration of a personal computer.

1.2 Concepts of systems software and applications software.

1.3 hands-on exercises in the computer lab to practice some of the popular applications software such as Lotus1-2-3 and WordPerfect.

Example Two: Design an Information Booth for a Shopping Mall

Preliminary Design

1. In this design, I will discuss the layout of an information booth and the services to be provided by an information booth.

2. Maybe I should also decide how many information booths there should be in a shopping mall.

Detailed Design

1.1 The layout of an information booth should adopt an open concept. That is, shoppers can approach the booth from any direction.

Further Detailed Design:

Further detail from [1.1]:
- A kiosk with big display signs that can be seen from all directions.
- Bright colors to attract shoppers' attention.

Summary

1. - Layout
   - Services
   - Number of booths

Design Exercise

Your design task is to

" Design a High School Counselling Office "

Please start your design with the sheet labelled Preliminary Design. You can then proceed to the remaining sheets in any order that you desire. Ycu will have 30 minutes to perform your design. Thank you.

Preliminary Design

1.

2.

3.

4.

5.

6.

7.

Detailed Design

Detail from [1]:
1.1
1.2
1.3

Detail from [2]:
2.1
2.2
2.3

Detail from [3]:
3.1
3.2
3.3

Detail from [4]:
4.1
4.2
4.3

Detail from [5]:
5.1
5.2
5.3

Detail from [6]:
6.1
6.2
6.3

Detail from [7]:
7.1
7.2
7.3

Further Detailed Design

Further detail from [ ]:

Further detail from [ ]:

Further detail from [ ]:

Summary

Summary 1:

Summary 2:

Summary 3:

Summary 4:

Summary 5:

Summary 6:

Summary 7:

Further Summary:

Further Summary 1:

Further Summary 2:

Further Summary 3: