6-3-1997

Design and evaluation of a consulting system for database design

Solomon Raj Antony
Florida International University

Florida International University

Miami, Florida

DESIGN AND EVALUATION OF A CONSULTING SYSTEM FOR DATABASE DESIGN

A dissertation submitted in partial satisfaction of
the requirements for the degree of

Doctor of Philosophy

in

Business Administration

by

Solomon Raj Antony

1997

DISSERTATION COMMITTEE APPROVAL

To: Dean Harold Wyman
College of Business Administration

This dissertation, written by Solomon Raj Antony, and entitled "Design and evaluation of a consulting system for database design", having been approved in respect to style and intellectual content, is referred to you for judgement.

We have read this dissertation and recommend that it be approved.

Joyce Elam

Paul Hart

Radhika Santhanam

Dinesh Batra, Major Professor

Date of defense: June 3, 1997

The dissertation of Solomon Raj Antony is approved.

Dean Harold Wyman
College of Business Administration

Dr. Richard Campbell
Dean of Graduate Studies

Florida International University, 1997

ACKNOWLEDGEMENTS

First, I thank God for all the blessing and wisdom He showered upon me during this research project. I am grateful to Dr. Dinesh Batra, my major professor, for his patience and perseverance in encouraging me to excel in the planning, preparation, and execution of this thesis. He has been extra-ordinarily quick in giving me very useful feedback on this thesis and his insightful comments helped me improve on earlier versions of this document. I thank him for that. I am pleased to state that Dr. Batra has positively influenced my research and teaching style. Next, I am thankful to Dr. Radhika Santhanam for her constant encouragement and invaluable professional advice. Dr. Santhanam has been instrumental in shaping the research model and enhancing the research methodology and I thank her for that.

Dr. Joyce Elam contributed to this research study by ensuring that the standards are set high. Her valuable thoughts and comments, helped me shape the research idea. I am thankful to Dr. Paul Hart for his insightful and useful comments on the research questions. His suggestions helped me make the theoretical foundations stronger. I wish to acknowledge the help and support provided by Dr. Dan Robey as the Chair of the department. I remain thankful to Mr. Amir Motamedi, Coordinator Computing Services at FIU for technical support.

I am very thankful to all of my graduate student friends without whose encouragement and help I would not have completed my graduate program. I am thankful, in particular, to Sundeep Sahay, Ivy Palit, Nagraj Sivasubramanian, Sandipa Dublish, Nicole Wishart, Line Dube, Guy Pare, Martin Santana, George Marakas, Manoel Olivera, Len Gomez, and Minnie Dunbar. Additional "Thank you"s go out to Nicole and Sandipa for their help during data collection.

The one person I cannot thank enough is my dear wife Chitra. She successfully played the very hard role of the spouse-of-a-Ph d-student. Her sacrifices and patience enabled me focus on my studies and research. Her selfless acts of love and care helped me keep my spiritual and emotional life alive and healthy. I will remain indebted to her lifelong, and happily so!

ABSTRACT OF THE DISSERTATION

DESIGN AND EVALUATION OF A CONSULTING SYSTEM FOR DATABASE DESIGN

by

Solomon Raj Antony

Florida International University, 1997

Professor Dinesh Batra, Major Professor

Database design is a difficult problem for non-expert designers. It is desirable to assist such designers during the problem solving process by means of a knowledge based (KB) system. Although a number of prototype KB systems have been proposed, there are many shortcomings. Firstly, few have incorporated sufficient expertise in modeling relationships, particularly higher order relationships. Secondly, there does not seem to be any published empirical study that experimentally tested the effectiveness of any of these KB tools. Thirdly, problem solving behavior of non-experts, whom the systems were intended to assist, has not been one of the bases for system design.

In this project, a consulting system, called CODA, for conceptual database design that addresses the above short comings was developed and empirically validated. More

specifically, the CODA system incorporates (a) findings on why non-experts commit errors and (b) heuristics for modeling relationships. Two approaches to knowledge base implementation were used and compared in this project, namely *system restrictiveness* and *decisional guidance* (Silver 1990). The Restrictive system uses a proscriptive approach and limits the designer's choices at various design phases by forcing him/her to follow a specific design path. The Guidance system approach, which is less restrictive, involves providing context specific, informative and suggestive guidance throughout the design process. Both the approaches would prevent erroneous design decisions. The main objectives of the study are to evaluate (1) whether the knowledge-based system is more effective than the system without a knowledge-base and (2) which approach to knowledge implementation – whether Restrictive or Guidance – is more effective. To evaluate the effectiveness of the knowledge base itself, the systems were compared with a system that does not incorporate the expertise (Control).

An experimental procedure using student subjects was used to test the effectiveness of the systems. The subjects solved a task without using the system (pre-treatment task) and another task using one of the three systems, *viz.* Control, Guidance or Restrictive (experimental task). Analysis of experimental task scores of those subjects who performed satisfactorily in the pre-treatment task revealed that the knowledge based approach to database design support lead to more accurate solutions than the control system. Among the two KB approaches, Guidance approach was found to lead to

better performance when compared to the Control system. It was found that the subjects perceived the Restrictive system easier to use than the Guidance system.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Technical knowledge is a top priority for today's information systems (IS) professionals, as evident from analysis of advertisements for IS positions (Todd, McKeen and Gallupe 1995). Among the technical skills required, knowledge of Database Management systems (DBMS) and analysis and design skills rate among the top (Todd *et al.* 1995). The increased availability and use of inexpensive database management tools have lead to wider use of database technologies by expert and non-expert designers.

A *database* is a collection of shared, inter-related data (McFadden and Hoffer 1991). Database design consists of (i) analyzing the data requirements of the users and developing a conceptual model, (ii) translating the conceptual design into logical design and (iii) creating a physical design i.e. optimizing the logical design to satisfy performance considerations like response time and maintainability. Conceptual database design is one of the most critical phases in database design, because it is the basis for logical and subsequently physical design of the database. It allows the designer to capture the structural aspects of an application without becoming enmeshed in the implementation details. The conceptual design is used as documentation of the data requirements also. Even if the target DBMS changes, the conceptual design remains useful to the designer. Thus, the additional developmental costs are minimized when there are changes later

(Batini, Ceri and Navathe 1992). Conceptual database design involves identifying objects in the application using a data model such as the Entity-Relationship (ER) model (Chen, 1976). The impact of ill-conceived database design is a serious concern, especially when the mission critical applications are based on such designs (Martin and Leben 1995; Salchenberger 1993).

The conceptual database design task is a difficult problem and it cannot be fully automated. The designer has full responsibility of ensuring its accuracy (Batini *et al.* 1992). To partially automate the design process, a number of knowledge-based (KB) tools have been proposed (Storey and Goldstein 1993). Although those systems were proposed to help non-experts there are a number of short-comings. These are (a) the problem solving behavior of non-experts were not studied prior to the system development, (b) the knowledge embedded in those systems is not comprehensive with respect to modeling relationships and (c) there are no empirical evaluation of the prototype systems.

In this research project a consulting system for Conceptual Database design (called CODA), based on a design framework from the decision support systems field has been developed. This project addresses the following shortcomings of earlier projects that involved design of knowledge based database design tools: (a) The system design oriented research project is based on a theoretical framework, (b) The source of knowledge embedded in the system includes reasons for error commission, rules and heuristics in

database design, and (c) The effectiveness of the system in assisting non-expert designers has been empirically validated through experimental procedures. The most widely used data model for conceptual design is the Entity-Relationship (ER) model proposed by Chen (1976) and hence, ER model is the domain of this research project.

CODA differs from earlier systems in the breadth of knowledge sources used and in the knowledge base implementation strategies. The knowledge sources used in the design of CODA include (i) Entity Relationship modeling approach (ii) some rules and heuristics for ER modeling, (iii) analysis of novice errors in ER modeling and (iv) strategies on how the errors can be prevented. The knowledge embedded in CODA is expected to help the non-expert designers by preventing errors commonly committed and by providing opportunities for following a systematic problem solving process.

In most KB systems, error prevention and systematic problem solving can be effected by one of two implementation strategies (Silver 1990). Firstly, the design support tool can be programmed to limit the designer's choices and force him/her in following a pre-specified "normative" design path. By forcing the designer to follow a "normative" design process, the final design can be expected to be relatively error-free. Secondly, the system can be programmed to be less restrictive and provide context-specific guidance to the designer. Since non-expert designers find it hard to manage the design problem solving process, guidance in verifying various ER constructs can be expected to reduce their cognitive load. The system would help them focus better on the task specific details

and provide them opportunities to prevent and correct errors. At the same time, the designer will not be constrained by the system to follow a specific sequence of procedures.

The objectives of this research project are (a) to develop the knowledge-based system for conceptual database design (b) empirically validate the effectiveness of the knowledge based systems in assisting non-expert database designers and (c) to compare the effectiveness of the two knowledge implementation strategies. Two types of CODA implementations were developed. The first implementation is the Guidance and the second is the Restrictive . To study the effects of the knowledge based systems (i.e. the two implementations of the CODA) another system that does not provide any knowledge-based support to the designers (hereafter referred as the Control system) was also developed. The effectiveness of these systems in supporting non-expert database designers was studied in an experimental setting.

In Chapter 2, research findings in problem solving are reviewed. More specifically, the cognitive limitations of novices, effects of external interventions during problem solving, and strategies for better management of the problem solving processes are reviewed. The research involves development and empirically validating of a software program. So, the research framework suggested by Wall, Widmeyer and El Sawy (1992) is used to present the literature review. In Chapter 3, important features the CODA implementations are described. The similarities and differences among the three types of systems (Control,

Restrictive and Guidance) are also described in Chapter 3. In Chapter 4, the research model is presented. The research model relates the effects of software intervention and user characteristics on performance. The research questions based on the model are also presented. In Chapter 5, the research methodology and research procedures are detailed. A pre-pilot and a pilot test were conducted before the main experiment. Details of statistical analysis and results are given in Chapter 6. In Chapter 7, the implications for researchers and practitioners are elaborated. Future research directions and limitations of this study are also discussed in Chapter 7. In the last chapter, the findings of the study are summarized.

Chapter 2

LITERATURE REVIEW

The research project involves the design and development of a knowledge based system for database design and the empirical testing of its effectiveness. Walls, Widmeyer and El Sawy (1992) have proposed an Information System Design Theory (ISDT), which describes how system development oriented research can be formulated as a scientific process. Using ISDT for system design projects would lead to improved comprehension of reality, just as scientific methods improve our comprehension of reality. Since *design* is a noun as well as a verb, it can be equated with the *design product* and *design process*. In this research project, the focus is on the final product, that is CODA, rather than the process of design of the software. The components of ISDT include (1) meta-requirements, (2) meta-design (3) kernel theories and (4) testable design product hypotheses. The theoretical research framework is shown in Figure 1. The components of ISDT are shown on the left. The fields from which theoretical and empirical support are drawn, are shown on the right column. There is congruence between the ISDT component and the corresponding research fields.

Meta-Requirements - Objectives of system development

Meta-requirements mean requirements of requirements. Meta-requirements refer to the class of goals which are to be achieved by the design of the system. Hence, the underlying objectives of design and development of the consulting system for conceptual

database design are described in this section. The main objective behind the development of such a system is to support non-expert designers during the process of database design. Non-expert designers can be supported by providing methods and opportunities for error prevention. To provide such support, knowledge of what types of errors are commonly committed and what causes them is essential. It is also necessary to demonstrate the need for development of such a consulting system and is necessary to contrast the system with any previously developed system.

ISDT Component	Literature source
Meta-requirements	Error prevention; Support for non-expert problem solving
Meta-design	Expertise in data modeling; Knowledge based systems
Kernel Theories	Problem solving; Software intervention
Testable design hypotheses	Knowledge based support

Figure 1: Theoretical framework

Error prevention

If the objective is to help prevent errors, we need to understand why it is important, what types of errors there are, why they occur and what can be done to prevent them. This line of reasoning concurs with Norman (1983), who suggested that analysis of errors that users make while using computer systems be used by designers of systems. He advocates that analysis of people's performance - but especially their errors - be used in construction of human machine interfaces. Although his recommendations were for designers of computer interfaces, they are applicable to system designs at large. A system thus designed could be expected to minimize the incidence of errors. Thus we find that effective system development warrant the analysis of erroneous behavior of non-expert problem solvers.

One of the prime objectives of information systems is that they should help overcome human limitations. Tendency to commit errors is a human limitation. Rouse (1991), based on a generic model of problem solving, classified human errors. The problem solving model consists of six stages, namely (i) observation of system state (ii) choice of hypothesis (iii) testing of hypothesis (iv) choice of goal (v) choice of procedure and (vi) execution of procedure. Erroneous actions at each stage are categorized with respect to the (i) completeness of the procedure, (ii) correctness of procedure, and (iii) correctness of variables used. For example, during the second stage - choice of hypothesis - the subject may not have considered all the hypotheses. The author recommends that process based

data be used to study errors. The author concludes that there are four very general causes of errors: (i) inherent human limitations (ii) inherent system limitations (iii) contributing factors, and (iv) contributing events. Similar causes have been found by other researchers as well (e.g. Reason 1988).

On analysis of a number of models of human performance in problem solving, Reason (1988) concludes that most errors arise from a natural tendency to minimize cognitive strain and a tendency to over-utilize stored knowledge structures, heuristics and short-cuts that help in simplifying complex problem solving processes. Errors arise from misapplied expertise, from confirmation bias, from spillage out of the limited workspace and from the use of an inaccurate or incomplete knowledge base. The behavior of a problem solver can be described as the utilization of attentional and cognitive resources. Since novices cannot allocate attentional resources efficiently, their performance suffers. Batra and Antony (1994a) also found that most of the errors committed by novice database designers were attributable to effort minimization strategy, misapplied heuristics and incomplete knowledge base. We can comprehend the effects of incomplete knowledge, if they are considered along with the level of difficulty of the problem. Since the domain where the system is useful is database design, research in database design, with a focus on errors, is reviewed next.

Errors in Data modeling

Researchers in database design have been interested in the usability of different data models and modeling methods. Typically the usability studies have compared designer performance between two data models. Many of them have compared the usability of ER model with other models and found that ER model rated higher in usability (Juhn and Nauman 1985; Batra, Hoffer and Bostrom 1990; Ridjanovic 1986; Jarvenpaa and Machesky, 1989). Typically these studies compared the performance of designers and did not detail the reasons for poor performance, if any. Batra *et al.*(1990) classified the errors as relationship, entity and attribute errors. Batra and Kirs (1993) found that most errors were related to degree and connectivity of relationships. They found that designers had little difficulty in modeling entities and attributes, but relationships were more difficult. Jarvenpaa and Machesky (1989) assessed the performance on the basis of number of entities, number of attributes and number of relationships in the solution, but they did not discuss errors. Mantha (1987) attributes the poor performance of designer to the lack of experience.

Comprehension of the underlying *process* would help the users correct design errors (Batra and Srinivasan, 1992). Batra and Davis (1992) compared the design processes of novices and experts. Their analysis was at episodic levels (i.e. enterprise, recognition and representation) levels. The authors found that novices tended to have more errors in their solutions. They have attributed novice errors to the inexperience and inability of

novices in pursuing a focused effort in integrating the various parts of problem description.

Srinivasan and Te'eni (1990) studied database design using process tracing techniques. The tasks they administered included data model building and testing of the models using feedback on queries the subjects constructed. The 'constant testing of the evolving design' approach was found to lead to more accurate data representations. The authors attributed erroneous design by some subjects to their prolonged modeling activities at the lowest levels of abstraction, i.e. at the attributes and entity level. Although they reported the frequency of errors committed by subjects, they did not discuss what *caused* the commission of those errors.

Database design is a complex task. While solving complex tasks, people use heuristics. Hence, database designers may be expected to employ heuristics. However, use of heuristics has an associated risk of committing 'biases' (Kahneman, Slovic and Tversky 1982). Hence, possible explanation for novice errors can be found from literature on the use of heuristics and related biases. Study of biases as a basis for classifying why people commit errors is an appropriate approach (Senders and Moray 1991). Batra and Antony (1994a) used protocol data to identify cognitive and process oriented causes that lead to database design errors. They classified the relationship related errors into (i) high-as-low degree errors, (ii) low-as-high degree errors and (iii) connectivity errors. The main causes of errors include (i) over-reliance on literal translation of case description into

relationships (ii) anchoring, i.e. designer's persistence in retaining initial design and (iii) incomplete conceptual and procedural knowledge. Their recommendations, in order to prevent some of the errors include (1) providing feedback based on the correctness of the design, (2) providing tools that can translate from one representation to another, (3) providing a tool that can help the designer with connectivity checks, (4) advising designers about the pitfalls with literal translation and better instruction of "do's and don'ts". They also recommend that the design support tools should (6) help in preventing derived relationships and (7) in preventing fragmented designs. Most of their recommendations can be implemented in a design support tool and in training manuals.

Usually the correctness and quality of a conceptual database design is evaluated by the final design outcome only. For example, the typology of errors in the Batra and Antony (1994a) study - high-as-low degree, low-as-high degree and connectivity errors - are based on the final designs only. The availability of process data enabled comprehension of the design decisions that were made during the problem solving process. By studying the problem solving behavior, it is possible to identify the specific decision(s) that manifest as errors in the final design. Since the errors that are identified in the final design could have been caused by more than one erroneous design decision, it is necessary to use process data to pinpoint the first and subsequent erroneous design decisions. The protocol data that were collected for the Batra and Antony (1994a) study were analyzed to identify the erroneous design decisions made by novice designers. While developing

the list of errors, only those instances which could be unambiguously identified are reported below. The list of erroneous designs decision can be categorized as entity related and relationships related errors.

Entity related errors:

- Defining entity without identifying the key and non-key attributes
- Assigning attributes to entities without ensuring that they are dependent on the identifier.
- Modeling an attribute that is not assigned to any entity as another entity. Not checking if its dependency on any of the key-attributes or verify if it is a multi-valued attribute.

Relationship related errors

- Deciding on *many-many* relationship without consideration of higher order relationships.
- Defining a relationship without determining and verifying the key and non-key attributes of the entities involved.
- Incorrect verification of connectivity.
- Incorrect verification of attribute(s) of relationship.

In most circumstances, these instances of erroneous design decisions cannot be discovered by analyzing the final solutions alone. These decisions may lead to other types of errors which are identifiable in the final solution. Consider, for example, a problem where there is a ternary relationship between the Employee, Project and Skill entities. Assume

the designer did not model Skills as a separate entity but modeled it as an attribute of Employee. Although Skills is multi-valued with respect to the Employee entity (An employee may have many skills), the designer might have "assigned the skill attribute to the Employee entity without ensuring whether Skills is single valued with respect to the key attribute of Employee". This design decision would lead to a design solution, where a ternary relationship is incorrectly modeled as binary relationship. Thus the original cause of an incorrect design solution can be determined.

Meta-design: Design strategies that meet meta-requirements

Meta-design describes the "class of artifacts hypothesized to meet the meta-requirements" (Walls *et al.* 1992). Most of the erroneous design decisions mentioned in the previous section can be prevented and non-experts can be supported by appropriate training and computer based interventions during the design process. There have been attempts at developing expert systems for database design. In this section, some of the knowledge based systems are reviewed and sources of expertise for SODA are detailed.

Knowledge based systems for database design

The tools that support conceptual design phase are discussed. Specifically, the focus of the discussion is on the extent of conceptual modeling expertise that is implemented in the tool and empirical evaluation of each system. Intelligent Interview Systems I^2S (Kawaguchi *et al.* 1986) emulates the interviewing process of a database design expert for

eliciting data requirements from users. The system relies on a number of 'template' applications and can be effective only for those applications.

View Creation System (Storey 1988) engages the user in a dialog to determine the entities and attributes. It can infer relationships between entities and model them. However, the system can model only binary relationships.

Computer Aided Requirements Synthesis (Demo and Tilli 1986) helps the users in integrating many Entity-Relationship models into one user view. The system can also model the relationships based on an analysis of the user data requirements. However, if there are sentences that relate two entities a binary relationship is modeled. The possibility of higher order relationships may be ignored. Modeller (Tauzovich 1989) translates the user requirements into entities and relationships. The requirement statements may include sentences that may lead to derived relationships. Modeller may not detect and prevent such instances. Expert Database Design System (Choobineh *et al.* 1988) analyzes *forms* and produces the E-R diagrams. The system automatically models binary relationships. However, higher order relationships are designated as exceptions and the user has the responsibility of modeling such relationships.

The source of knowledge for these knowledge based systems include standard text books and personal experience. The knowledge required for specifying and verifying entity definition have been embedded in some of these tools. However, the knowledge base

required for modeling relationships seems to be incomplete for a number of reasons. Firstly, most of these systems can handle only binary relationships. However, in real business problems, higher order relationships are common place. For example, there are many business applications that call for ternary, 4-way and even 5-way relationships. Scheer (1989) provides examples from manufacturing, purchasing, sales and other business functions that have higher order relationships. This artificial limitation to problems that have only binary relationships is a serious concern. For non-experts modeling relationships is more difficult than modeling entities and attributes (Batra and Kirs 1993). An expert tool developed to support the non-experts should help them with modeling higher order relationships for several reasons. During problem solving non-experts may represent higher degree relationships as lower degree and vice versa. Their models typically include relationships that can be derived from other relationships also. Hence, any support tool should help the non-experts manage such problematic design decisions.

Although the expert tools can suggest relationships based on the user data requirements, they do not have the complete semantic knowledge of the application. The decision of selecting between binary and ternary relationships eventually rests on the users only. Thus it is possible that a simple consulting system literally translates the requirements into relationships constructs without verifying the correctness of such representations.

Storey and Goldstein (1993), in their review of the expert tools, conclude that tools cannot replace an expert and that they can only be used as intelligent assistants to the designer. The lack of empirical validation of any of the expert tools has also been noted. In the next section, the extent of knowledge or expertise that can be implemented so that the final designs are error free, is discussed.

Expertise in database design

Since most of the applications are based on the Relational model (Codd 1970) the design methodology that is adapted should enable the designer in identifying normalized relations. The ER modeling methodology has been analyzed and well understood. An accurate attempt at following the ER methodology should enable the designer to truly represent the data requirements as entities and relationships. Translation from ER diagrams to the relational model is an algorithmic approach and it can be automated (see. e.g. Ram 1995). The expertise needed to arrive at the correct ER diagram is detailed next.

Each entity should capture information about a certain object about which there are some descriptive information. Each entity should have a key attribute that has unique value for each instance of the entity. The functional dependency between key and non-key attributes should be in the third normal form.

A relationship is the association between entities. Problems may have combinations of unary, binary, ternary and 4-way relationships. Unary relationships is a relationship

with two instances of the same entity. Binary relationship involves two entities, ternary involves three entities and so on. The number of entities participating in a relationship is called the *degree of the relationship.* Another characteristic of the relationship is the *connectivity.* Connectivity refers to the combination of cardinalities of entities in the relationship. Cardinality of an entity in a relationship can either be *one* or *many.* It is the number of instances of the entity that is associated with an instance of each other entity in the relationship. For example, if there is a relationship, called Employs between Department and Employee entities, the cardinality of the Department entity is the number of instances Department that may be associated with one instance of Employee. Similarly, the cardinality of Employee is the number instances of Employee that is associated with one instance of the Department entity. So, the degree of the Works relationship is *one-many.*

The choice of relationships must primarily be based on the data requirements. Not all of those relationships would be part of the final solution. This is so, because some relationships can be derived from others. Batra and Zanakis (1994) report that there is a specific procedure which would prevent derived relationships. They state that the binary relationships that have at least one *one* cardinality should be modeled first. After defining all the binary *one-many* and *one-one* relationship, the entities that are on the *one* sides of the relationships will not be considered for further relationships. The other entities that are available for relationships are called free entities. If there are three or

more free entities, the designer should model *one-one-many* or *one-many-many* relationships next. The designer should model ternary *many-many-many* relationships next. The designer should model the binary *many-many* relationships in the end. The entities that participate in a relationship should not be a sub-set of or equal set or super-set of entities from another relationship. This restriction prevents the possibility of derived relationships. The ER diagram based on the above set of guidelines can be translated into relational model using established methodologies (e.g. Teorey, Yang and Fry, 1986) and the resulting relations will automatically be in third normal form. In the next, section the consulting system for database design is described.

Kernel Theories - Useful theories governing meta-requirements

Kernel theories refer the set of theories from natural or social sciences that govern the meta design requirements. Since the focus of this study is in assisting non-experts in database design which is a problem solving exercise, theories from novice behavior and problem solving are representative kernel theories. In this section relevant aspects of those theories and findings are described.

Problem solving

Problem solving is defined as a behavior directed towards achieving a goal (Anderson 1985). In an effort to understand the mechanisms of human problem solving, Newell and Simon (1972) analyzed the processes of problem solving in certain domains, such as chess and crypt-arithmetic. Their main findings, based on the analysis of the processes

of solving problems indicate that (i) people use limited short-term memory, (ii) they retrieve relevant knowledge from long-term memory, and (iii) they exhibit means-ends strategy. Researchers who studied physics, mathematics and other semantically rich (Bhaskar and Simon 1977) domains found that problem solving involves searching one's knowledge in order to find schema and necessary operators for solving the problems. A schema is defined as a cognitive construct that permits the problem solver to recognize a problem as belonging to a specific category which the problem solver has encountered previously (Sweller 1989).

The knowledge required for problem solving consists of conceptual and procedural knowledge (Anderson 1985; Chi, Feltovich and Glaser 1981). Conceptual knowledge is the knowledge of the principles of the domain. Procedural knowledge is the knowledge of the operators and operations to be carried out to solve the problem. In addition to conceptual and procedural knowledge, the utilization of strategic knowledge during problem solving behavior is also found (Heller and Hungate 1985). Strategic knowledge refers to formulation of the sequence of procedures for problem solving. Application of the means-ends strategy is an example of strategic knowledge used to solve problems with known goal state (Newell and Simon 1972).

Evidence for use of different strategies for problem solving can be found from research on expert-novice differences. Novices typically follow an effort minimization strategy. When the goal state is known in advance they work backwards using means-ends

strategies (Larkin, McDermott, Simon and Simon 1980), whereas experts work forward from the basic principles (Hinsley, Hayes and Simon 1977). Expert problem solvers have easy access to conceptual and procedural knowledge that is stored in a pre-compiled form in their long-term memory (Dreyfus and Dreyfus 1986).

The construct of cognitive resources and attentional resources plays an important role in explaining problem solving behavior (Kahneman 1973; Norman and Bobrow 1975). Performance in problem solving behavior is affected by (i) the limitations of the cognitive resources that can be devoted to the task and (ii) the task characteristics, .i.e. the level of difficulty of the problem (Norman and Bobrow 1975). During problem solving the subject has to be aware of and make use of the declarative knowledge as well as comprehend the problem domain specific information. An expert designer is less likely to be overwhelmed with allocating attentional resources between the tasks of comprehending the domain specific information and utilizing the declarative knowledge that are specific to the database design than a novice designer. There is evidence that experts take a different approach to manage their cognitive resources.

McKeithen, Reitman, Reuter and Hirtle (1981) report that expert computer programmers are capable of organizing information more meaningfully. Bouwman (1983) found that expert financial decision makers periodically summarized their results and formulated useful hypotheses. However, novice decision makers were found to have less control over their problem solving process. Experts in software development also

display better control of their cognitive resource allocation. They (i) break the problem down to manageable parts (ii) understand a problem before breaking it down to parts and (iii) they retrieve known solution if one exists (Jeffries, Turner, Polson and Atwood 1981). Similar to findings from other domains, Batra and Davis (1992) found that expert performance in conceptual database design also reflects a relative ease in categorizing the problem.

Although most times, allocation of cognitive resources is done sub-consciously, the problem solver can execute explicit steps for better cognition. Self-monitoring is one such method (Kanfer and Ackerman 1989). It refers to the individual's allocation of attention to specific aspects of his or her behavior. By conscious effort at management of the cognitive resources, we can expect the problem solver to perform better. Management of one's own cognitive processes is recommended by researchers in education also. Presley *et al* (1990) recognize the importance of the deliberate use of various cognitive strategies in an effort to help the students gain knowledge and skills necessary for managing their own learning.

In summary, we find that the novice problem solvers can improve their performance by exerting conscious control over their cognitive activities. They can improve the management of their cognitive resources by explicit self-monitoring activities. However, since their knowledge base is not in a pre-compiled form they expend more resources at retrieving and processing of declarative and procedural knowledge. It is possible to help

them with external processing resources such as computational aids. In the next section, research on effect of software intervention to aid in non-expert problem solvers are presented.

Software interventions

When a computer system is used for problem solving, it is possible to create interventions that would improve the performance. Computational resources can provide the problem solver with external processing and memory capacity. Thus the system may enable him/her to manage the problem solving process more efficiently. In addition the system may be proactive by preventing erroneous problem solving steps. In this section, research on concepts, development and effectiveness of systems for supporting problem solving are detailed.

Carroll and Kellog (1987) have compiled research on many issues that expert system developers face. Their focus is on design of the interface of the advice giving expert systems. They express concern for the lack of empirical validation of the many advisory strategies. The authors, describe the "control blocking" feature as design features where some options having been made inaccessible to the user. If the system's particular feature is not applicable to the current task the user is performing, then it makes sense to block those features off. Similarly, system controlled dialogue, where the system asks the questions and the user answers them, has been proposed by Clancy (1982).

Duffield (1991) identifies the factors that influence the effectiveness of computer software designed to support problem solving. The factors include the computers' ability to (i) to retrieve and present data and information that can enhance problem solving (ii) to monitor and suggest problem solving strategies and (iii) to provide immediate and context sensitive explanation. She specifies that the software for problem solving should free the learners of unnecessary demands on short-term memory.

Many facts of problem solving are affected by cognitive biases that people inadvertently use (Sage 1981). It is desirable to develop systems that prevent the occurrence of cognitive biases during problem solving. Paradice and Courtney (1988) developed a prototype system (called SmartSLIM) for problem formulation that prevent many of the biases. SmartSLIM incorporates step-by-step procedure to acquire knowledge about each variable and relationships between the variables in a systematic manner.

It is possible not only to develop a system that prevents occurrence of biases, but positively affect the performance by aiding the user with error prevention mechanisms. Anderson, Corbett, Fincham, Huffman and Pelltier (1992) report their experiences in developing an intelligent tutoring system for students in a programming language class. Their learning model is based on Anderson's (1983) ACT* theory. They recommend that the students must follow a predefined correct path in solving problems. If the student selects an incorrect path a feedback mechanism informs him/her of the potential error and the student is not allowed to proceed unless the error is remedied. In the

description of their tutoring system Anderson, Boyle and Reiser (1985) recognize the need to help students with managing the memory load. This is accomplished by having the system encode and display on the screen much of the information the student is likely to forget.

The system can be used to (i) direct the user with step-by-step prompts or (ii) allow more independence to the user, but provide assistance and guidance when he/she faults. Town and Munro (1992) report the use of a system to explain complex avionics devices.

Which of these two methods - making the user follow a step-by-step procedure or providing assistance and guidance to user - is more effective? Leeuw (1983) tested to the performance of two groups of subjects in solving *number series extrapolation* problems. One group was trained in using an algorithmic approach and other was trained in a heuristics approach. The algorithmic method involves specific steps to be followed in finding a regularity in the series of numbers. The heuristic method is aimed at helping the students in systematically testing various hypotheses about the regularity in series extrapolation. The help was facilitated by different levels of feedback (some generic and other specific) messages. The author reports that the performance of students using the algorithmic method were superior to those using the heuristic method. In the same study, (de Leeuw 1983) the author reports the performance of two groups of students solving syllogism problems. The author found that the students trained in using an algorithmic method performed better than those who used a heuristic method in solving

simple problems. However, for more complex problems the performance of subjects trained in heuristics was better than those using the algorithmic method. The test problems were given to the students later (after a few weeks) to test for differences in memory retention. Students trained in using the heuristic method performed better in the retention tests than those trained in the algorithmic method. The heuristic method was aimed at prompting the subject in formulating and testing various hypotheses. In other words, the subjects were provided with opportunities to formulate questions and find answers to them.

King (1991) reports that meta-cognitive "questioning and answering" during problem solving help the subjects in finding the correct solution. Meta-cognitive questioning and answering refer to explicit attempts in controlling the cognitive processes by the problem solver. Typically the questions include such open-ended context free questions such as "What is the plan?", "What do we know about the problem so far?" and "Do we need a different strategy?". Such self-questioning prompts the subjects to take a more focused look at the problem and helps them improve their performance. In the experimental setup King (1991) used a control group and two treatment groups. One of the treatment groups received training on questioning and answering and were provided with a list of questions that can be used while solving the problem (Guided questioning). The other treatment group received training only on questioning and answering (unguided questioning), but were not given a list of questions. The control group received neither

training nor a list of questions. One of the major findings is that guided questioning strategy results in superior performance than unguided questioning strategy.

Heller and Reif (1984) studied the effects of *prescriptive theory* based training. The prescriptive model specifies that the exact steps to be carried out in applying the declarative knowledge to generate a theoretical description of the mechanics problems. While training, considerations were given to incorrect preconceptions and deficient knowledge that novices possess, so that the common errors could be prevented. In a controlled experimental setting, the authors compared effects of prescriptiveness. They found that the performance of subjects in the treatment group was significantly better than the performance of subjects in the control group.

Tarmizi and Sweller (1988) compared the performance of two groups of students solving geometry problems. The treatment group was trained to use a pre-specified precise set of transformation rules. The students in the control group were not trained to use any systematic rule. They did not find a significant difference in performance between the two groups. They theorize that additional cognitive load of following the precise steps may have dissipated any reduction in demand for cognitive resources. Thus we find there are two approaches to helping problem solver. Firstly, the users can have access to the step-by-step, algorithmic procedural knowledge either by training or through the software. Secondly, the users can be made to exercise better control of their cognitive

activities with the assistance from the system. These approaches appear very similar to the approaches suggested by for developing decision support systems.

Researchers in decision support systems are interested in effectiveness of artificial aids that help in solving managerial decision problems. Decision support systems are viewed as *change agents*, that propel the decision makers from using a non-normative model of problem solving to using a normative model of problem solving. The change in user's problem solving behavior can be effected with *system restrictiveness* and *decisional guidance*, that can be implemented in decision support systems (Silver 1990).

Systems restrictiveness refers to "the degree to which and the manner in which a decision support system limits its users' decision making [problem solving] processes to a subset of all possible processes" (Silver 1990). The concept of system restrictiveness can be applied to other support systems that aid problem solving. For most problems, there are usually a very few systematic and correct ways to the solution. A system which restricts the problem solver from selecting inappropriate intermediate steps would help in steering the problem solver in the right direction. Thus, a user of a system with restrictiveness is likely to find the correct solution using a correct sequence of solution steps. The objectives that favor greater system restrictiveness are (i) prescription of desirable techniques (ii) proscription i.e. preventing the user from choosing undesirable techniques, (iii) providing structure to the problem solving process (iv) promoting ease of learning

and ease of use of the system and (v) fostering structured learning of the problem solving technique (Silver 1991).

Decisional guidance refers to "the degree to which and the manner in which a decision support system guides its users in constructing and executing decision making [problem solving] processes, by assisting them in choosing and using the operators". It refers to the mechanisms for enlightening the user of the appropriate choices that can be made and possibly informing him/her about the consequences of selecting any of the choices. The objectives of developing a system with guidance are as follows (i) It can prevent the user being influenced by undesirable cognitive biases (ii) it prescribes the use of desirable problem solving process and (iii) it fosters structured learning (Silver 1991).

Silver (1990) has elaborated on how these concepts of restrictiveness and guidance can be implemented in decision support systems. He recommends that research be conducted in evaluating effectiveness of support system that incorporate system restrictiveness and/or decisional guidance.

Similar to managerial decision making, systems development is a complex task and a methodology is often used to manage complexity (Pressman 1982). Since there is a normative manner of solving system development problems the system developers usually are trained to use the normative methodology. However, due to various reasons the developers may not be able to follow the normative method. With the advent of

computer-aided design tools, it is possible to incorporate restrictiveness and guidance in system design support systems.

In an effort to find the extent of restrictiveness and guidance in CASE tools Vessey, Jarvenpaa and Tractinsky (1992) evaluated a number of commercial CASE tools. The tools support structured analysis and/or structure design techniques. The authors compiled a sets of rules for each technique from popular texts. They developed sets of checks (for example, "Must a process have at least one input and one output data flow?") for each structured technique. They evaluated the presence or absence of these checks and *how* they were implemented in each of those commercial CASE tools. The checks were (a) implemented as warnings or errors, (b) activated during the design phase or after the design phase and (c) set to be active on request or automatic. Based on an analysis of the checks implementation strategies the authors classified the tools as either guided or restrictive or flexible. They note that a clearly defined tool philosophy would be beneficial to the vendors as well as the users.

Dos Santos and Bariff (1988) compared the performance of subjects (treatment group) who were required to follow certain sequence in selecting variables in a DSS environment with the performance of subjects whose systems did not require them to follow any specific sequence in selecting variables. Thus, the types of systems they compared are similar to what are referred to as Restrictive and Control systems in this study. They found that the performance of treatment group subjects was significantly

higher than the control group. They report that the more restrictive DSS usage resulted in better performance. Computer logs of subjects who used the less restrictive system showed that they had not adopted a structured process in solving the problem. The difference in performance is attributed to the restrictiveness of the system used by the treatment group.

Summary of literature review

The domain of this research project is data modeling with the Entity Relationship model. Researchers in data modeling have attributed poor performance to (a) incomplete knowledge and insufficient experience (Mantha 1987) (b) systematic errors that are attributable to over-reliance on naïve heuristics (Batra and Antony 1994a) (c) effort minimization strategies such as making early design decisions and not revising them later (Batra and Antony 1994a). The meta-requirements of system development is to assist the non-expert designers by (a) providing opportunities for error prevention (Anderson *et al.* 1992; Carroll and Kellog 1987) (b) external memory and processing power so that the their cognitive load is reduced (Anderson *et al.* 1985; Duffield 1991) and (c) provide help with structuring the problem solving process (Reason 1988; Rouse 1991; Paradice and Courtney 1988). The meta-design component of this research project includes compilation of expertise on conceptual data modeling as reported by Chen (1976), Teorey (1990), and Batra and Zanakis (1994). The earlier knowledge based tools for database design, lacked certain capabilities such as modeling higher degree relationships.

Further more, those systems could not prevent errors like those resulting from biases like literal translation and anchoring. Previous knowledge based tools have not been empirically validated (Storey and Goldstein 1993). Hence, it is appropriate to develop a knowledge based consulting tool, that incorporates mechanisms for error prevention. The possibility of developing such error preventing systems, particularly for preventing cognitive bias based errors, has been demonstrated by Paradice and Courtney (1988). Such systems should prevent errors (Carroll and Kellog 1989), and help users manage their cognitive load (Anderson *et al.* 1985). Studies have reported tools that impart step-by-step procedures and provide guidance (Town and Munro 1992; Dos Santos and Bariff 1988) are described. The embedded step-by-step procedure was found to be effective in certain domains (*e.g.* Leeuw 1983). Systems that provide tools for better management of their cognitive resources have been found to be effective too (King 1991). The two distinct approaches - restrictive and guidance - in assisting users have been elaborated by Silver (1990) for systems for decision support. The Restrictive system provides a step-by-step procedure for problem solving with little allowance for deviation from the pre-specified path. The Guidance system is less restrictive and enable the users to manage the problem solving process by providing context specific guidance and assistance through messages and warnings. Research in system assisted problem solving points to the benefits of both approaches. In the next chapter features of the consulting system for database design are described.

Chapter 3

DESCRIPTION OF THE CONSULTING SYSTEM IMPLEMENTATIONS

Three systems for conceptual database design were developed in a programming language for GUI applications. They are Control, Guidance and Restrictive types. The Control system does not offer any knowledge based assistance to the user. The Guidance and Restrictive systems do offer knowledge based assistance. This chapter includes (a) a description of the knowledge embedded in the system, (b) a description of the Restrictive system, (c) a description of the Guidance system and (d) an articulation of the differences between the two knowledge based systems using a framework proposed by Silver (1988).

Knowledge base in the systems

The knowledge implemented in the system is summarized below. The source documents include Teorey (1990), Batini, Ceri and Navathe (1992) and Batra and Zanakis (1994).

Entities: Each defined entity would have to have an attribute as the key. Each non-key attribute value must depend on the key attribute and no other non-key attribute. The system allows only one key attribute to each entity. An attribute used for an entity cannot be used for any other entity. There cannot be two attributes with the same name.

Sequence of Relationships: After all entities have been defined (*i.e.* key and non-key attributes have been assigned), model all binary *one-many* type relationships first. After, all binary *one-many* relationships have been defined, only those entities which have not

been assigned to any relationships and those which are on the *many* side of the binary relationships will be considered for other relationships. Those entities are called "free" entities. If there are less than 2 free entities, no more relationships are possible. If there are only 2 "free" entities, only a binary *many-many* type relationship is possible. If there are 3 or more "free" entities, ternary relationships can be modeled. Among, ternary relationships *one-one-many* and *one-many-many* relationships are modeled next. Then, if there are three or more "free" entities, ternary *many-many-many* relationships can be modeled. After modeling ternary relationships, binary *many-many* relationships will be modeled.

Connectivity: The system would frame and display the question to the user. For example, if the user models a binary relationship called WORKS between EMPLOYEE and DEPARTMENT, the system would frame the question "Given one instance of employee, how many instances of department can there be in the WORKS relationships?" with an option to specify either *one* or *many*. If it is possible to infer the connectivity, without asking the user, the system would automatically fill the connectivity.

Derived relationships: The system would not allow (or issue a warning) if the user attempts to select a set of entities for a relationship if that set is a subset of, or superset of or equal set of entities in another relationship.

Attribute of a relationship: The system would help determine the correctness of assigning an attribute to a relationship.

On-line assistance: (For Guidance only). The system has an on-line guide which can be accessed from the main menu of the program. The on-line guide informs the user about the sequence of relationship types to be modeled.

Description of the Restrictive system

In this section, the description of the Restrictive system is shown as per the user's perspective. As soon as the software is loaded, the user is prompted to enter his/her name and other details (See Figure 2). This information will be stored in the log file and used in printing the E R diagram.

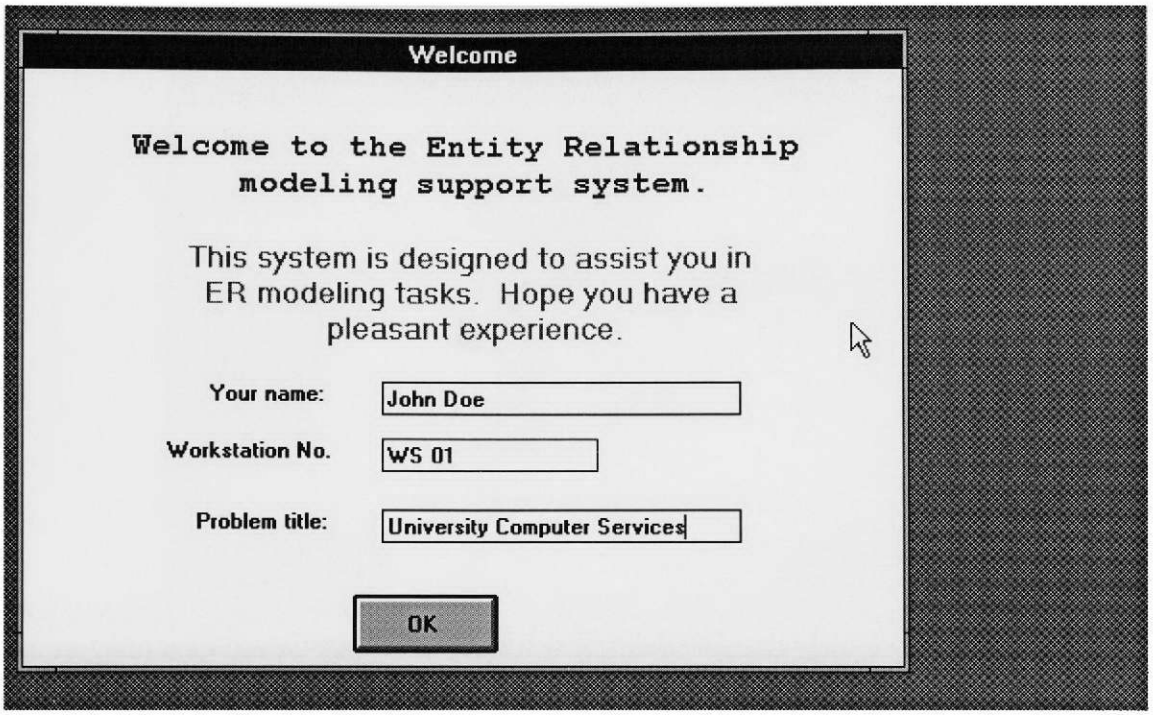


Figure 2 Restrictive system: Welcome screen

The user can then input the attribute names by choosing the Attributes option from the main menu. Notice that the only relevant menu options are enabled. The system can store up to 40 attributes. No two attributes can be the same. The Entities option will be enabled only after at least two attribute names are entered in the system. The attribute entry screen keeps track of which attribute has been assigned to an entity. An attribute that is assigned an entity cannot be assigned to another entity. Snapshot of the main screen and the Attributes entry window are shown in Figure 3 and Figure 4. The user can rectify any entry errors by removing the attribute from the list and re-entering the value.

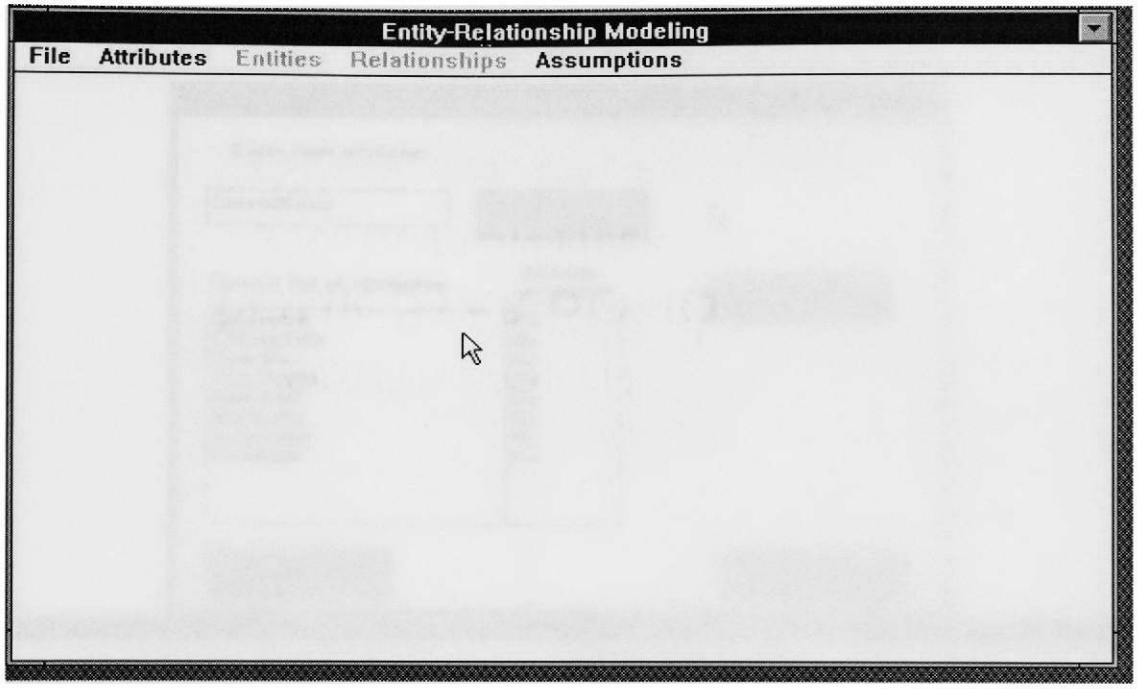


Figure 3 E R diagramming window

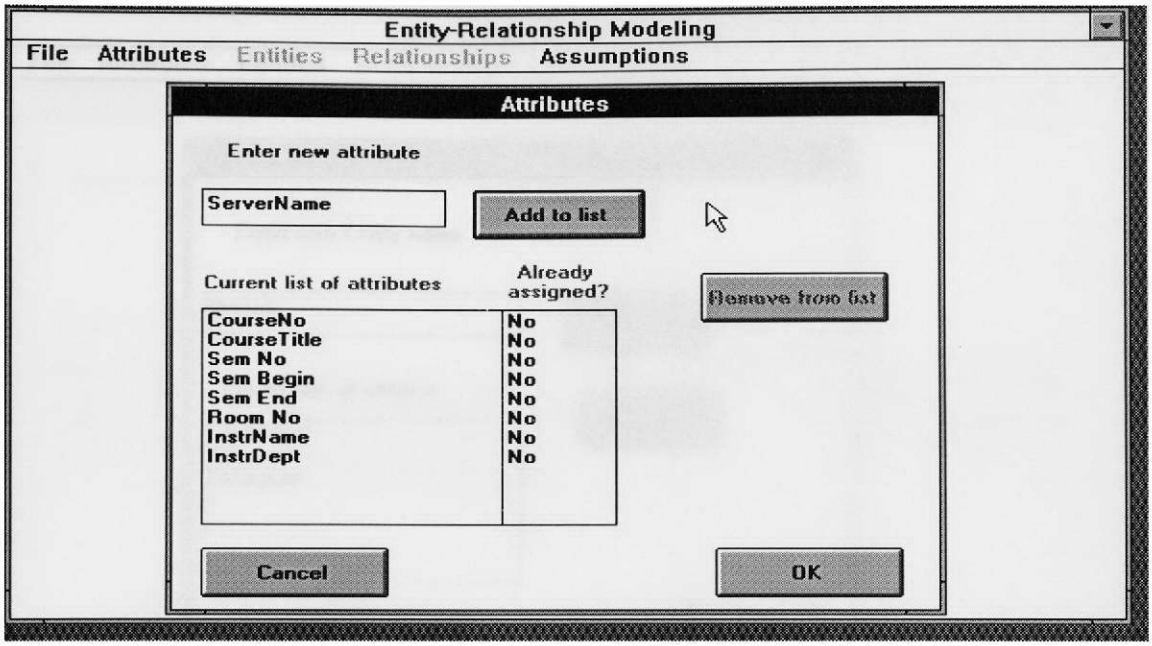


Figure 4 Attributes entry window

After entering all the attribute names, the user can enter Entity names in the Entity Declaration window. No two entities can have the same name. The assignment of attributes to entities will be done in a subsequent window. Similar the Attributes window, here too, the user can correct entry errors with "Remove" and "Add" buttons. A snapshot of the Entities window is shown in Figure 5.

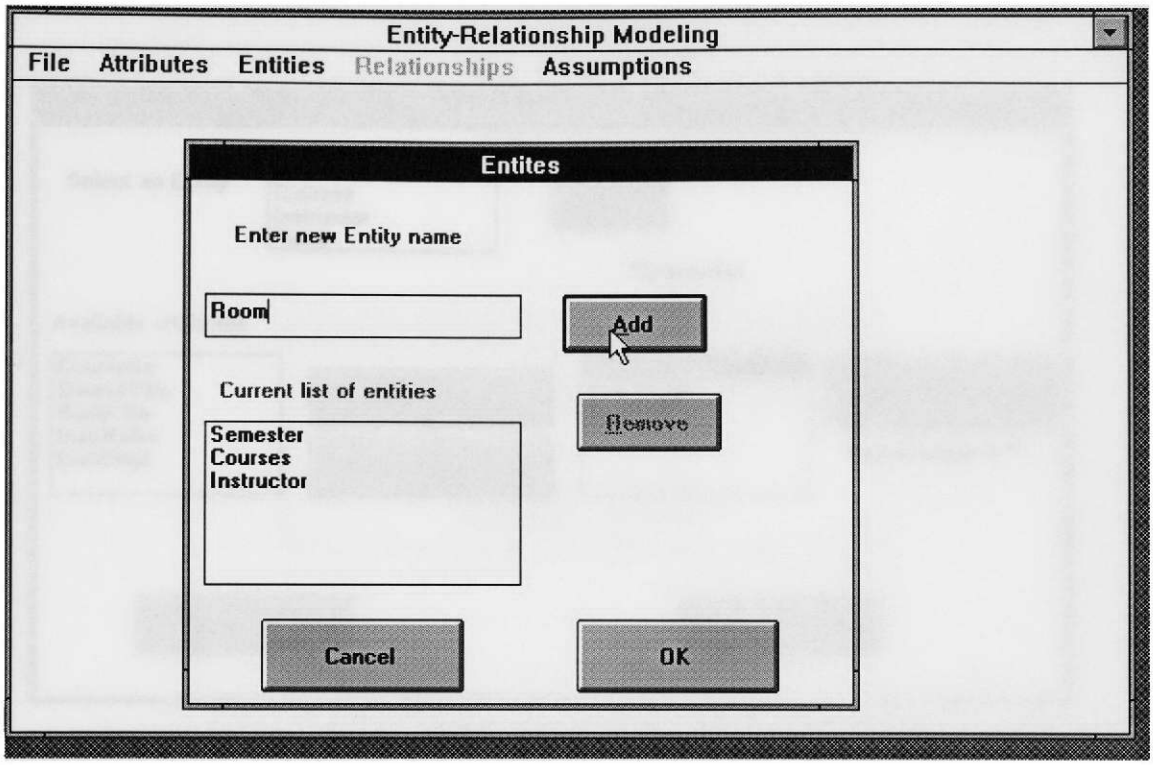


Figure 5 Entities declaration

Once the entity names have been entered, the menu option for assigning attributes to entities become enabled. This will be done in the Entity definition window. The user can pick an entity from the list of entities and assign attributes to them. The list of attributes is maintained as a dynamic list, where only unassigned attributes are listed. This feature prevents an attribute being assigned to more than one entity. A sample Entity definition window is shown in Figure 6.

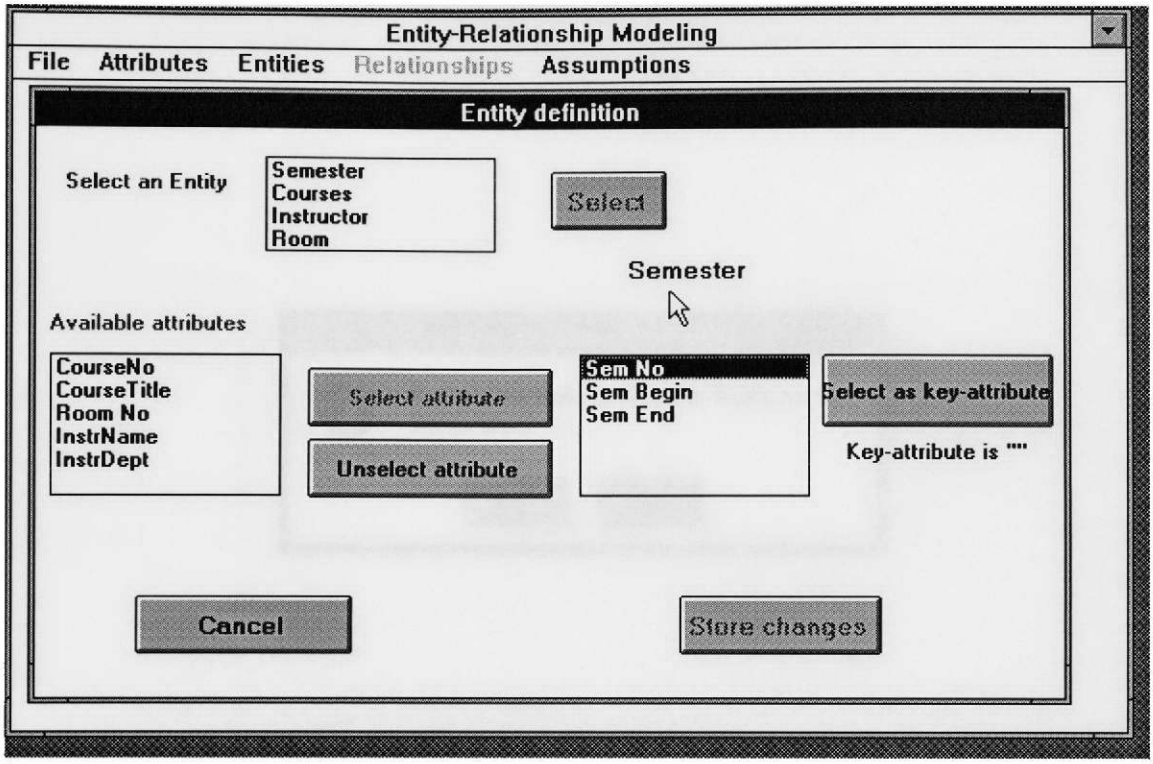


Figure 6 Entity definition

When the user selects a key attribute, the system verifies the correctness with a message. Similarly, when the user attempts to save the entity definition, the system forces the user to verify the correctness of that the assignment. (See Figure 7 and Figure 8).

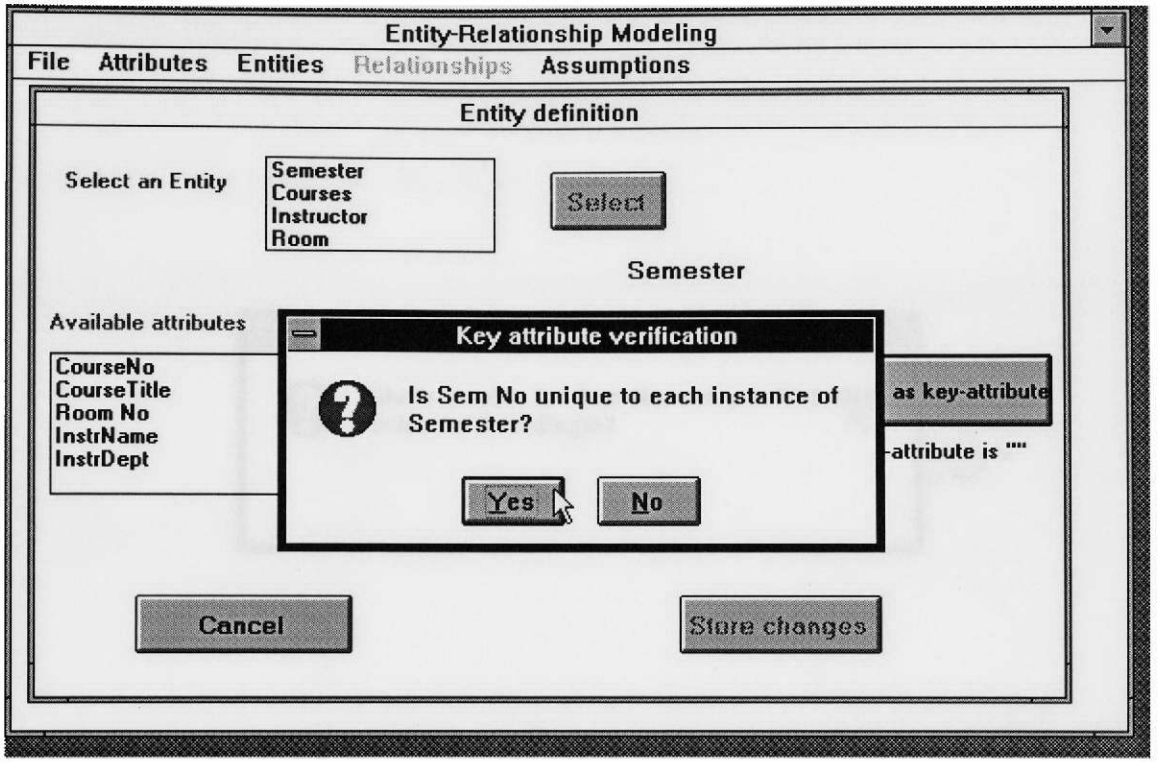


Figure 7 Key attribute verification

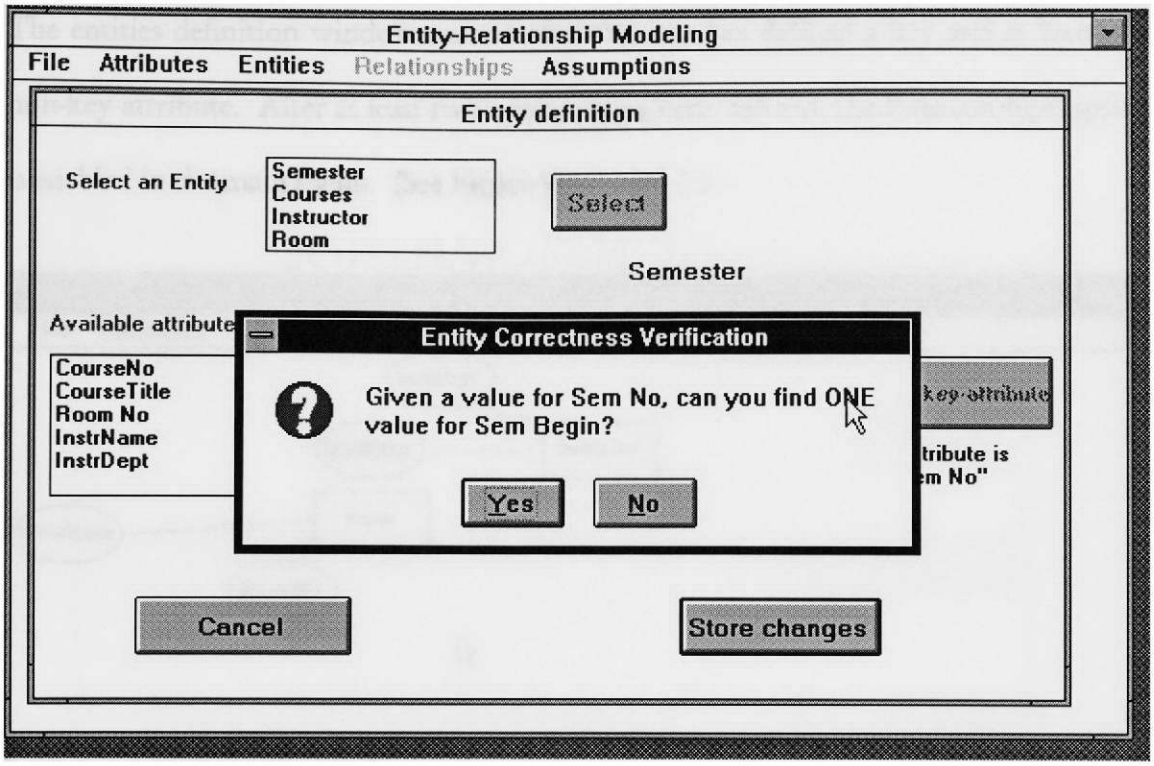


Figure 8 Attribute assignment verification

The entities definition window ensures that the user has defined a key and at least one non-key attribute. After at least two entities have been defined, the Relationships option is enabled in the main menu. (See Figure 9)

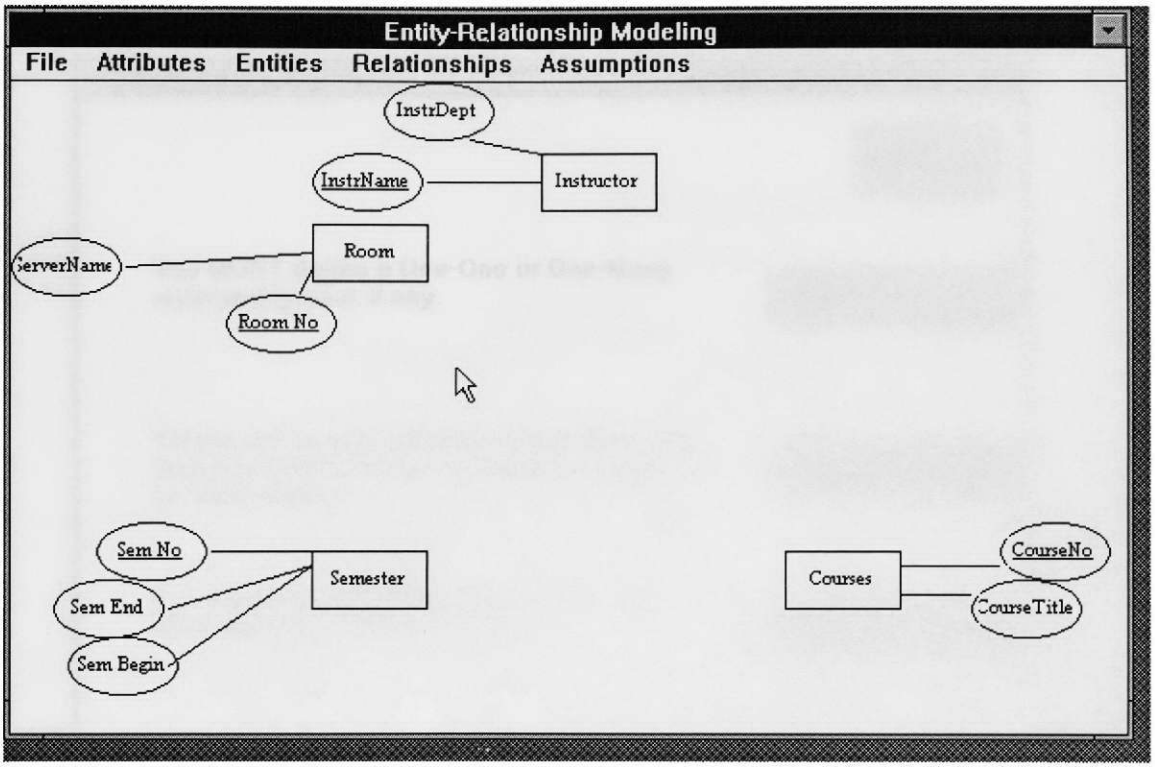


Figure 9 E R Diagram window

The user cannot define a binary or ternary relationship at will. The user is forced to follow a specific sequence in modeling relationships. In the beginning, the user can define binary *one-many* type relationships only. See Figure 10 for a screen print.

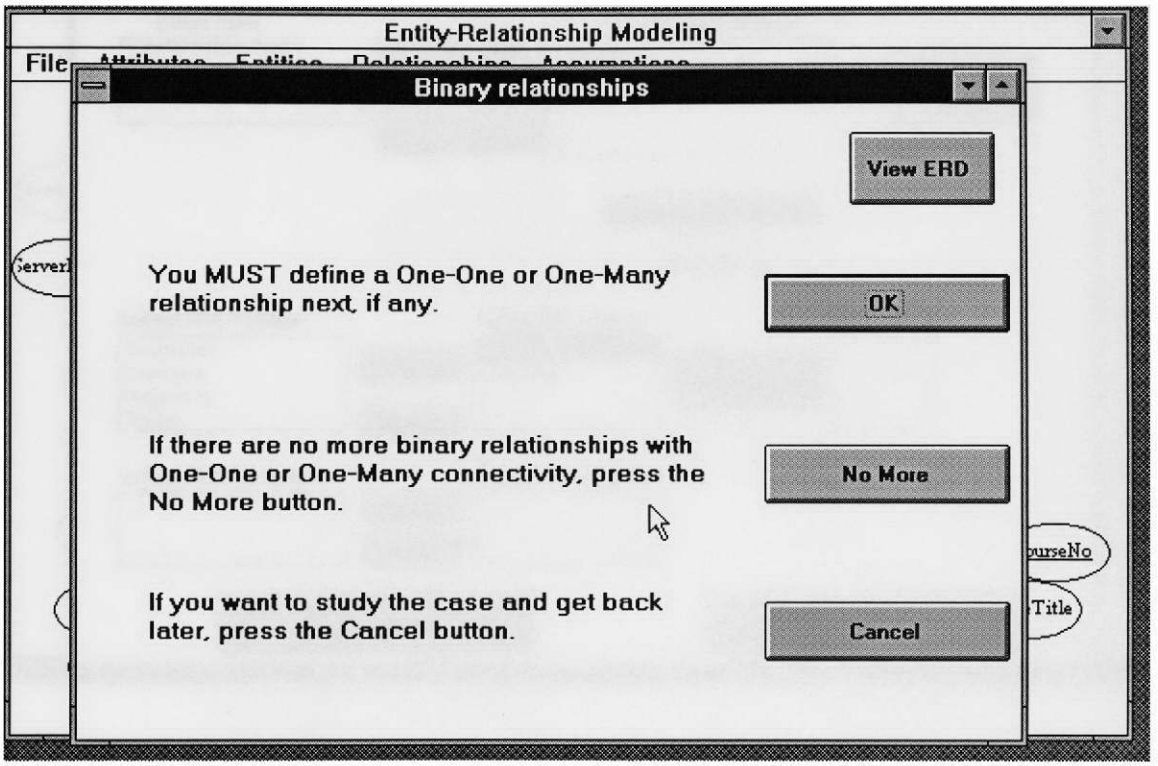


Figure 10 Restrictive system: Relationships modeling

At this point, the user has to either go ahead and model the binary *one-many* relationship or press the "No more" button to model other types of relationships. The screen shot of the Relationship definition window is shown in Figure 11. Notice that the title of the window reminds the user to define a certain type of relationship. Each relationship has to be given a unique name. The list of selectable entities are only those which have not

been assigned to any relationship. Although there are many buttons on this window, only a few are enabled at any given time (See Figure 11).

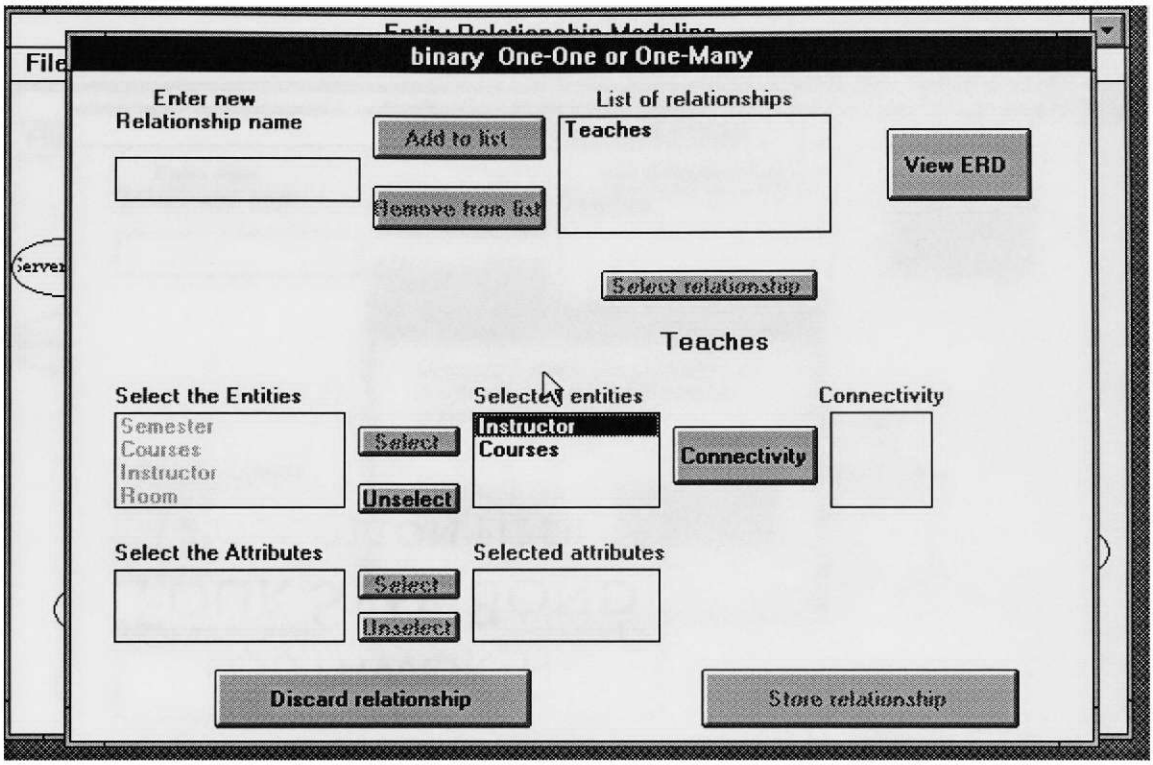


Figure 11 Restrictive system: Relationships definition

While defining the relationships the system assists the user in determining the connectivity of the relationship (See Figure 12). If the relationship is of a type that is different from the required type, the user would not be able to store the design.

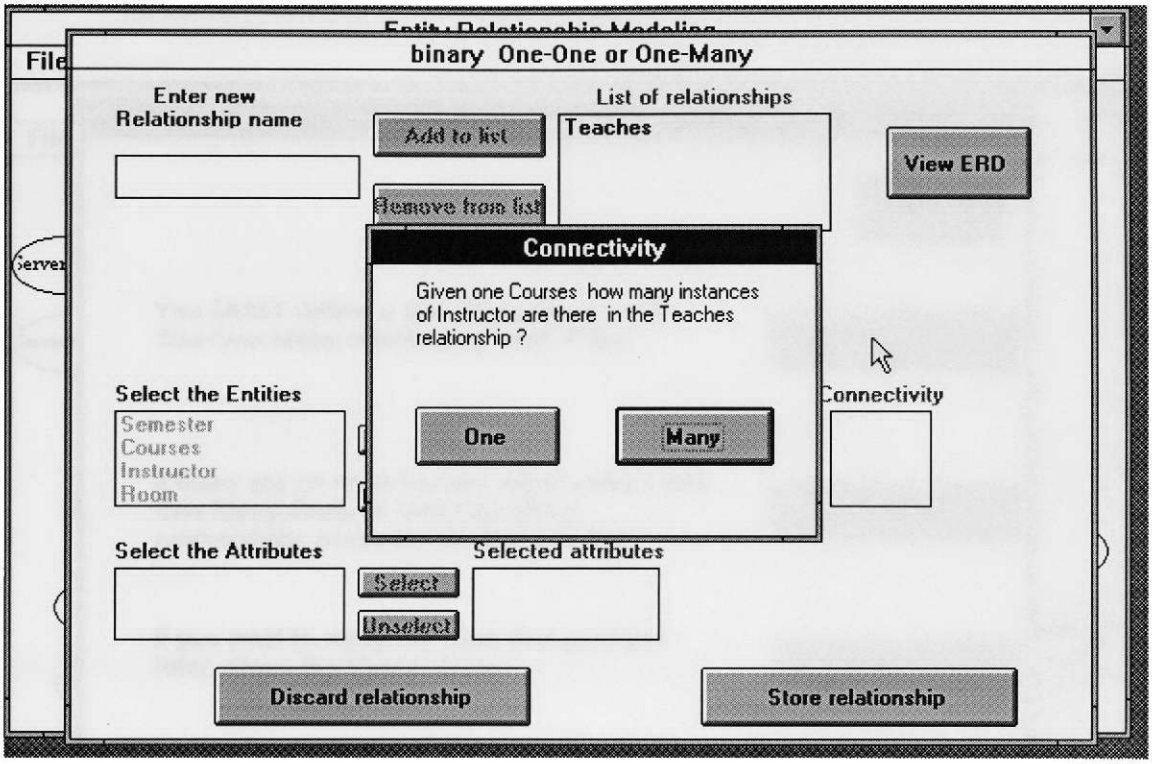


Figure 12 Restrictive system: Connectivity window

After the user defines all binary *one-many* relationships, he/she would press the "No more" button to model other types of relationships. The system forces the user to define ternary *one-one-many* or *one-many-many* relationships next. The user's choice is recorded from the relationships window as shown in Figure 13.

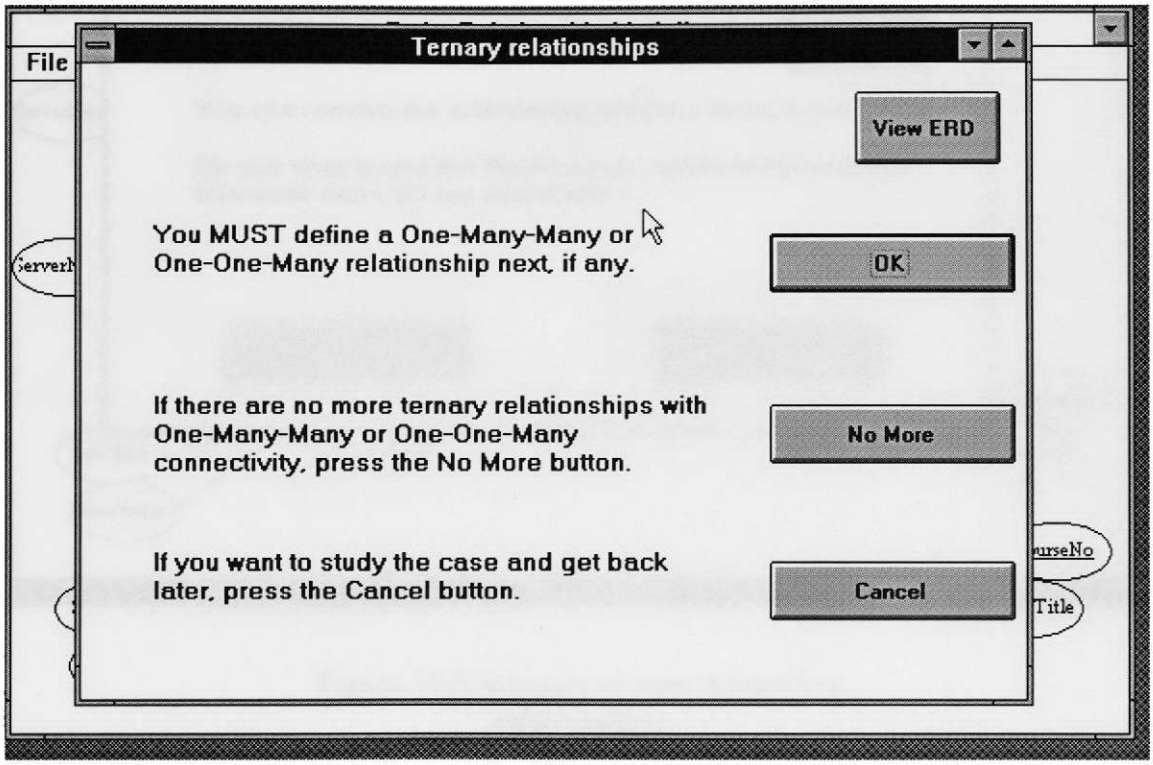


Figure 13 Restrictive system: Ternary relationships

After defining some relationships, if the user wishes to remove an entity from the model, he/she will have to delete the relationships in which the entity participated. This restriction is necessary to prevent errors in relationships. Similarly, the user can delete only the relationship that was defined last. Thus, with the restrictive system, any desired

modification will have to be done as a series of user initiated *undo* operations. (See Figure 14 and Figure 15 for illustration of the implementation of restrictiveness).

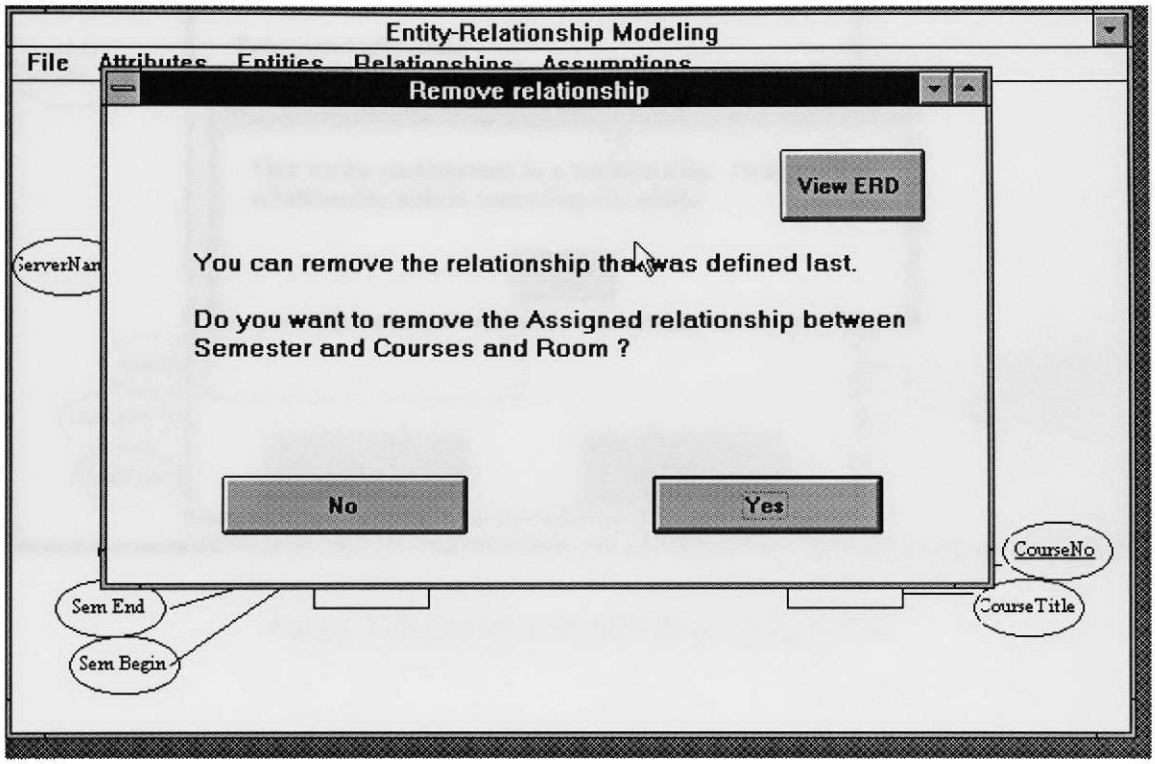


Figure 14 Restrictive system: Removing relationships

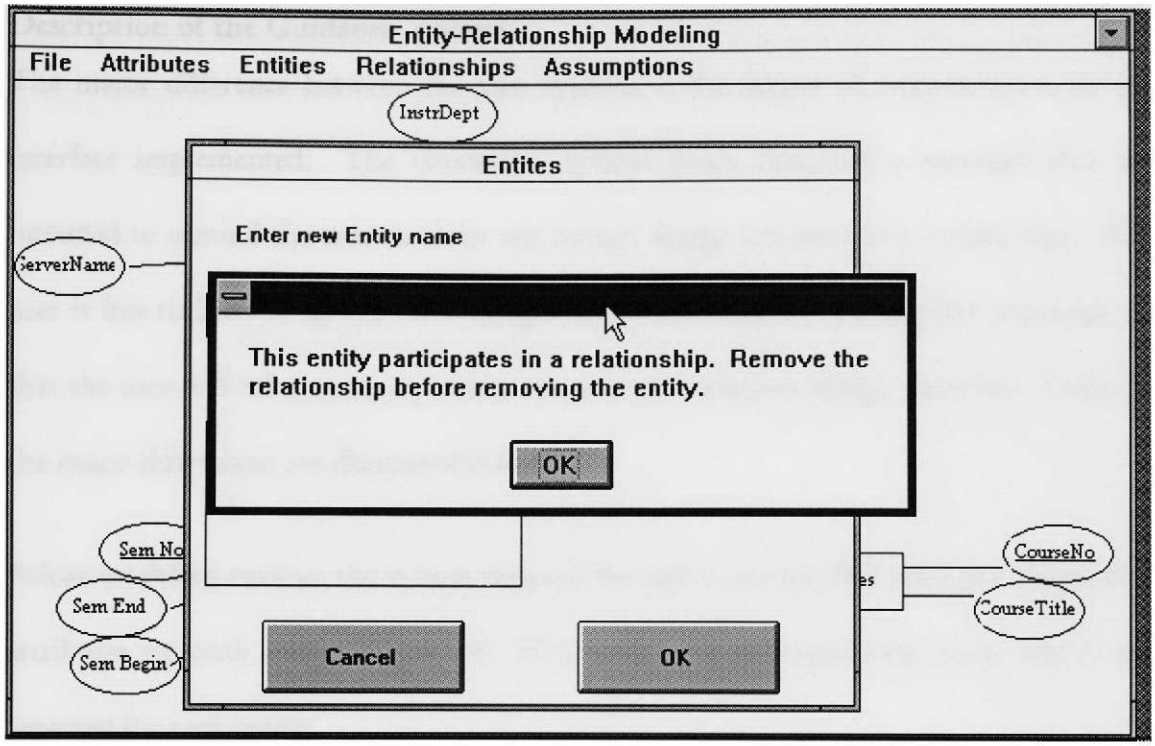


Figure 15 Restrictive system: Removing entities

Description of the Guidance system

The major difference between the two systems is the degree of restrictiveness in the interface implemented. The Guidance system issues informative messages that are intended to remind the user to carry out certain design activities in a certain way. The user is free to heed or ignore the messages. Some messages are worded like warnings, so that the user will take necessary action to prevent erroneous design decisions. Some of the major differences are illustrated below.

Before modeling entities, the system reminds the user to ensure that there are at least two attributes for each entity (Figure 16). This message is displayed only once, and is not repeated for each entity.

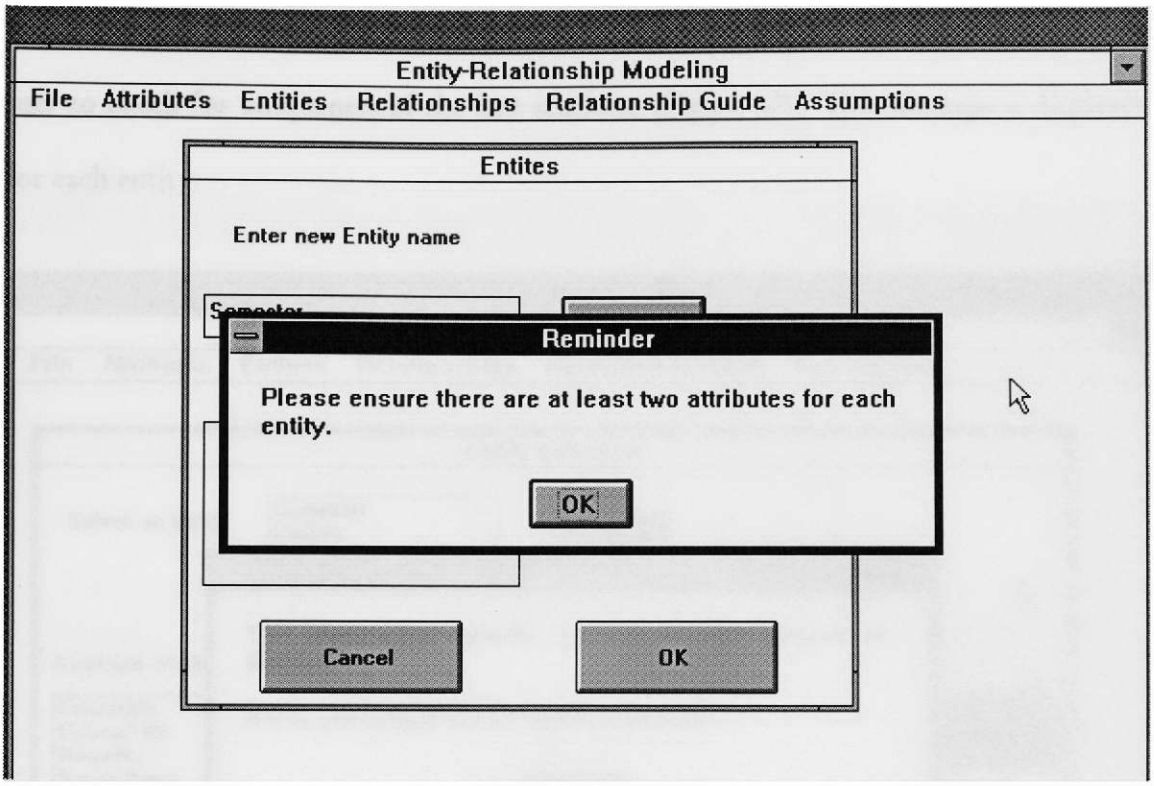


Figure 16 Guidance system: Entity declaration

When the user defines an entity and pick an attribute as the key, the system reminds the user to check for uniqueness of the key attribute (Figure 17). This message is displayed for each entity.

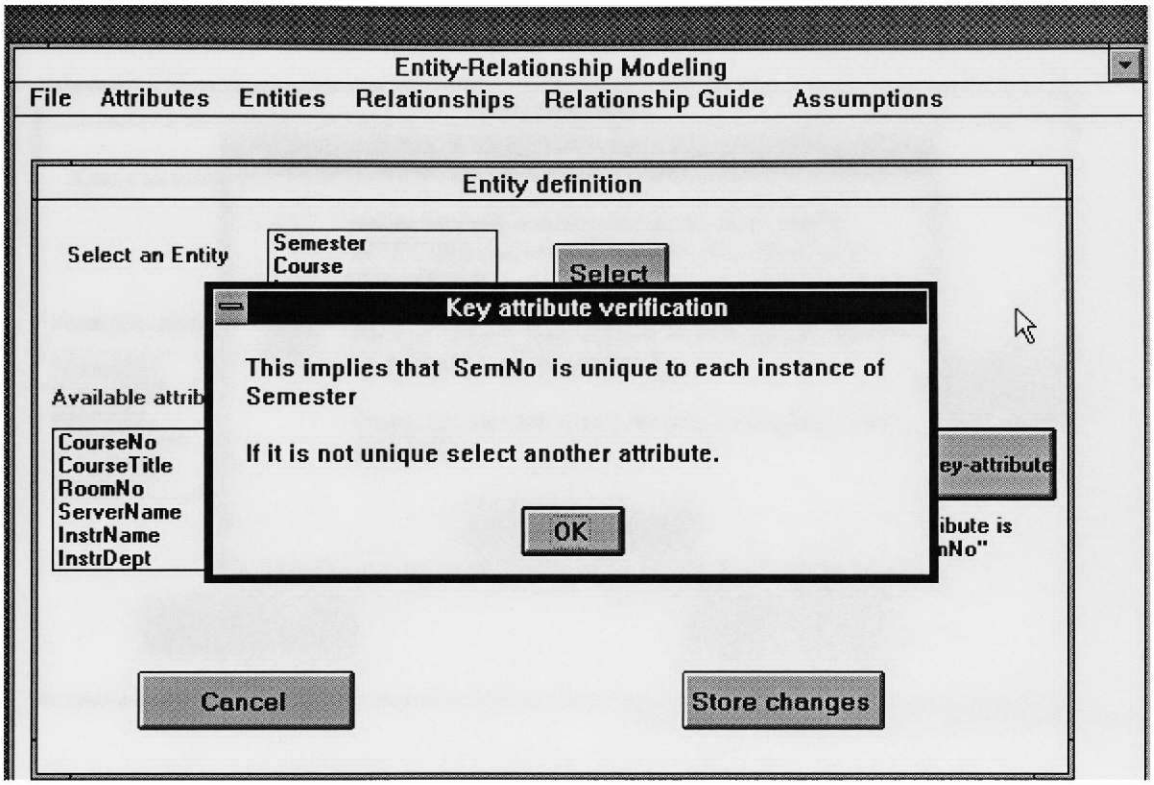


Figure 17 Guidance system: Key attribute verification

When the user attempts to save the entity definition, the system prompts the user to verify the functional dependency between the key and non-key attributes (Figure 18). The user is given an opportunity to discover any error and take corrective action and the user may press "Cancel" to go back to the attributes assignment window. This message

is displayed only once for each entity, unlike the case with the Restrictive system where the user is prompted as many times as the number of non-key entities.

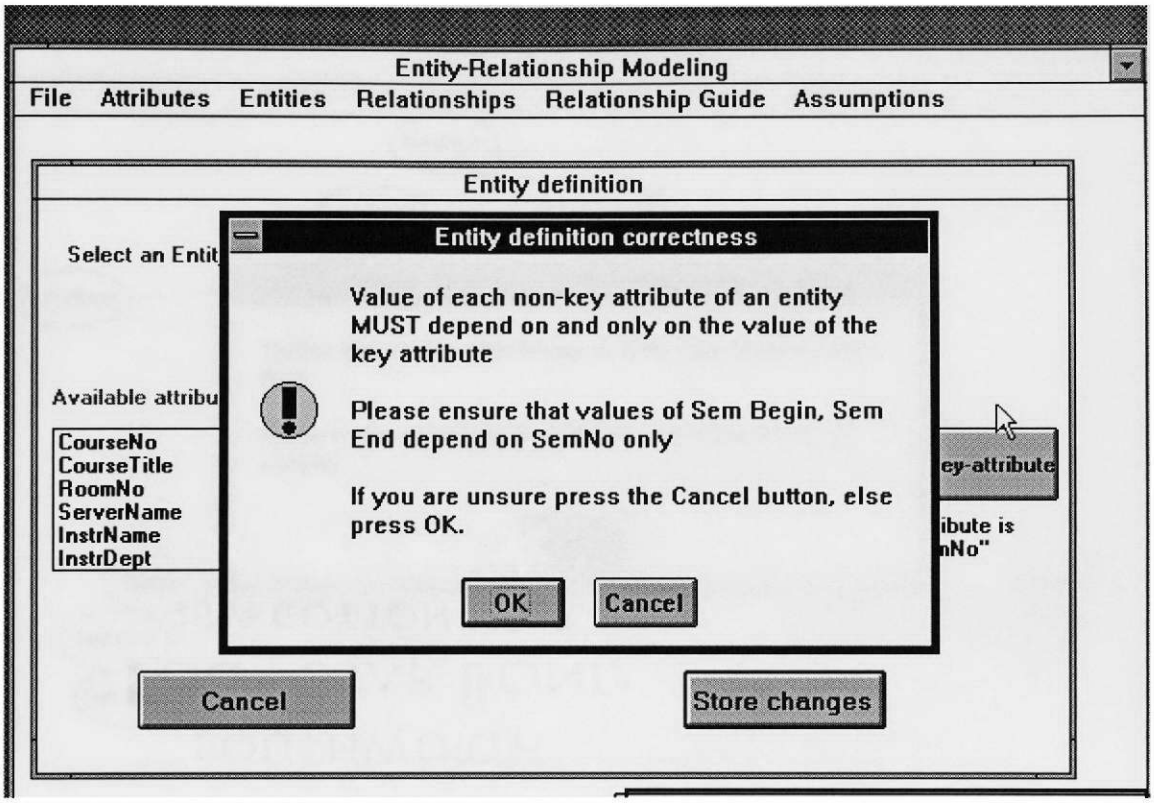


Figure 18 Guidance system: Non-key attribute verification

Before modeling the relationships the user is reminded to define binary *one-many* type relationships first. The user, of course, is free to ignore the suggestion. The suggestive messages are context sensitive. If the user has already defined a *one-many* relationship, the system would recommend the user to either model more *one-many* relationships if any, or model any ternary relationships (Figure 19). The exact contents of the message

depends on the number of 'free' entities also. The user can use an on-line guide to follow the sequence of relationships (Figure 23).

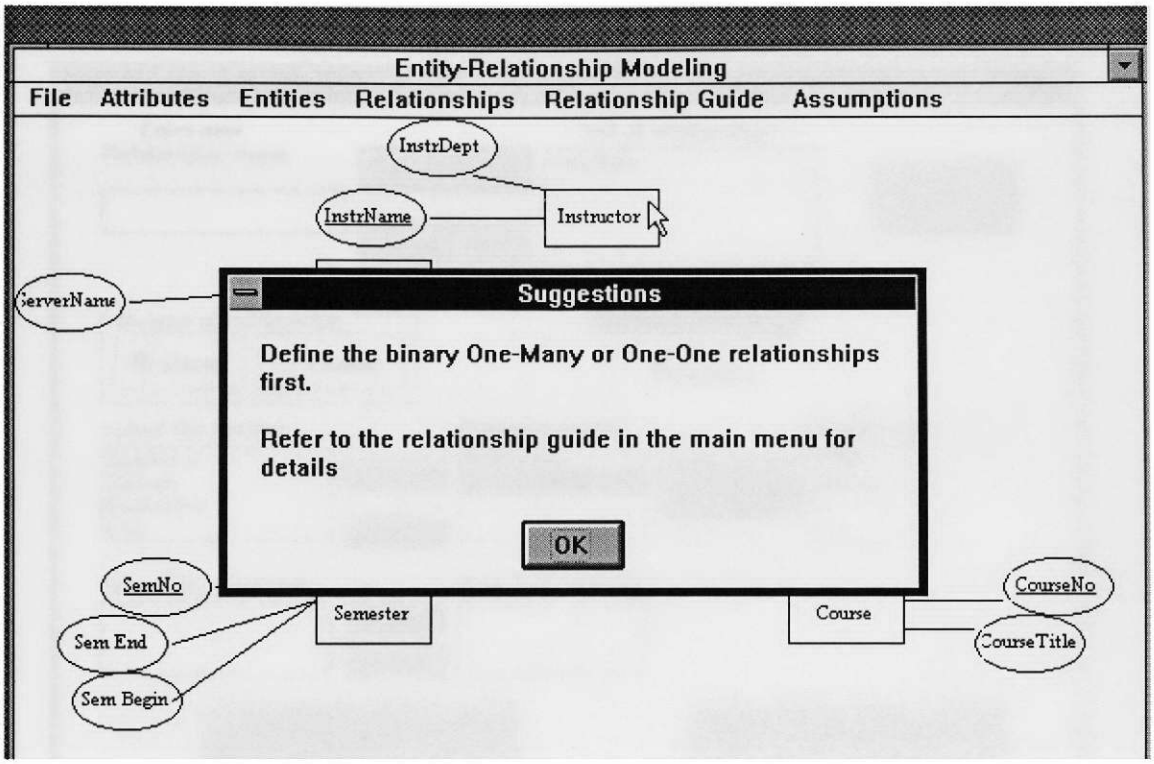


Figure 19 Guidance system: Relationship modeling

Although the system recommends a certain course of action, it does not monitor whether the user followed the recommendation or not. Hence, the relationship definition window does not reflect the recommendation message, the user viewed seconds ago. The user is free to select either binary or ternary type for relationships (Figure 20). He/she may also select an already defined relationship and modify it or delete it. The system helps the user with determining the connectivity of the

relationship, by framing an appropriate question. The relationship window helps the user with determining the correctness of the attribute of relationship also.

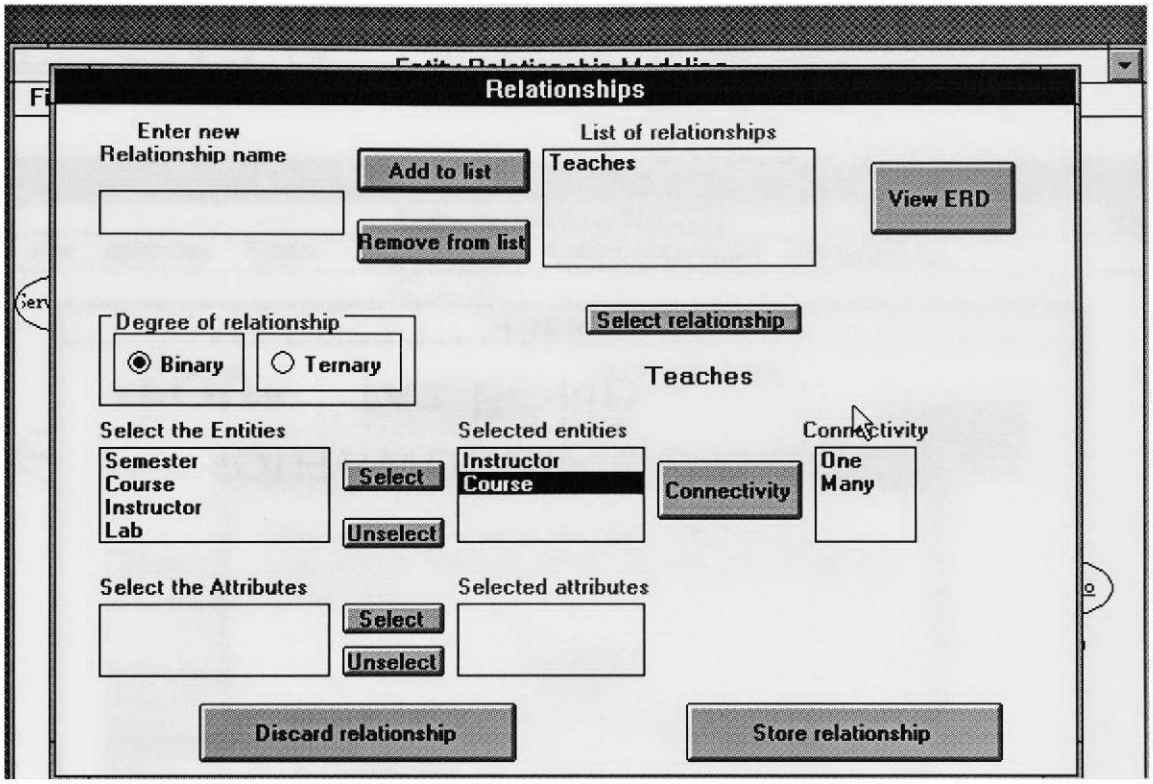


Figure 20 Guidance system: Relationship modeling Window

The list of entities from which the user chooses for the current relationship includes all the entities. It includes both 'free' and 'non-free' entities. If the user picks a non-free entity, the system issues a mild warning to the user. This warning is expected to make the user think harder before selecting that entity. See Figure 21 for an illustration.

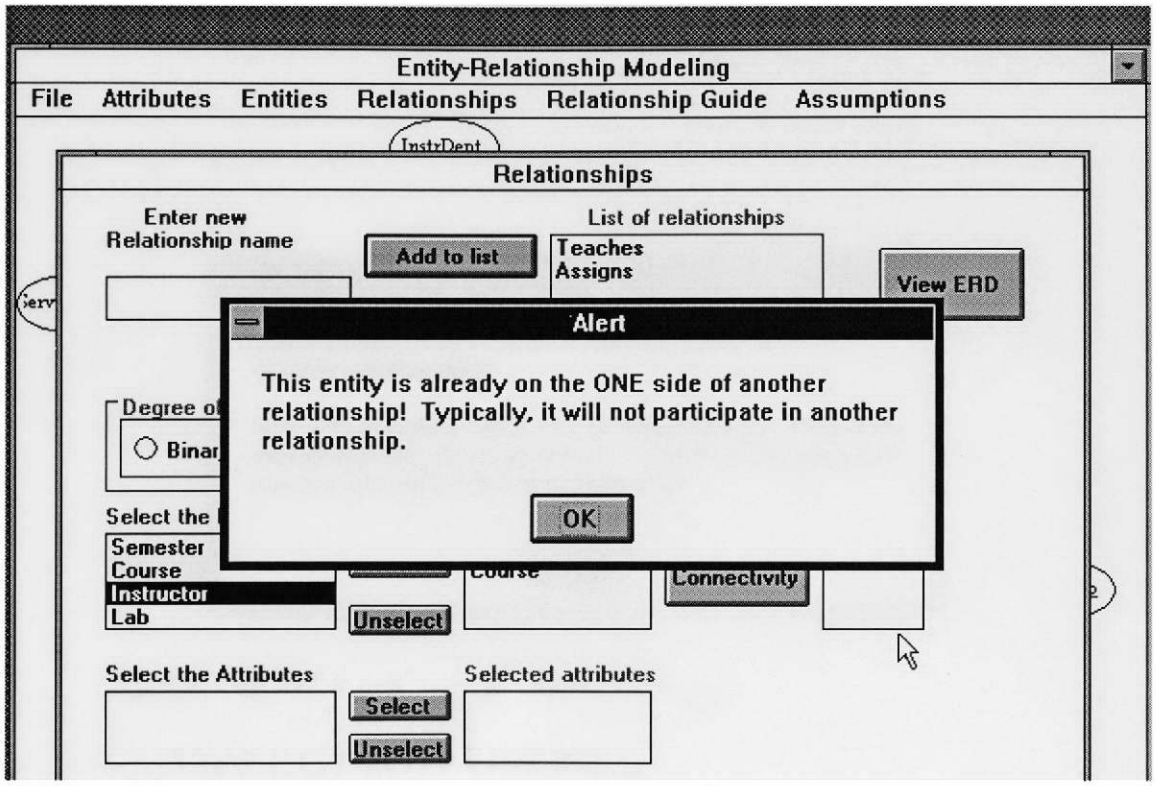


Figure 21 Guidance system: 'Free entities' implementation

Similarly, the system would warn the user of potential derived relationships. Suppose the user had earlier defined a relationship between two entities. If the user defines another relationship where the set of entities is a subset or equal set or super set of the

entities from the earlier relationship, it would most likely lead to derived relationships. The system would determine the potentiality of derived relationship and alert the user (see Figure 22). However, the user is free to ignore the message and continue according his/her own plan.

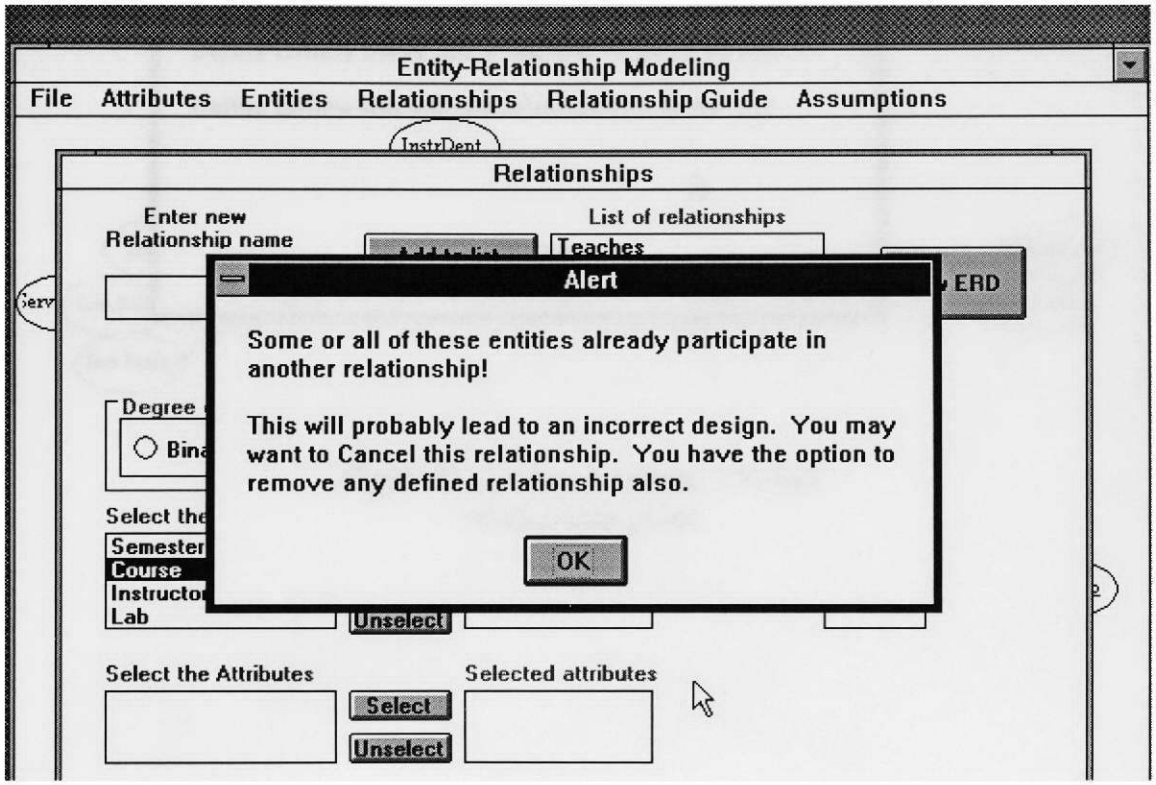


Figure 22 Guidance system: Derived relationships prevention

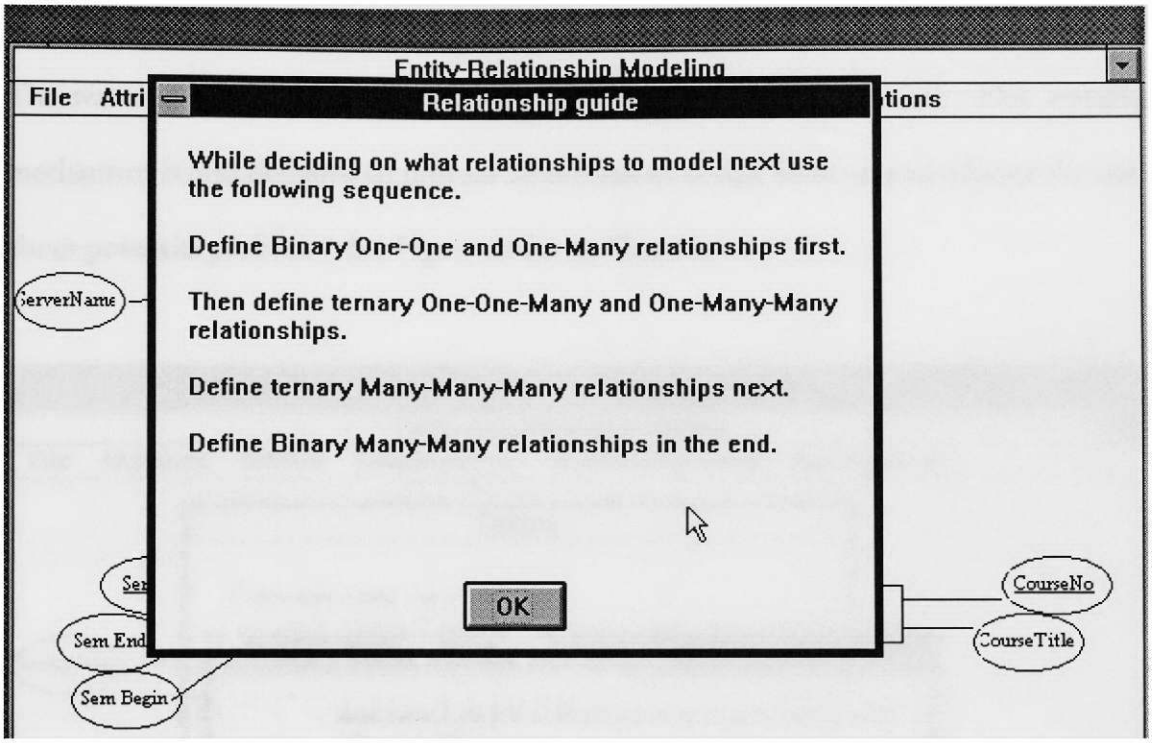


Figure 23 Guidance system: On-line relationship guide

When the user wishes to remove an entity from the model, the system issues a warning. The warning informs that some relationship may have to be removed. This warning mechanism is also designed to prevent an erroneous design decision and educate the user about potential problems. See Figure 24 for an illustration.

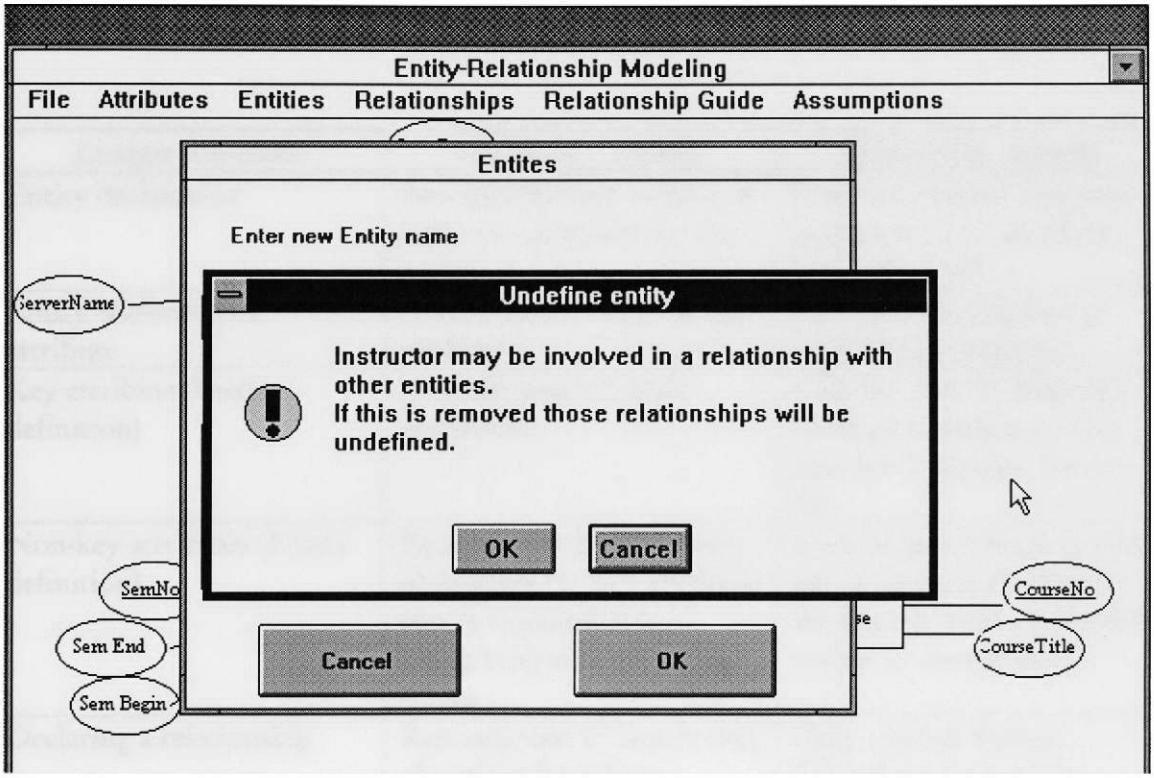


Figure 24: Guidance system: Removing an entity from the model

Comparison of Guidance and Restrictive systems

The details of the differences are shown in Table 1. Silver (1988) offers a framework for descriptive analysis of decision support systems. As presented earlier, the conceptual

database design problem involves decision making at each intermediate phases and hence Silver's framework can be used to describe the design support system also. Silver proposes a three-tiered approach in describing the system. The three tiers are (a) functional capabilities (b) user-view of the system and (c) system attributes.

Table 1 Differences between Guidance and Restrictive systems

Design sub-tasks	Guidance system	Restrictive system
Entity declaration	Reminds the user to have at least two attributes for the entity	Does not remind user; but entities without attributes cannot be used
Entity without Key attribute	Reminds user to define key attribute	Prevents defining entity without key attribute
Key attribute (Entity definition)	Reminds user to verify uniqueness	Asks the user whether key attribute is unique or not; user *has* to answer Yes or No
Non-key attributes (Entity definition)	Reminds about dependence of non-key on key attribute; user is responsible for identifying and correcting errors	Ensures dependence of non-key on key attributes one attribute at a time with and option to correct error
Declaring a relationship	Reminds user to ensure that all entities have been defined	Only entities that are defined can be used for relationships
Relationships	Suggests the use of a sequence of relationships to model	Forces the user in following the sequence of relationships
Entities for relationships	User may use entities without attributes in a relationship	User cannot use entities without attributes in a relationship
Use of non-free entities in a relationship	Warns user against use of non-free entities	Blocks user from accessing non-free entities
Binary relationships (One-	Suggests user define one-	Forces user to define one-

Many)	many relationship first	many relationship first
Ternary relationships (One-one-many or one-many-many)	Reminds user to ensure all binary one-many relationships have been defined prior to ternary	Allowed only after all binary one-many relationships are defined
Ternary relationships (Many-many-many)	Reminds user to ensure all other ternary relationships have been defined	Allowed only after other types of ternary relationships are defined
Binary relationships (Many-many)	Reminds to ensure that all other types of relationships are defined	Allowed only after all other types of relationships are defined
Determining connectivity of a relationship	Helps user by framing the question for connectivity check	Helps the user by framing the question for connectivity check and automatic connectivity definition whenever possible
Derived relationships - due to multiple use of entity groups	Warns the user about derived relationship; but does not prevent using group of entities that lead to derived relationship	Prevents user from using the group of entities that lead to derived relationship
Attribute of a relationship	Reminds the user to check dependency	Ensures dependency by asking explicit questions to the user
Deleting a relationship	User may delete any relationship	User can delete only the relationship that was modeled last
Modifying a relationship	User may modify any relationship	User cannot modify any relationship
Help with next step in design	Suggestions on next possible step	User is forced to select from few enabled options

The first tier description refers to the functional capabilities of the system. Each of the three types of design support systems has the same functional capabilities. The functional capabilities include declaring attributes, declaring entities, assigning attributes

to entities, defining relationships between entities and drawing the Entity-Relationship diagram.

The next tier of description refers to the user-view of the system. From the user's perspective the restrictive and the guided system offers a pre-determined sequence of steps to be followed. The control system does not provide the user with pre-determined sequence of steps to be followed. The users of the control system are free to define entities without any key or non-key attributes. They can define a relationship between entities and later assign attributes to those entities. They are free to select any set of entities for a relationship. Whereas the guidance system prompts the user to define a certain type of relationship, whenever the user attempts a relationship. The system helps the user in determining the connectivity of the relationship also. The restrictive system forces the user to enter at least two attributes before he/she can define an entity. They can define a relationship only after they define at least two entities. While defining relationships they are required to define binary *one-many* type of relationships before attempting other relationships. Thus the user view of the systems are different.

The third tier - System attributes - refers to the mechanisms that affect the user's behavior. The guided system and the restrictive system differ in this perspective. The guided system offers context sensitive guidelines and the concepts that justify the guidelines. The user of the guided system is given opportunities to ponder before deciding on the next step. Whether the user heeds the warning and guidance messages or

not is left to the user's discretion. When the user attempts to store an entity without a key-attribute or when he/she attempts to select an entity that is not "free" for a relationship, a warning is issued. The potential problem with such actions will be explained by the system. However, the user is free to ignore such messages. When the user is left with only one free entity and if he/she tried to define another relationship, the system informs the user that he/she need not define any more relationships, but nevertheless the system does not prevent the user from defining more relationships.

The restrictive system affects the user's behavior by offering fewer choices to the user. The user is forced to ensure the functional dependency of each of the non-key attribute on the key attribute. While defining relationships, the user is forced to define all the binary *one-many* type relationships first. The user has to explicitly inform that there are no more binary *one-many* relationships before he/she will be allowed to define the next type of relationships. After defining all the binary *one-many* relationships, if there are three or more free entities, the user will be asked to define ternary *one-one-many* relationships. if the user declares that there are no more such relationship and if there are at least three free entities, the user is expected to define a *many-many-many* type relationship. if the user declares there are no more such relationships then the user is prompted to define binary *many-many* type relationships. At any time the user may not define a relationship that is out of the pre-specified sequence. The system will automatically exclude non-free entities from participating in any new relationships. If

the user attempts to select a set of entities that are a subset or super set or equal set of entities from another relationship, the system will not allow the user to store such a relationship. The user cannot modify any relationship already defined, but can only delete the relationship that was defined last. This restriction had to be implemented in the system, because the user may inadvertently change a relationship to another type that precludes the possibility of another relationship that is already defined. If there are less than two free entities, the user will not be able to define any more relationships. The system helps the user in determining the connectivities also.

Thus it can be seen that the guided, restrictive and control systems, albeit having the same functionalities are different in user's perspectives and different in ways they are expected to affect the user's behavior.

Chapter 4

RESEARCH MODEL AND RESEARCH QUESTIONS

The research model is based on a consolidation of relevant theoretical findings. In this section the findings from research areas that are of importance are summarized. The research hypotheses and corresponding statistical hypotheses are presented next.

The performance of novices (non-experts) in a problem solving task would be affected by (a) misapplied expertise, and (b) inaccurate or incomplete knowledge (Reason 1988). Researchers who analyzed why errors occur also report that incomplete knowledge base is a main reason (*e.g.* Batra and Antony 1994a). Hence, user's knowledge level, which includes the descriptive and procedural knowledge, is an important variable in explaining the user's performance in solving a problem.

Poor performance of non-experts in problem solving can be attributed to their limited allocation of cognitive resources to comprehending task domain. Norman and Bobrow (1975) reason that performance in problem solving would be affected by degree of allocation of cognitive resources to the task and problem solving methodology. Reason (1988) attributes poor performance of novices to spillage out of limited workspace. Batra and Davis (1992) while comparing experts and novice attributed novice errors to their inability in pursuing a focused effort in integrating various parts of the problem description. Batra and Antony (1994a) recommend that the database design support

system provide help with connectivity check and help in preventing derived relationships. Bouwman (1984) found that novices have less control over their problem solving process. Anderson *et al.* (1992) recommend that novices follow predefined correct path during problem solving and the system should help them do so. The two types of CODA implementations, Guidance and Restrictive, impart many of the desired characteristics of a system that would ease the cognitive load of the user, help with problem structuring and prevent some types of errors. Hence the type of system that is used will also be determinant of performance in the design problem solving task. The design problem solving involves a user of certain characteristics, with certain level of knowledge, using a certain type of system. Hence there may be other user characteristics that affect the user's performance. The research model is shown in Figure 25.

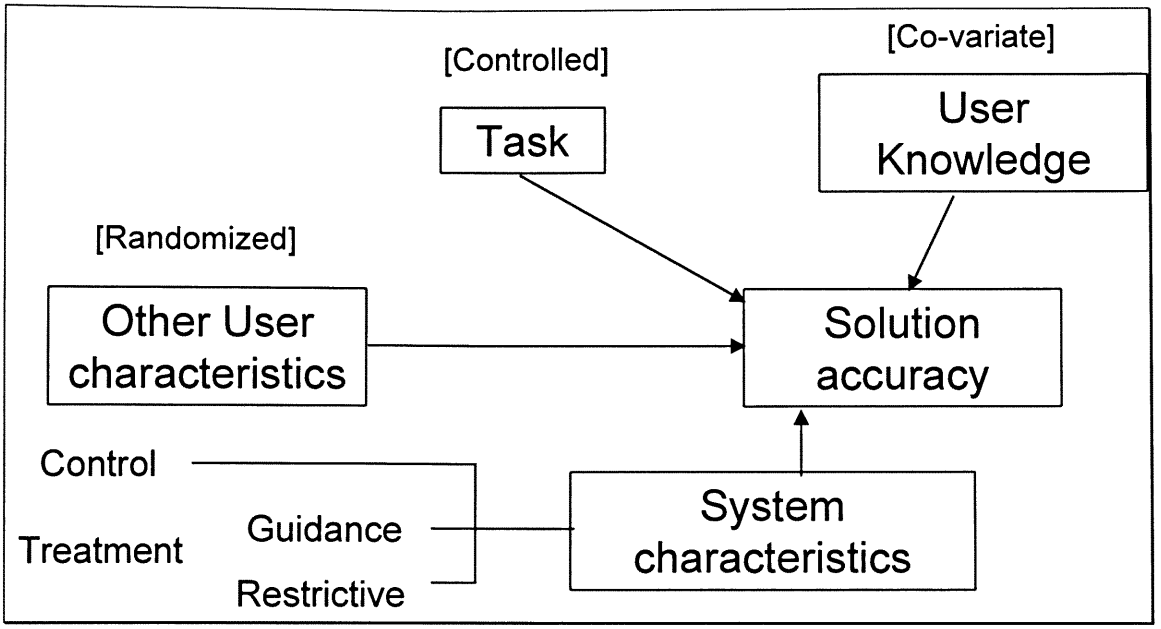


Figure 25 Research Model

The outcome of system assisted design activity is affected by (a) the presence of embedded knowledge in the system (b) the type of interface (i.e. restrictive or guidance) (c) designer's knowledge of data modeling and (d) other designer characteristics. The presence of embedded knowledge and the type of interface are the treatments and are controlled experimentally. The user characteristics are controlled by (a) selecting the subjects from a specific type population, (b) imparting training from the same training script by the same trainer and (c) randomizing the assignment of subjects to treatments. The solution accuracy is affected by the task characteristics also. It is controlled by assigning the same task to all the subjects. Although the same script is used in training the subjects, the subject's level of comprehension of the design procedure may not be the

same. It is possible that some subjects comprehend the design procedure better than others. Since it will not be possible to control subject's comprehension, an indicator of the subject's comprehension will be used in the analysis.

A design support system that has embedded knowledge will be more helpful than a system that does not have embedded knowledge. Hence the accuracy of solutions of the knowledge based system users can be expected to be higher than solutions of the control system users. The first research hypothesis stated as alternate hypothesis is

H1: The correctness of designs obtained by using the knowledge based (Restrictive and Guidance) systems will be higher than the correctness of designs obtained by using the control system.

Zook and Di Vesta (1989) report that overt controlled verbalization during a problem solving process has been found to facilitate the speed and efficiency in certain types of problems. They report also that conscious attention through overt verbalization facilitates learning during problem solving. The Guidance system, by providing opportunities for conscious attention to sub-tasks, would help the subjects perform better than a system that does not provide such opportunities. If a system provides cognitive feed back, it provides the user with information that allows the user compare their judgement with information in the task. Balzer, Sulsky, Hammer and Sumner (1992) found that if a system provides cognitive feedback, it enables the user to exercise better control over his/her cognition and thus perform better than someone who does not receive cognitive feedback.

Between the two knowledge based systems, the Restrictive system ensures that the designer follow a pre-specified and "normative" design path. However, the system is more punishing, than the guidance system, if the user deviates from the specific path. With the restrictive system the designer's early design decisions can affect later design decision situations. Suppose, if the user (wrongly) modeled a binary *one-many* relationship, the entity on the *one* side will not be available for other higher order relationships. Even if the user wishes to model (correctly) another relationship that involves the *one* side entity, he/she would not be able to do so. Thus use of the Restrictive system, although reducing the user's cognitive load, requires that the user does not make erroneous design decisions. If the designer modeled the entities correctly and modeled the binary *one-many* relationships correctly, then the restrictiveness of the system will be found beneficial. On the contrary, if the designer committed an error early during the design problem solving process, the system's subsequent behavior may lead to inaccurate solutions. However, the Guidance system is more flexible with respect permitting changes to designs, we can expect that the non-expert designer would benefit more with the Guidance system than the Restrictive system. Hence the next research hypothesis is

H2: The correctness of final designs obtained by using the Guidance system will be higher than the designs obtained by using the Restrictive system

Dufresne *et al.* (1992) found that when the problem structuring is done by the computer system, the subjects' performance were significantly higher than others whose systems

did not provide structuring to the problem solving process. Novices, unlike experts, do not have access to knowledge in a *pre-compiled* form (Dreyfus and Dreyfus 1986). Hence, they have to spend proportionately more cognitive effort in storage and retrieval of problem solving knowledge. If the non-experts had external assistance with managing the problem solving process, they can be expected to perform better than those who do not receive external assistance. The Control system does not offer any assistance to the user. It is merely a drafting tool. Where as the Guidance and Restrictive systems are knowledge based and they take some load off the user's cognitive responsibilities by providing structure to the problem solving process. Hence, we may expect that the Guidance system users would perform better than the Control system users. Laurel (1986) reports that unconstrained user actions lead to poor performance. Hence, the Restrictive system users would perform better than the Control system users. This leads us to the third and fourth research hypothesis.

H3: The correctness of final designs obtained by using the Guidance system will be significantly higher than the designs obtained by using the Control system

H4: The correctness of final designs obtained by using the Restrictive system will be significantly higher than the designs obtained by using the Control system

Catrambone (1990) compared the performance of subjects in executing certain tasks using the word processor. The treatment was in the specificity of instructions the subjects followed. The group of subjects who received specific instructions found it easier to follow than those who received general instructions. Mental efforts involved in

execution of tasks may be expected to contribute to perceive ease of use (Morris 1987). It has been reported that novice users prefer the computer controlled interaction (Benbasat, Dexter, and Marulis 1981) and they prefer less flexibility in the dialogue style (O'Neil and Walther 1974). The Restrictive system offers fewer options at any time and the instructions are more specific than the Guidance system. Messages and prompts are more directed and the dialog style is less flexible in the Restrictive system than the Guidance system.

Silver (1991) posits that a restrictive system would be favored over a non-restrictive system by non-expert problem solvers, because the users are not burdened with considering various choices during their use of a restrictive system, they would find restrictive system easier to use than a non-restrictive system. However, if a problem solver is using a system that provides informative and suggestive messages, the problem solver may have to expend more cognitive resources in comprehending the message to make an informed decision. Hence, users of the Guidance system can be expected to execute more cognitive processes than a user who uses the Restrictive system. The requirements of more cognitive processes may be perceived as being less easy to use by non-expert designers. Thus the fifth hypothesis is

H5: The perceived ease-of-use of the Restrictive system will be significantly higher than the perceived ease-of-use of the Guidance system.

The users of the guidance system are free to use the guiding messages and would perceive the system to be leading towards correct solution. Since they are free to pursue the design steps of their choice and use the system's messages, they would be more likely to feel in control than the restrictive system users. In a study of expert system usage, Gill (1996) found that the more discretion the users have the more in control they feel. This perception of control would positively influence their satisfaction with the system. Hence, it can be expected that the guidance system users would feel more satisfied with the messages than the restrictive system users. Thus the next hypothesis is

H6:The users of guidance system users would be significantly more satisfied with the system's messages than the restrictive system users.

As per the research model, the performance, which is measured as the correctness of design solution, depends on the type of system used, user knowledge level and other user characteristics. The effects due to "other user characteristics" can be minimized by random assignment of users to treatments. The measure of user knowledge involves grading the user's performance in solving a design problem (pre-treatment task) without the use of any type of system. Solution Accuracy is measured by theuser's performance in solving another task using the system (experimental task). The scores in the pretreatment task (PTS) and experimental task (ETS) will be used in analysis. PTS is a measure of subject's knowledge and it is necessary to partial out the effect of prior knowledge from the dependent variable. In other words, PTS is used to *adjust* the values

of ETS. By using the adjusted ETS, the effects of the type of system can be isolated. The first four research hypotheses are restated as statistical hypotheses as below:

H1: The mean ETS of the population of the Guidance and Restrictive system users is significantly larger than the mean ETS of the Control system users, after adjusting the scores for PTS

H2: The mean ETS of the population of the Guidance system users is significantly larger than the mean ETS of the Restrictive system users, after adjusting the scores for PTS

H3: The mean ETS of the population of the Guidance system users is significantly larger than the mean ETS of the Control system users, after adjusting the scores for PTS

H4: The mean ETS of the population of the Restrictive system users is significantly larger than the mean ETS of the Control system users, after adjusting the scores for PTS

All the tests of significance arecarried out at an alpha of 0.05. Since, the knowledge based systems would prevent some errors, an exploratory analysis of types of errors that were not prevented - was also done. The exploratory analysis would reveal relationships between the types of errors found and the type of the systems, if any. In the next chapter, the research methodology is presented.

Chapter 5

RESEARCH METHODOLOGY

In this chapter the choice of research strategy, details of pre-pilot test, pilot test, subjects, experimental tasks, independent variable, dependent variables and other measures are described. The experimental procedures that were followed are also described here.

Choice of strategy

In this research study it is proposed to compare the effect of types of knowledge based support on users' performance in database design tasks. Types of knowledge based support systems can be classified (a) on the level of embedded knowledge and (b) on the mode of support offered by the knowledge based system. There are two levels of embedded knowledge namely (a) no knowledge and (b) some knowledge. There are two modes of support namely (a) Guidance and (b) Restrictive type. Since, the systems that need to be tested are not commercially available products, it is not possible to observe their effectiveness in a natural setting. To economically capture the effect of the type of system on user performance, it is desired to have a sample of users whose characteristics are similar. It is desired to control the effect of user characteristics on their performance on design tasks as much as possible. In summary, the objective is to test predictions, that certain type of knowledge based support will lead to better performance, by providing means for studying the relationships under controlled and unconfounded conditions.

Under these circumstances, the appropriate research strategy would be a laboratory experiment Kerlinger (1986).

Characteristics of Laboratory Experiments

According to Stone (1978, p 118) a laboratory experiment is a research strategy that is characterized by (1) artificiality of the setting of (2) assignment of subjects to treatments (3) manipulation of the independent variables by the researcher and (4) virtual control on all independent and intervening variables by the researcher.

System development

The consulting system for conceptual database design was developed as stated. The system was written in Visual Basic, a language for developing systems with graphic user interfaces. The menu options are very similar across the three types of systems *i.e.* control, guidance and restrictive. The user is prompted to enter his/her name and a title for the problem in the beginning. The control system users can access any menu option any time. The guidance system users too can access any menu option, however a context specific message will be displayed. The restrictive system user has fewer menu options enabled at any time. The knowledge base for the restrictive and guidance system was implemented using selection (IF THEN ELSE) and choice (SELECT CASE) constructs. The user can save, retrieve or print ER designs. The system captures the process trace unobtrusively. Each action, such as clicking on a button, choosing a menu option or entering data in a text box are time-stamped and a log of user actions are maintained

automatically. None of the three systems has explicit Help menu. Printed user manuals are available for looking up help. For further details on the system see Chapter 3.

Pre-pilot Study

Nine students from an under-graduate class in an Introductory Information Systems course volunteered to participate in the pre-pilot study. The purpose of the pre-pilot was to validate the software and the training script. The pre-pilot study was completed in one session. The subjects were lectured on the concepts of Entity Relationship modeling for an hour and forty minutes. Each of them were assigned to a specific version of the software (Control or Restricted or Guidance) and were trained on using the system. For training purposes, a problem and its solution were provided. The explanation of the solution was also given. The system training and training on problem solving were combined in one script. By following the procedures given in the script they could learn the problem solving methodology and at the same time get trained on the software. After the training, the subjects were assigned to solve one of two problems. The first problem involved a binary and a ternary relationship and an artificial entity. The second problem involved only binary relationships. Apparently, the second problem was not difficult to the subjects. The first problem was too hard to all but one subject, immaterial of what system they used. After the pre-pilot lab session de-briefing with the subjects revealed the inadequacy of the practice problems. Some technical problems with the software were discovered for the first time. The experience from the pre-pilot study

was used to improve the instructional training, problem solving methodology training and system training techniques.

Pilot Study

The purpose of pilot study was to test the software, test the revised training on ER modeling and to get a preliminary understanding of the effect of the knowledge based systems. More students from the same under-graduate course volunteered to participate in the pilot study. Those who had already participated in the pre-pilot were exempted from the pilot study. They were trained on the concepts of Entity Relationship modeling in a regular class session. The class met during a week day for 2 hours and 40 minutes. During the in-class session, they were exposed to the fundamental concepts of Entities, binary and ternary relationships. Printed guidelines for identifying and defining the ER constructs were provided (Appendix 1). After the lecture and a break for 10 minutes, the problem solving methodology was demonstrated. The demonstration involved problem solving by the instructor (Appendix 2). The problem involves a binary and a ternary relationship. The notes for the design methodology were condensed into a "quick reference" list and the subjects were given a copy each (Appendix 3). After the demonstration, the subjects solved a problem without the help of the instructor. This problem involved two binary relationships only. Thus the subjects got an opportunity to learn the ER modeling method and solve a problem before they got to use the software.

A number of lab sessions were scheduled during the subsequent days. In all, twenty three subjects showed up for the lab sessions. They completed a questionnaire that elicits their background information (Appendix 4). Then they were randomly assigned one of the three types of the system. After they were trained on how to use the system, they solved a problem that involves one ternary and two binary relationships (Appendix 6). After problem solving they completed the Ease-of-Use and Information Satisfaction Questionnaires (Appendix 11,12).

The summary of performance of the subjects from the three groups are shown in Table 2. Each solution was graded on a scale of 1 to 100. The mean score of performance of the guidance group was found to be higher than the restrictive and control groups. Using t-test, the difference between the scores of guidance and control subjects was found to be significant ($p < 0.05$). Summary measures of ease-of-use (EOU) and Satisfaction are shown in Table 3. The mean score EOU and Satisfaction scores of the guidance group were higher than the restrictive groups.

Table 2 Summary of performance in Pilot study

System Type	N	Overall	
		Mean	Std Dev
Control	6	45.17	19.21
Guidance	7	65.71	23.45
Restrictive	10	48.10	18.84

Table 3 Ease of Use and Satisfaction measures
(Pilot study)

System Type	N	EOU	SAT
Control	6	4.78	4.27
Guidance	7	5.67	4.02
Restrictive	10	5.40	3.37

The pilot study was conducted like a trial run for the actual experiment. The training script, the instruction delivery, the experiment task and the procedures of the experiment would be the same as the actual experiment. The user manual for the three systems, proved to be inadequate and had to be modified for the actual experiment. The pilot study provided valuable insights into the conduct of the experimental study procedures. The analysis of the pilot study data also revealed the superiority of the Guidance system over the Control system. The difference in scores was nearly 20%. The performance of the Restrictive system users was as good as the Guidance system users. Next, the characteristics of experiment subjects are detailed.

Subjects

The participants in a lab experiment should be drawn from a population that is representative of intended criterion population (Fromkin and Streufert 1986). Criterion population refers to the specific population to which the results of the experimental study will be generalized. The objective of this experimental research is to find the

effectiveness of knowledge based system when used by *non-expert* designers and so, the criterion population is the group of non-expert database designers. Non-expert designers are the type of knowledge workers who have basic training on information system development methodologies with little experience. They have to use productivity tools such as micro computer based database management systems in a typical work environment. Since the employers of such knowledge workers are more likely to be small businesses, each individual in such an enterprise would be expected to know a wide variety of computer related skills.

The choice of subjects also depends on the level of control the researcher wishes to have on extraneous variables. The performance of subjects in solving the experimental task would be affected by how knowledgeable they are in the domain, *i.e.* database design. However, the researcher wishes to minimize the variance in the level of knowledge the subjects would have in database design domain. The researcher could ensure all the subjects would have equal training on database design, if the subjects did not have *any* training in database design *prior* to the experiment. The undergraduate students from sections of an introductory information system course were found to have the desired level of knowledge in database design. Thus the choice of subjects was driven by the desire to control extraneous variables also.

Students from two sections of an introductory information systems course volunteered to participate in the study. This course is a core course for business undergraduate

students. They had earlier completed a course on microcomputer software where they were taught productivity tools such as word processor, electronic spreadsheet and database management system. In the Introduction to Information Systems class, they had been taught database concepts such as tables, records, fields and key fields. This introduction to database concepts was given in a session that met for 2 hours and forty minutes during a week day. The number of students enrolled in the classes were 60 and 65 respectively for the first and second sections. The subjects were trained on Entity Relationship modeling during the next class (for Training script see Appendix 1). After training on ER modeling, the students attended the lab session where the experiment was conducted. Only those students who attended all three - database, ER Modeling and the lab - sessions were considered subjects of the study. In all 89 of the participants qualified to be the subjects. The demographic data are provided in Table 4. Most of the subjects were employed and were representative of the criterion population.

Table 4: Subjects Profile

Number of subjects	89
Males	50
Females	39
Average age in years	26.8
Average number of computer courses taken	1.69
Average number of packages familiar with	2
Proportions with	
... Word processing knowledge	100%
... Spreadsheet knowledge	86%
... Programming knowledge	31%
... 4GL experience	20%
... DBMS use experience	32%
... Database application experience	22%

The subjects were well motivated to participate in the experiment. Their motivation was effected by two incentives. First, they were informed about the importance of database systems, learning the modeling concepts and that they have the opportunity to use a prototype CASE tool. They were told their feedback about the system will be valuable for the system designer. Second, they could earn up to 10 percent credit toward their course grade. To earn the 10 points, they were required to attend both the ER Modeling training session in class and the lab session. For students who could not attend the lab session and who did not wish to participate in the study, alternative means of earning the credits were available.

Training on Entity Relationship Modeling

As a topic of the course, the student subjects learnt the concepts of relational database system in a classroom setting that lasted for 2 hours and 40 minutes. In that class session learnt the concepts of tables, fields, key fields, foreign key fields, and need for links between the tables in a relational database system. After the subjects had learnt the basics of relational database, they were taught the Entity-Relationship Modeling methodology in another class meeting. The session included (i) instructions of the concepts, which includes attributes, entities, binary relationships and ternary relationships, (ii) demonstration of the design problem solving process, and (iii) practice problem solving by the students themselves. The complete training script used for instruction is shown in Appendix 1. The author conducted the training in both class sections using the same script. During the training the subjects were exposed to the concepts of attributes, entities and relationships. After instruction of the basic concepts, the author demonstrated how to solve a database design problem (Appendix 2). The problem involved three entities and a ternary relationship. The subjects were given opportunities to ask questions. After demonstration of the ER modeling methodology, the subjects solved a problem themselves without help from the instructor. The second problem involved four entities and three binary relationships (Appendix 9). The training session ended after the subjects completed the second problem. The session lasted for 2 hours and 40 minutes. After the training, the subjects were reminded to attend the lab session to participate in the experiment. The subjects were not aware of the experimental design. They were told

that they would use a computer program that had been designed to help them database design and that the researcher wished to find how effective the software would be in helping them.

Experimental procedures

After the subjects arrived for the lab session, they completed and signed the Consent form (Appendix 5). They completed a questionnaire that recorded their background information such as major, age and computer experience (Appendix 4). Then they were asked to solve a pre-treatment task (Appendix 10) using paper and pencil. After they completed the paper and pencil problem, each of the subjects was provided with the solution and an explanation to the problem they attempted in class (Appendix 11). Then subjects were randomly assigned to either the Control group or one of the two treatment groups. After the random assignment, each subject was handed a diskette containing a specific type of the software program. They were not aware of experimental control or treatments. Each diskette was accompanied by a user's manual. The manuals for the three different systems were different in content (see Appendix 12 for Control group, Appendix 13 for Guidance group, and Appendix 14 for Restrictive group). However, the look and feel of the three manuals were similar. The presentation format such as page layout, font used, graphics used were similar. Each subject was asked to model the solution (copy of which was given to the subjects) to the problem done in class using the system. They completed it by following the specific procedure outlined in the system

training manual. Each subject was free to use as much time as he/she desired . During the system training the subjects were free to ask for any help that would assist them in learning to use the software. The experimenter and his associates were available to help them.

After the subjects perceived to have understood how to use the software, they were given the experimental task (Appendix 6) to be solved using the system. They were explicitly told to *use* the system that had been assigned to them to solve the experimental task. They were instructed to complete the task by themselves with no external assistance. During problem solving, the system unobtrusively captured the process trace and kept track of the time used in completing the task. After they completed solving the experimental task, they handed the diskette, manual and the task sheet back to the experimenter. The experimenter distributed the Ease-of-use and User satisfaction questionnaires (Appendix 7 and Appendix 8). After completion of these questionnaires, the subjects were thanked and allowed to leave.

Independent Variable

The type of system used will be the only independent variable. It is a categorical variable whose value depicts the type of system. There are three types of systems (a) Restrictive, (b) Guidance and (c) Control. The Restrictive and Guidance systems have the same embedded knowledge base. The Control system does not have any embedded knowledge. The implementation strategy of knowledge based assistance is the main

difference between the Guidance and Restrictive systems. The Restrictive system forces the user to select from few choices at any time and follow a predefined sequence of operations. The Guidance system uses a prescriptive approach, whereby it informs and guides the user in various operations. The information and guidance is provided through messages and prompts. It does not prevent the user from executing any operation he/she wishes. The differences between the restrictive and guidance systems are detailed in Table 1. The Control system uses no knowledge base for prescription or proscription. It essentially is a drafting program that lets the user model entities and relationships. Functionally all the three systems are similar. The content validity of this variable can be verified by the details given in Chapter 3.

Grading scheme

A grading scheme was used to score the pre-treatment task and the experimental task. The scheme assigns 75% weight to modeling relationship and 25% to modeling entities. The different weights assigned to model constructs reflect the importance of the constructs. The grading scheme is consistent with earlier studies in data modeling (*e.g.* Batra *et al* 1990; Batra and Antony 1994b; Hardgrave and Dalal, 1995; Bock and Ryan 1993).

Each E-R diagram was graded out of 100 points. Since a database represents inter-related data, task of determining the relationships will be given more importance. The weighing scheme gives a weight of 3 to a relationship and 1 to an entity. A relationship is deemed

correct if both its degree and connectivity are correct. An error in connectivity will result in a 50% penalty and an error in degree will result in a 100% penalty. If an entity was modeled correctly but a non-key attribute was misplaced, a penalty of 20% was imposed to that entity's grade. The details of the scoring scheme are shown next.

The grading scheme was designed to be comprehensive in capturing all contingencies. Each entity can be graded as being one of (0) Missing (1) correct, (2) Correct key but incomplete non-key attributes, (3) Correct key, but incorrect non-key attributes (4) missing key attribute and correct non-key attribute and (5) missing key attribute and incomplete non-key attributes. Similarly, the relationships grading also is comprehensive enough to be usable for different levels of correctness. Each relationship can be graded as being one of (0) missing or wrong set of entities (1) correct (2) correct entities but incorrect connectivity (3) Correct entities and connectivity but with an extra derivable relationship (4) Correct entities and incorrect connectivity with an extra derivable relationship (5) Relationship represented as an entity but with incorrect implied connectivity.

Pre-treatment task

The pre-treatment task was completed by each subject using paper and pencil only. The score in the pre-treatment task was used as the co-variate in the analysis of variance. There are three reasons for requiring the subjects to complete the pre-treatment task. First, the subjects were from two different sections and received the conceptual training

at different points in time. Although, conceptual training was imparted by the same instructor and using the same training script and practice problems, it is possible that the two classes of subjects did not receive identical training. The subjects' performance in the paper and pencil task would be used as a co-variate while determining the effects of the system.

Second, it is possible that some subjects did not understand the conceptual data modeling process well. The subjects were thought to be well motivated and the training was perceived to be sufficient, it is possible that some of them did not quite devote full attention and hence, comprehend ER modeling well. CODA is designed to help non-experts, but the subjects need to have comprehended procedures of determining entities and relationships. The system will not help them *find* the entities and relationships. CODA does not help in interpretation and comprehension of the task domain. The system would only help by preventing errors. Database design problem solving requires that the subject retrieves and uses his/her declarative knowledge and procedural knowledge. If a subject lacks either declarative and/or procedural knowledge, then it would be inaccurate to attribute the subject's poor performance in the experimental task solely to the treatment. Hence, it is necessary to have a measure of subject's problem solving capability *prior* to the treatment.

Thirdly, a measure of comprehension of specific aspects of ER modeling, such as entities or relationships, can be used in determining whether the subject has the *minimum*

required comprehension of the problem solving process. Thus, their score in the pre-treatment task can be used against a benchmark to distinguish between subjects who 'passed' the pre-test and whose who did not. Relationships are far more important and difficult to model correctly than entities. So, if a subject did not score any points for relationships can be considered as having less than minimum knowledge required to model databases. So, the relationships score portion of the pre-treatment score can be used as a measure in determining the appropriateness of using the observation. The grading of the pre-treatment task solutions was completed by the author using grading scheme described earlier.

Task

The experimental task (Super Systems Inc.) involves 10 attributes, 5 entities and 3 relationships. There is one binary *one-many*, one binary *many-many* and a ternary *many-many-one* relationships. Among all the problems provided to the subjects this was the most complex. The task problem was completed with the help of the system assigned to them.

Other variables

The subject's knowledge about information systems was measured by the (i) number of computer courses the subject had taken (ii) number of distinct software packages the subject had knowledge about and (iii) a dichotomous variable that measures whether the subject had (a) word processing (b) spreadsheet (c) 4GL (d) DBMS basics and (e) DBMS

applications knowledge. Other variables are whether the subject is an MIS major or not, age, and sex. Since subjects were randomly assigned to the treatments, it is expected that these variables would have been randomized.

Dependent Variable

The objective of this research project is to determine which of the three types of design support systems is the most effective in helping the designer model the problem accurately. For this purpose, the score subjects received in the experimental task is used as the dependent variable.

Scoring methodology

The ER diagrams of the subjects were printed on separate sheets which also had the subject's ID on it. The type of system that was used was not printed on the ER diagram. Each subject's design was graded by two raters independently. While grading the design, each entity was assigned one of the six (0 to 5) grade codes. Similarly each relationship was assigned one of the six (0 to 5) grade codes. Since the grade codes are from a nominal scale the reliability of the grading can be measured by Cohen's (1960) *kappa*. Cohen's *kappa* is a coefficient of agreement between two or more raters.

Inter-rater reliability

The *kappa* measure is defined as follows: Suppose p_o is the proportion of units in which the raters agreed; p_c is the proportion of units for which the agreement is expected by chance. Then, *kappa* is defined as the proportion of chance expected disagreements

which did not occur, or alternatively it is the proportion of agreement *after* chance agreement is removed from consideration. $\kappa = \frac{p_o - p_c}{1 - p_c}$. It is equivalently expressed as frequencies as $\kappa = \frac{f_o - f_c}{N - f_c}$. For example, suppose the two raters classified 100 items into either type A or type B. Their classifications and other calculations are tabled below:

	Rater 2 # of A's	Rater 2 # of B's	Column total	Marginal probability
Rater 1 # of A's	40	25	65	0.65
Rater 1 # of B's	5	30	35	0.35
Row total	45	55	100	
Marginal probability	0.45	0.55		
Chance agreements	29.25	19.25	48.50	
Observed agreements			70	

In the above example, both raters classified 40 observations as A and 30 observations as B (the diagonal entries). Rater 1 classified 25 as A but rater 2 classified them as B and rater 1 classified 5 as B and rater 2 classified them as A (non-diagonal entries). Overall they agreed on the classifications of 70 on 100 observations. However, this 70% also includes chance agreements. The number of chance agreements on classifying observations as A can be calculated by computing the joint probability of an observation being classified as A by both the raters. In the table above, the probability of rater 1 classifying an observation as A is 65 on 100 *i.e.* 0.65. Similarly the probability of rater 2 classifying an observation as A is 45 on 100 *i.e.* 0.45. The joint probability of an observation being

classified as A is 0.45*0.65. The number of observations that will be classified as A by chance alone is 0.45*0.65*100 = 29.25. Similarly the number of observations that will be classified as B by both raters is 19.25. The inter-rater agreement as measured by Cohen's *kappa* is calculated by the expression (70 - 29.25 - 19.25)/(100 - 29.25 - 19.25) which evaluates to be 0.42. For large N, the sampling error of *kappa* can be computed as $\sqrt{fc/N(N-fc)}$ and the standard deviate *z* can be computed as *kappa/samperror* which can be referred to the normal curve to determine the significance. For the above example, the significance of *kappa* value can be found to be high ($p < 0.01$).

Inter-rater reliability measures

The experimental task involved 5 entities and 3 relationships. Each of those constructs could be classified to one of 6 possible categories. Each coding of each construct is independent of the coding other constructs. The inter-rater reliability as measured by Cohen's *kappa* is shown in Table 5. Each the *kappa* values were found to be significantly different from 0. The correlation between the scores obtained the two raters were found to be 0.99. Since the inter-rater reliability measures are sufficiently high, it can be safely stated that the measures are reliable and if the grading were done by a third rater, the scores would not be significantly different. For analysis purposes, average of the two raters' scores were used.

Table 5 Inter-rater reliability measures

Construct	Kappa
Entities	
Project	0.74
Programmer	0.64
Platform	0.84
Company	0.80
Skills	0.87
All Entities	0.83
Relationships	
Comp-Proj	0.96
Prog-Skill	0.89
Prog-Proj-Plat	0.95
All Relationships	0.94
Overall	0.89

User perception variables

The other measures of the user-system interaction includes (a) perceived ease of use and (b) user satisfaction with the interaction with the system. Perceived ease-of-use (EOU) was measured by a six item measure that is based on the instrument developed by Davis (1989). This instrument has been validated by other researchers in other domains also (Adams, Nelson and Todd 1992, Segers and Grover 1993, Hendrickson, Massey and Cronan, 1993).

The user satisfaction measure is based on an instrument that was developed by (Doll and Torkzadeh 1988). This instrument has been validated and its test-retest reliability has

been established also (Torkzadeh and Doll 1991, Hendrickson, Glorfeld and Cronan, 1994).

Summary of research methodology

In this research project, lab experimentation research strategy was chosen to compare the effectiveness of the three CODA implementations on the performance of non-expert database designers. Undergraduate students enrolled in an information systems course volunteered to participate in the study. A pre-pilot test was conducted with a small group of subjects to validate the software program and training methodology. After rectifying the problems with the software and training methodology, a pilot test was run with a larger, different set of student subjects. The main experiment involved (a) training the subjects on data modeling concepts, (b) demonstration of E R Modeling technique, (c) practice problem solving by subjects, and (d) solving the experimental task using one of the three implementations of CODA.. The accuracy of their design solutions were graded by two independent graders and the inter-rater reliability was found to be satisfactorily high. Each subject's performance in a task *prior* to the actual experiment was captured also. Subjects rated their perceptions on ease of use of the system and their satisfaction with the software program. The details of data analysis and results are given in the next chapter.

Chapter 6

RESULTS

In this section, the composition of subjects' background is reported through a number of demographic and IS knowledge measures. Next, the analysis of experimental data is presented. Thirdly, the analyses of Ease of Use and Information Satisfaction measures are reported.

Characteristics of subjects

The demographic characteristics of the subjects is summarized in Table 4 in the previous chapter. In all 89 subjects completed the experimental procedures. That is, they had attended class on database management, entity-relationship modeling and completed the lab session. Their average age was 26.8 years. Most of the student subjects were working as well. There were 50 male subjects and 39 female. A typical student would have completed an introductory course in personal productivity systems, where he/she was introduced to the popular personal computer software packages like word processing, spreadsheet and database management systems. In the current course, he/she was learning about information systems in organizations. All the subjects had knowledge of word processing software. More than 85% of them had spreadsheet knowledge also. More than 30% of them had programming experience in a higher level language. However, only 20% had experience with fourth generation languages. Since some of the more popular 4GLs are database oriented, it can be stated that only a few of them

actually had any programming experience in database environments. However, approximately 20% of them reported that they used database applications also.

The experiment was conducted over nine lab sessions. The break up of attendance for the lab sessions are shown in Table 6. Those 89 subjects are from two class sections of the same subject. There were 48 from one section and 41 from another.

Table 6 Sessions and attendance

Session No	Number of subjects
1	18
2	12
3	4
4	12
5	2
6	8
7	12
8	8
9	13

Summary statistics

Next, the summary statistics of the performance measures in the pre-treatment task are given in Table 7. The maximum score for modeling entities can be 25 points, however, the mean score is only 13.25 with a standard deviation of nearly 6.5. The performance in modeling relationships appear to be worse than that of entities. Of a possible 75 points for relationships, the mean score is only 20.01, with a very high standard deviation of

18.71. This implies that the comprehension of the modeling procedure had not been fully internalized by the subjects. The overall mean score for the pre-treatment task (PTS) is 33.27 with a standard deviation of 19.91. The poor performance in the pre-treatment task can be attributed to the large proportion of no-score for modeling relationships, which accounted for 75% of the total score. Nearly a third of the subjects, twenty eight in number, did not manage to correctly model even one relationship among the two relationships in the task. Since modeling relationships is the most important aspect of E R Modeling, a 'no-score' performance in modeling relationships is of a big concern.

Table 7 Performance in the Pre-treatment task

	Mean	Std Dev
Entities	13.25	6.48
Relationships	20.01	18.71
Total	33.27	19.91

The summary statistics for performance measures in the experimental task are shown in Table 8. When we compare the scores for modeling entities, we find that there is little difference among the three scores. The largest difference (which is less than 1.0 on 25 points) is between the Restrictive and Control system. When we compare the scores for modeling relationships the largest difference (more than 10 on 75 points) is between the Guidance and Control systems. The score for Restrictive system falls in middle. The

differences are in the directions that we expected with Control system scores less than both the Guidance and Restrictive scores.

Table 8 Performance in Experimental task

Performance in experimental task							
System Type	N	Entities		Relationships		Overall	
		Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
Control	29	18.72	5.77	26.72	24.25	45.54	27.91
Guidance	32	19.91	4.53	37.89	20.93	58.1	23.64
Restrictive	28	19.39	4.29	32.58	20.22	52.07	22.01
Overall	89	19.36	4.87	32.58	22.11	52.11	24.91

The mean score for the Guidance system users is higher than the Restrictive and Control system users. The differences in mean scores between Control and Guidance systems is more than 10 points. The difference between Restrictive and Guidance systems and Restrictive and Control systems are even smaller.

Analysis of variance

The main analysis in this research project will use the analysis of variance (ANOVA) technique and its accompanying *F*-test of significance. One of the assumptions of the ANOVA techniques is that the variance *within the classes* are homogeneous, i.e. do not differ significantly differ among themselves. The random assignment of subjects to treatments usually results in homogeneity of variance. The ratio of maximum variance to minimum variance among the three groups was found to be 1.61 which is not

significant (p=0.291). This implies that the variances of the three groups of dependent variable values are not significantly different from each other. So, it is appropriate to use the Analysis of variance technique.

The objective of this study is to determine if use of the knowledge-based design support tool results in performance that is significantly higher than the performance from use of a tool that does not have embedded knowledge. The designer performance would be affected by the designer's own modeling expertise and assistance from the system. Hence it is necessary to 'partial' out any effect that can be attributed to the designer's expertise. This measure of designer expertise has to be measured *before* the designer gets to use the system. The ANOVA technique can use the measure of expertise as a *co-variate* variable. The expertise variable co-varies with the performance measure variable. ANOVA technique when used with a *co-variate variable* is also called the Analysis of Covariance (ANCOVA).

In the analysis of co-variance, a measure of expertise which is the score each subject scored in the pre-treatment task (PTS) is the co-variate variable. The measure of subject's performance in the experimental task (ETS) is the dependent variable. The PTS scores were obtained under uniform conditions prior to random assignment and application of treatments. The ETS measures are adjusted or "corrected" by eliminating the variability due to PTS measures. The adjustment is carried out by regression of ETS on PTS.

The test of significance of the regression of the dependent variable (ETS) on the adjusting variable (PTS) was conducted *before* proceeding with the analysis as recommended by Hill and Kerber (1967). If the two variables are found to be correlated, then the analysis of covariance technique would be employed and if the correlation between the two variables were not found to be significant, then the need for adjusting or 'correcting' the dependent variable measure is not warranted and a simple analysis of variance would suffice. The correlation between PTS and ETS was found to be nearly significant ($rho=0.192$, $p=0.074$). Hence, it is appropriate to use ANCOVA, instead of ANOVA. The results of the analysis are shown in Table 9. As the results of ANOVA indicate, there is no significant difference *among* the ETS of the three groups ($p=0.242$). Hence the null hypothesis that the mean ETS scores of all the three groups (Control, Guidance or Restrictive) are same cannot be rejected. In other words, the data does not support the alternative hypothesis - that at least one of the three systems is more effective than others.

Table 9 Analysis of variance

Analysis of variance of Adjusted ETS with PTS					
Source of variation	**SS**	**DF**	**MSS**	**F**	***p* value**
Main effect System type	1708	2	854	1.443	0.242
Residual	50306	85	592		
Total	54636	87	620		

Post-hoc analysis

Since it is desired to isolate the effect of the type of system that is used, the knowledge of subjects in data modeling - which is measured by their scores on the pre-treatment task - was used as the co-variate. All the subjects had equal opportunity for learning the data modeling methodology. The data modeling procedure involves identifying and modeling entities and determining and modeling relationships between the entities. Modeling entities was relatively easy for all the subjects. As it can be seen from Table 7, the mean score for modeling relationships was only 27% (20 points on a maximum 75) whereas modeling entities was apparently easier with 53% (13.25 on a maximum of 25). This observation is in accordance with observations from earlier studies in data modeling, where it was found that novice designers find modeling entities easier and that they find modeling relationships to be difficult (e.g. Batra and Kirs 1993). Modeling relationship is the most crucial aspect of the entity relationships modeling method, because relationships result in tables that act as links to other 'master' tables in the relational database. If the links are defined accurate, the database design will be rendered useless. Hence, it is highly important that the modeling of relationships be done accurately. This importance, of course, is reflected in the weight assigned to relationships - 75% versus 25% for entities.

An analysis of the distribution of the relationship scores in PTS (PTS Relationship Score) reveals that there were 28 subjects, nearly of third of all subjects, did not score any

points for modeling relationships. The pre-treatment problem involved two relationships - a binary and a ternary - and only 61 subjects were able to determine at least one of the relationships. Thus it is apparent that the subjects' aptitude in modeling relationships was not homogeneous (mean 20 and a standard deviation of 18.71). Hence, it was decided to discontinue using those observations where the PTS Relationships score was zero and repeat the analysis on the 'reduced' sample.

Summary statistics of the reduced sample

The summary statistics of PTS for the reduced sample is shown in Table 10. The overall mean has increased, approximately, by nearly 10 points (on hundred) from 33 and there is less variance in the data set now (Standard deviation reduced from 19.91 to 16.86).

Table 10 Pre-treatment Task Scores (Reduced sample)

	Mean	Std Dev
Entities	13.37	6.59
Relationships	29.20	15.51
Total	42.57	16.86

However, the differences in summary statistics of ETS are not as apparent (See Table 11). There is a slight increase in the over all score from 52.11 to 52.19 (see Table 8 and Table 11). However, the differences *among* the three groups are more pronounced. The scores for the Control and Guidance systems have grown apart from an earlier difference of 13

to more than 20 points. The gap increased because, there is a slight increase in $ETS_{Guidance}$ and a marked decrease in $ETS_{Control.}$ Similar to the Guidance system scores, $ETS_{Restrictive}$ has increased from 52.07 to 55.85.

Table 11 Experimental Task Scores (Reduced sample)

System Type	N	Entities		Relationships		Overall	
		Mean	Std Dev	Mean	Std Dev	Mean	Std Dev
Control	19	18.55	4.95	21.70	19.43	40.25	21.51
Guidance	25	20.17	4.60	40.00	21.94	60.54	24.55
Restrictive	17	19.68	4.39	36.03	24.16	55.85	26.99
Overall	61	19.52	4.64	33.19	22.92	52.91	25.53

Analysis of variance - Reduced sample

The test of homogeneity of variance was conducted on the reduced sample to determine if the ANOVA technique can be used. It was found that the ratio of largest to the smallest variance was only 1.574 and there is no significant differences in the variance (p value = 0.603). So, the application of ANOVA technique is appropriate for the reduced sample also.

It is desirable and appropriate to repeat the same analytical technique, i.e. ANCOVA, on the smaller sample also. The correlation between the co-variate variable (PTS) and the dependent variable (ETS) was not found to be significant ($rho = 0.180$, $p = 0.17$) and so, the use of ANCOVA technique is not warranted. A simple ANOVA was performed

using the reduced sample. The results are shown in Table 12. The ANOVA results indicate that there is significant ($p < 0.05$) difference between the mean scores of the three groups of users. Hence, the null hypothesis that there is no significant difference among the scores of the three systems can be rejected. The results indicate that at least one of the three mean scores is different from others.

Table 12 Analysis of Variance (Reduced sample)

Analysis of variance of Adjusted ETS with PTS					
Source of variation	**SS**	**DF**	**MSS**	**F**	***p* value**
Main effect System type	4646	2	2323	3.91	0.026
Residual	34450	58	594		
Total	39096	60			

Multiple comparisons

The tests of other hypotheses call for comparison of pairs of group means. Orthogonal contrasts for comparing (a) Control to Guidance, (b) Control to Restrictive, (c) Guidance to Restrictive and (d) Control to Guidance and Restrictive were created and the sum of squares for each contrast was computed. The results of multiple comparisons are shown in Table 13. The results indicate that the mean score of the knowledge based system users is significantly higher than the mean score of Control system users, thus providing us reasons to reject the null hypothesis that there is no difference in the mean scores of Control and other type system users. The comparison of Guidance to Control system also is found to be statistically significant. Hence, the alternative hypothesis that the

mean score of Guidance system users is significantly higher than that of the Control system users is supported. The difference between Control and Restrictive system scores was not found to be significant at 0.05 alpha, had a low p value (0.067). Similarly, the comparison between Guidance and Restrictive also did not indicate significant difference

Table 13 Multiple comparisons

Comparison	t statistic	p value
Control Vs Guidance and Restrictive	2.798	0.008
Control Vs Guidance	2.914	0.006
Control Vs Restrictive	1.901	0.067
Guidance Vs Restrictive	-0.574	0.570

Ease of use

It was hypothesized that the Restrictive system would rate higher on perceived ease of use than the Guidance system. The perceived Ease-of-user is a 6 item scale with a minimum of 1 (least ease of use) to 7 (most ease of use). The mean values are shown in Table 14. The mean score for the Restrictive system is higher than that of the Guidance system. The variance of the Guidance system EOU values was 3.08, much higher compared to the variance of 0.354 of the Restrictive system. A test of homogeneity of variance revealed that the variances are indeed different (F = 8.706, p <0.01). The knowledge of non-homogeneity of variance was used in selecting the appropriate t-test. The observed test statistic was 1.748. Since, it was hypothesized that the mean score of the Restrictive system would be higher than the mean score of the Guidance system, a one-tailed test was chosen. The observed test statistic was found to be significant (p

< 0.05). Hence, it can be concluded that the data supports the alternative hypothesis that the Restrictive system would be perceived to be easier to use than the Guidance system.

Table 14 Summary of Perceived Ease of Use

Perceived Ease-of-Use			
	N	Mean	Std Dev
Guidance	23	5.50	1.755
Restrictive	17	6.19	0.595

User Satisfaction

It was hypothesized that the users of the Guidance system would feel more satisfied with the messages provided by the system than the Restrictive users. The SAT measure is the mean the user scores in 6 item questionnaire. Each item was scored on a scale of 1 (least satisfied) to 5 (most satisfied). The summary statistics of User Satisfaction measures are shown in

Table 15. The mean SAT score for the Guidance system, 4.41, was higher than that for the Restrictive system, namely 4.16. The variances were comparable and an F test revealed that the variances were not significantly different ($p = 0.85$). The results of the t-test that compares the means indicated that they are not significantly different. The observed test statistic was 1.41 and the corresponding p value is 0.084.

Table 15 Summary of User Satisfaction measure

	N	Mean
Guidance	23	4.41
Restrictive	17	4.16

Summary of statistical analysis

The results shown are based on the analysis of performance of the subjects who scored some points for modeling relationships. The effect of the knowledge based system is apparent by examining first, third and fourth hypotheses. The perceived Ease-of-use hypothesis was also supported by the observations. The user Satisfaction hypothesis was also supported, but only at 0.10 alpha level.

Table 16 Summary of statistical results

No.	Hypotheses	*t*-Statistic	*p*-value
H1	Use of knowledge based system leads to better performance than use of control system	2.798	0.008
H2	Use of Guidance system leads to better performance than use of Restrictive system	-0.574	0.570
H3	Use of Guidance system leads to better performance than use of Control system	2.914	0.006
H4	Use of Restrictive system leads to better performance than use Control system	1.901	0.067
H5	Perceived ease-of-use would be higher for the Restrictive system than for Guidance system	1.748	0.044
H6	Perceived user satisfaction would be higher for the Guidance system than for the Restrictive system	1.410	0.084

Error analysis

Since relationships are more important than entities they carry more weight in the grading scheme. Each relationship was assigned one of six grade codes. Each grade code represents the degree of correctness of a relationship. A count of different grade codes assigned to the solutions is used in this error analysis. Table 17 displays the actual and expected frequencies for each grade code and system type combination. Since this analysis is based on 61 subjects and there are 3 relationships in the solution, a total of 183 relationships were graded. The table has two sections, Observed (top section) and Expected (bottom section). Since the purpose of this analysis is exploratory in nature, a *Chi-square* analysis is not conducted. The difference in observed and expected frequencies for each 'System type' and 'Grade code' reveals the effectiveness or ineffectiveness of the system. For example, by chance alone, the Control system users should have nearly 20 instances of 'Correct' relationships, whereas they managed only 11. At the same time Guidance system users should, by chance, have only 26 instances, but actually had 32 instances. The comparison of observed and expected values tend to support the over all findings. Another positive result is that the number of instance of 'Wrong connectivities' is lower for Guidance and Restrictive systems. This seems to support the expectation that system help in determining connectivities will be useful.

Table 17 Relationships correctness: Observed and Expected values

Observed frequencies								
System type	Incorrect	Correct	Wrong connectivity	Extra relationship	Multiple errors	No relationship	Row Total	Row Tot as Proportion
Control	28	11	11	0	1	6	57	0.31
Guidance	22	32	16	0	1	4	75	0.41
Restrictive	16	20	9	0	0	6	51	0.28
Column total	66	63	36	0	2	16	183	
Col Tot as proportion	0.36	0.34	0.20	0.00	0.01	0.09		
Expected by chance frequencies								
System type	Incorrect	Correct	Wrong connectivity	Extra relationship	Multiple errors	No relationship		
Control	20.56	19.62	11.21	0.00	0.62	4.98		
Guidance	27.05	25.82	14.75	0.00	0.82	6.56		
Restrictive	18.39	17.56	10.03	0.00	0.56	4.46		

Chapter 7

DISCUSSION

This chapter includes discussion of the results, some thoughts on generalizability of the research findings and implications for researchers and practitioners. In this research project two types of knowledge base implementation were compared with each other and with a system that does not use knowledge based support. The Information Systems Design Theory as proposed by Walls *et al* (1992) was used in articulating the meta-design requirements, streamlining inputs from the kernel theories, and formulation of testable hypotheses. Since it was desired to control the implementation of types of knowledge based support, a prototype consulting system for database design was developed. One of the major objectives of the study was to determine the effectiveness of knowledge based support from the system. So, it was desirable to use subjects whose knowledge in the database design domain is homogeneous. A laboratory experiment research strategy was used with subjects from an undergraduate IS class were trained on Entity-Relationship modeling, a database design methodology. The subjects were randomly assigned to either one of the three groups (two treatment groups and the control group). In this section, summary of findings and their implications are presented.

Discussion of results

When the sample included all the subjects, including those who did poorly in the pre-treatment task, the ANOVA technique revealed that there is a no significant differences

between the means of the three groups. It is highly desirable to investigate the reasons behind this finding. The subjects did not perform well enough in solving the pre-treatment task. Their mean score in the pre-treatment task was only 33.27 points on a maximum of 100. As expected, they had performed satisfactorily in modeling entities (approximately 13 on a possible 25). However, modeling relationships was more difficult for them, as is evident from the low value (only 30 on a possible 75) of mean of relationships score. In the pre-treatment task there were two relationships, a binary and a ternary, each worth 37.5 points. Twenty eight of the 89 subjects did not score any point for relationships. The performance of those subjects who did poorly in the pre-treatment task also did poorly in the experimental task. Table 18 depicts the correlation coefficients between ETS and PTS for various sets of samples. The first row represents the correlation in the sample that includes all the subjects. The second row displays the coefficients of the sample which included only those subjects whose relationships were not totally erroneous. The third row displays the coefficients for the sample whose subjects did not model any relationship correctly, not even partially. The correlation in the sample that includes all the subjects, is 0.219 which was found to be statistically significant ($p < 0.05$). The significance supports the alternative hypothesis that the observed *rho* is significantly different from zero. However, since the value is very low, it is not of any practical significance. Similarly, we find the correlation coefficient from the second row almost significant ($p = 0.07$) it is not of any practical value (*Rho* = 0.234). The only instance when the correlation is statistically significant *and* is a high value (*Rho*

= 0.507, $p < 0.01$) is when we consider the correlation between ETS and PTS of those subjects who did poorly in the pre-treatment. In other words, subjects who performed poorly in the pre-treatment task also performed poorly in the experimental task, immaterial of which system they used. Since, the number of such subjects is large (28) it is likely that the true effect *due* to the type of system can not be correctly measured. These subjects have not demonstrated satisfactory comprehension of modeling relationships. So, it was decided to analyze the data without considering the subjects who scored no points for modeling relationships.

Table 18 Correlation between ETS and PTS

		Overall	***Control***	***Guidance***	***Restrictive***
All subjects	*N*	89	29	32	28
	Rho	0.219	-0.047	0.284	0.303
	p value	0.039	0.807	0.115	0.117
Subjects with positive Relationship scores	*N*	61	19	25	17
	Rho	0.234	-0.052	0.219	0.244
	p value	0.069	0.831	0.293	0.344
Subjects with zero Relationship scores	*N*	28	10	7	11
	Rho	0.507	0.744	0.144	-0.234
	p value	0.006	0.014	0.757	0.489

The mean scores were recalculated for the reduced sample (see Table 11). The difference between the largest mean (Guidance group, 60.54%) and the smallest mean (Control, 40.25%) is more than 20%. This difference is larger than the difference we had with the inclusion of poorly performing subjects, which was little more than 13% (see Table 8).

The analysis of variance with the reduced sample of subjects yielded significant results (Table 12). The *F* value was large enough to reject the null hypothesis that the mean scores of the three groups were the same. This implies that the mean score of at least one of the three groups was significantly different from others. The pair-wise comparison results (see Table 13) indicate that (a) the mean score of subjects who used the knowledge based system (*i.e.* Guidance or Restrictive) was significantly higher than those who used the control system and (b) the mean score of the Guidance system users was higher than that of the Control system users. The results also indicate that there was no support for the hypothesis that the Guidance system will lead to better performance than the use of the Restrictive system. The comparison between the Restrictive and Control system yielded results that were nearly significant ($p=0.067$).

The hypotheses and results are shown in Table 19. The findings are based on an alpha of 0.05. The first hypothesis evaluates the effectiveness of knowledge based design support. The Control system offers no knowledge based support. It would allow the user to define an entity with no key attribute, allow the user to define a relationship between two entities that *already* participates in another relationship. It is essentially a drafting tool which draws ER diagrams for the user. The knowledge based tool gives structure to the problem solving process, helps with verifying appropriateness of key and non-key attributes, and helps with determining the connectivities of relationships among other

assistance features. Since the subjects in the groups had undergone the same training, the differences in scores are attributable only to the system characteristics.

Table 19 Summary of research findings (at $\alpha = 0.05$)

No.	Hypotheses	Finding
H1	Use of knowledge based system leads to better performance than use of control system	Supported
H2	Use of Guidance system leads to better performance than use of Restrictive system	Not supported
H3	Use of Guidance system leads to better performance than use of Control system	Supported
H4	Use of Restrictive system leads to better performance than use Control system	Not supported
H5	Perceived ease-of-use would be higher for the Restrictive system than for Guidance system	Supported
H6	Perceived user satisfaction would be higher for the Guidance system than for the Restrictive system	Not supported

The second hypothesis that compares the Guidance and Restrictive system was not supported by the observations. The restrictiveness feature of the system *requires* that the subject complete *each* step of the modeling process without an error. If an error is committed early in the process, effects of the erroneous design decision would lead to a compounding effect. For example, if a user wrongly modeled a binary *one-many* relationship between Programmer and Project entities, the Programmer entity would no longer be "free" and hence the designer would not be able to use it in a ternary relationship. To correct the mistake, that the user would have to *remove* the binary

relationship and attempt the ternary relationship. Although not impossible, it is hard for the subjects to recover from such errors. If the user's mental model of the system does not match the conceptual model of the system, error recovery would be harder.

The fourth hypothesis that compares Restrictive and Control groups also was not supported by the data. Although, the mean score of the Guidance group is higher than the mean score of Restrictive group (61 versus 56), the difference is not statistically significant. Both groups did equally well in modeling entities (approximately 20 points each - see Table 11). The largest difference is between Guidance and Control groups and Restrictive group mean falls in between, closer to the mean of the Guidance group. The small spread of means between the groups can be attributed to the small number, which is 61 among three groups, of subjects we have to use in the analysis. By increasing the sample size to a more reasonable level it would be possible to confirm if the Guidance system would yield better performance than the Restrictive system. The pair-wise comparison of Restrictive and Guidance systems was nearly significant ($p = 0.067$) and the difference in scores is approximately 16%.

The measure of ease-of-use was found to be higher for the Restrictive system, than the Guidance system, as hypothesized. The Restrictive system provides few number of choices and it automatically provides a structure to the problem solving process. These features were seen favorably by the non-expert users and hence they rated the Restrictive system higher than the Guidance system. The sixth hypothesis was that the Guidance

system would be rated higher in information satisfaction than the Restrictive system. This hypothesis was supported by the observations, although not at the 0.05 level, but at 0.1 level ($t = 1.41, p = 0.08$). The highest value possible is 5.0 and the Guidance system scored 4.41, where as the Restrictive system scored lower (4.16). Although it is not statistically significant, it can nevertheless be useful to system designers.

Overall, the Guidance system lead to better performance. However, the user found the Restrictive system to be easier to use and were more satisfied with the system. This finding throws up an interesting problem for system designer. Making the system restrictive may result in lower performance, but would be rated higher than making the system less restrictive and more guidance oriented. Since, the difference in scores of the Restrictive and Guidance systems is not very high, the recommendations to a system designer would be make the system restrictive and offer fewer choices to the user. These recommendations are more appropriate if the users are non-experts. Users who are expert in the domain may find the restrictiveness too stifling.

Generalizability

Generalizability refers to what extent the effects that were observed in the experimental setting will also occur in the untested universe (Fromkin and Streufert 1976). By definition, generalizability requires extrapolation to realms not considered in this study. The idea here is to determine if the relationship between the type of system used and the performance in design problem solving holds good outside the particular confinements of

the laboratory experiment. The concepts of *internal validity* and *external validity* can be used determining the generalizability of the research findings (Campbell and Stanley 1963). Internal validity refers to the correctness of the claim of the relationship between the independent and dependent variables. In this laboratory experiment, internal validity refers to the extent of the effect of system type on performance. In other words, are the differences in performance in the experimental task (measured by ETS) attributable to the system type? In this experiment a number of procedures were followed which improved the internal validity. The effects of uncontrollable designer characteristics, such as knowledge in information systems and other demographic characteristics, were minimized by randomly assigning the subjects to treatments. The performance in the experimental task would also be affected by the subject's knowledge in database design. None of the subjects had been exposed to E R modeling before this study. They were trained on database design by the same instructor who followed the same script. Although all the subjects received the same training, it is possible that their knowledge in design problem solving was not homogeneous. Hence, a measure of their prior knowledge is taken before the treatment. This measure (PTS) was used as the co-variate in the analysis. Hence, it can be argued that any differences in the performance are attributable only to the type of system that was used. Thus, the internal validity of the experimental findings can be established.

According to Fromkin and Streufert (1976) "external validity refers to the degree to which the experimental effects can be generalized to other populations, settings, treatment variables, and measurement dependent variables". In this study, the target population for whom the results are generalizable would consist of non-expert or beginner level database designers. The settings which were used, by nature of the experiment, is artificial. However, the criterion setting is likely to be different from the experimental setting. In a real database design project, the designers may not have same type of environment as faced by the subjects. For example, the designer may be working in a team of designers and users. In this study, it was necessary to control the expertise of the designer so that the effects of the system types can be studied.

Will the system type have the similar effect when used by designers of different expertise? The knowledge based system assists the designer by behaving like an external assistant. It shares the designer's cognitive load by reminding and prompting the designer at appropriate situations. Thus the designer can devote more cognitive and attentional resources to the semantics of the task. Hence, the assisting capabilities of the system will be found useful during and effective in easing the problem solving process. However, for the system to be effective, the designer needs to have some qualifying knowledge in the design problem solving. For example, would the system be effective in helping designers who are 'absolute beginners', or people who have even less knowledge than the participants of this study? As it is evident from this study, designers who substantively

lack knowledge on modeling relationships, the system was of little help. On the other hand, if the designer, is an 'advanced beginner' who has more expertise than the subjects considered in this study, the system would still be effective in helping the designer manage the design process. However, if the task is not complex, the designer may be able to solve it with little help from the system. The system would be particularly more effective in solving more complex problems by 'advanced beginners'. Problems that have multiple higher order relationships (ternary, 4-way) are difficult problems even for 'advanced' database designers. Hence the effect of the system would continue to be pronounced for more difficult problems.

Will similar strategies of knowledge implementation (restrictive or guidance), yield similar results in another domain? The concepts of system restrictiveness and decisional guidance have been applied in classifying CASE tools also. Any CASE tool supports a certain system design methodology and hence it is a methodology companion. A CASE tool, by definition, does have certain knowledge of the specific methodology implemented in it. Hence, it is a knowledge based tool. Vessey *et al.* (1992) have used a typology that in classifying CASE tools as one of (a) guidance (b) restrictive and (c) flexible. The applicability of knowledge implementation strategy is thus a well accepted notion among both the researchers and practitioners. Hence, the external validity of effects of system type on performance can be established in domains other than database

modeling. Next, some future research directions and other research implications of this study are discussed.

Implications

Information systems researchers have traditionally relied on "reference disciplines" as sources for intellectual capital, research methods, folkways and mores (King 1993). They have contributed to the cumulative knowledge and IS field itself have many theoretical explanations for the IS phenomena. Concepts that are specific to IS are include Information Systems Design Theory and System restrictiveness and decisional guidance. In this research project it has been demonstrated that these concepts can be applied in a scientific framework. This demonstration would contribute, although in a small way, to what Benbasat and Weber (1996) call as, an important goal of attaining disciplinary status.

There are some implications for practitioners, that is, CASE tool designers also. The general focus of database researchers has been primarily on database management, design and modeling (Lai 1996). This study has contributed to data modeling research. It has been demonstrated that data modeling knowledge can be implemented in a system and that subjects with little training can use the system to solve moderately difficult problems. The finding - that the users of the knowledge based system outperformed the control group - lends credence to the rules and heuristics that were embedded in the knowledge base. Developers of database design tools would find this result useful. The

design tool that was used in this study is similar to many CASE tools used by practitioners today. One of the findings of this study imply that a restrictive tool would be rated high on ease of use. This finding attains importance in the light of the report that the Ease-of-use construct has a large influence on CASE acceptance (Chau 1996). CASE tool designers and vendors might find this result useful.

Future research

Various degrees of system restrictiveness and decisional guidance can be found in most CASE tools (Vessey *et al.* 1993). In this study all the subjects had similar expertise, i.e. non-experts and they solved only one problem using the system. Since, performance depends on, among others, user expertise, user characteristics, problem complexity and system type a number of important and interesting research ideas may be pursued.

First, it is possible that the more advanced designers would find the restrictive system too stifling, and they would prefer to receive 'guidance on demand'. Hence, a series of experiments may be designed to compare the performance of experts versus non-experts. The user expertise and system type interaction effect on performance, will help us ascertain if non-experts need more restrictiveness more than experts. It is possible to consider different levels of expertise. In this study the designers can be called novices. They had only a few hours of training on database design and Entity Relationship modeling. Would the knowledge based system be effective for advanced beginners, such as subjects with more experience in database design? An affirmative answer to the above

question will validate the expertise embedded in the system. This type of validation must be given high priority. The subjects used in this study were not instructed on all the rules and heuristics embedded in the system. For example, the subjects were not taught *why* a certain sequence should be followed while modeling relationships. Would the system still make a difference if the subjects *had* been instructed on the all the heuristics? Answer to this question can be found by comparing their performance between a system assisted problem solving and manual problem solving exercises.

Second, the problem complexity can be varied. An increase in the number of entities will lead to a disproportionate increase in the number of possible relationships. Designers would be more in need of help from the system for more complex problems. Hence, it would be desirable to find if use of the system would yield significantly better performance for complex problems also.

Third, the database design support system can be improved in a number of ways and its effectiveness can be studied. First, it is possible to include the concepts of generalization and aggregation (Smith and Smith 1977) in data modeling. This enhancement would make the system suitable for object oriented analysis. Even in extended ER modeling, the importance of relationships remain high and hence we can expect that the system would continue to be effective. Second, the extent of design support can be expanded to include logical and physical design also. Teorey *et al* (1986) provide a *one-to-one* mapping from ER constructs to relations. The system can be programmed to create SQL code for

creation of the data tables. Research has revealed that designers perform better if they received feedback during design process. Since for each conceptual model there is one unique logical model, the designer would be able to *preview* the design and modify it if necessary. The opportunity of viewing the table structures during conceptual design has its advantages and disadvantages. It can lead to more cognitive load and the designer's overall performance may deteriorate. On the other hand, viewing the table structures along with the graphical representation can be expected to improve the subject's overall understanding of the problem. Third, the system can be enhanced to include concepts of *clustering* entities, because in real life projects it is necessary to cluster the entities and model the relationships among the entities in each cluster and then integrate the clusters of ER diagrams. Fourth, the system can be programmed to detect derived relationships. Algorithms for such detection of derived relationships are available (see *e.g.* Maier 1988). Currently, the users have to discover and model relationships. The system would not assist them with determining if a relationships should be binary or ternary. With the derived relationships detection feature, the user may enter all the relationships that he/she thinks are representative. Among those relationships, the system will exclude the derivable relationships. This enhancement would make the system more effective even when used by non-experts.

The analysis in this project focused only on the final design. Since the users of the guided system performed better than the control group, it is apparent that the guidance messages

were useful. Yet, there was high variance in their performance. From the logs, it is possible to find to what extent the subjects followed the guidance messages. Further analysis can be done using only the guided system group to find whether there is a relationship between 'degree of adherence to guidance messages' and performance. The process based analysis will throw more light at the user-computer interactions and will be beneficial to interface developers.

Limitations

A number of limitations can be attributed to this study. First, the database design tool has been developed for a specific user population, non-expert designers. Although the embedded knowledge base will be found useful, performance of expert designers may not improve significantly by using this system. Secondly, as revealed by the experiment, the system is not suitable for total novices; *i.e.* people who have little knowledge about modeling relationships. Thirdly, in retrospect, the level of training was perceived to be insufficient. Prior to the experimental session, subjects had solved only one practice problem. Had the subjects been given more practice problems with feedback, it could have been ensured that more subjects attained the minimum level of comprehension.

Chapter 8

CONCLUSION

In this research project, a knowledge based consulting system for conceptual database design was developed. The interaction of the user with the knowledge base can be facilitated through (a) a Guidance type interface or (b) a Restrictive type interface. The Guidance type interface, provides context sensitive and informative guidance messages to the user throughout the design problem solving process. The Restrictive system limits the options the user can access at any given time. It embeds a 'normative' design process and forces the user to follow the normative process. The knowledge sources for design methodology embedded in the system include (a) analysis of novice errors in database design (b) a list of rules and heuristics and (c) theories of database design. The objective of this research project is to compare the effectiveness of the knowledge based system with a system that does not have embedded knowledge base. For comparison purposes, a third system called Control system was also developed. The control system nor restricts the user's choices neither provides guidance messages during the design process. Undergraduate students enrolled in an Information Systems class were the proxies for non-expert database designers. After being trained on Entity Relationship modeling and on how to use the system, each subject solved a complex database design problem. Their solutions were graded by two independent graders and the reliability of the grading scheme was established.

Statistical analysis of the scores of subjects on the data modeling problem revealed that (a) the knowledge based systems resulted in better performance and (b) among the two implementations, the Guidance knowledge base implementation resulted in better performance than the Control system. These findings lend credibility to the knowledge embedded in the system, at the same time validating the knowledge based tool also. The restrictive system was perceived to be more easy to use and where as the subjects recorded higher satisfaction with the Guidance system than the Restrictive system. These findings would be useful to system designers. Particularly, designers of systems for problem solving - including decision making, database design, data flow diagramming etc. - would perceive the findings that (a) Restrictive system rates higher in ease-of-use and user satisfaction and (b) Guidance system leads to better performance than the Restrictive system.

Contributions

There have been many research projects where the main research objective was to develop a knowledge based tool for conceptual modeling (Storey and Goldstein 1993). Although, their objectives are of genuine interest to the user community, there is no empirical validation of the tool in a scientific setting. This project distinguishes itself by an empirical demonstration of the validity of tool.

The knowledge implementation is based on well established concepts of Guidance and Restrictiveness. The implementations, have demonstrated the prospects of developing

similar knowledge based systems in other system design problem domains as well. CASE tools designers can successfully use the concepts of Guidance and Restrictiveness in developing tools for user populations with specific characteristics.

The Wall *et al* (1990) framework was used to clearly articulate the meta-requirements. Since it was desired to assist non-expert designers, research findings about non-experts and their problem solving behavior were effectively used in formulating this research project.

The use of published set of 'rules and heuristics' (Batra and Zanakis 1994) and other public domain expertise as the source of knowledge renders this research project more ingenuous. The embedded knowledge ensures that derived relationships are prevented. The guidance system warns users about the potential dangers of derived relationships and the restrictive system prevents the user from making design decisions that would lead to derived relationships. So, the resulting database structures are automatically in the third normal form. The implementation of rules are aimed at preventing *literal translation* (see Batra and Antony 1994a) of case description to data structures, and the implementation was found to be effective in this research project.

Future research directions

The behavior of experts and novices have been studied in many domains. Use of the concepts of system restrictiveness and decisional guidance along with the various levels of

user expertise would be very useful in building information system design theories. For example, questions like "Do novices like to use and perform better if they use restrictive systems?" or "Do advanced beginners find the guidance system easy to use and perform better?" can be answered with the design support tool. Another research dimension is the complexity of the database design task. The question of whether problems of increased complexity are more easily solved with the restrictive system would also be of interest.

An interesting finding of this research project is that CODA's effectiveness is more pronounced when used by designers who had at least the minimum comprehension of modeling relationships. If the designer is a total novice, the system will not be effective in assisting the user. This finding leads to more interesting research ideas such as "What level of expertise is needed before the system can be effective?" and "Is there an interaction effect of expertise and system characteristics on performance?". MIS researchers, practitioners and CASE tools designers will benefit by using the current findings and findings from the future research also.

REFERENCES

1. Adams D.A, Nelson R.R. and Todd P.A. (1992). Perceived usefulness, ease of use and usage of information technology: A replication. *MIS Quarterly*, **16**, 227-247

2. Anderson J.R (1980). *Cognitive Psychology and its implications.* W H Freeman and Co., New York.

3. Anderson J.R. (1983). *The architecture of cognition.* Harvard University Press, Cambridge, Mass.

4. Anderson J.R. Bole C.F and Reiser B.J (1985). Intelligent Tutoring Systems. *Science*, **228**, 456-462.

5. Anderson J.R., Corbett A.T., Fincham J.M., Hoffman D. and Pelltier R. (1992). "General principles for an intelligent tutoring architecture" in *Cognitive Approaches to Automated Instruction.* V.J. Shute and J.W. Regian. (Eds)

6. Balzer W.K, Sulsky L.M, Hammer L.B and Sumner K.E. (1992). Task information, cognitive information, or functional validity information: Which components of cognitive feedback affects performance?. *Organizational Behavior and Human Decision Processes*, **53**, 35-54.

7. Batini C., Ceri S., and Navathe S.B. (1992). *Conceptual Database Design: An Entity-Relationship Approach.* Benjamin Cummings. Redwood City, CA.

8. Batra D. and Antony S.R. (1994a). Novice errors in conceptual database design. *European Journal of Information Systems*, **3**, 57-69.

9. Batra D. and Antony S.R (1994b). Effects of data model and task characteristics on designer performance: A laboratory study. *International Journal of Human Computer Studies. 41, 481-508.*

10. Batra D. and Davis J. (1992). Conceptual data modelling in database design: Similarities and differences between expert and novice designers. *International Journal of Man-Machine Studies*, **37**, 83-101.

11. Batra D. and Hoffer J.A. and Bostrom R.P. (1990). Comparing representations with the Relational and EER models. *Communications of the ACM*, **33**, 126-139.

12. Batra D. and Kirs P. (1993). The Quality of Data Representations Developed by Novice Designers: An Experimental Study. *Journal of Database Management*, **Fall 1993**, 17-29.

13. Batra D. and Srinivasan A. (1992). A review and analysis of the usability of data management environments. *International Journal of Man-Machine Studies*, **36**, 395-417.

14. Batra D. and Zanakis S.H (1994). A conceptual database design approach based on rules and heuristics. *European Journal of Information Systems*, **3**, 228-239.

15. Benbasat I, Dexter A.S. and Marulis P.S. (1981). An experimental study of the human computer interface. *Communications of the ACM*, **24**, 752.

16. Benbasat I. And Weber R. (1996). Research commentary: Rethinking "Diversity" in Information Systems Research. *Information Systems Research*, **7**, 389-399.

17. Bhaskar R. and Simon H.A. (1977). Problem solving in semantically rich domains: An example from engineering thermodynamics, *Cognitive Science*, **1**, 193-215.

18. Bock D. B. and Ryan T. (1993). Accuracy in modeling with extended entity relationship and object oriented data models. *Journal of Database Management*, **4**, No. 4, 30-39.

19. Bouwman (1984). Expert versus novice decision making in accounting: a summary. *Accounting, Organizations and Society*. **9**, 325-327.

20. Campbell D.T. and Stanley J.C. (1966). *Experimental and quasi-experimental designs for research*. Chicago: Rand McNally.

21. Carroll J.M. and Kellog W.A (1989) Artifact as theory-nexus: Hermeneutics meets theory-based design. *CHI'89 Proceedings*, Austin, Texas.

22. Catrambone R. (1990). Specific versus general procedures in instructions. *Human Computer Interaction*, **5**, 49-93.

23. Chau P.Y.K. (1996). An empirical investigation of factors affecting the acceptance of CASE by systems developers. *Information and Management*. **30**, 269-280.

24. Chen P.P. (1976). The Entity-Relationship Model - Toward unified view of data. *ACM Transactions on Database Systems*, **1**, 9-36.

25. Chi M.T.H., Feltovich P.J. and Glaser R. (1981). Categorization and representation of physics problems. *Cognitive Science*, **5**, 121-152.

26. Choobineh J., Konsynski B.R., Mannino M.V. and Nunamaker J.F. (1988). An expert system based on Forms, *IEEE Transactions on Software Engineering*, **14**, 242-253.

27. Clancy W.J. (1982). "Tutoring rules for guiding case method dialog". In *Intelligent Tutoring Systems*, D. Sleeman and J.S. Brown (Eds). Academic Press, New York.

28. Codd E. (1970). A relational model for large shared data banks. *Communications of the ACM*, **13**, 377-387.

29. Cohen J (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, **1**, 37-46.

30. Davis F.D. (1985). A technology acceptance model for empirically testing new end-user information systems: Theory and results. Ph. D Dissertation, Massachusetts Institute of Technology, Sloan School of Management.

31. de Leeuw L. (1983). Teaching problem solving: An ATI study of the effects of teaching algorithms and heuristic solution methods. *Instructional Science*, **12**, 1-48.

32. Demo B. and Tilli M. (1986). "Expert system functionalities for database design tools" in *Applications of Artificial Intelligence in Engineering Problems : Proceedings of the 1st International Conference*, D. Sriram and R. Adey (Eds.) April 1986, Springer-Verlag, Berlin, pp. 1073-1082.

33. Doll and Torkzadeh (1988). The measurement of end-user computing satisfaction. *MIS Quarterly*, **June 1988**, 259-274.

34. Dos Santos B.L. and Bariff M.L. (1988). A study of user interface aids for model oriented decision support systems. *Management Science*, **34**, 461-468.

35. Dreyfus and Dreyfus (1986). *Mind over machine : the power of human intuition and expertise in the era of the computer.* Free Press, New York.

36. Duffield J.A. (1991). Designing computer software for problem solving instruction. *Educational Technology Research and Development*, **39**, 50-62.

37. Dufresne R.J., Gerace W.J., Hardiman P.T. and Mestre J.P. (1992). Constraining novices to perform expert-like problem analyses: Effects on schema acquisition. *The journal of the learning sciences*, **2(3)**, 307-331.

38. Fromkin H.L, Streufert S. (1976). "Laboratory Experimentation" in Dunette, Marvin D. (Ed.) *Handbook of Industrial and Organizational Psychology* (Chapter 10). Rand McNally College Publishing CO., Chicago, Illinois.

39. Gill T.G. (1996). Expert systems usage: Task change and intrinsic motivation. *MIS Quarterly*, **20**, 301-329.

40. Hardgrave B.C. and Dalal N.P. (1995). Comparing Object-oriented and Extended Entity Relationship data models. *Journal of Database Management*. Summer 1995, 15-21.

41. Heller J.I. and Hungate H.N. (1985). Implications for mathematics instruction of research on scientific problem solving. In E.A. Silver (Ed.) *Teaching and learning mathematical problem solving: Multiple research perspectives*, Lawrence Erlbaum Associates, Hillsdale, NJ pp. 83-112.

42. Heller J.I and Reif F. (1984). Prescribing effective human problem solving processes: Problem description in physics. *Cognition and Instruction*, **1**, 177-216.

43. Hendrickson A.R., Glorfeld K. and Cronan T.P. (1994). On the test-retest reliability of end-user computing satisfaction, A comment, *Decision Sciences*, **25**, 655-667.

44. Hendrickson A.R., Massey P.D. and Cronan T.P. (1993). On the test-retest reliability of perceived usefulness and perceived ease-of-use. *MIS Quarterly*, **17**, 227-230.

45. Hill J.E. and Kerber A. (1967). *Models, methods and analytical procedures in education research*. Wayne State University Press, Detroit.

46. Hill W.C. (1989). How some advice fails. *CHI'89 Proceedings*, Austin, Texas.

47. Hinsley D.A., Hayes J.R. and Simon H.A. (1977). From words to equations: meaning and representations in algebra word problems. In P.A. Carpenter and M.A. Just (Eds.) *Cognitive processes in comprehension*, Lawrence Earlbaum, Hillsdale, NJ.

48. Jarvenpaa S.L and Machesky J.J. (1989). Data analysis and learning: An experimental study of data modeling tools, *International Journal of Man-Machine Studies*, **31**, 367-391.

49. Jeffries R., Turner A., Polson P. and Atwood M. (1981). Processes involved in designing software. In J.R. Anderson (Ed.), *Cognitive skills and their acquisition*, Erlbaum, Hillsdale, NJ.

50. Juhn S. and Naumann J.D. (1985). The effectiveness of data representation characteristics on user validation. *Proceedings of the Sixth International Conference on Information Systems*, Indianapolis.

51. Kahneman D. (1973). *Attention and effort*. Prentice Hall, Englewood Cliffs, NJ

52. Kahneman D., Slovic P., and Tversky A. (1982). *Judgement under uncertainty: Heuristics and biases*, Cambridge University Press, New York.

53. Kanfer R. and Ackerman P.L. (1989). Motivation and cognitive abilities: An integrative aptitude treatment interaction approach to skill acquisition. *Journal of Applied Psychology*, **74**, 657-690.

54. Kawaguchi A., Taoka N., Mizoguchi R., Yamaguchi T. and Kakusho O. (1986). An intelligent interview system for conceptual design of database. *ECAI'86: The 7th European Conference on Artificial Intelligence,* Conference Services Ltd., London, pp. 1-7.

55. Kerlinger N.D. (1986). *Foundations of Behavioral Research*. 3rd Edition. Holt, Reinhart and Winston, New York.

56. King A. (1991). Effects of training in strategic questioning on children's problem solving performance, *Journal of Educational Psychology*, **83**, 307-317.

57. King J.L. (1993). Editorial notes. *Information Systems Research*. **4**, 291-298.

58. Lai V.S. (1996). An assessment of database research interest in MIS. *The database for advances in information systems,* **27**, No.2, 37-43.

59. Larkin J., McDermott J., Simon D. and Simon H. (1980). Models of competence in solving physics problems. *Cognitive Psychology*, **83**, 307-317.

60. Laurel B.K. "Interfaces as Mimesis" in *User-Centered System Design*, D.A. Norman and S.W. Draper (Eds.). Lawrence Erlbaum, Hillsdale N.J. pp 67-86.

61. Maier D. (1988). *The theory of relational databases.* Computer Sciences Press, Rockville, Maryland.

62. Mantha R.W. (1987). Data flow and data structure modeling for database requirements determination: A comparative study. *MIS Quarterly*, **1987, December**, 531-545.

63. Martin J. and Leben J. (1995). *Client/Server Databases: Enterprise computing*, Prentice Hall, Upper Saddle River, NJ.

64. McFadden F.R. and Hoffer J.A. (1991). *Database Management* 3rd Edition, Benjamin/Cummings, Menlo Park, CA.

65. McKeithen K.B., Reitman J.S., Reuter H.H. and Hirtle S.C (1981). Knowledge organization and skill differences in computer programmers. *Cognitive Psychology*, **13**, 307-325.

66. Morris N.M. (1987). "Designing for user acceptance of design aids" in W. Rouse and K. Boff (Eds.) *System Design* Elsevier Publishing Co Inc. New York.

67. Newell A. and Simon H. (1972). *Human Problem Solving*, Prentice Hill, Englewood Cliffs, NJ.

68. Norman D.A. (1983). Design rules based on analysis of human error, *Communications of the ACM*, **26**, 254-258.

69. Norman D.A. and Bobrow D.B. (1975). On data-limited and resources limited processes, *Cognitive Psychology*, **7**, 44-64.

70. O'Neil H.F and Walther G.H. (1974). "On-line user-computer interface - The effects of interface flexibility, terminal type and experience on performance" in *Proceedings of the National Computer Conference*, San Diego, ACM, 379.

71. Paradice D.B. and Courtney Jr. J.F (1988). SmartSLIM: A DSS for controlling biases during problem formulation. In *Human factors in information systems*, Jane M. Carey (Ed.) pp83-100. Ablex Publishing Corp, New Jersey.

72. Presley M., Woloshyn V., Lysynchuk L.M., Martin V., Wood E., and Willoghby T. (1990). A primer of research on cognitive strategy instruction: The important issues and how to address them. *Educational Psychology Review*, **2**, 1-58.

73. Pressman R.S. (1982). *Software engineering: A practitioner's approach*, Prentice-Hall, Englewood Cliffs, NJ.

74. Ram S. (1995). Deriving functional dependencies from the Entity-Relationship model. *Communications of the ACM*. **38**, 95-107.

75. Reason J. (1988). "Framework models of human performance and error: A consumer guide". In *Tasks, errors and mental models*, L.P. Goodstein, H.B. Anderson and S.E. Olsen (Eds.) pp. 35-49, Taylor and Francis, UK.

76. Ridjanovic D. (1986). Comparing quality of data representations produced by non-experts using logical data structure and relational model. Ph D. Dissertation, University of Minnesota, Carlson School of Management.

77. Rouse W.B. (1991). *Design for success.* John Wiley & Sons, New York.

78. Sage A. (1981). Behavioral and organizational considerations of information systems and processes for planning and decision support. *IEEE Transactions on Systems, Man and Cybernetics*, **SMC-11**, 640-678.

79. Scheer A.W. (1989). *Enterprise-wide data modelling: Information systems in industry*, Springer Verlag, Berlin.

80. Segers A.H. and Grover V. (1993). Re-examining perceived ease-of-use and usefulness: A confirmatory factor analysis, *MIS Quarterly*, **17**, 517-525.

81. Senders J.W. and Moray N.P. (1991). *Human error: Cause, prediction and reduction*, Lawrence Erlbaum Associates, Hillsdale, NJ.

82. Silver M.S. (1990). Decision support systems: Directed and non-directed change. *Information Systems Research*, **1**, 47-70.

83. Silver M.S. (1991). *Systems that support decision makers: description and analysis.* John Wiley & Sons, Chichester, England.

84. Smith J.M and Smith D.C.P. (1977). Database abstractions: Aggregation and Generlization. *ACM Transactions on Database Systems.*, **2**, 105-133.

85. Srinivasan A. and Te'eni D. (1992). Modeling as constrained problem solving: An empirical study of the data modeling process. In *Proceedings of the Eleventh International Conference on Information Systems*, Copenhagen.

86. Stone E. (1978). *Research Methods in Organizational Behavior.* Scott Foresman, Glenview, Illinois.

87. Storey V.C (1988). *View Creation: An expert system for database design.* ICIT Press, Washington D.C.

88. Storey V.C. and Goldstein R.C. (1993). Knowledge-based approaches to database design, *MIS Quarterly*, **17**, 25-46.

89. Sweller J. (1989). Cognitive technology: Some procedures for facilitating learning and problem solving in mathematics and science, *Journal of Educational Psychology*, **81**, 457-466.

90. Tarmizi R.A and Sweller J. (1988). Guidance during mathematical problem solving, *Journal of Educational Psychology*, **80**, 424-436.

91. Tauzovich B. (1989). An expert system for conceptual data modeling. *Proceedings of the eighth International Conference on the Entity-Relationship Approach*, Toronto, Ont., October 1989, pp. 329-344.

92. Teorey T.J. (1990). *Database modeling and design/* Morgan Kaufmann Publishers, San Mateo, California.

93. Teorey T.J., Yang D., and Fry J.P (1986). A logical design methodology for relational databases using the extended entity relationship model. *ACM Computing Surveys.* **18(2)**, 197-222.

94. Todd P.A., McKeen J.D., and Gallupe R.B.(1995). The evolution of IS job skills: A content analysis of IS job advertisements from 1970 to 1990. *MIS Quarterly*, **1995 March**, 1-28.

95. Torkzadeh G. and Doll W.J. (1991). Test-retest reliability of the end-user computing satisfaction instrument. *Decision Sciences*, **22**, 26-37.

96. Towne D.M. and Munro A. (1992). Supporting diverse instructional strategies in a simulation-oriented training environment. In V.J. Shute and J.W. Regian (1992) pp 107-134.

97. Walls J.G., Widmeyer G.R., and El Sawy O.A. (1992). Building an information system design theory for vigilant EIS. *Information Systems Research*, **3**, 36-59.

98. Vessey I., Jarvenpaa S.L., and Tractinsky N. (1992). Evaluation of vendor products: CASE tools as methodology companions. *Communications of the ACM*, **35**, No. 4, 90-105.

99. Zook K.B. and Di Vesta F.J. (1989). Effects of overt controlled verbalization and goal-specific search on acquisition of procedural knowledge in problem solving. *Journal of Educational Psychology*, **81**, 220-225.

APPENDIX

Appendix 1: E R Methodology Training Script

Notes on Conceptual Database Design

Database design is the process of determining the organization of a database, including its contents, structure and the applications to be run. Database design is normally done in three phases. The first phase, called the *conceptual design*, produces an abstract representation of reality. The second phase, called *logical design*, translates this representation into specifications that can be implemented on and processed by a computer system. The third phase, called *physical design*, determines the physical storage structures and access methods required for efficient access to the contents of the database from secondary storage devices. In this document, procedures for conceptual database design are explained.

During the conceptual database design, typically a description of the business data requirements - the case description - is the input document. The outcome of the conceptual database design process is a graphical representation of the data requirements using Entity-Relationship Modeling concepts. The representations in the ER diagram can then be converted into logical design. In this instructional script, the Entity-Relationship model (ER Model), the most widely used data model for conceptual database design, is explained. The basic concepts provided by the ER model are *attributes*, *entities* and *relationships*.

Attributes: Attributes represent elementary properties or characteristics of the objects that we wish to represent in a data model. For example, if we wish to represent information about a student, the student_number, student_name, address and major will be the attributes. Attributes are graphically represented as ellipses attached to entities. (see Figure 1).

How to determine attributes?

Each attribute typically has a simple atomic structure and no further property of the structure seems of interest. Typically, to decide whether an object is an attribute, use the following question: "Can I think of *a value* for the attribute?". A value can either be of alphanumeric string (e.g Student_name, SSN, Product_code) or a numeric (e.g. Price, Quantity, GPA) or a date (e.g. Order_date, Date_of_Reservation) type. It is possible to determine what attributes are of interest from the case description.

Entities: Entities represent classes of real world objects. Usually, we represent a person, place, object, event or a concept as an entity. For example, EMPLOYEE, PATIENT, EMPLOYEE, STUDENT, DEPARTMENT, COMPANY, MACHINE, BUILDING, AUTOMOBILE, COURSE, INVOICE, ORDER may be

represented as entities. Each entity typically, has at least 2 or more attributes associated with it. One of the attributes of an entity is called the *key-attribute*. A key-attribute (also called an identifier) has unique value for each instance of the entity. For example, Employee_number would be the key-attribute of the EMPLOYEE entity, since each employee has a unique employee number. Employee_name may not qualify as the key-attribute since there may be more than one employee with the same name. The other attributes of the entity are called non-key attributes. Entities are graphically represented as rectangles. The attributes of the entity are represented as ellipses attached to the rectangle. The key-attribute is distinguished by the underline (See Figure 1).

How to determine the entities?

If there are some properties associated with an object, it can be modeled as an entity. In other words, if we can determine that a set of attributes describe an object, then the object can be modeled as an entity. In a case description an entity can be identified by the grouping of some descriptor objects. After identifying an entity, determine its key-attribute and non-key attributes. Each entity must have a key attribute. The key-attribute should represent a unique value for each instance of the entity. For the STUDENT entity the SSN would be an ideal key-attribute, since it is unique to each student. Another constraint on key-attribute is that it cannot be a combination of two or more attributes. Assign attributes to an entity that most directly describe the entity. After determining the key-attribute, ensure that each of the non-key attributes is functionally dependent on the key-attribute and no other attribute. To ensure the functional dependency, ask the following type of question for each non-key attribute: "Given a value of the key-attribute can I find one instance of the non-key attribute?". For example, if we have a set of attributes SSN, Name, Address, Major and Course_Number that we think belong the STUDENT entity then the following set questions would help in determining the correctness of assigning those attributes to the entity (Assume SSN is the key-attribute): (i) "Given a SSN can I find *one* instance of Name?" (ii) "Given a SSN can I find *one* instance of Address" (iii) "Given a SSN can I find *one* instance of Major?" and (iv) "Given a SSN can I find *one* instance of Course_number?". The answer is "Yes" to the first three questions and "No" for the last one. Either from the case description or common sense we know that given a SSN we cannot determine just *one* course number. Hence, Course_No does not rightly belong to the STUDENT entity (see Figure 3). Each non-key

attribute must depend on the key-attribute and no other attribute.

Relationships: Relationships represent an association of two or more entities. Each entity should participate in at least one relationship. In the final ER model there should not be any entity which does not participate in a relationship. If there are two entities that participate in a relationship, it is called a binary relationship, and a relationship between three entities is called a ternary relationship. The number of entities involved in a relationship is called the *degree* of the relationship. The following example illustrates a binary relationship: Assume we have two entities - EMPLOYEE and DEPARTMENT and it is known that an employee works for a department and a department may employ many employees. This suggests that there is a relationship between EMPLOYEE and DEPARTMENT. Let us call the relationship WORKS-FOR (See Figure 2a). Similarly, an example of a ternary relationship - MEETS that relates COURSE, CLASSROOM and SEMESTER - is given in Figure 2b. Binary relationships are represented as diamonds and ternary relationships as triangles in the E R Diagram.

Connectivity of a relationship: A relationship definition is incomplete without specifying the *connectivity* of the relationship. Connectivity is another characteristic of a relationship. It is represented as the combination of *cardinalities* of the entities involved in the relationship. Cardinality represents the number of instances of an entity that is associated with the combination of instances of each other entity in the relationship. Cardinality can either be *one* or *many*. For example, consider the WORKS-FOR relationship between EMPLOYEE and DEPARTMENT. In this relationship, *one* instance of DEPARTMENT is associated with an instance of EMPLOYEE and *many* instances of EMPLOYEE are associated with an instance of DEPARTMENT. Hence, the connectivity of WORKS-FOR is *one-many* with DEPARTMENT on the *one* side. Graphically, a *many* cardinality is represented as a dark triangle pointing the entity and a *one* cardinality is represented as an unshaded triangle pointing the entity (See Figures 2a and 2b).

How to determine the relationships?

The relationships can be determined from sentences in the case description that seem to relate entities. Usually relationships can be identified by the presence of an action between two or more entities. Consider the following sentences: (I) a salesperson *deals* with customers (ii) a customer *places* orders (iii) a student *enrolls* for courses in a semester and (iv) an employee is *assigned* to a project. These sentences suggest

existence of relationships. Although it is a useful rule-of-thumb, using information from just once sentence to decide on a relationship might not lead to correct solution. The presence of a sentence that relates three entities may not really represent a ternary relationship. For example, consider the following sentence: "A student enrolls in courses that are taught by instructors". This sentence seems to suggest a ternary relationship between STUDENT, COURSE and INSTRUCTOR. However, the correct answer would involve two binary relationships, one between STUDENT and COURSE and another between COURSE and INSTRUCTOR.

It is equally important to remember that information from all sentences in the case description should be used in determining relationships. Do not consider any sentence in isolation to determine the presence of relationships. For example, consider the following sentences from a case description: (I) "A patient may be treated by many doctors." (ii) "A doctor may prescribe many medications" and (ii) "A particular medication can be prescribed to a patient by only one doctor." The first sentence relates PATIENT and DOCTOR and it seems appropriate to define a binary relationship between doctor and patient. The second sentence suggests a relationship between DOCTOR and MEDICATION. Thus it seems correct to define 2 binary relationships (Figure 4a). However, the third sentence exhibits more complete information and suggests a ternary relationship between DOCTOR, PATIENT and MEDICATION and it seems a ternary relationship is more appropriate. From the third sentence we can infer that there is some kind of constraint. Whenever, there is a constraint, higher degree relationships are more accurate than lower degree relationships. Hence, in this case, it is incorrect to define the binary relationship, but correct to define a ternary relationship between DOCTOR, PATIENT and MEDICATION (see Figure 4b). Thus extreme care should be taken in determining the degree of relationship.

After the degree of a relationship is determined, i.e. whether it is a binary or ternary, determine the connectivity of the relationship. It can be determined by asking yourself a number of questions. For example, in the WORKS relationship between EMPLOYEE and DEPARTMENT (Figure 2a) the cardinality of the EMPLOYEE entity can be determined by asking the following question: "Given one instance of DEPARTMENT how many instances of EMPLOYEE are involved in the WORKS relationship?" or an equivalent and in more conversational English "How many employees work for a department?". The

answer to such question can either be *one* or *many*. In this example, the answer is "Many", since many employees work for a department. Thus the cardinality of Employee is *many*. The cardinality of the DEPARTMENT entity can be determined by asking "Given an instance of EMPLOYEE how many instances of DEPARTMENT are involved in the WORKS relationship?". The answer is "One", since an employee can work for only one department. The connectivity of the WORKS relationship between EMPLOYEE and DEPARTMENT is *many-one*. The connectivity of MEETS relationship between COURSE, CLASSROOM and SEMESTER (Figure 2b) can be determined as follows: (I) "Given an instance of a CLASSROOM, and an instance of a SEMESTER how many instances of COURSE are involved in the MEETS relationship?" In other words, "During a particular semester how many courses may meet in the same classroom?". The answer is "Many". (ii) "Given an instance of a SEMESTER, and an instance of a COURSE how many instances of CLASSROOM are involved in the MEETS relationship?" The answer is "One", provided the course meets in only one classroom for an entire semester. (iii) "Given instance of a CLASSROOM, and an instance of a COURSE how many instances of SEMESTER are involved in the MEETS relationship?" In other words, "Can a course be meeting in a particular classroom during only one semester or any number of semesters?". The answer is "Many". Hence the connectivity of the MEETS relationship between COURSE, CLASSROOM and SEMESTER is *one-many-many*.

Design procedure

Determine and verify the attributes. Determine and verify entities. Assign key and non-key attributes to the entities. Next determine the relationships from the case description. Model the relationships one-by-one. In the final ER diagram each entity should be connected with all other entities through the relationships. There should not be any relationship(s) or entities that are not connected to the rest of the diagram.

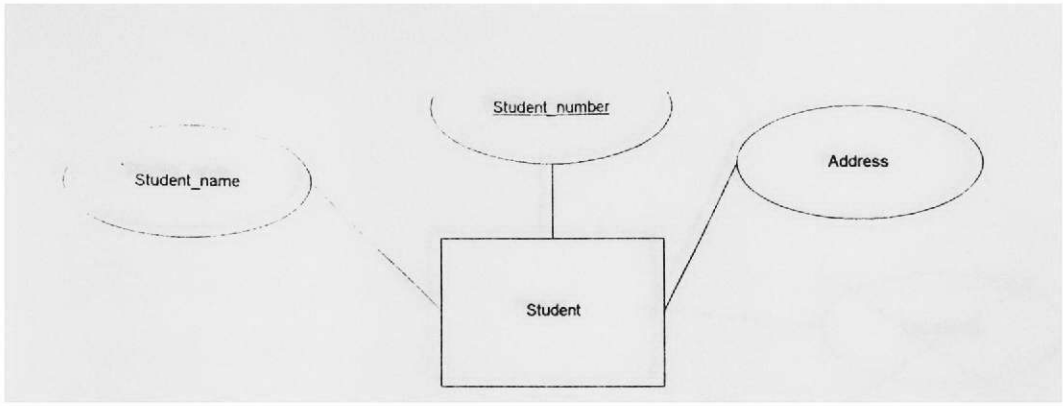


Figure 1 : Entity and Attributes

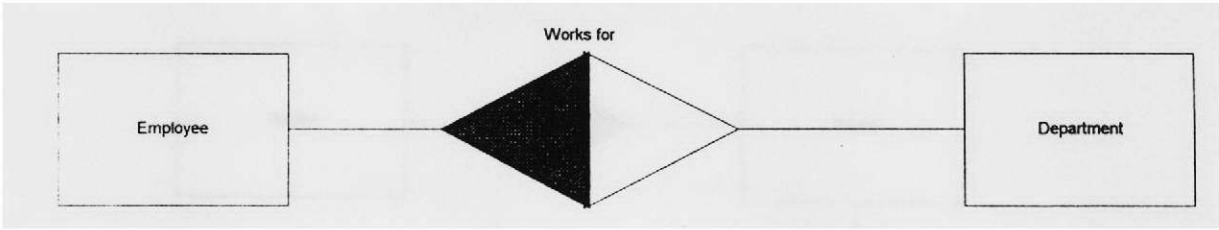


Figure 2a: Binary relationship

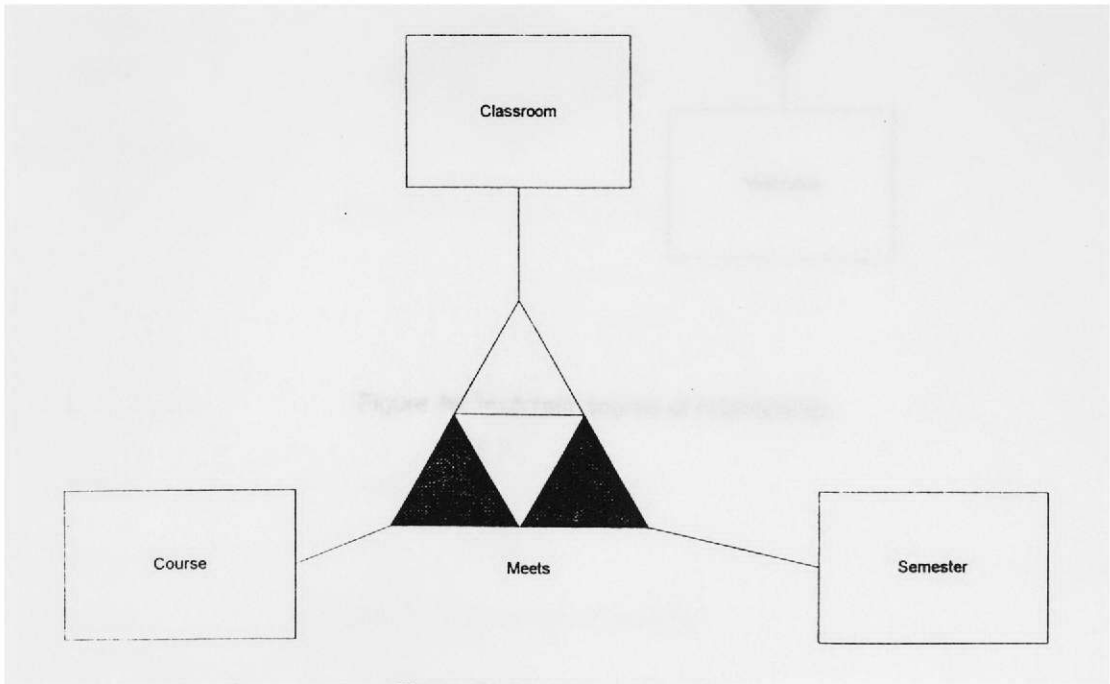


Figure 2b: Ternary relationship

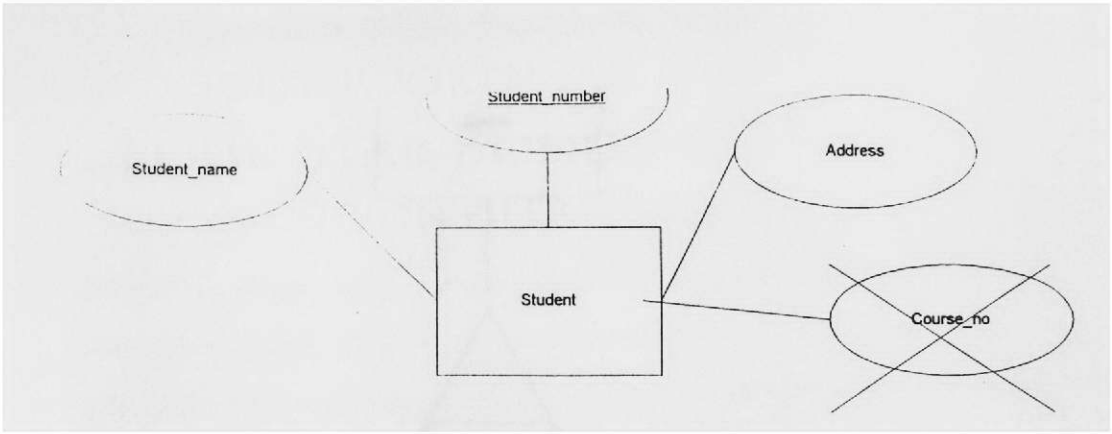


Figure 3 : Entity STUDENT

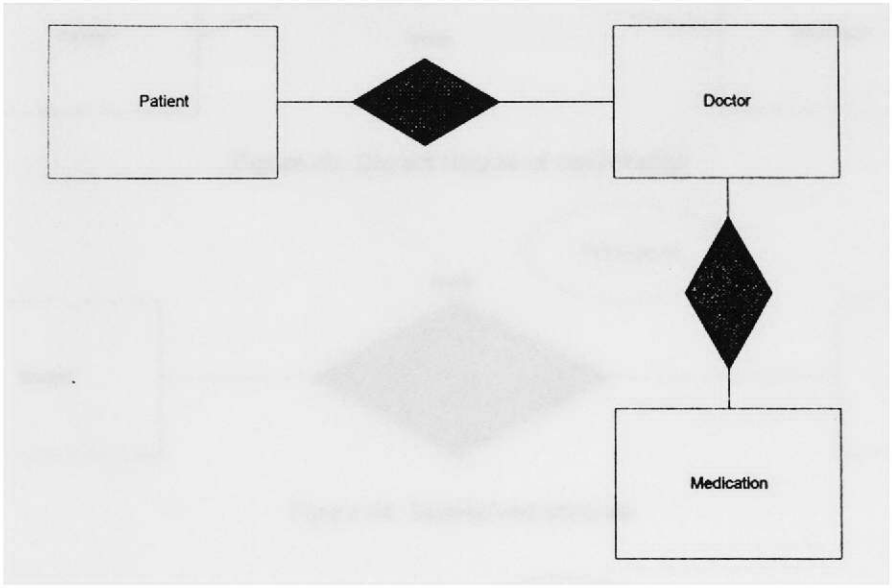


Figure 4a: Incorrect degree of relationship

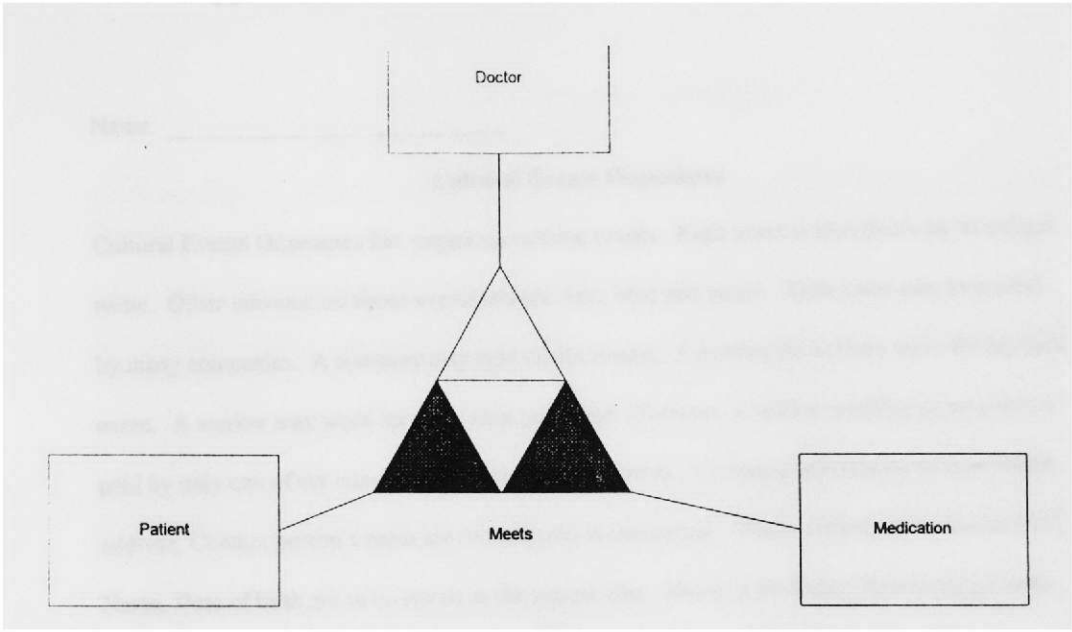


Figure 4b: Correct degree of relationship

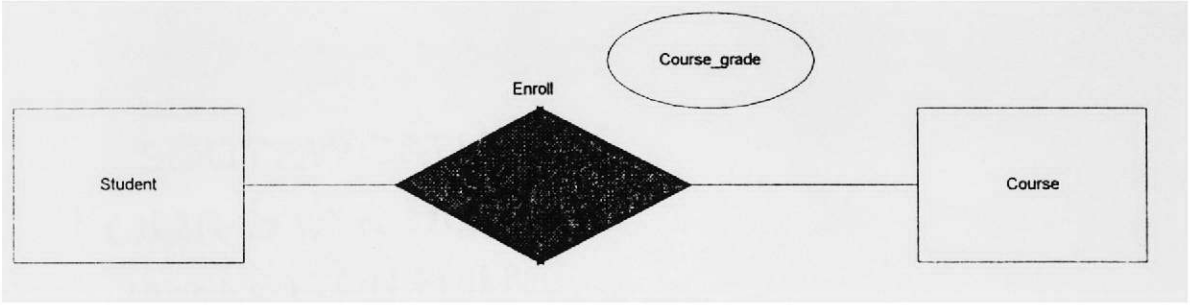


Figure 5a: Unresolved attribute

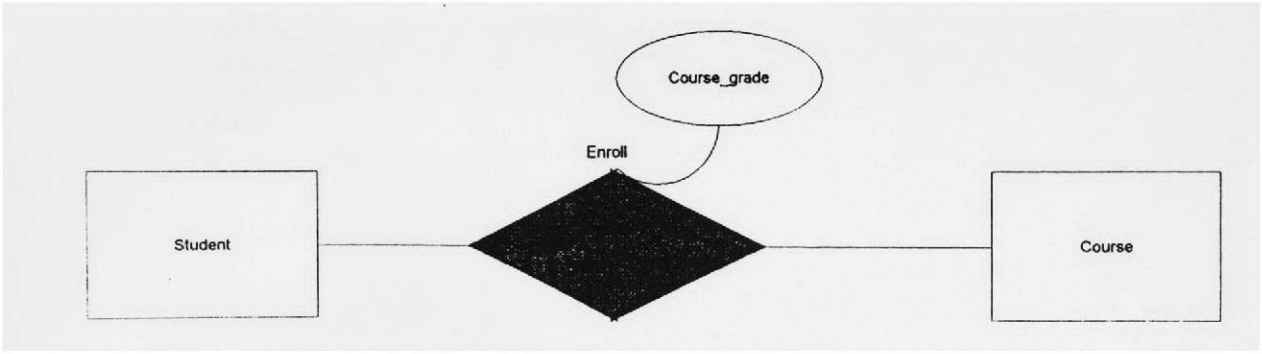


Figure 5b: Attribute of relationship

Appendix 2: In-Class Demonstration Problem

Name: _____

Cultural Events Organizers

Cultural Events Organizers Inc. organizes cultural events. Each event is identifiable by its unique name. Other information about events include date, time and venue. Each event may be hosted by many companies. A company may host manyt events. A number of workers work during each event. A worker may work for more than one event. However, a worker working on an event is paid by only one of the companies that sponsor the events. Company information such as Name, address, Contact person's name are to be stored in the system. Worker information such as SSN, Name, Date of birth are to be stored in the system also. Develop the Entity-Relationship model for this case description. Make any assumptions that deem appropriate.

Appendix 3: Quick Reference Guide

Quick reference - E R Modeling
How to determine....

Attribute: Can you think of a *value* for it? The value can be either numeric or character type or date type.

Entity: Are there attributes that describe it? Is there a key-attribute?

Relationship: Is there a sentence that suggests the presence of this relationship? Are there more sentences that relate the same set of entities? If yes, which is most appropriate?

Key - Attribute: Does each instance of the entity have a unique value for the key-attribute?

Assigning non-key attributes to an entity: Does the attribute directly describe the entity? Can one value be determined for this attribute, given a value for the key attribute? Does the value of this attribute depend on the key attribute only and no other attribute of the entity?

Connectivity of binary relationship: Given one instance of B, how many instances of the A participate in the relationship? (Ans = connectivity of A). Given one instance of A, how many instances of B participate in the relationship? (Ans = connectivity of B)

Connectivity of ternary relationship: Given one instance of B and an instance of C, how many instances of A participate in the relationship? (Ans = connectivity of A). Given one instance of A and an instance of C, how many instances of B participate in the relationship? (Ans = connectivity of B). Given one instance of the A and an instance of B, how many instances of C participate in the relationship? (Ans = connectivity of C).

Attribute of a relationship: Can a value for the attribute be determined for a given value of the key attribute of each entity individually? If the answer is NO to all the above questions, then Can a value of for the attribute be determined if you have a combination of the key attributes of all the entities in the relationship? If yes, then it is an attribute of the relationship.

Overall Procedure: Identify the attributes. Identify the entities. Determine and verify the key and non-key attributes of the entities. Assign the attributes to entities. Determine all the relationships from the case description. Use the system to enter the model and draw the ER diagram.

Appendix 4: Background Questionnaire

Background information

Name:_____ **Major** : _____

Age : _____ **Gender**: Male / Female

Computer related courses taken:

No.	Name	When (Approximately)
1		
2		
3		
4		

Name the software packages you are familiar with:

No.	Name	DOS/Windows/Mac/ Mainframe	Familiarity (Expert=5;beginner=1)
1			
2			
3			
4			
5			

Computer experience: (Check all that is applicable)

	Word Processing		4th Gen. Languages
	Spreasheet		DBMS basics
	Programming		DBMS applications

Appendix 5: Consent Form

Computer Aided Database Design

Purpose of the study

You have volunteered to participate in a study being conducted to train users on the use of a database design support system. The study will be conducted in a single lab session.

Description

During today's session you will (1) solve a database design task without using the software and (2) solve another database design task using the software. Your interaction with the computer will be recorded during this task session. You will also be asked to complete some questionnaires. The session will last about 2 hours.

Statement of consent

I agree to participate in the Database design study with the understanding that all the information collected from me will be kept confidential. I agree to follow all instructions laid down for the study. I understand that I can withdraw from the study at any time without any penalty. Should I withdraw, I realize that I will not get any participation points towards my CGS 3300 grade. However, I may earn these points through other options provided by my CGS 3300 instructor.

_____

Name (please print)

_____ _____

Signature Date

Appendix 6: Experiment Task

Name : _____

Super Systems Inc.

The people at Super Systems Inc (SSI) are in the business of developing software for large companies. They have a number of projects underway. Each project has a unique name is and is for a specific company. SSI may have many projects from the same company. Each project has a deadline before which it has to be completed. They wish to keep track of the company details such as company name, contact person and contact phone number. On each project a number of skilled programmers are employed. Each project may involve work on different platforms such as Vax, AS/400, Mac etc.. A programmer may work on many platforms, but for a given project, a programmer works on only one platform. Information about each platform such as Name, Operating system, date of last upgrade and name of manufacturer are to be stored in the database. It is necessary to store programmer information such as name and wage rate also. Each programmer has many language skills like C, C++, SmallTalk etc.. SSI wishes to code each language skill (with a unique code) and some description. Develop the Entity Relationship model for this case. Make any assumptions necessary.

Appendix 7: Ease-of-Use Questionnaire

User feedback questionnaire

Note : For questions 1 through 6 place a check mark in the box that is most representative of your opinion. For example, if you feel that "**Learning to use the system was easy for me**" is slightly correct place a check mark as shown below:

correct			X					incorrect
	extremely	quite	slightly	neither	slightly	quite	extremely	

1. **Learning to use the system was easy for me**

correct								incorrect
	extremely	quite	slightly	neither	slightly	quite	extremely	

2. **I find it easy to get the system to do what I want it to do.**

correct								incorrect
	extremely	quite	slightly	neither	slightly	quite	extremely	

3. **My interaction with the system was easy for me to understand.**

correct								incorrect
	extremely	quite	slightly	neither	slightly	quite	extremely	

4. **I find system to be flexible to interact with**

correct								incorrect
	extremely	quite	slightly	neither	slightly	quite	extremely	

5. **It would be easy for me to become skillful at using the system**

correct								incorrect
	extremely	quite	slightly	neither	slightly	quite	extremely	

6. **I find the system easy to use.**

correct								incorrect
	extremely	quite	slightly	neither	slightly	quite	extremely	

Appendix 8: User Satisfaction Questionnaire

User feedback questionnaire

Note: For questions 7 through 12 circle the choice that is most representative of your opinion.

7. **Do you feel the messages of the system are reliable?**

Almost never	some of the time	about half the time	most of the time	almost always

8. **Do you find the system dependable?**

Almost never	some of the time	about half the time	most of the time	almost always

9. **Do you think the messages are presented in a useful format?**

Almost never	some of the time	about half the time	most of the time	almost always

10. **Are the messages clear to understand?**

Almost never	some of the time	about half the time	most of the time	almost always

11. **Is the system user friendly?**

Almost never	some of the time	about half the time	most of the time	almost always

12. **Is the system efficient?**

Almost never	some of the time	about half the time	most of the time	almost always

Appendix 9: Practice Problem

Name: _____

Art Gallery

The Art Gallery at New South Wales in Australia is very popular for its impressive collection of artifacts from around the world. They acquire artifacts from many donars around the world. Each donor may provide many aritfacts. Upon receipt of an artifact a unique code is assigned to it. Information such as title of the artifact, its estimated age, donor's name, donor's address are to be stored in the database. The Gallery also lends its artifacts to museums. Whenever an artifact is lent to a museum, a Loan document, with a unique number, is prepared. The date of loan, return date are noted on the loan document. The address of the borrowing museum, contact person and phone number are to be stored in the system. A loan to a museum may involve many artifacts. Whenever an artifact is loaned, its condition is noted also.

Appendix 10: Pre-treatment Tasks

Name: _____

Instructional Lab Assignment

The University Computing Services (UCS) manages the assignment of instructional labs to different classes. A class is identified by the Course Number; information such as class size, Instructor's name, Instructor's phone number are also to be stored in the database. Information about labs, such as Room number, capacity and name of the main server are to be stored in the database also. A course may use many labs. But for a given semester the course uses only one lab. Information about each semester such as semester number, year, beginning and ending dates are to stored in the system also. Assume that the same instructor teaches the course every semester. Make any assumptions that deem appropriate. Draw an ER diagram for the above description.

Appendix 11: Practice Problem solution

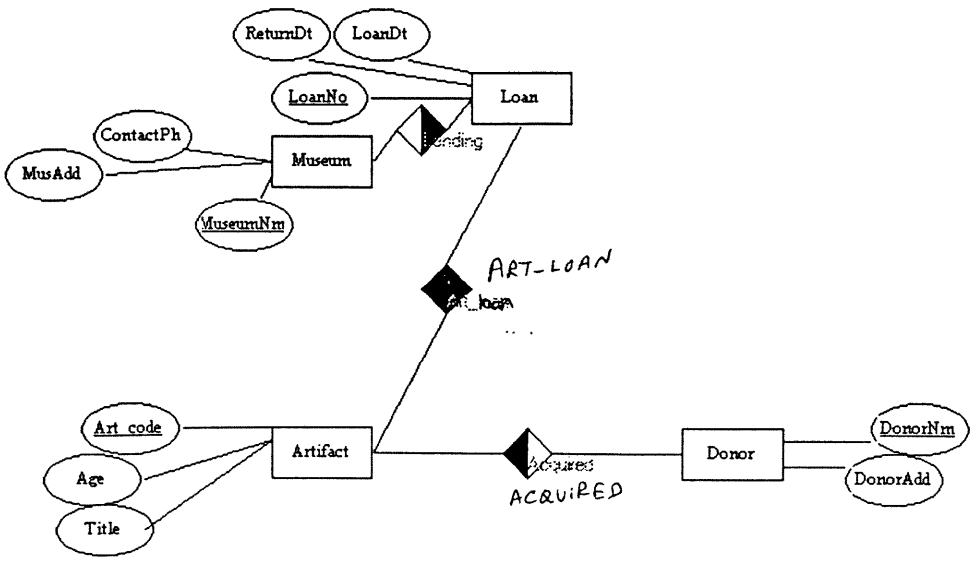
Note: A review of the problem you did in class, and its solution is given below. Read the problem over, read and understand the comments and the solution.

Art Gallery

Problem : The Art Gallery at New South Wales in Australia is very popular for its impressive collection of artifacts from around the world. They acquire artifacts from many donars around the world. Each donor may provide many aritfacts. Upon receipt of an artifact a unique code is assigned to it. Information such as title of the artifact, its estimated age, donor's name, donor's address are to be stored in the database. The Gallery also lends its artifacts to museums. Whenever an artifact is lent to a museum, a Loan document, with a unique number, is prepared. The date of loan, return date are noted on the loan document. The address of the borrowing museum, contact person and phone number are to be stored in the system. A loan to a museum may involve many artifacts.

Comments: (i) Since Donor Name and Donor address "describe" a donor, DONOR becomes and entity. (ii) Museum also becomes an entity, with Museum name as the key-attribute. (iii) Since there is no sentence suggesting a need for a ternary relationship, the solution involves only binary relationships. (iv) A loan is for a specific museum and there can be many loans towards a museum (v) An artifact may be involved in many loans (on different times, of course) and a loan may involve many artifacts.

The solution is given below.



Use the software to model the Art Gallery problem. Follow the instructions given on the next page. Once you have finished modeling the Art Gallery problem ask the instructor for the next problem.

Appendix 12: User Manual for Control system users

How-to use the software?

1. Run the Windows program
2. Click on the File option in the menu and choose Run option.
3. In the command line text box, type A:ERD.EXE and press enter
4. Once the program is running, minimize the other windows. DO NOT remove the disk when the program is running. You may remove the disk only AFTER exiting the Windows program. WHILE USING THE SYSTEM READ ALL THE MESSSAGES AND TAKE APPROPRIATE ACTION. If you wish to start over during a problem, exit the program and run it again.

Entity-Relationship modeling using the software

1. Refer to the page titled "Main menu options" to get to know what the menu options are for.

2. Choose the Attributes menu option and enter the all the attributes. Refer to the page titled "Attributes window" for an explanation of the interface.

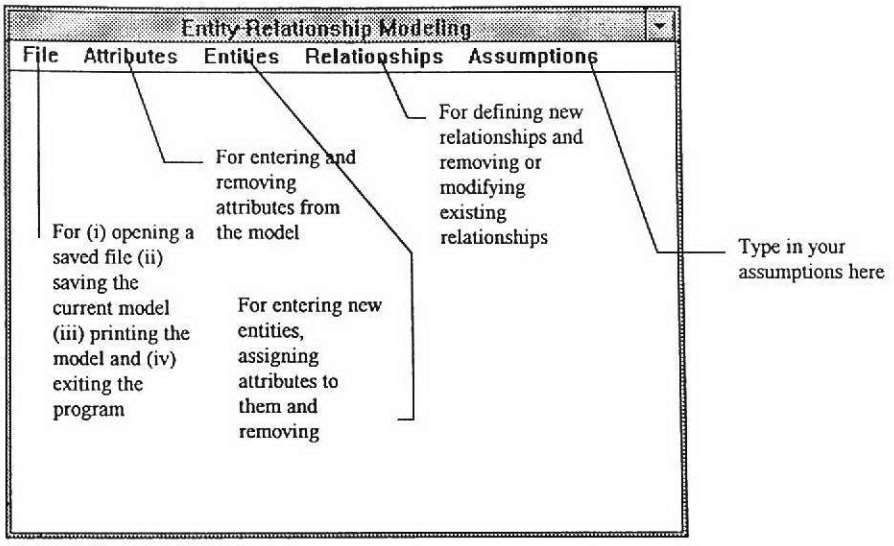
3. Choose the Entities menu option and select "Enter entity names" option and enter all the entity names. Refer to the "Entering entity names" page for an explanation of the interface.

4. Choose the Entities menu option and select "Assign attributes" option. Refer to the "Assigning attributes to entities" page for an explanation of this window features. Select one of the entities and assign the attributes to it. Remeber to specify the key-attribute to the entity. Store the entity definition. Repeat step #3 for each entity involved.

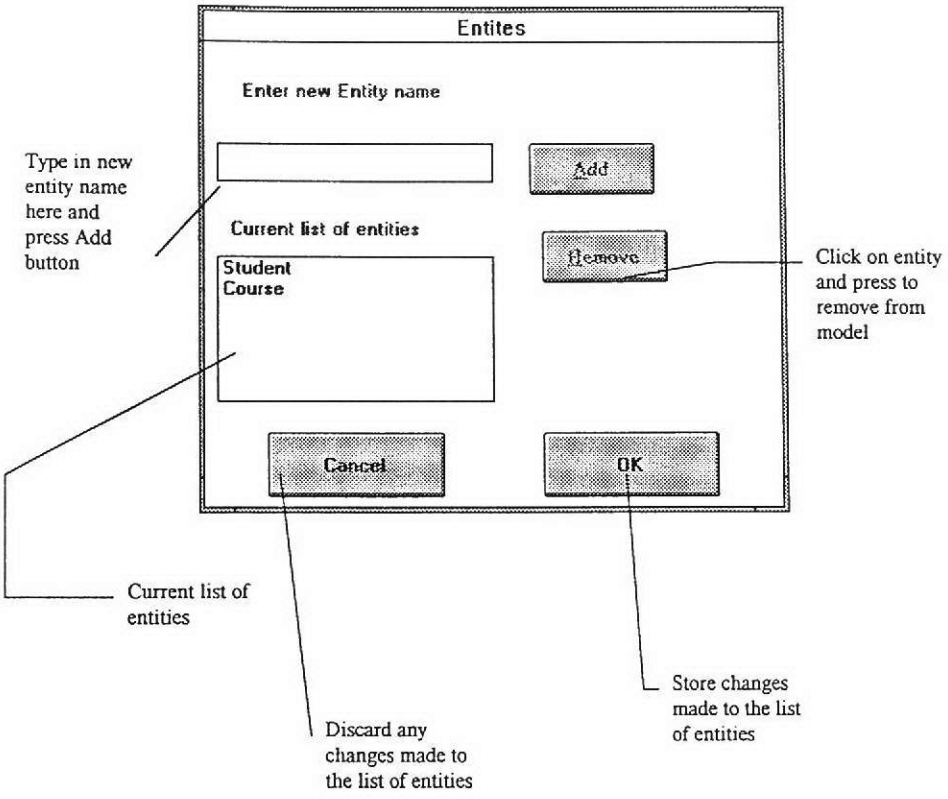
5. After all the entities have been defined, choose the Relationship menu option and select the "Define" option. Refer to the "Defining relationship page for a description of the various features on this window. Type in a name for the relationship and select appropriate entities. Specify the connectivity of each entity in the relationship and store the relationship definition. Repeat step #4 for each of the relationship.

6. Once you have defined all the relationships, save the file and exit the program.

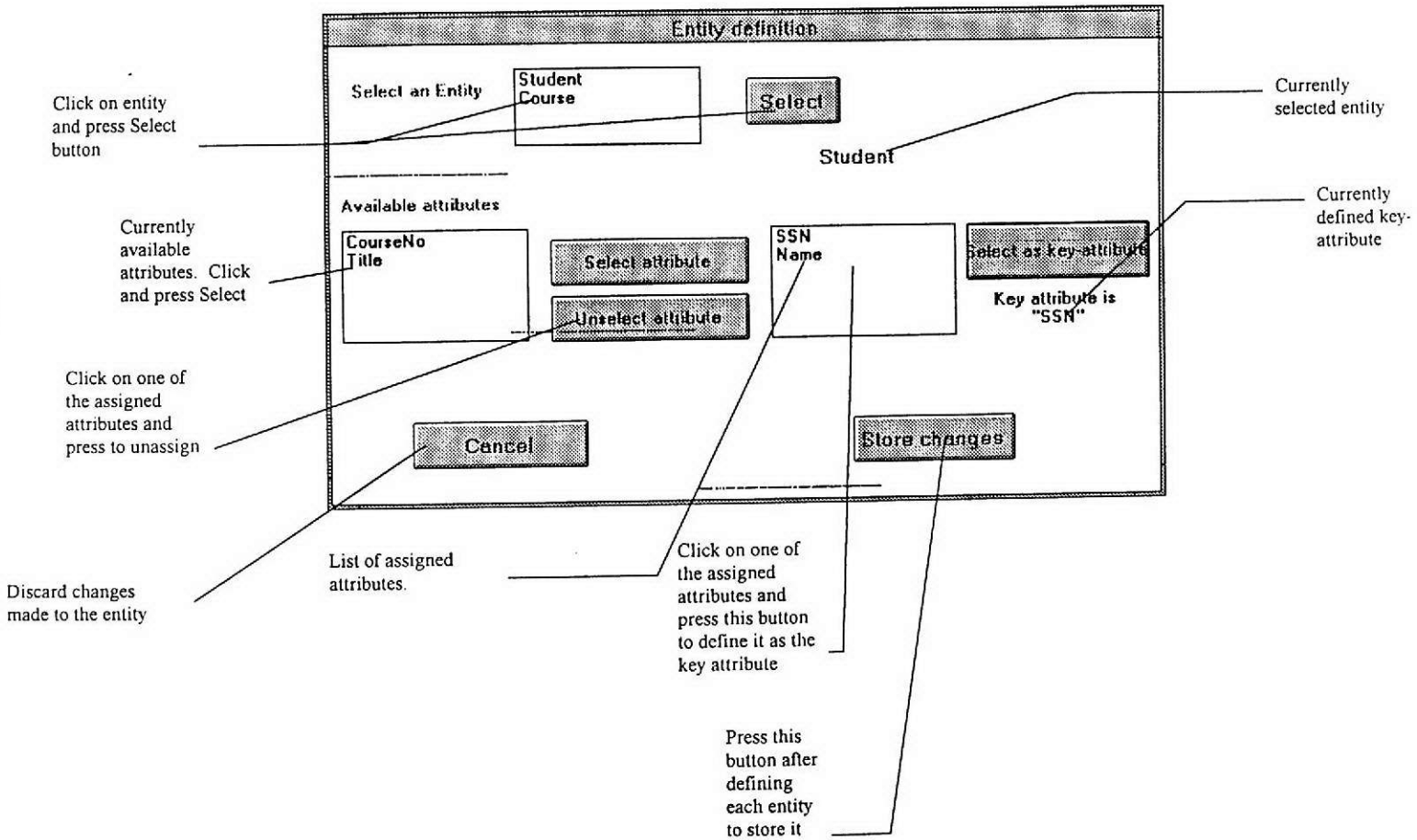
Main Menu Options



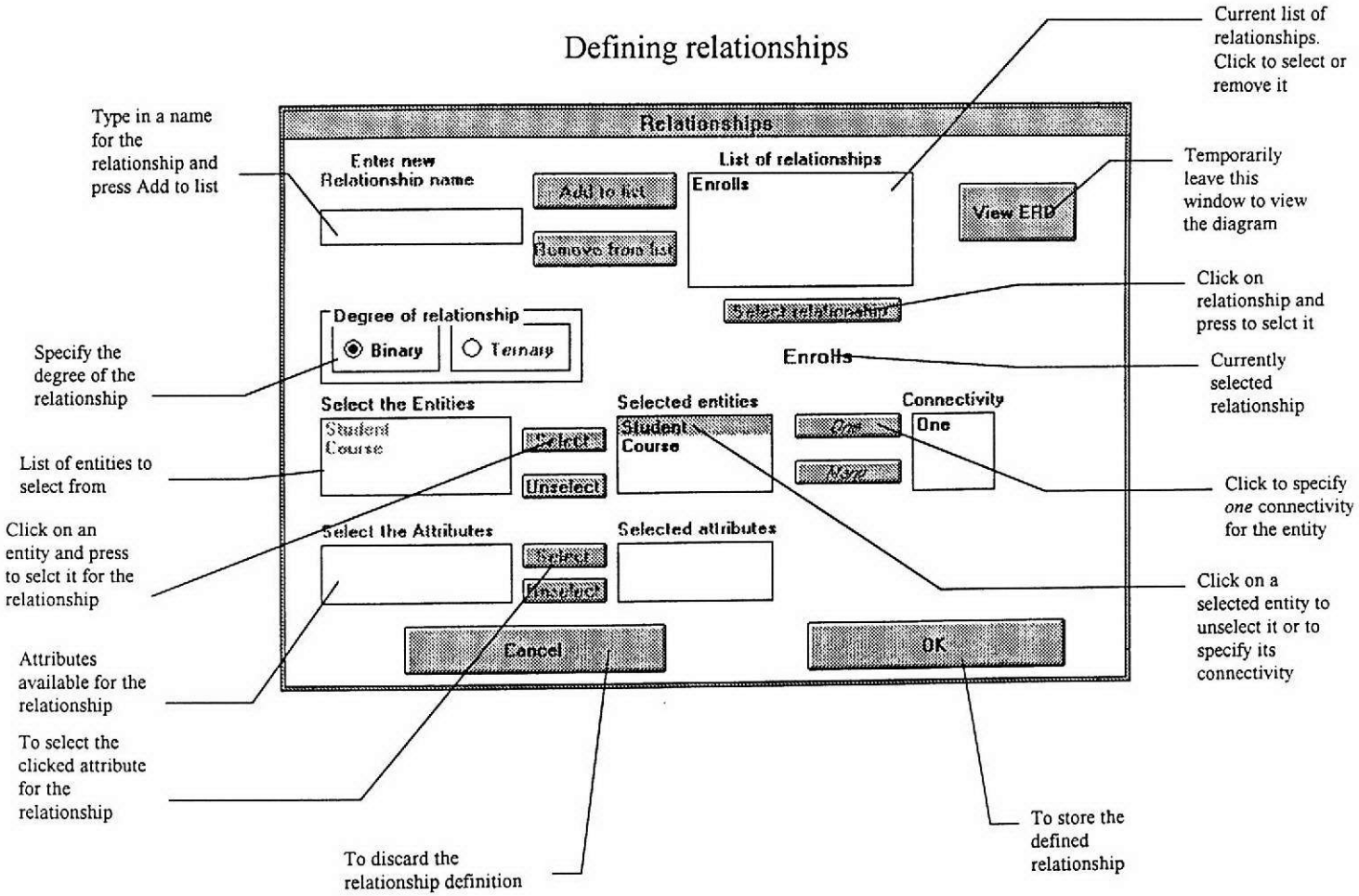
Entering entity names



Assigning attributes to entities



Defining relationships



Appendix 13: User Manual for Guidance system users

How-to use the software?

1. Run the Windows program
2. Click on the File option in the menu and choose Run option.
3. In the command line text box, type A:ERD.EXE and press enter
4. Once the program is running, minimize the other windows. DO NOT remove the disk when the program is running. You may remove the disk only AFTER exiting the Windows program. WHILE USING THE SYSTEM READ ALL THE MESSSAGES AND TAKE APPROPRIATE ACTION. If you wish to start over during a problem, exit the program and run it again.

Entity-Relationship modeling using the software

1. Refer to the page titled "Main menu options" to get to know what the menu options are for.

2. Choose the Attributes menu option and enter the all the attributes. Refer to the page titled "Attributes window" for an explanation of the interface.

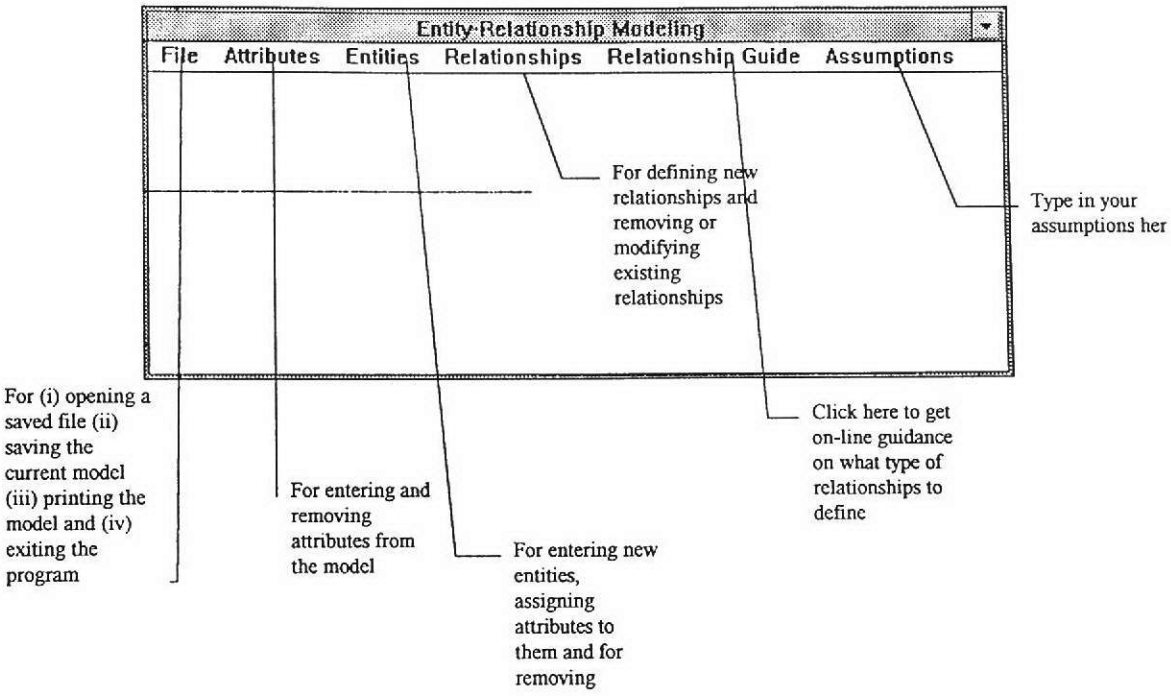
3. Choose the Entities menu option and select "Enter entity names" option and enter all the entity names. Refer to the "Entering entity names" page for an explanation of the interface.

4. Choose the Entities menu option and select "Assign attributes" option. Refer to the "Assigning attributes to entities" page for an explanation of this window features. Select one of the entities and assign the attributes to it. Remeber to specify the key-attribute to the entity. Store the entity definition. Repeat step #3 for each entity involved.

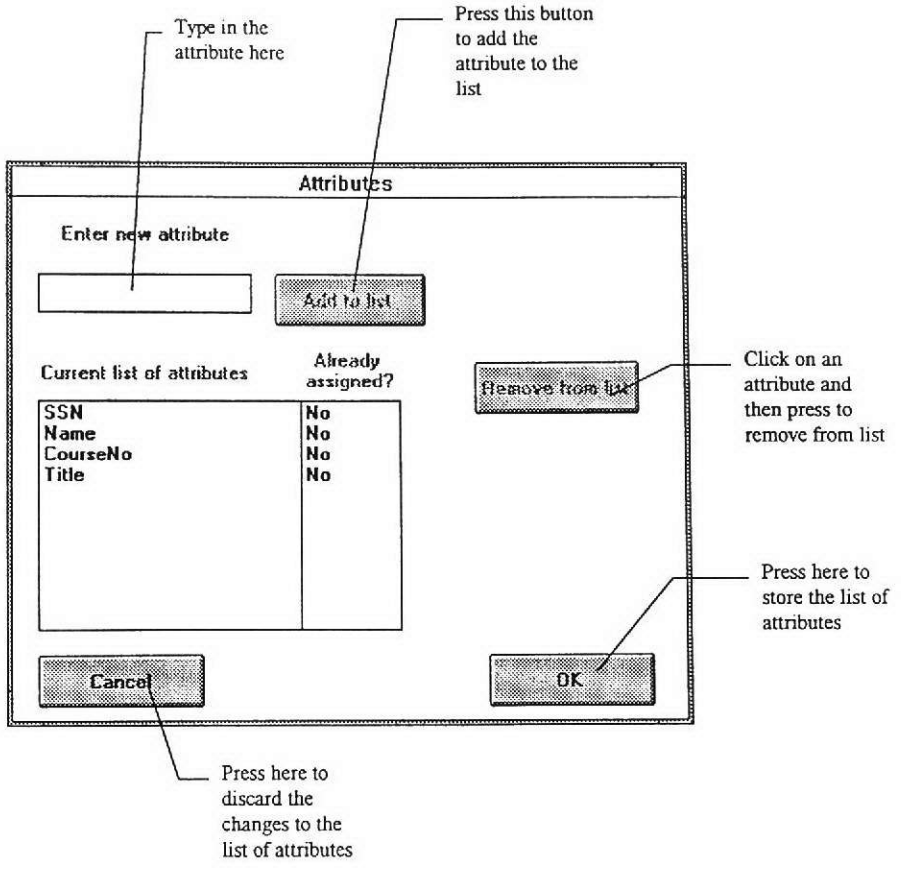
5. After all the entities have been defined, choose the Relationship menu option and select the "Define" option. Refer to the "Defining relationship page for a description of the various features on this window. Type in a name for the relationship and select appropriate entities. Specify the connectivity of each entity in the relationship and store the relationship definition. Repeat step #4 for each of the relationship.

6. Once you have defined all the relationships, save the file and exit the program.

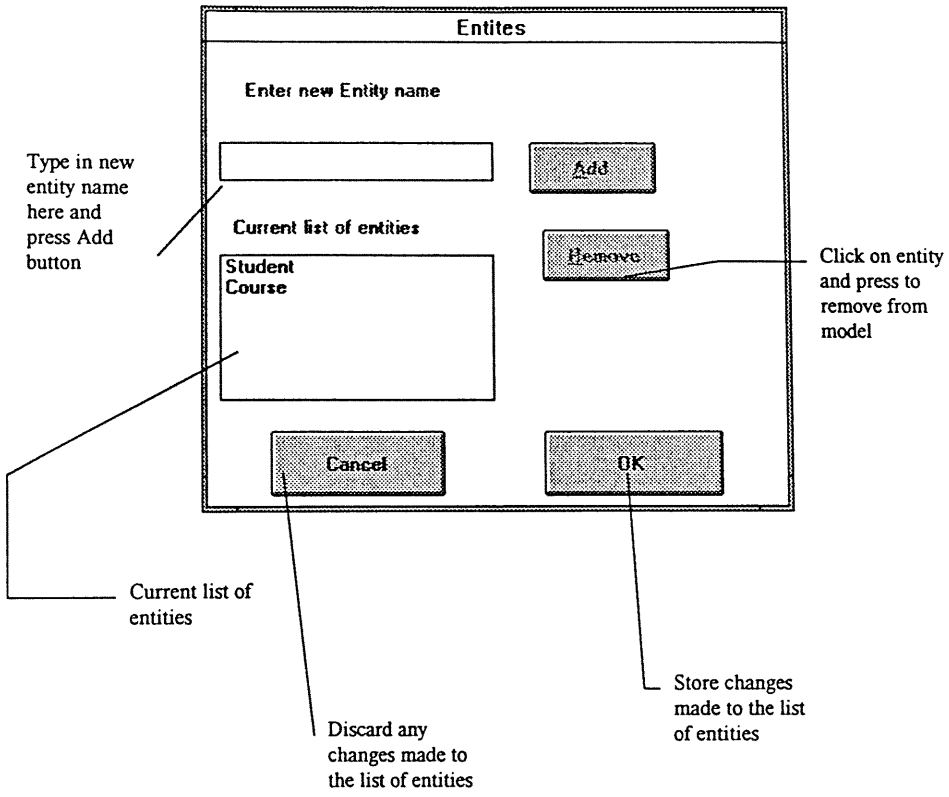
Main menu options



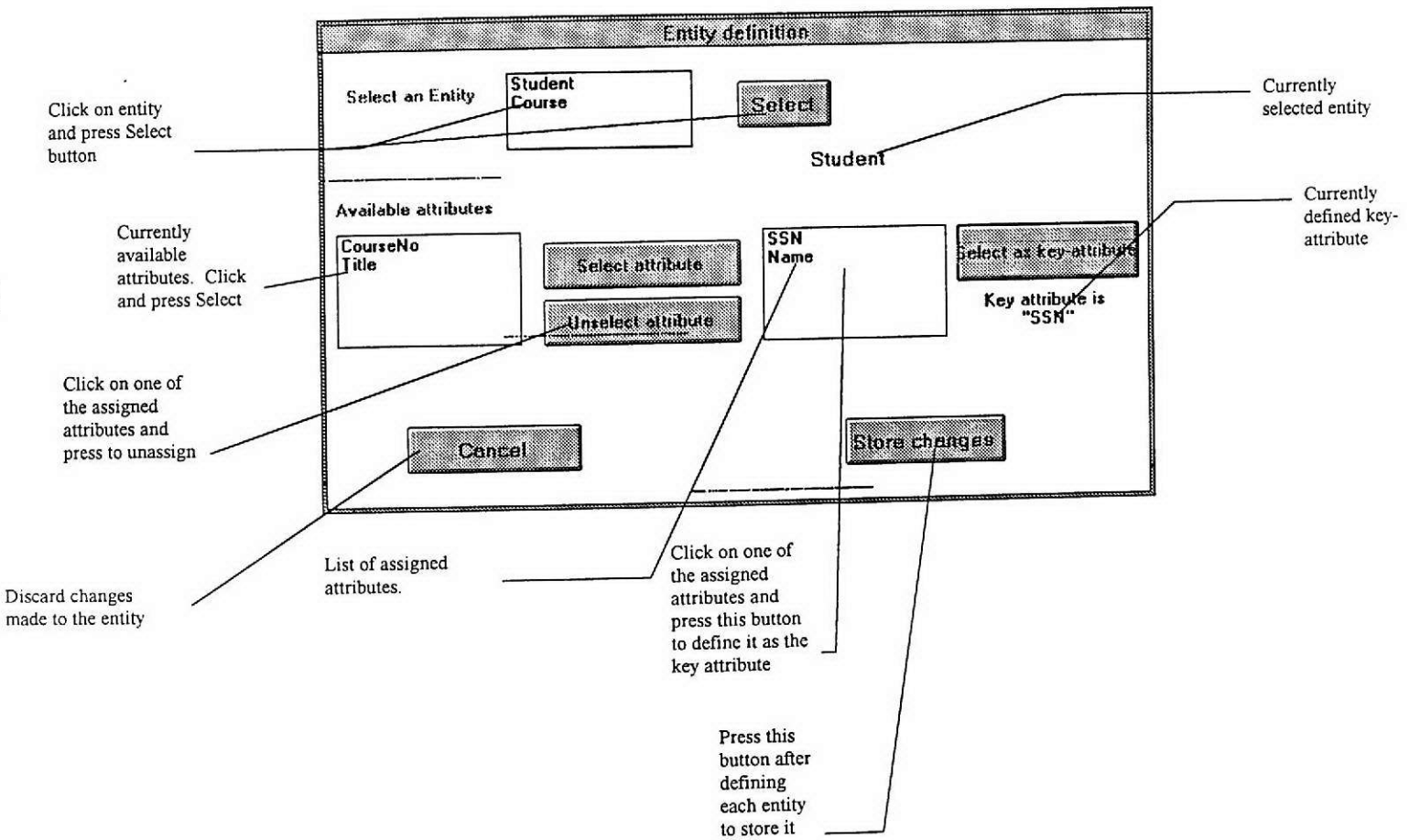
Attributes window



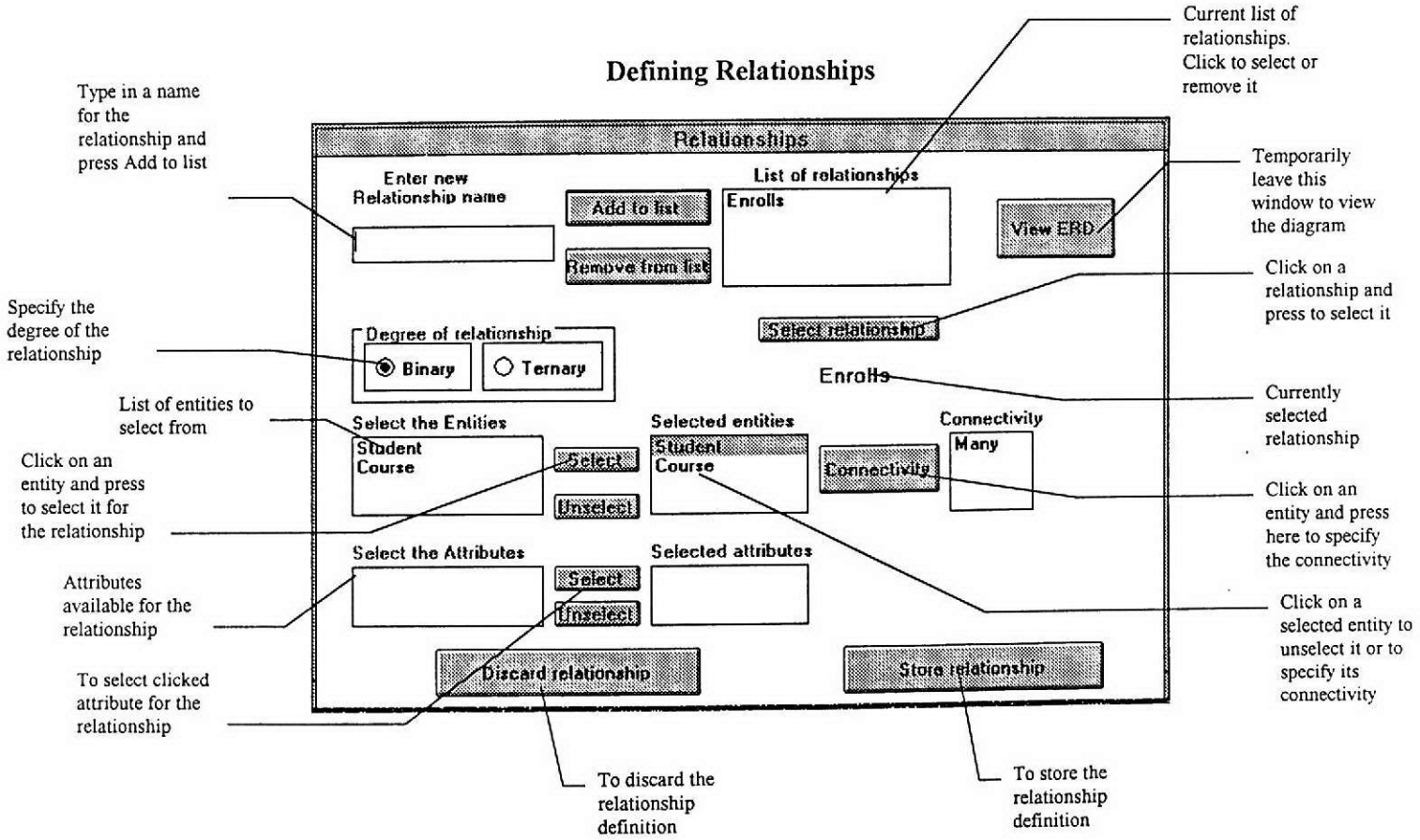
Entering entity names



Assigning attributes to entities



Defining Relationships



How-to use the software?

1. Run the Windows program
2. Click on the File option in the menu and choose Run option.
3. In the command line text box, type A:ERD.EXE and press enter
4. Once the program is running, minimize the other windows. DO NOT remove the disk when the program is running. You may remove the disk only AFTER exiting the Windows program. WHILE USING THE SYSTEM READ ALL THE MESSSAGES AND TAKE APPROPRIATE ACTION. If you wish to start over during a problem, exit the program and run it again.

Entity-Relationship modeling using the software

1. Refer to the page titled "Main menu options" to get to know what the menu options are for.

2. Choose the Attributes menu option and enter the all the attributes. Refer to the page titled "Attributes window" for an explanation of the interface.

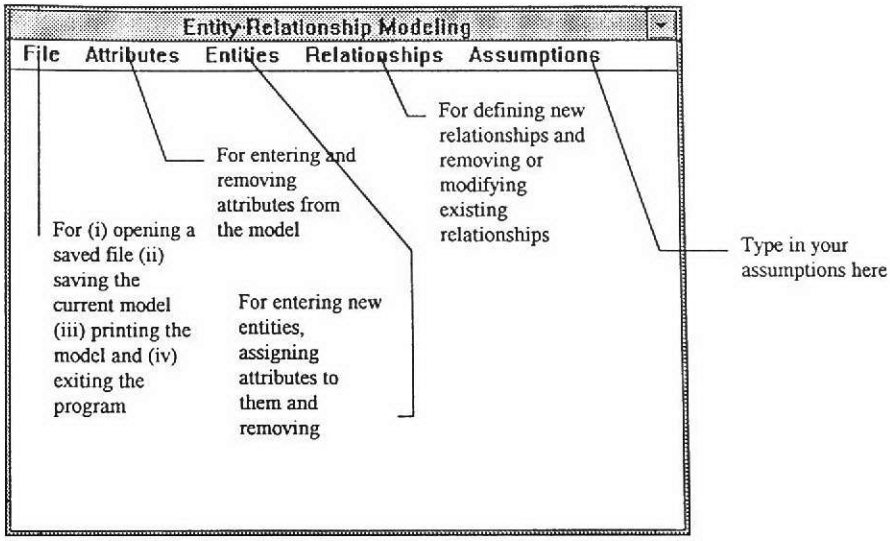
3. Choose the Entities menu option and select "Enter entity names" option and enter all the entity names. Refer to the "Entering entity names" page for an explanation of the interface.

4. Choose the Entities menu option and select "Assign attributes" option. Refer to the "Assigning attributes to entities" page for an explanation of this window features. Select one of the entities and assign the attributes to it. Remeber to specify the key-attribute to the entity. Store the entity definition. Repeat step #3 for each entity involved.

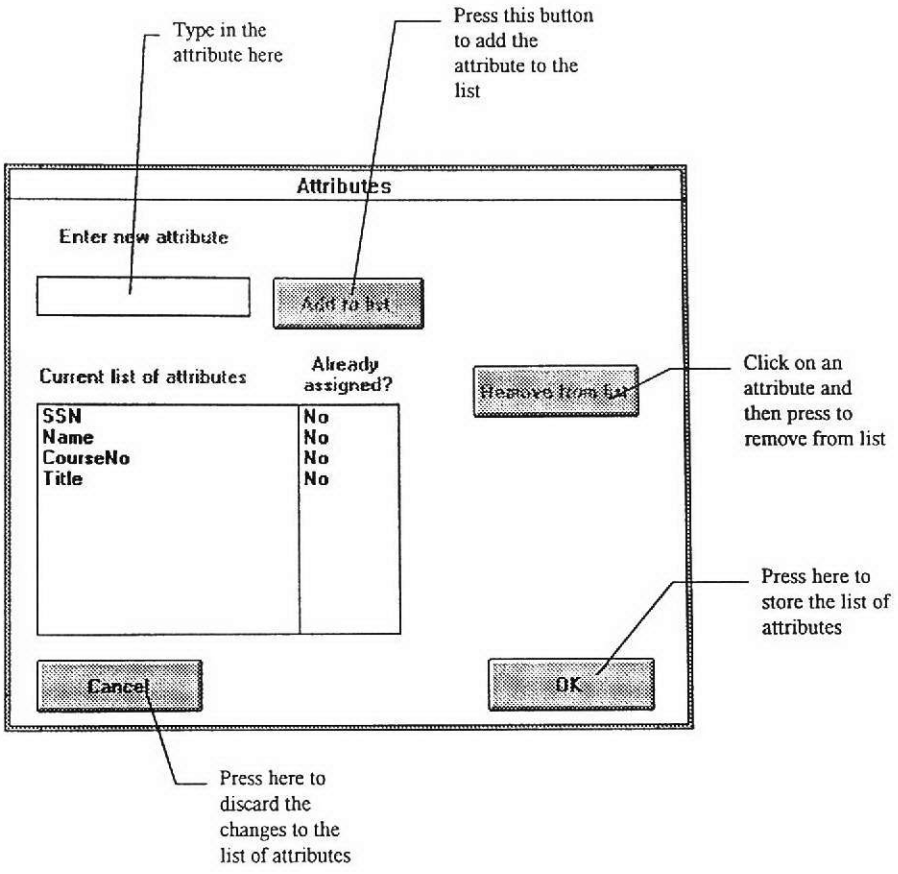
5. After all the entities have been defined, choose the Relationship menu option. The system expects you to do the relationships in a certain sequence. The sequence is : binary *one-many*, ternary *one-many-many* or *one-many-many*, then ternary *many-many-many* and binary *many-many*. Read the messages and take appropriate action. While following the sequence, if you wish to remove a relationship, you can remove the relationship that was defined last. Refer to the "Defining relationship page for a description of the various features on this window. Type in a name for the relationship and select appropriate entities. Specify the connectivity of each entity in the relationship and store the relationship definition. Repeat step #4 for each of the relationship.

6. Once you have defined all the relationships, save the file and exit the program.

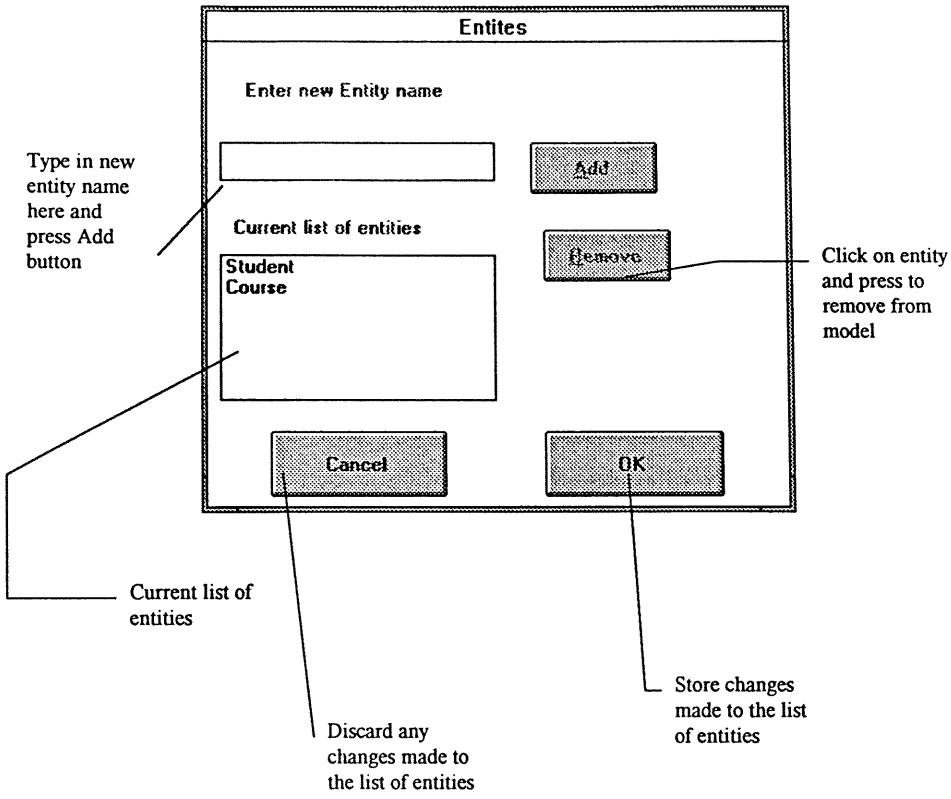
Main Menu Options



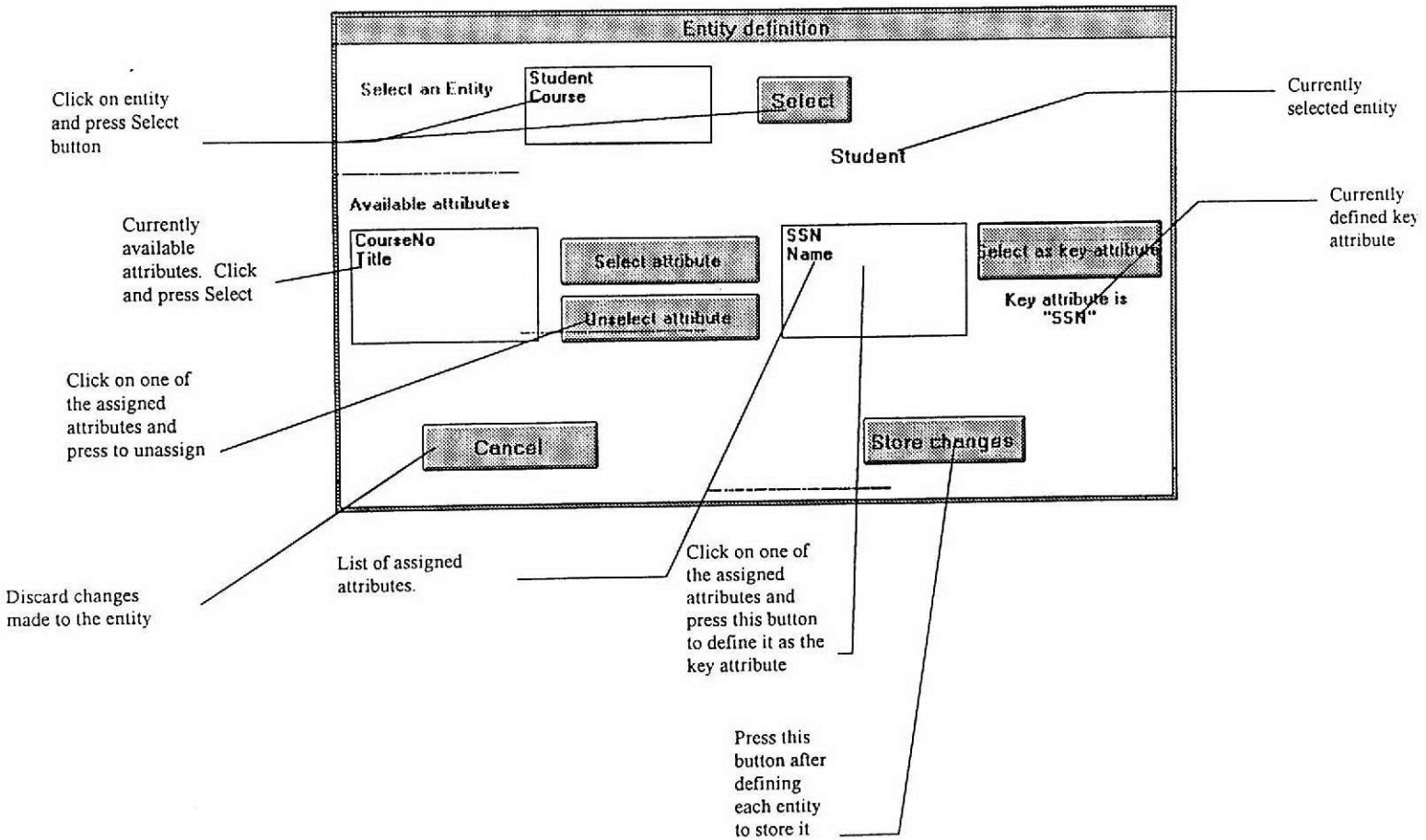
Attributes window



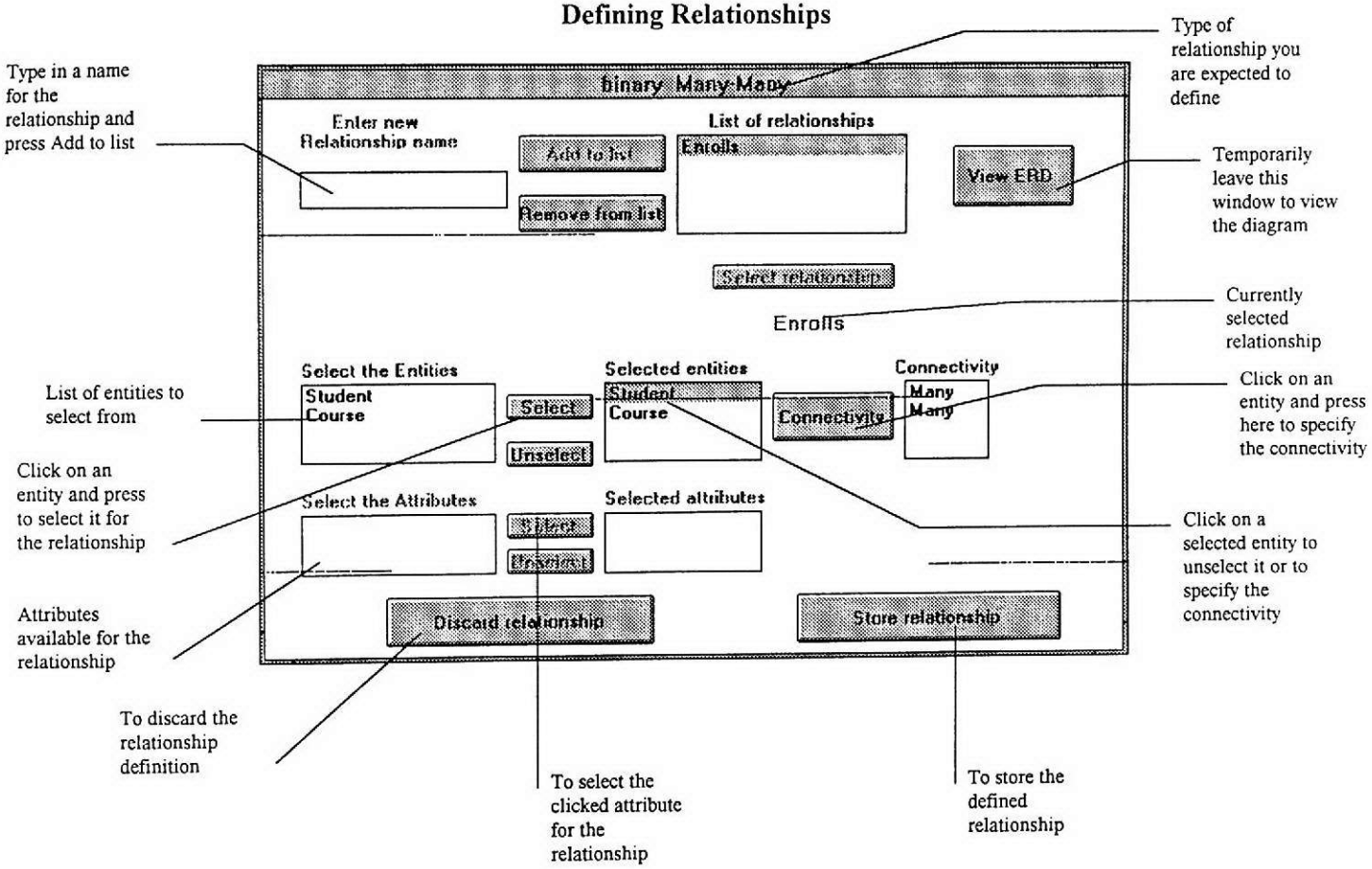
Entering entity names



Assigning attributes to entities



Defining Relationships



VITA

Solomon Raj Antony
960 Oakwood Drive #141
Rochester MI 48307

Date of birth: May 30, 1961
Place of Birth: Sivakasi, India

EDUCATION

Bachelor of Engineering (Hons.) **1978-83**
BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE PILANI, INDIA

Specialization: Chemical Engineering

WORK EXPERIENCE

1983-85 Research Associate, Center for Development Studies, India;
1986 Research Associate, National Council for Applied Economic Research, India;
1987-91 Research Associate and Systems Analyst, Indian Council for Research on International Economic Relations, India.
1996 Aug - Present : As a Visiting Instructor at the Decision and Information Sciences Department, School of Business Administration, Oakland University. Courses: Database Management (Graduate and Undergraduate level), Introduction to Information Systems and Decision Support Systems.

PUBLICATIONS AND PRESENTATIONS

1. Solomon R Antony (1996). Design support system for Entity Relationship Modeling. Accepted for presentation at *ICASE*96*, Crete

2. Solomon R Antony and C Koulamas (1996). Simulated Annealing applied to the total tardiness problem. *Control and Cybernetics*, **25**, 121-130.

3. D Batra and S R Antony (1994). Novice errors in conceptual database design. *European Journal of Information Systems*, **3**, 57-69.

4. Dinesh Batra and Solomon R Antony (1994). Effects of data model and task characteristics on designer performance. *International Journal of Human-Computer Studies*, **41**, 481-508.

5. C Koulamas, S R Antony and R Jaen (1994). A survey of simulated annealing applications to operations research problems. *Omega*, **22**, 41-56.

6. Radhika Santhanam, Sungwan Hong and Solomon Antony (1996). The Reexamination of Information System Project Selection Bias. Proceedings of the *1996 Annual Meeting of the Decision Sciences Institute*, Orlando, Florida

7. Steve Zanakis, Solomon R Antony, Sandipa Dublish and Nicole Wishart (forthcoming). Multi-criteria decision making: A comparison of select methods. *European Journal of Operational Research.*