A METHODOLOGY AND SUPPORTING FRAMEWORK FOR FUNCTIONAL MODELING AND CONFIGURATION IN CONCEPTUAL DESIGN

by

Janis Pinchefsky Terpenny

Dissertation submitted to the Faculty of the

Virginia Polytechnic Institute and State University

in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Industrial and Systems Engineering

APPROVED:

Dr. Paul Torgersen, Co-chair

Dr. Jan Helge Bøhn, Co-chair

Dr. Bartholomew Nnaji

Dr. Arvid Myklebust

Dr. Subhash Sarin

Dr. Wolter Fabrycky

December 1996
Blacksburg, Virginia

Key words: Conceptual Design, Concurrent Engineering, Configuration

A METHODOLOGY AND SUPPORTING FRAMEWORK FOR FUNCTIONAL MODELING AND CONFIGURATION IN CONCEPTUAL DESIGN

by

Janis Pinchefsky Terpenny

Chairmen: Paul E. Torgersen and Jan Helge Bøhn

Industrial and Systems Engineering

(ABSTRACT)

Concurrent Engineering has brought much attention in recent years to engineering design and its impact on issues such as costs, cycle-time, quality, and other life-cycle processes. Coupled with global markets and the rapid rate of technology advancements, the need for improved methods and supporting tools for engineering design is significant. To date, advances for engineering design have predominantly focused on tasks that are well into the latter stages of product development. Advances for early design (where over 60% of life-cycle costs are committed) still remain largely investigational, specialized, and rarely consider the requirements for functional abstraction *and* detail necessary in a concurrently engineered development process. In general, methodologies have taken either a top-down or a bottom-up approach to design, and as such, have virtually guaranteed the continued separation of abstraction and detail during conceptualization.

This research proposes a methodology based on the blending of top-down and bottom-up approaches. Toward this end, a framework for integrated conceptual design is presented. Three mechanisms are central to the framework definition, including: 1) a functional

modeling environment supporting concept model building, knowledge capture, and reuse, 2) a components knowledge-base supporting configuration, and 3) an integrated design domain accessible from the functional modeling environment interconnecting tools, analysis routines, and data sources necessary for design synthesis, analysis, and evaluation.

Based upon an object-oriented paradigm and semantic reasoning, the framework for functional modeling and configuration has been designed in detail. A Windows-based user interface has been prototyped and enables designers to both visualize and compose conceptual models using a building-block approach. An example design problem in the domain of power conversion systems is provided and demonstrates the methodology for technologically sophisticated products where conceptualization crosses electrical, mechanical, and software domains (mechatronics). A use case model and object class diagrams describe and document the framework function and architecture.

Dedication

In memory of Jason and in honor of Jonathan

... *my sons*

... *my inspiration*

Reminding me that life is a fragile gift.

Inspiring me to bring love, laughter, courage, and passion to all of life's challenges.

ACKNOWLEDGEMENTS

I offer my sincere appreciation and gratitude to numerous individuals and organizations in the successful completion of my doctoral studies. First, I would like to thank the members of my committee for their guidance and support. To Dr. Myklebust, Dr. Fabrycky, and Dr. Sarin, I am most grateful for their probing questions and validation of the worthiness of my research. Thank you, Dr. Bøhn, for the deadlines, attention to administrative concerns, and your edits of the document. Although no longer a member of my committee following his departure from Virginia Tech, thanks are also due to Dr. Eyada for his challenge and support. I would also like to offer my gratitude to Dr. Deisenroth who provided me with guidance and professional opportunities early in my doctoral program. Thank you for your continued encouragement. Words fail me as I try to express the thanks and gratitude that are due Dr. Nnaji and Dr. Torgersen in the completion of my doctoral work. Dr. Nnaji, thank you for the unwavering confidence that you always bestowed upon me, your professional expertise, and your guidance. Dr. Torgersen, thank you for your confidence in my abilities and willingness to support me through very rough periods. I could ask for no better role model. You are an inspiration in your noble pursuit of excellence as a professional and in the care that you offer to every individual. Thank you for the opportunity.

I would also like to express my appreciation to those who have so generously funded my dissertation work. Thank you to Dr. Vorster, Dr. Watford, and the College of Engineering at Virginia Tech for the support that was provided through the Charles Minor Fellowship. In addition, thanks are due to Dr. Casali, Dr. Dryden, and the Department of Industrial and Systems Engineering at Virginia Tech for the funding and opportunity to

serve as a graduate teaching assistant and then as an instructor. I am also most grateful to General Electric Motors and Industrial Systems (GEMIS) in Salem, Virginia for not only providing funding for this research, but also for the work space (facilities, software, equipment, desk), and expertise of numerous engineers. In particular, thank you to the following individuals (either now or previously) with General Electric who provided resources and/or their time to this research: Tom Brock, Russ Shade, Scott Donnelly, Brian Smith, Andy Stevenson, Ross Edmunds, Al Scalera, Bruce Wittmeier, Bill Bowman, Dave Black, Paul Espelage, Bobby Compton, Bill O'Brien, Bill Giewont, Bob Walton, Jay Gordon, and Marshall Aurnou. Thanks are also due to Virginia's Center for Innovative Technology (CIT) who co-sponsored this research along with the funding provided by General Electric.

As with all graduate students, my dissertation studies and research have been both a happy time and a trying time. For me, these years have been blessed with new found friendships that will likely last a lifetime and opportunities for self-discovery and growth. Unfortunately, these have also been very sad times and a time of crisis with the loss of my eldest son Jason following a courageous battle with Leukemia. For you Jason and your brother Jonathan I dedicate this work. I love you dearly.

I feel sure that I will omit the name of someone in the list that follows. Please forgive me for those unintentional oversights. There have been so many people who have supported me through this process. I am inspired by and thankful to the following friends and colleagues: Alice McCaffrey, Dr. Jin Ong, Sherry Heron, Lovedia Cole, Leslie Graham, Patty Cox, Sherry Hollar, Judy Howell, Sandra and Homer Duff, Ann Fox, Nei Mueller, Joni Quesenberry, Steve Strong, Dr. Simon Hurley, Richard Turner, Dianne Henry-

Leggette, Bill Rossberg, Lori Caudill, Dr. Suvrajeet Sen, Dr. Bob Davis, Dr. Marilyn Jones, Dr. John Burns, Dr. Bill Sullivan, Dr. Garry Coleman, Barbara Mabe, Joy Taylor, friends at the University of Virginia Children's Medical Center, friends at the University of Iowa Pediatric Bone Marrow Transplant Unit, friends at UUCR, friends at General Electric, and the many supportive faculty and staff in the Department of Industrial and Systems Engineering at Virginia Tech.

Finally, but by no means lastly, I acknowledge the many family members whose support throughout this period has been tremendous. I am especially grateful to my sister, Marci Rochkind, for her friendship and to my parents, Herb and Betty Pinchefsky, for their encouragement. For my husband Wally and my son Jonathan, 'thanks' is a small word when expressing the appreciation due for your patience, tolerance, and support. It has been hard on all of us. My heart felt thanks and love.

TABLE OF CONTENTS

List of Figures

List of Tables

1. INTRODUCTION

The primary purpose of this research is to develop a framework for functional modeling and configuration in engineering design. It provides the foundation for an integrated design environment capable of improving the efficiency and effectiveness of conceptual design processes. Specifically, the framework for functional configuration supports a design methodology based upon the blending of top-down and bottom-up approaches to engineering design. A brief contrast of the two approaches points to the need for this blended approach.

In the top-down approach, characteristic of a systems engineering process, design is driven from functional requirements toward solution alternatives. While design solutions using this approach are likely to meet functional requirements, there is no guarantee that solutions are realizable in terms of physical manifestations. In contrast, using the bottom-up approach, characteristic of traditional engineering design practices, designs are built from known components (embodiments) in anticipation of satisfying functional requirements. In this scenario, physical realizability is guaranteed, but there is no immediate assurance that functional requirements are met. Clearly, there are merits and limitations inherent in either approach.

The premise of this research is that a blended approach is required to provide versatility to early design processes to overcome the shortcomings typified by the exclusive practice of either the top-down or bottom-up approach. The blended approach caters to both the need for abstraction during conceptualization and the requirements for detail for informed decision processes. This approach not only aids the designer in functional modeling and

synthesis of new product concepts, but should result in reduced cycle times, lower costs, and support other life-cycle processes.

The domain of power conversion systems, comprised of electrical, mechanical, and software components, is used to demonstrate the utility of the framework for the design of technologically sophisticated products. Three mechanisms are utilized in defining the framework to meet these objectives:

- a ***functional modeling environment*** supporting concept model building, knowledge capture, and reuse; capturing design intent, requirements, specifications, and dependency relationships,

- a ***components knowledge-base*** supporting conceptualization and configuration from the functional model description, and

- an ***integrated design domain*** accessible from the functional modeling environment allowing for the interconnection of tools, analysis routines, and data sources necessary for design synthesis, analysis, and evaluation.

The remaining sections of this chapter further introduce and outline the research presented in this dissertation. Motivation is provided first. Included in this discussion is the need for improved methods and supporting tools for engineering design. In addition, justification is offered for the focus on power conversion systems in examples and prototypes. Next, the significance of the problem is considered from a broader problem context. The problem statement is then provided and includes the objective as well as the research questions addressed in this work. Finally, the research approach is presented, followed by scope and the outline for the remaining chapters of the dissertation.

1.1 Motivation

In recent years, technology advancements have had dramatic effects on both products and processes. For manufacturers in high technology markets in particular, information, automation and frequent change have become expected. Advances have caused not only the rapid evolution of technically sophisticated products, but have allowed for their rapid obsolescence as well. Coupled with global markets, these advances have generated intensely competitive environments. Customers now expect and demand products which are on the leading edge of innovation, flexible to a variety of needs, reliable, serviceable, minimize their costs, and are backed by commitments to customer satisfaction. To remain viable, manufacturers must respond to these requirements, minimize costs for competitive pricing, and reduce time to market for new products. For these reasons, product and process improvements are currently receiving substantial research attention. Within engineering design in particular, the need for improved methods and supporting tools to meet these competitive challenges is significant.

1.1.1 Why Engineering Design?

Today, it has become apparent that the impacts, and therefore responsibilities, of designers go well beyond designing technical systems to satisfy customer needs. Designers are challenged to complete their tasks quicker, at reduced costs, and push the edge of technology with innovation. In addition, designers are challenged to actively incorporate the objectives of company strategies in their decision processes. Consideration must be given to costs and cycle time of other life-cycle processes. Numerous examples now document the direct impact of product design decisions on issues such as manufacturability, maintainability, reliability, testability, disposability, and environmental concerns [Banc92], [Dice93], [Hern91], [Kapu94], [OGra91a]. These examples show

that quality cannot be inspected into a product, nor can manufacturing or other downstream processes make up for poorly developed products.

Researchers who have sought to quantify the impacts of design decisions estimate that by the completion of product design stages, while only 8% of total life-cycle costs have been incurred, 80% of total life-cycle costs have been committed [NRC91]. Moreover, estimates indicate that these impacts occur very early in the product development cycle with 60% to 75% of total life-cycle costs committed by the completion of conceptual design [Nevi89]. Clearly, there is significant motivation to justify research which explores methods and tools for improved design processes; processes which reduce costs, minimize time to market, increase product quality, and enable the creation of innovative solutions to customer needs.

1.1.2 Why Power Conversion?

Within the domain of technically sophisticated products, such as power conversion products, the need to improve and support engineering design processes is particularly important. Containing electrical, mechanical, as well as software components, these products require that designers from across engineering disciplines and specialties work together. Not only are designers faced with life-cycle, cost, time, and quality issues, but they must coordinate their efforts as they configure complex functional system designs. Often approachable from a variety of engineering perspectives, configuring and evaluating *feasible* solutions, much less competitively *optimized* concepts, can be quite difficult. Designers must consider AC to DC and DC to AC conversions, control, protection and safety issues across electrical, mechanical and thermal domains, environmental operating

conditions, product packaging, integration with existing customer systems, and requirements which are often ill defined and/or changing.

With widespread use in the control of industrial and commercial processes, motors, and the generation of electric power, the demand for power conversion products is substantial. Thus the motivation for research into enabling technologies to improve design processes for these products is significant. The framework of this research provides the foundation to define a rich vocabulary of functional objects for conceptual modeling, synthesis, and evaluation. In addition to providing the vehicle for demonstration and proof of concept of this research, a prototype based on a power conversion system also provides insight into the specific functional objects and principles for this application domain.

1.2 Significance of the Problem

Researchers have spent many years studying design theory and methodology seeking to understand and capture the *how* and *why* of experienced designers. With some variation in defining the exact stages of design, most agree that design processes can be characterized as hierarchical, with synthesis and transformations from abstraction to finalized specification known to be very iterative and information intense. Typical process definitions have included stages such as conceptualization, configuration, embodiment, and detailed design.

For many companies, this separation of design into distinct stages has been implemented in organizational structures where engineering disciplines, associated specialties, and other business areas are both physically and functionally grouped. This has led to design practices which are largely sequential, involving several hand-offs as well as transferals of

responsibility. As a result, the iterative processes of design have become difficult, costly, fraught with significant delays, and have suffered from poorly coordinated objectives. These practices are a frustration for designers and counter to remaining competitive in today's marketplace.

In reality, the definitive steps of sequential stages, though iterative, are contrary to natural human reasoning processes which require *frequent* movement between abstraction and detail. Conceptual designers naturally need to alternate between steps of reasoning in which detail is initially ignored to focus on high level functional abstractions, and steps in which the addition of detail is necessary for evaluation and design representation [PazS89], [Szyk92]. The separation of design tasks along organizational boundaries combined with the pressures to reduce cycle times, has led to processes that are very specialized. For technologically sophisticated products in particular, this can mean the parallel design efforts of electrical, mechanical, and software engineers. Often, with little or assumed information, decisions are often made using "good faith" intentions only to require major revisions when functional areas do converge for design reviews. Rather than resulting in reduced cycle times, these practices have increased costs and required extended deadlines for product development.

As confirmed by estimates indicating that 60-75% of costs are committed by the completion of conceptual design, there is a direct and measurable relationship between abstraction and detail. Abstractions necessarily narrow the means available to configure functional solutions. It becomes evident that improved efficiencies (cycle time and costs) and effectiveness (innovation and quality), simply cannot occur without sufficient detail for informed decisions in early design.

As evident in the volume of research literature, there is much research interest in developing enabling technologies for improved product design processes. Though motivated by market pressures, much of this interest is attributable to the advances in computer hardware and software technologies over the last decade. Advances such as computer networks, powerful and inexpensive personal computers (PCs), workstations, artificial intelligence, and object-oriented software and database technologies have made profound changes to both the availability of information and the ability to develop advanced software tools. Feature and parametric based techniques, advanced engineering analysis tools, and application developments for design automation are now the focus of much research.

While showing great promise, tools have demonstrated the feasibility of automating particular tasks, but have not necessarily led to an increased understanding or improvement to *overall* product realization processes [Balk93], [Keir90]. Like so many automation efforts in other parts of a business, tools for the automation of existing tasks without investigation of underlying issues and process problems can consume a lot of resource and produced little, if any, gain. Further, advances have most often been for specific tasks which are well into the latter stages of product development. Developments for early design processes still remain largely investigational, specialized, and rarely consider the interconnectedness of abstraction and requirements for detail necessary in a concurrently engineered development process. Much work remains to develop enabling technologies capable of supporting the concurrency and integration requirements so crucial to improved design processes for the early stages of conceptual design.

1.3 Problem Statement

The inability of previous research to adequately address the early stages of conceptual design becomes evident when one considers the apparent dichotomy in current methodologies. Largely, methodologies have approached design either from a top-down or a bottom-up approach, and as such, have virtually guaranteed the continued separation of abstraction and detail during conceptualization.

The top-down approach, as it implies, begins with a top-level view of the design problem. Ideally, based upon a requirements specification and functional characterization, design proceeds iteratively through successive levels of refinement toward the ultimate goal of identifying feasible alternatives. Inherently, this approach satisfies the functional requirements of the design problem. Unfortunately however, there is no assurance that the process will lead to realizable solutions (i.e., real-world embodiments). Frustrated with the time consuming and perhaps fruitless efforts of this approach, designers frequently abandon the effort seeking solutions in the combinatoric domain of detail.

The bottom-up approach to design is based on the selection and configuration of solutions from known components. This approach relies heavily on individual expertise and ingenuity. While realizable solutions are implicit, embodiments which meet functional requirements are not immediately assured. The process is known to be highly iterative. In the case of large or complex designs, the problem quickly leads to combinatorial explosion, inefficiency in the design process, and difficulties assessing the effectiveness and merits of design alternatives.

The objective of this research is to reduce the separation of abstraction and detail in the early stages of engineering design. Specifically, a framework that supports a blended approach to design, drawing upon the strengths of both top-down and bottom-up approaches, is proposed. Research questions addressed in this work include:

- How can top-down and bottom-up approaches be blended?
- Will the blended approach improve the quality of design?
- Will the blended approach reduce human resistance to the rigor of top-down design?
- Will the blended approach reduce the combinatorial explosion that is often present in bottom-up design?

1.4 Research Approach

The approach of this research centers around the development of a framework for integrated conceptual design processes. Such a framework will provide the foundation for the proposed blended approach to engineering design; catering to the diverse needs for abstraction and detail in early design processes. As mentioned previously, three mechanisms are key to defining this integrated methodology. A summary of each of these primary framework components follows.

1.4.1 Functional Modeling Environment

An object-oriented architecture provides the foundation for knowledge capture as well as functional modeling. Functional knowledge can be defined and classified in terms of function objects. These objects are reflective of purpose (design intent) and can be lower level or higher level (decomposable) functions. A building-block approach to modeling

makes use of these functional objects. This approach permits modeling at a variety of levels of abstraction, promotes reuse and flexibility, helps speed the search for feasible alternatives, adds consistency to design practices, and provides a rich model for solution synthesis.

Modeling systems based on the proposed framework can accommodate the visual placement, manipulation, and interconnection of functional objects. Relevant design parameters and requirements specifications can be captured graphically or in readily accessible tables. In particular, a prototype example for power conversion systems has been created using the proposed framework. It includes functional objects that are indicative of those natural to these products. Both higher level and lower level functional objects have been defined.

1.4.2 Component Knowledge-Base

A prototype knowledge-base has been constructed to demonstrate the ease of design configuration from a functional model description. Recurrent engineering calculations that are used in sizing and selection are captured in the knowledge-base and then utilized in the configuration process. Component selections are *not* explicitly tied to functions, but emerge with the satisfaction of requirements. Object-oriented techniques permit variety in the classifications of components; adding versatility to their use for a wide range of functional applications. In this manner, the framework provides an adaptable environment to support new developments and uses of components in engineering functional product solutions and assists the form follows function configuration process.

1.4.3 Integrated Design Domain

The functional modeling environment serves as a foundation for the interconnection of design tools, analysis routines, and sources of information relevant to recording and making decisions in design. This interconnection creates a focal point for designers as they propose, evaluate, communicate, and iterate toward feasible solutions. In addition to reducing the valuable time spent in search of sources of input to design, this architecture could potentially serve as an integrated problem solving domain whereby dependencies between tools and data are captured and used for immediate feedback. Knowledge of the dependency relationships could alert designers of the affected modules and/or re-optimization requirements resultant from their decisions. A potential scenario depicting the variety of knowledge, information, and analysis sources supportive of the product conceptual design process is shown in Figure 1.

The figure is representative of a focused design environment composed of existing and future application developments to aid new product realization processes. As depicted, the functional object modeling environment is central to the interconnection of a variety of tools and data. During the conceptual modeling process users may enter requirements and specifications and/or access data as it is described in a marketing database.

Configuration is supported with access to a components database, analysis programs, engineering calculations, and cost data. An interconnection to 3D graphics calls upon libraries of approved components and subassemblies to support the spatial consideration of configurations and evaluate conceptual designs. As the figure indicates, automatic schematic generation should also be a readily obtainable result following the functional product description.

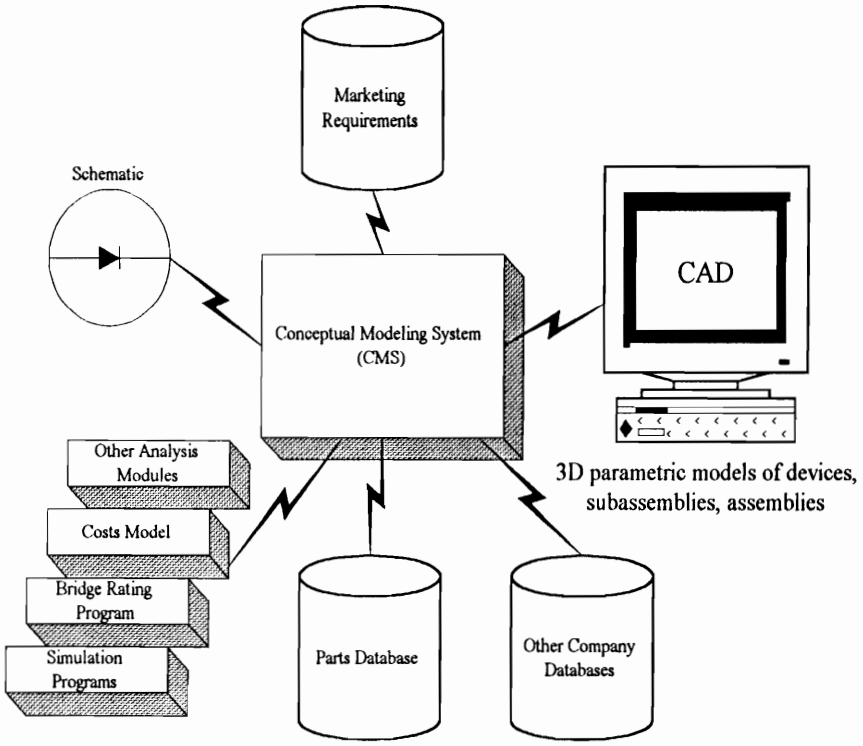


Figure 1. Conceptual Design Environment.

1.5 Scope

While developing all of the applications represented in Figure 1 is beyond the scope of this research, the framework is a significant first step toward realizing an integrated environment supportive of new design processes. The immediate focus of this research has been on the design and specification of the functional modeling environment and on demonstrating the ability of the framework to serve as an integrated problem solving environment.

1.6 Dissertation Outline

The remaining chapters of this dissertation will proceed as follows. Chapter 2 provides a review of the literature. First, an overview of engineering design is presented. Several topics are included in this discussion including: definitions of engineering design, design theory and methodology, design processes, and knowledge in design (types and sources). Technology for design is addressed next with particular emphasis on concurrent engineering and recent advances for conceptual design.

Chapter 3 describes the framework and resultant system developed for conceptual design from a functional viewpoint. A brief introduction to use case modeling is first offered. Next, the use cases and accompanying scenarios that describe the conceptual modeling system are presented. Visuals (screen shots) of the prototyped system are also included.

Based on the functional view presented in Chapter 3, Chapter 4 provides an architectural view of the framework for integrated design. A brief overview of the object-oriented analysis and design (OOAD) methodology utilized to develop the framework introduces the chapter. Several diagrams are then presented detailing the resultant architecture.

Chapter 5 evaluates the framework against the original research premise that calls for the blending of top-down and bottom-up approaches in conceptual design. The design of power conversion systems assists this discussion and provides the foundation for problem examples. Conclusions drawn from this research are presented in Chapter 6. Recommendations for extensions and further considerations are also discussed.

2. *Literature Review*

The review of literature is divided into two major sections. Engineering design is covered first, followed by a closer examination of technology for design. Topics included in these sections are both for background and review purposes. The intent of this broad coverage is to provide sufficient background and an appreciation of the approaches that have been investigated to improve product development processes. Summary discussion is offered at the conclusion of the chapter.

2.1 Engineering Design Overview

The subject of engineering design is certainly broad; having quite a lengthy history and one for which there are a substantial number of texts and research publications. Its inclusion as a topic for discussion here is in support of the belief that, to be successful, technological developments which purport to improve design processes must be rooted in fundamental principles and a clear understanding of the subject matter. It is not the intent of this section to provide an in-depth background, but to present sufficient discussion for understanding and an appreciation for the need and approach of this research. For this discussion, topics include: definitions of engineering design, design theory and methodology, design processes (process models, functions and design, and synthesis approaches), and knowledge in design (types as well as sources.) A more extensive background in engineering design can be obtained from notable texts such as [Suh90], [Pahl84], or [Hubk82].

2.1.1 Defining Engineering Design

Although numerous definitions of engineering design can be found in the literature, most include common elements. Consider the following definitions:

> "Design, as the epitome of the goal of engineering, facilitates the creation of new products, processes, software, systems, and organizations through which engineering contributes to society by satisfying its needs and aspirations." or "Design may be formally *defined* as the creation of synthesized solutions in the form of products, processes or systems that satisfy perceived needs through the *mapping* between the functional requirements (FRs) in the functional domain and the design parameters (DPs) of the physical domain, through the proper *selection* of DPs that satisfy FRs." [Suh90].

> "Engineering design is concerned with the design of products and systems from an engineering perspective." [Nobl93]

> "... a process performed by humans whereby information in the form of requirements is converted into a description of technical systems and other forms of abstractions, such as physical models and mock-ups so that these systems meet certain human needs. Moreover, this process is aided by technical and mathematical tools." [Verm91]

When considered separately, defining the words *engineering* and *design* also lend insight into answering the question: What is engineering design? Webster's dictionary [Wool76] offers the following definitions:

"*engineering* - the application of science and mathematics by which the properties of matter and the sources of energy in nature are made useful to man in structures, machines, products, systems, and processes."

"*design* - a mental project or scheme in which means to an end are laid down."

Key words across all of these definitions promote the notion that engineering design is a means to an end, scientifically based, creative, and noble in its purpose as contributions are sought which satisfy human and/or societal needs. Whether it be a system, product, process, etc., engineering design serves to translate need into concepts which are realizable. Implicitly, and often understated, engineering design is also *responsible*; responsible for the impacts, positive and negative, on the world it serves.

Certainly, engineering design is responsible for major contributions which have defined our modern world: transportation, medicine, utilities, communication, and agriculture, among many others. Yet, the inceptions of engineering design are also directly responsible for failures which are capable of causing death and destruction: collapsing bridges, chemical leaks, electrical fires, nuclear power accidents, and automobile accidents. Further, and with much attention in recent years, engineering design is held increasingly responsible for the impact it has on life-cycle issues such as costs, usability, safety, manufacturability, serviceability, disposability, and quality. It is no surprise that the subject of engineering design is the focus of much research and considered an issue of national importance [NRC91]. Pointing to the ill preparation of today's engineering graduates to meet these challenges, Dixon speaks to the need for doctoral programs in design [Dixo92].

Good design practices can be observed, but *how* can the expertise and methodologies employed be captured, transferred, taught, implemented, formalized or improved? *Why* are some people more likely to be good designers? To seek the answers to these questions is to study design theory and methodology.

2.1.2 Design Theory and Methodology

The study of design theory and methodology is not new. From a historical perspective, writings on the subject of mechanical design methodology are dated from ancient Greek and Alexandrian authors somewhere between 300 BC and 100 AD [Dima93]. Today, progress and debate continue.

Suh [Suh90] describes a *good* designer as one with the ability to identify only the most important requirements of a design task, ignoring those of secondary importance until later stages. In addition, a good designer possesses an in-depth grasp of the issues involved, and is able to operate in the conceptual world as well as the physical domain. *Creative* qualities include the ability to be a risk-taker, having a good memory and much knowledge from many fields, knowing how to use analogies and extrapolate and interpolate from known applications to a new situation [Shaw86].

Agogino, Cagan, and Molezzi [Agog88] report on observations of design teams. Listing a few of these observations:

- major design decisions are made very early in the design process,
- only one or two concepts are pursued at a time,
- designers move alternatively from detailed design to global design,

- design work is divided into sub-tasks, and
- decisions made by a team are sometimes forgotten and rehashed.

Dixon [Dixo89] discusses the inherent difficulties in the formulation of a scientific theory of design. Considered as a process, there is reason to scrutinize the validity of a single, generalizable, theory of design. Dixon emphasizes the importance and enlightenment to be gained in such a pursuit, yet places great significance on discovering and organizing design knowledge and relating it to various types of design problems [Dixo88].

Kota and Ward [Kota90] present opposing views in a debate which ultimately scrutinizes the validity and utility of design research based upon experimental or intuitive ("scruffy") approaches as opposed to design methodologies founded upon mathematically precise ("neat") theory. In the end, the authors agree that it is likely that both approaches will have to coexist, perhaps forever, given the complex subject of design.

With some variation, procedural design methodologies have been widely adopted by many researchers. In Germany, the Society of German Engineers (VDI) has published procedural guidelines which are cited to have been the culmination of nearly thirty years of university research and industry use [Hund92], [VDI87]. The systematic design method prescribes definitive procedural steps to guide design processes [Pahl88], [Rode84]. Steps are grouped into four phases of design: (1) clarification of the task and development of the design specifications, (2) conceptual design, (3) preliminary or embodiment design, and (4) final or detail design. The steps and phases overlap using feedback loops supporting iteration. To date, the majority of work in procedural approaches have focused on design representation, synthesis mechanisms and knowledge representation

schemes which will lead to scientifically based and/or computable methodologies. Much of this research effort has been for the embodiment and detail design phases. Research for conceptual design is very immature by comparison. A more detailed treatment of these topics will be given in subsequent sections of this chapter.

Suggesting an axiomatic approach, Suh defines four distinct *aspects* of engineering and scientific endeavor [Suh90]: "the *problem definition* from a 'fuzzy' array of facts and myths into a coherent statement of the question; the *creative process* of devising a proposed physical embodiment of solutions; the *analytical process* of determining whether the proposed solution is correct or rational; and the *ultimate check* of the fidelity of the design product to the original perceived needs." The creative process is described as an ideation process which is highly subjective and dependent upon the specific knowledge of the designer and their ability to integrate this knowledge. The analytic process is deterministic, based upon basic principles, and serves to evaluate the concepts of the creative process. Suh provides two axioms used in the analytic process for the purpose of distinguishing good designs from bad. Without these axioms, Suh considers design decisions to be made at best on an "ad hoc" or "empirical" basis. Axiom 1 is the *Independence Axiom* which states: maintain the independence of functional requirements. Axiom 2 is the *Information Axiom* which states: minimize the information content. Sekimoto [Seki94] demonstrates the utility of the axiomatic design theory for the design of a paper handling mechanism.

Others have sought to improve upon generalized methods through the classification of design problem types. Pahl and Beitz [Pahl88] categorize design into three types: original design, adaptive design, and variant design. Original design involves elaborating an

original solution principle for a system, adaptive design involves adapting a known system to a changed task, and variant design involves varying the size and/or arrangement of certain aspects of the system with the solution principle remaining unchanged. Condoor *et al.* [Cond92] use a similar approach, but categorize design problems according to a cognitive framework resulting in four classifications: variant design, developmental design, adoptive design, and original design. These categories help to evaluate and guide processes based upon conceptual and configurational novelty. Ullman [Ullm92] decomposes mechanical design processes into original design, parametric design, configuration design, and catalog selection. Again, design complexity serves to categorize the problem difficulty and focus the emphasis of process steps [Snav93].

More recently, the methods of artificial intelligence have received considerable attention as design methodologies are sought which are computable and reflective of reasoning similar to human creative processes [Boyl93], [West93]. Waldron *et al.* [Wald89] propose a theoretical framework for representing the mechanical design process based on the use of systemic theory. Cagan and Agogino [Caga91] offer a methodology for inducing trends in a first principle reasoning system for design innovation. Takeda *et al.* [Take92] propose a logical design process model with the potential to serve as a framework for integrating design models and design processes in the definition of an intelligent CAD system. Williams [Will90] suggests an approach for designing novel devices from first principles. Design is viewed as a process of building interaction topologies.

2.1.3 Design Processes

This section provides a closer examination of design processes. Process models are central to defining the stages or phases of design and are therefore considered first. The

concept of function and its role in the design process is then examined. Finally, synthesis approaches (solution methods) are discussed.

2.1.3.1 Process Models

From a very high level, researchers agree that the processes of engineering design can be grouped according to purpose. As described previously, Suh [Suh90] defines four distinct *aspects* of engineering and scientific endeavor. Pahl and Beitz [Pahl88] group design processes into four *phases*. Maher [Mahe90] describes three general phases: design formulation, design synthesis, and design evaluation. Cited as the most widely used engineering design text in the United States [Ward94], Shigley [Shig77] describes a process model which includes six broadly defined steps. In this model (see Figure 2), design processes begin with a recognition of need and end with a presentation of the solution.

Upon examination, all of these process models are very similar, if not the same. Following the recognition of a need, be it market, customer, or internally driven, the need is formalized into functional requirements and constraints which define the problem in terms of specifications. Solutions are then sought to satisfy the specifications (synthesis). These solutions are evaluated against the specifications resulting in: satisfaction, the need to continue the search, or the need to modify the original requirements. It is with much iteration and the application of information, knowledge, and constraints on the problem domain that the feasible design space is ultimately reduced to realize the final product definition (see Figure 3).

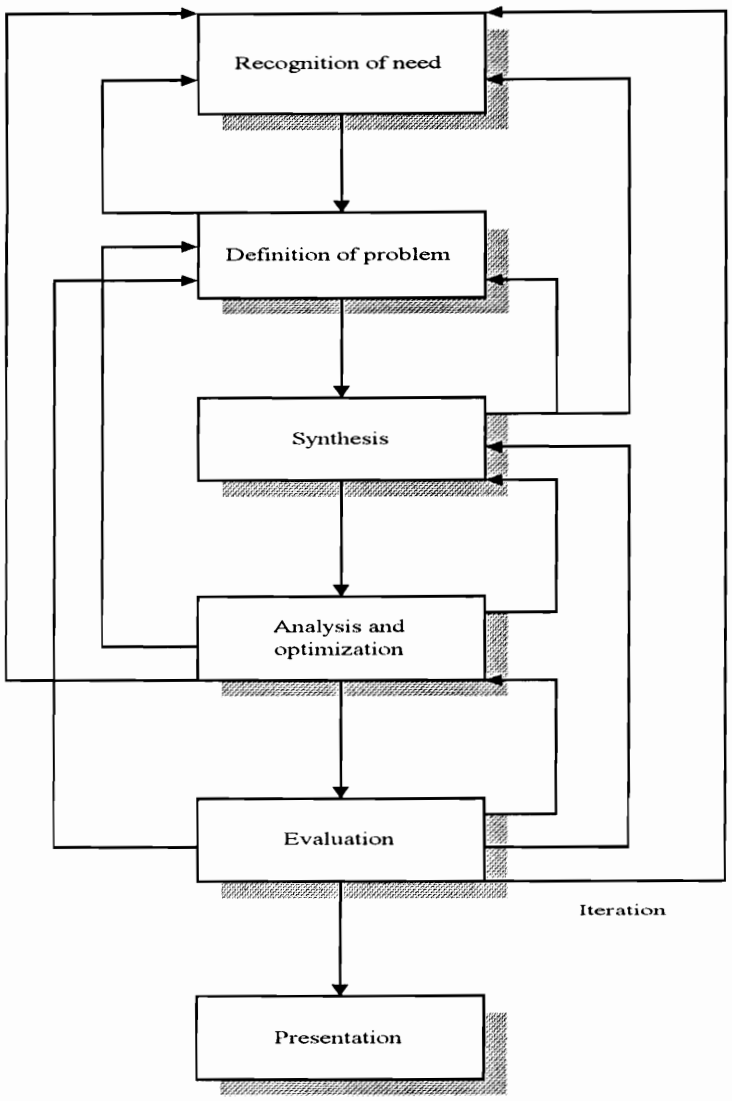


Figure 2. Model of the Design Process [Shig77]

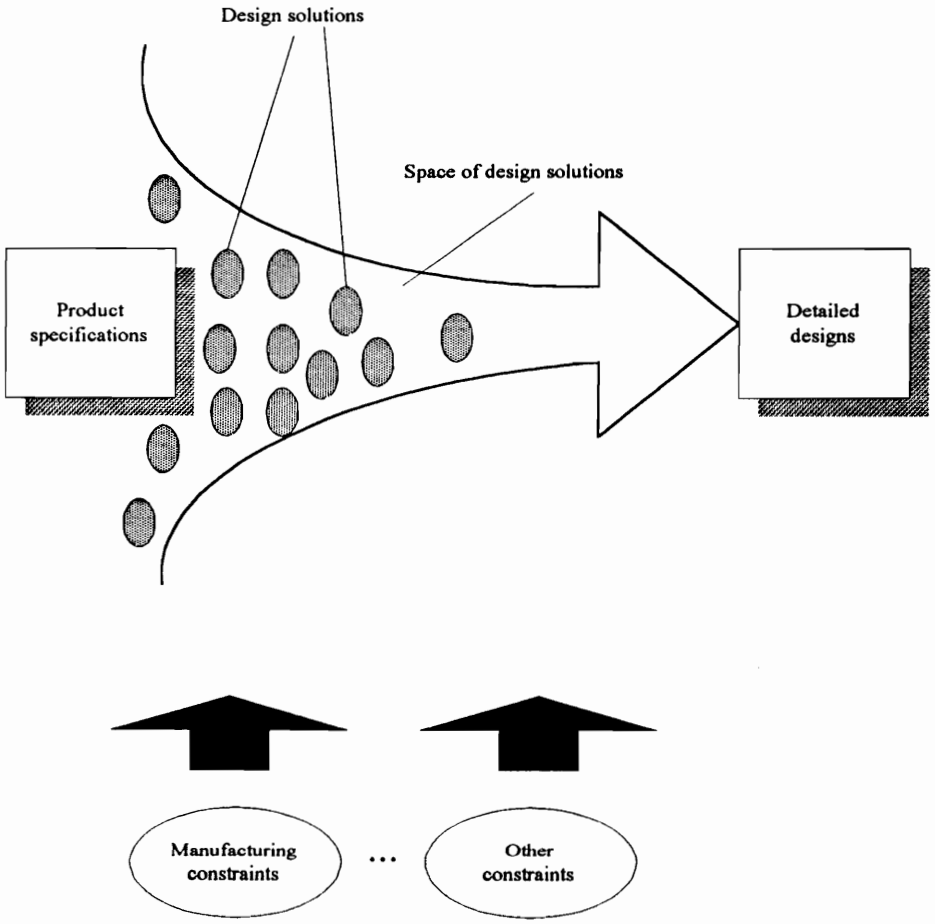


Figure 3. Design Process and Pruning of Feasible Solutions [Schm89]

Additionally, throughout the stages of development, new design problems emerge and must also be addressed as the product design progresses to its eventual detailed definition. Finally, the designer presents the results which document the design in terms of material selections, geometric models, circuit diagrams, software, physical component selections, and other documentation sufficient for product realization.

2.1.3.2 Functions and Design

As noted in the previous section, *function* is a concept that is fundamental to engineering design processes. Suh [Suh90] describes design as "a continuous interplay between what we want to achieve and how we want to achieve it"; function commonly describing the need, desired intent, or purpose. The objective of design is stated in a *functional domain* and the solution is generated from a *physical domain.* The design process involves mapping from the functional domain, specified as functional requirements, to an embodiment in the physical domain, characterized in terms of design parameters. Figure 4 illustrates a simplified representation of this mapping.

Note that the physical domain is not necessarily restricted to solutions which are physically tangible, but may also include, among others, solutions such as software, organizational plans, or process descriptions. Though not explicitly evident in Figure 4, both the functional space and physical space are likely to contain decomposable hierarchies. Further, the two domains are inherently independent of each other and only related through the design. Ultimately, it is through the ongoing interlinking between the functional and physical domains, at every level of design, that solutions are iteratively refined and realized with greater detail.

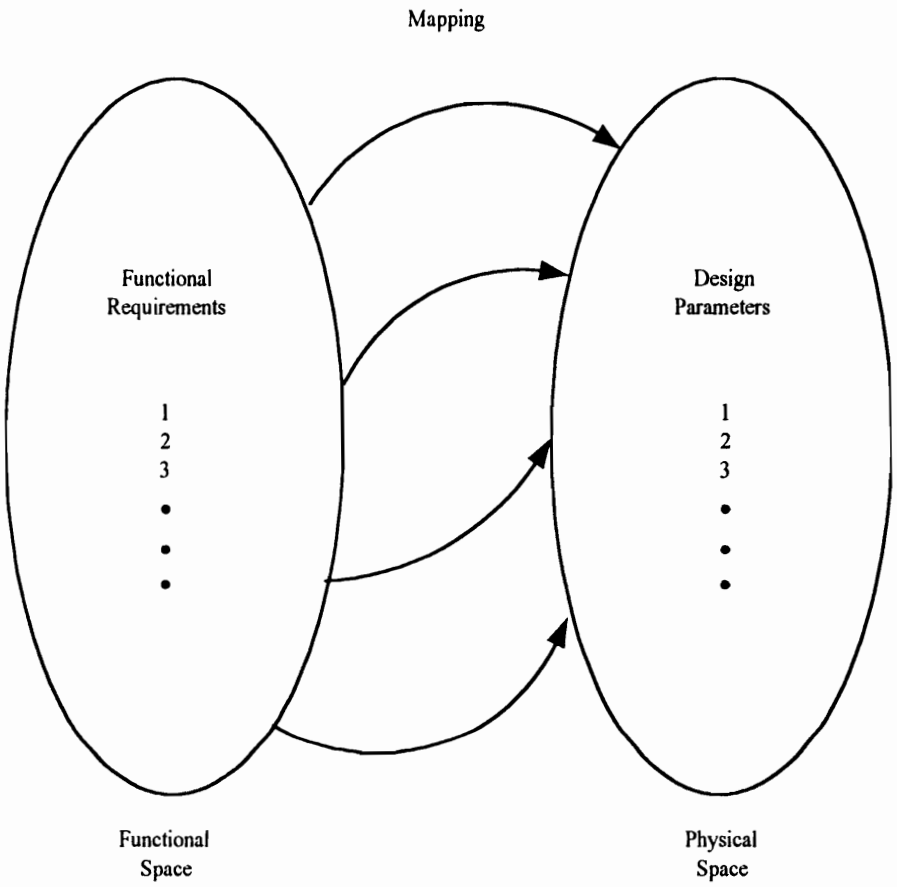


Figure 4. Design as a Mapping Process [Suh90].

Beginning with problem definition, it is crucial to understand and identify the *appropriate* set of functional requirements. Discussed with greater detail in subsequent sections, the impacts of decisions that are made very early in design based upon high-level functions are significant. Early design decisions restrict, and in many cases determine, much of the remainder of product design as well as matters related to life-cycle issues. In today's competitive marketplace it is therefore imperative that systems developed to support early design processes consider functional modeling (abstraction) and the information requirements for informed decision making (detail).

2.1.3.3 Synthesis Approaches

Boyle [Boyl93] maintains that design in general is still a very poorly understood activity where too often researchers have limited their focus to particular design domains, notably very large scale integrated circuits (VLSI) and mechanical engineering design. However, Boyle also states that a universal and formal model of engineering design is not realizable given the diversity of different design domains. In an effort to "escape" from domain specific research, Boyle suggests a classification that splits design into three broad methodologies:

- analytic,
- procedural, and
- experimental.

The distinction in approaches is made according to the amount of analytical knowledge that can be applied to the process of obtaining design solutions. This categorization directs the designer to the most appropriate method (synthesis approach) for solving the problem.

The *analytic approach* can be used for problems where the objective(s) and constraints can be specified with precise and complete models, and solved for optimality using algorithmic methods. The analytic approach can be found in the work of Fabrycky [Fabr92], who formulates and synthesizes design solutions in what is termed the "design dependent parameter approach."

The *procedural approach* is a trade-off process. It is appropriate for problems where the objectives are rarely fixed, but tend to be modified as the design proceeds and the designer obtains more information about what can realistically be achieved. All objectives are satisfied, but are rarely optimal.

The *experimental approach* is a search oriented process. An available set of solutions are systematically searched to find the best match of attributes with design objectives. This results in the 'best' design solution from the known search space. Noble and Tanchoco [Nobl93] describe these approaches pictorially as shown in Figure 5.

Lin [Lin93] also categorizes synthesis models into three distinct categories. Here, the approaches are given to include:

- decomposition,
- case-based reasoning, and
- transformation

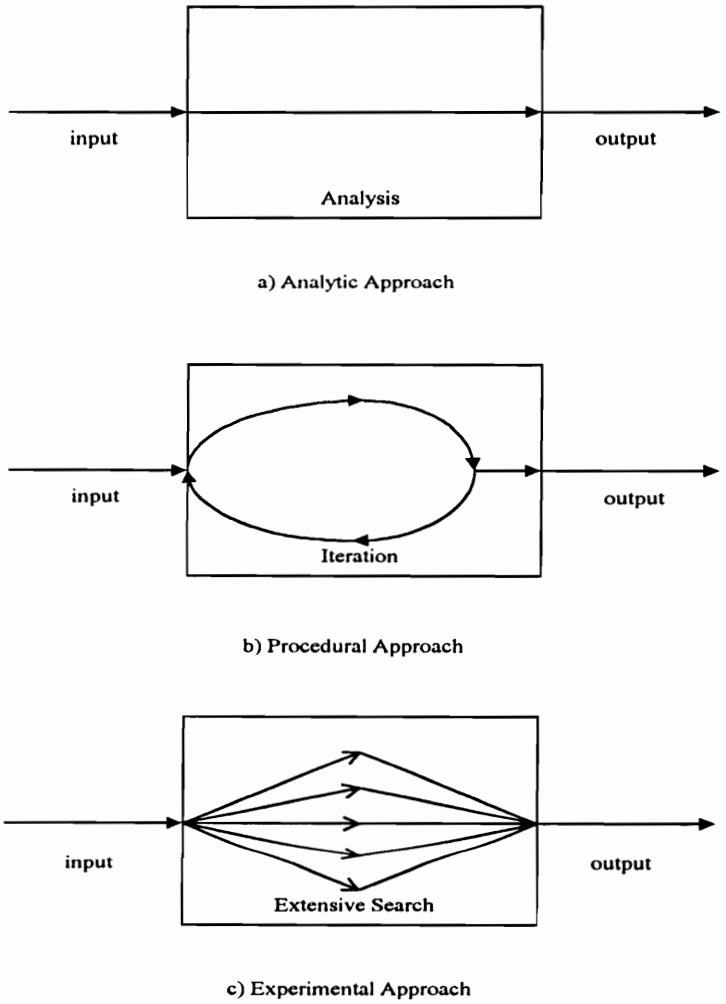


Figure 5. Engineering Design Approaches [Nobl93]

Decomposition is described as a sequence of refinement steps wherein large or complex design problems are progressively broken down into subproblems. It is considered to be a "divide and conquer" technique which eventually leads to problem solution. Decomposition is particularly suitable for problems where the subproblems are loosely coupled.

Case-based reasoning approaches rely heavily upon an extensive database of previous case (solution) histories. Near matches are sought with existing cases and then adapted to define a new case for the given problem situation [Pu93].

Transformation involves making progressive steps from the initial set of requirements toward a design solution. At each step the complete model is transformed from the previous state on to the next. This type of synthesis model is appropriate when there is a strong connection of artifacts which cannot be decomposed into subproblems.

Contrasting the categorizations offered by Boyle and Lin, one notes the similarities. The experimental approach and case-based reasoning are both search oriented. Similarities are also noted between the analytic approach and transformation. Less obvious, are the parallels which can be drawn between the procedural and decomposition methods.

A more important distinction to made in this discussion is that in practice, engineering design problems usually require some combination of these approaches. The exact approach used often is related to the level of abstraction and stage within the design process. Over the life of the design project, designers may call upon all of the synthesis methods. This is certainly the case where satisfying a recognized need implies developing

a complex system, such as an electro/mechanical system. It is likely that the problem will undergo decomposition into smaller subsystem design problems. Transformation (analytical) methods will be used to analyze and determine feasible solutions; as in the case of thermal models for example. Procedural approaches will be used throughout the design effort as a greater understanding of the system and firming of the specifications occurs. Search and adaptation methods, regardless of whether formally implemented in computer based tools, will be used as designers inherently draw upon previous experience when presented with similar design problems.

Numerous examples of "hybrid" design methods can be found in the literature. Roderman and Tsatsoulis [Rode93] integrate iterative refinement and decomposition in what is considered a case-based approach to aid novice mechanical designers. Bardasz and Zeid [Bard93] also use a case-based approach for mechanical design, but point to the need for a hybrid approach which could also reason from first principles of engineering. Navinchandra *et al.* [Navi91] also use a case-based approach for mechanical design. Transformation rather than pre-defined indexing is used for case retrieval. Again, combining case-based methods and transformation, Maher and Zhang [Mahe93] use a hybrid approach where case transformation is treated as a constraint satisfaction problem.

2.1.4 Knowledge in Design

Engineering design can be characterized as an information-based process [Colt94], [Ullm94]. From the earliest recognition of need, proceeding through to the completion of finalized design and documentation, information is gathered, manipulated and generated. Considered in a broader sense, information is a form of knowledge. A closer examination of knowledge types and sources follows.

2.1.4.1 Knowledge Types

Miles and Moore [Mile94] offer two generally accepted categories of knowledge:

- declarative, and
- procedural.

Declarative knowledge is described to include facts, concepts, and relationships. This type of knowledge is gained via lectures, tutorials, and design practicals; perhaps in a university setting.

Procedural knowledge is described as relating to how one performs tasks. It is most commonly acquired with experience and is therefore difficult to verbalize. Further, Miles and Moore relay that 'real expertise' consists mostly of procedural knowledge.

Another classification of knowledge offered by Miles and Moore [Mile94] includes:

- algorithmic,
- heuristic, and
- meta-knowledge.

These classifications deal with the type of approach taken to design processes. *Algorithmic* utilizes equations based on Newtonian physics whereas *heuristic* relies on 'rules of thumb' from experience. *Meta-knowledge* deals with the knowledge of how to control procedures and methods. For artificial intelligence (AI) applications, this would be the knowledge used to control the inferencing procedure.

Schmekel [Schm89] takes a different approach to the classification of knowledge types. Four categories distinguish knowledge types ranging from the most fundamental, and therefore generalizable, to knowledge which is highly specific to particular company practices. Briefly, these types include:

- *physical knowledge* - concepts of physical reality and basic sciences,
- *design knowledge* - concepts of shape, function, structure, and relations used in developing the product description,
- *application knowledge* - describes the application of a design. Includes design knowledge (shape and function) and physical knowledge (behavior), and
- *company knowledge* - company standards and specific product descriptions.

Similarly, in a less technical publication [Wisd92], knowledge is classified into several types along a spectrum from broad, generic knowledge, to product specific knowledge:

- *generic* - basic scientific concepts such as force, moments of inertia, etc.,
- *domain* - particular to a field of engineering (electrical, mechanical, etc.),
- *industry* - applicable to a specific industry (automobiles for example),
- *company* - information and standards of how products are designed specific to a given company, and
- *product* - particular to a given product or product-line.

Pictorially, the concepts and relation of generality and abstraction to knowledge types are represented in Figure 6. Following the directed arrows, it can been seen that abstraction increases as knowledge types become increasingly specific. Inversely, generality increases as knowledge types become less abstract.

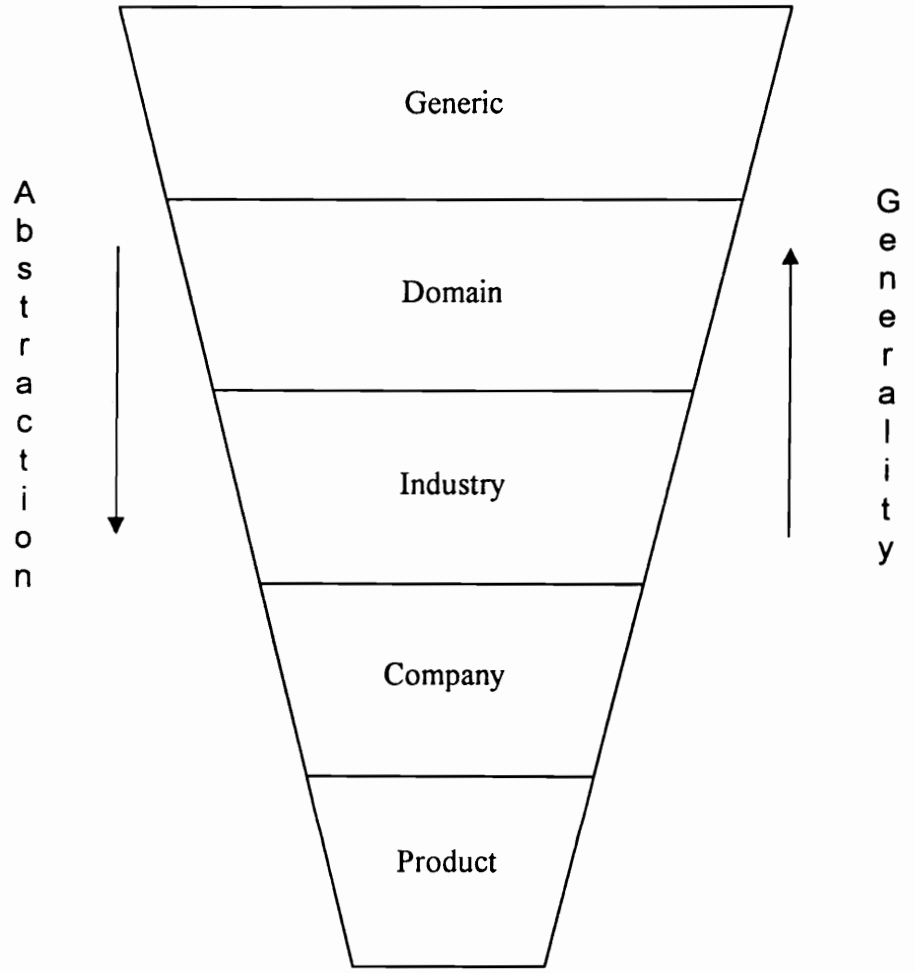


Figure 6. Generality and Abstraction in Design Knowledge

2.1.4.2 Knowledge Sources

Even more varied than the types of knowledge in design, are the sources from which this knowledge is obtained. Whether manually kept or stored in a computer, numerous sources are accessed as information is gathered, manipulated, and generated throughout design processes.

Knowledge sources can include people, databases, files, catalogs, equations, analysis results, handbooks, and other hard-copy (paper) documentation. Customer requirements, marketing requirements, existing designs, manufacturing constraints, costs, vendor data, availability, part data (ratings, features, etc.), geometric models, schematics, evaluation and/or simulation software, and personal expertise are just some of the typical inputs to informed decision making.

Clearly, availability, format, and timeliness of knowledge sources are all important factors affecting both the efficiency and effectiveness of design processes. Time spent tracking down information which is not readily available, unreliable, ambiguous, or otherwise difficult to use, extends cycle-times, leads to misinformed decisions, and is time which could have otherwise been applied to creative processes.

2.2 Technology for Design

The major motivation for this work, and an abundance of research in the literature, is that today's competitive marketplace dictates the development of enabling technologies for improved design processes. In particular, advances which serve to improve early design, integrating abstract and detailed processes, hold significant promise for competitive

advantage. This section will promote this assertion and justify the proposed research through an examination of recent advances of technology for design.

The review begins with a discussion of concurrent engineering. Following this introduction, advances which have been fundamental to enabling computable methods for engineering design are presented. The concept of architecture is presented followed by topics including feature and parametric based techniques, object-oriented methods, and architectures for integrating design processes and tools. Next, research for design evaluation and advice is reviewed. Model representation and synthesis methods for configuration and conceptual design are also reviewed. The chapter concludes with sections covering the topics of mechatronics and engineering analysis, and final summary discussion.

2.2.1 Concurrent Engineering

Recognizing the inherent complexity and interdisciplinary nature of engineering design, researchers have taken a variety of approaches to *enable* improved design processes. Perhaps overapplied, *concurrent engineering* (CE) is the term commonly associated with these research endeavors. Hence, it is important to define and understand its meaning. In a definition frequently cited by others, the Institute for Defense Analysis [Winn88] define concurrent engineering as follows:

"Concurrent engineering is a systematic approach to the integrated, concurrent design of products and related processes, including manufacture and support. This approach is intended to cause the developers, from the outset to consider all elements of the product life cycle from conception through disposal, including quality, cost, schedule and user requirements."

Though as a whole, concurrent engineering is still more of a goal than a reality, it has brought much research attention to competitive issues and the role engineering design must assume in achieving these objectives. As cited in Chapter 1, the impacts of design decisions on life-cycle costs are now recognized as significant motivators for research. Estimates indicate that as much as 80% of total life-cycle costs are committed by the completion of product design stages. Further, 60-75% of total costs are committed by the completion of conceptual design.

Using manufacturing as an example, it is easy to observe how costs are directly affected by product design. Design configuration naturally limits the feasible set of machines and tools which can be selected to manufacture a part. With limited manufacturing knowledge, a designer may develop a product which requires non-standard, or expensive processes to manufacture. The scrap, time, and missed promise dates which result from rework, special processes or late changes in the design can be quite costly. Kapur [Kapu94] relays how process variation, manufacturability, and quality are all interrelated, and must therefore be considered within product design.

Current literature abounds with research claiming to advance the practice of concurrent engineering. It would seem that almost anything which serves to ease the tasks of design

and/or integrates knowledge into the design process is said to be for concurrent engineering.

Few have truly kept to the original definition of concurrent engineering; designing both product and processes concurrently. More commonly, processes are given to be fixed and consideration of life-cycle issues is supported through design evaluation and advice systems, access to databases containing life-cycle information, or model representations which ease the flow out of design and on to downstream life-cycle processes.

O'Grady and Young [OGra91a] offer a somewhat different definition where little, if any, emphasis is placed on the actual design of supporting processes. Rather, the authors take more of a reactive perspective and stress the need to design *for* other life-cycle issues. Their definition follows:

> "Concurrent engineering (CE) is the consideration, during the design phase, of the factors associated with the life-cycle of the product. These include manufacturing, assembly, testing, maintenance, reliability, cost, and quality. ... Subsets of CE include design for manufacture, design for assembly, design for testability, design for maintainability, etc."

With this definition, most any research for engineering design could be qualified as concurrent engineering. Disregarding other life-cycle processes for a moment, one must ask the question: What is *good* design, if consideration has not been given to reliability, quality, and costs? Suh [Suh90] maintains that the life-cycle issues of concurrent

engineering are merely additions to the functional requirements and constraints of the design problem definition.

Rather than debating semantics, in what is or is not concurrent engineering, suffice it to say that its popularity as a research area has led to not only the consideration of life-cycle process issues, but more generally, a renewed interest in improving engineering design.

2.2.2 Architecture

The concept of architecture is also an important topic for computable methods for engineering design. Certainly, it is not a new concept, but one which should be familiar to all development type projects. Whether it be software development, the construction and building industry, printed circuit board (PCB) design, or complex products developed from across engineering disciplines, a sound architecture supports not only the definition of immediate solutions, but also provides the foundation for future development efforts.

On the subject of object-oriented software engineering, Jacobson [Jaco94] discusses the importance of understanding what goes into making "industrial processes" successful and applying this knowledge to systems development. As shown in Figure 7, *architecture* is the fundamental building block upon which a "rational enterprise philosophy" is based. Relating to the construction industry, Jacobson suggests that there must be rationality in all phases of building construction. This is accomplished with a well established philosophy to guide the work of all participants and activities in the project. The philosophy is grounded in the form of an architecture and related activities which establish the way of doing business, i.e., an approach.

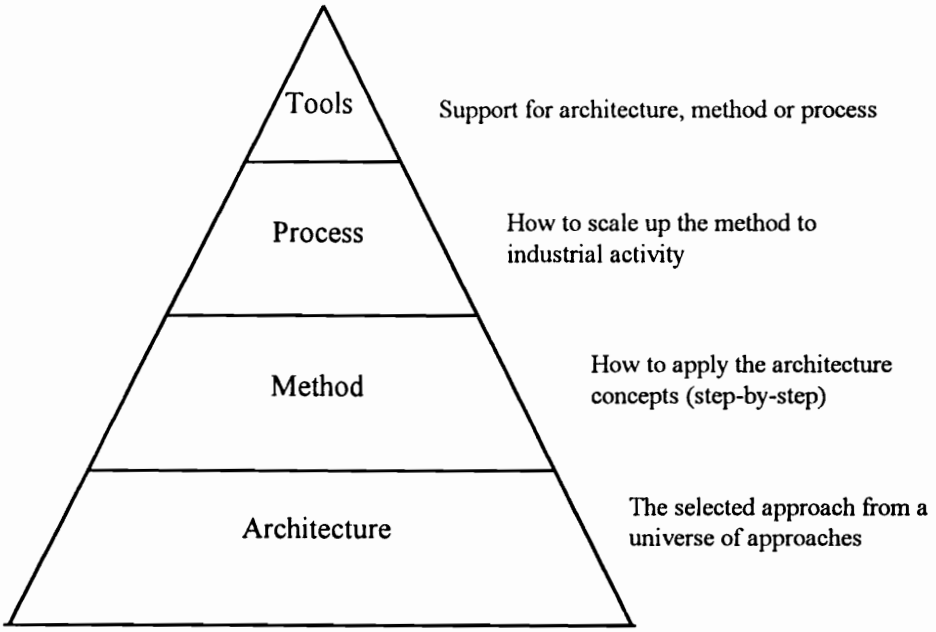


Figure 7. The Constituents of a Rational Enterprise Philosophy [Jaco94].

Jacobson defines *architecture* as a foundation of concepts and techniques, selected from a universe of potential foundations, that define the characteristic structure of all instances using the approach. The architectural approach adopted could be based on using building blocks and components, treating every new instance as a customized construct, or some combination thereof.

Again referring to Figure 7, *method* prescribes the step-by-step procedures, broken into activities, to follow in applying the architecture to projects. The *process* permits the method to be scaled up when applied to projects with many interacting activities and participants. It lasts as long as the product, and describes the interaction of activities throughout the product life-cycle. *Tools* support the activities of architecture, method, and process.

In the construction example, building blocks could be prefabricated sections of a house. Components would likely include basic off-the-shelf devices. Examples of some of these devices might include windows, doors, lighting fixtures, and tubs. In the case of different material types, bricks or concrete for example, each approach would have associated methods to describe how to work with these constructs using step-by-step procedures.

Through analogy with the construction example, one can see how these principles could also be applied to software development. In particular, important aspects of object-oriented software engineering would permit making software development an industrial process. This philosophy moves beyond what has traditionally been a strong emphasis in the creative initial development of software systems to one that also promotes viability in an industrial environment.

Jacobson suggests the need for reusable software based upon building blocks and components. Reuse should occur at several levels of granularity, with components representing the most primitive constructs, and multiple components, larger modules, or entire programs providing the building blocks necessary for the greatest reuse benefit. Such a software development environment is an on-going challenge that the software industry has yet to realize.

On a larger scale, reuse is an important topic. Whether it be in the domain of software engineering or in some other development domain, reuse supports an industrial process philosophy. Establishing a sound architecture where reuse is integral makes new development economical. Knowledge and best practices are retained which have been proven to promote quality, safety, manufacturability, and other life-cycle considerations. Moreover, the architecture provides the mechanism through which standards and norms can be put into practice.

The challenge for reuse is, of course, in discovering and understanding the components and building blocks that are appropriate for the domain of interest. The architecture should provide a foundation which is rich in building blocks while also providing the basic components for freedom and flexibility in new developments. It is from this approach that the proposed architecture for conceptual modeling is founded. Demonstrated in the domain of power conversion systems, the architecture makes use of basic and higher level functions to configure conceptual model design representations.

2.2.3 Feature and Parametric Based Techniques

One of the most significant influences in computer based methods for concurrent engineering has been the concept of *features*. Originally inspired by feature extraction applications for manufacturing (Lee and Fu [Lee87] for example), feature and parametric based techniques have more recently received considerable attention as researchers have sought to integrate form, function, and life-cycle issues in design.

For geometric modeling in particular, features have offered a higher level of abstraction above that of more traditional modeling systems. Designers are relieved of the tasks needed to reduce intent to low level details of topology and geometry for representation. Rather, models can be constructed with features reflective of purpose. A few examples of form features include: holes, slots, flanges, louvers, and keyways. In addition, when parametrically defined, features can be sized, located, and geometrically related within a part or assembly model [Keir90]. Refer to Figure 8 for a pictorial representation of a feature based description of a sheet metal part offered by Eversheim *et al.* [Ever82]. Having advanced rapidly, feature and parametric based techniques are now an expected, if not required, capability within commercially available computer aided design (CAD) systems [Huss90].

Nonetheless, features and their use in design have not been without controversy. While many researchers agree that intelligent design systems must be capable of capturing both form and function, and that features are ideally suited to this purpose, there has yet to be agreement on this representation. In fact, there is no single agreed upon definition of what

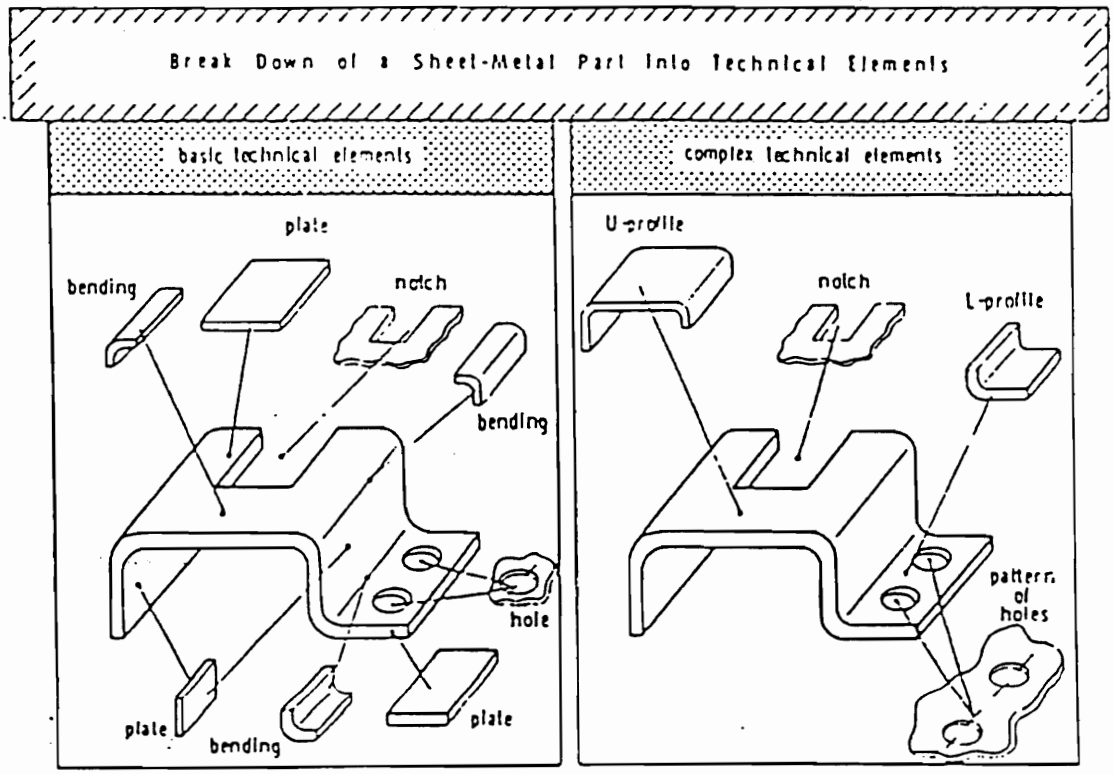


Figure 8. Features in a sheet metal part [Ever82]

constitutes a feature, how it should be represented [Fing90], [Rose92], [Subr92], or whether it is better to design with features [Maye94] or employ feature extraction and recognition strategies [Pete92].

Nnaji and Kang [Nnaj90] define feature to be a set of geometric entities together with specification of the bounding relationships between them which together imply an engineering function on an object. Cunningham and Dixon [Cunn88] state "A feature is any geometric form or entity that is used in reasoning in one or more design or manufacturing activities (i.e., fit, function, manufacturability evaluation, analysis interfacing, tool and die design, inspectability, serviceability, etc.)." To be useful, i.e., computable, in a CE environment, features must contain at a minimum, geometric, topological, and technological data for both the product definition and function of supporting life-cycle processes. In general, the completeness and content of feature information will depend on both the application domain and the degree to which function can be inferred through intelligent reasoning systems [Terp93]. Mayer *et al.* relay how features provide a language (vocabulary, grammar and semantics) which is inherently context-dependent [Maye94].

Researchers in concurrent engineering have utilized feature and parametric based techniques in a variety of applications. As discussed in a later section, features have been investigated to integrate and evaluate life-cycle concerns in design. In addition, design feature representations have been investigated for automating the interface to and function of downstream processes such as assembly, inspection, manufacturing planning, and rapid prototyping. For example, Chang and Anderson [Chan92] discuss a research prototype developed to implement and study the integration of feature based design and process

planning. The system combines feature based design and feature refinement processes to translate design features for CAM purposes.

Gu *et al.* [Gu89] have investigated the development of a language to associate feature based design and manufacturing tasks planning. The developed system consists of a number of lexical analyzers, a parser and three code generators. It has been applied to feature based modeling, parts/manufacturing cells assignment, expert CMM inspection planning, and mechanical assembly sequence planning.

2.2.4 Object-Oriented Methods

Object-oriented methods have also had significant influence on the evolution of computer based methods for concurrent engineering. Based on methods which encourage structure and a building-block approach, many researchers believe object-oriented methods to be a *natural* approach for knowledge representation and reasoning processes in design [Kim90], [Mile94], [Boyl93].

With this approach, knowledge is viewed in terms of objects; each object being an instance of some class, possessing certain behaviors and properties based upon class membership and position in a hierarchical class structure [Pars88]. Consequently, objects can contain both data and methods for data manipulation [Lin93]. Important characteristics of object-oriented methods include:

- *encapsulation* - also known as information hiding, allows for the separation of how an object behaves from how it is implemented [Mart93]. This is an important concept allowing for the creation of modular, as well as reusable object definitions. Objects communicate with each other by way of message passing. In this manner, the requester has no burden or need for knowledge of the actual method by which the request is honored.

- *inheritance* - is the ability of lower level objects in a class hierarchy structure (children) to assume and apply (inherit) the characteristics of higher, more general object classes (parents). Hierarchies allow for coping with complex issues; promoting structure and reuse.

- *polymorphism* - is the ability for two or more objects (different classes) to respond to the same request with different methods. Here again, the requester does not need to know, or care about, who or how the request is met, only that it is satisfied.

- *dynamic binding* - also known as late, delayed, or virtual binding [Jaco94], implies that the linking (binding) of a requested procedure to an appropriate operation (method) occurs at run-time rather than at compilation time. Again, this is a characteristic which supports software reuse and flexibility.

Object-oriented methods have been widely applied in the construction of intelligent design modeling systems. Applications have ranged anywhere from defining feature based graphic modeling systems to artificial intelligence applications for the generation of complex product assembly models. Discussed further in subsequent sections, object-oriented methods have been central to knowledge organization and representation, the

development of engineering design software, and architectures for integrating design processes and tools.

Kim *et al.* [Kim91] apply an object-oriented approach; offering a taxonomy for rotational parts. Nnaji *et al.* [Nnaji91] address classification in the domain of sheet metal components. Rosen *et al.* [Rose92] discuss a feature-based design environment using features containing form, intent, and relevancy. The authors describe how form contains geometry and connectivity with other features, intent captures designer specified limitation on form, and relevancy identifies which computer integrated manufacture (CIM) viewpoint from which the feature is defined. Intent is expressed by the tolerances, dimensions, and size parameters used by the designer. Wierda [Wier91] discusses how a feature based modeling system, again object-oriented, can be used to link design, process planning, and cost.

2.2.5 Architectures for Integrating Design Processes and Tools

As discussed previously, engineering design is very much an information-based process involving an extensive variety of data sources, knowledge, and tools. Information useful to design can be found throughout an entire organization. Customer requirements, marketing requirements, existing designs, manufacturing constraints, costs, vendor data, availability, part data (ratings, features, etc.), geometric models, schematics, evaluation and analysis software, and personal expertise are just some of the inputs to informed decision processes in design.

Undeniably, computer networks, the proliferation of powerful and inexpensive personal computers (PCs) and workstations, as well as readily available advanced software and

database development tools have had profound effects on both the availability of information and an ever increasing number of specialized tools.

Given this situation, one would expect designers to be operating effectively in such a rich environment. This is not the case. In fact, many consider this wealth of tools and information to have contributed to poorly defined design processes and unwanted complexity [Salu94], [Simo91]. Integration, software maintainability, distributed processing, and data integrity are just some of the issues which are on-going concerns.

Papanikolopoulos [Papa91] has investigated distributed problem solving techniques for collaboration among different tools. Tools are organized into classes and stored in an object-oriented database. Tong [Tong92] also uses object-oriented methods to manage tools in a blackboard architecture for coordinated design automation efforts. Kolb and Bailey [Kolb93] address the need for integrating engineering analysis codes using a constraint-based object modeling approach. Addressing the maintainability and development of engineering software in general, Woyak [Woya95] suggests the integration of software modules both in terms of functionality and data. An object-oriented approach is used in what is called the dynamic integration system. Dynamic variables along with dynamic responders identify dependencies across software modules.

Rather than focusing on methods for integrating analysis and optimization software, Sakthivel and Kalyanaraman [Sakt93] present a knowledge based expert shell to integrate AI technologies problem-solving strategies for engineering processes. Here again, the shell is based on a blackboard architecture. Knight and Kim [Knig91] apply the integration of diverse knowledge bases as a means to improve design processes.

Knowledge is separated into domain-independent and domain-dependent modules specific to particular fields of application. Users progress through four module levels as the design and supporting processes evolve with increasing detail. Eppinger *et al.* [Eppi94] tackle process improvement through methods which serve to organize the tasks of product design. A matrix structure is used to alert project managers to issues of sequencing and opportunities for decoupling tasks to speed design.

Others have researched the creation of integration platforms to serve as intermediary between applications in a CIM environment. Weston *et al.* [West91] suggest a soft integration which requires only a knowledge of how to use the platform of services, not requiring knowledge of other system entities. Glicksman *et al.* [Glic91] create what is characterized as a meta-CIM system whose goal is to tie together existing CIM systems and other resources. Membership in the centralized system (MKS) facilitates the integration and inter-operability amongst members.

2.2.6 Design Evaluation and Advice Systems

Feature and parametric based representations have been well suited to computer based methods for design evaluation and advice systems. Numerous examples of these systems can be found in the literature. Hernandez *et al.* [Hern91] couple feature based product modeling with assembly analysis methods for critiquing part shapes and part count. Yannoulakis *et al.* [Yann91] describe a method for developing quantitative indicators of manufacturability. Feature-based design and estimation of machining parameters are used for ascertaining the manufacturing requirements for a given part. The designer is led to features that must be redesigned for improved manufacturability. Rosen *et al.* [Rose92b] have developed a methodology for converting a design features representation of a

component into a tooling cost features representation. Cost drivers are then computed for the cost features. Evaluation is done directly from cost data tables.

Rather than using design by feature methods, Nnaji *et al.* [Nnaj91] incorporate manufacturing knowledge using feature extraction, geometric reasoning, and rules which check the proximity of features (cutouts, bends, etc.) for sheet metal fabrication. An intelligent CAD interface assists the evaluation process; prompting the user with suggestions and assisting the modification process. Jakiela and Papalambros [Jake89] offer a suggestion-making CAD system that evaluates designs for automated assembly based upon the handbook rules of Boothroyd and Dewhurst [Boot83].

Outside of graphically based systems, Lawlor-Wright and Hannam [Lawl89] rely on an expert system to guide designers as they answer questions which are ultimately checked against a tool library to insure manufacturability. Young, Greef and O'Grady [Youn91] use constraint networks to advise design decisions using a knowledge based environment for concurrent engineering. Bowen and Bahler [Bowe91] also use a constraint-based approach, targeting the need for negotiation when users conflict in their decisions.

Iyengar *et al.* [Iyen92] introduce the notion of connectors for top-down design and bottom-up evaluation. Connectors serve to combine individual components into functional groups. The information content of connectors varies depending on the criteria used for evaluation (e.g., cost, reliability, efficiency.) Bradshaw and Young [Brad92] suggest the need for evaluating designs using knowledge of purpose and knowledge of structure, i.e., how well have the functional requirements of the design problem been met.

Generally, design evaluation and advice systems have most often used reductions in costs and/or cycle time to guide or alert designers to better decisions in product definition. In doing so, implementations have sought to incorporate strategies such as simplification, standardization, minimizing manufacturing processes and setups, minimum material, and easily replaceable modules [Lu88], [Mill92], [Colt91], [Mile93]. Knowledge bases of preferred alternatives, feature based design, rule bases, and analysis software have all served in defining systems to guide and restrict designers toward meeting these objectives. Though these methods have shown some progress in defining design for assembly (DFA) and design for manufacture (DFM) type systems, progress in other design for 'X' (DFX) life-cycle concerns has been very limited. Moreover, systems have been restrictive in their ability to cope with conflicting measures of effectiveness across life-cycle concerns, have lacked the means to accurately measure or estimate costs, or have suffered from a limited capability to incorporate non-graphic information in an integrated design environment.

2.2.7 Computable Methods for Design Capture and Solution Synthesis

Beyond evaluation and advice systems, design automation has also been investigated as a means to reduce cycle time and costs; supporting CE objectives and improving design processes in general. Varying success has been achieved as researchers have applied computer-based methods to modeling and mechanisms for solution synthesis. Highlights of the advances for configuration design and conceptual design follow.

2.2.7.1 Advances for Configuration Design

To date, applications possessing the greatest degree of automation have been for problem types where there is a high degree of well understood recurrent tasks and/or problems where new designs are a variant of existing design solutions. Often, pre-defined product

models have served as the basis for such systems. Artificial intelligence methods, commonly referred to as knowledge based engineering (KBE), have been applied in constructing relationships representative of product structure. Rule-bases, constraint networks, and object models have been used in these efforts [Park94], [Elli92], [Grew92], [Wisd92], [Schm89]. Case-based reasoning methods, typically relying heavily on historical data, have also been applied to automating the generation of new design instances through the adaptation of existing case solutions [Bard93], [Dome94].

Kota and Lee [Kota93] present a general framework for configuration design. The authors define configuration design as a type of conceptual design activity where physical systems are synthesized from a set of *predefined* components that can be combined only in certain ways. The framework separates specifications into functions, performance goals and constraints. Decomposition is used to map functional requirements to embodiments. Design knowledge is organized into three tiers including domain-independent functions, generic devices, and catalog components. An example is offered for the configuration of hydraulic systems (HYSYN). In a manner reminiscent of numbering schemes in group technology, the hierarchical structure is maintained with subscripts; causing search procedures for single components to become combinatorial in nature.

Grigely and Billatos [Grig93] also use decomposition to break functional requirements down into a less abstract set of requirements or design requirements (DRs). Case based reasoning is used to search for single component instances meeting the given requirements. A natural language approach is used for search and case retrieval. While this does appear to be useful for organizing and retrieving devices based upon functional categories, it is in reality a search using keywords.

Murakami and Gossard [Mura92] propose a computerized method for retrieving mechanisms based upon kinematic behavioral characteristics. Locus patterns described in a two dimensional parameter space are used for matching desired characteristics to existing patterns in a library structure. The authors relay how this is a novel approach to configuration, as kinematic behavior has typically not been known until evaluation, i.e., after the model has already been created with best guess practices. Again, retrieval is for a single component instance. Moreover, all library instances are tested for the potential match; indicating a combinatorial approach.

Certainly, the potential benefits of design automation can be significant for recurrent tasks and variant design problems. Yet, as with automation efforts of any type, these systems must be applied with caution. One must be wary of the architecture used, its extensibility, and the maintenance requirements to product model data. Too often, these models have not been generalizable, so specific and burdensome to maintain, that long-term benefits have been limited.

2.2.7.2 Advances for Conceptual Design

Novel design and the early phases of product development have presented challenging problems for design automation and support tools. Still considered ill-understood, researchers have approached these problems with methods to assist model capture and synthesis using concepts for abstraction such as function, purpose, intent, and behavior. Functional diagrams, bond graphs, matrix-based methods, graph-based methods, object-oriented methods, and case-based reasoning are just some of the methods which have been investigated [Welc94], [Colt94a], [Hund92], [Gero92].

Many needs, requirements, and issues are central to the success of such efforts [Liba88], [Kutt93], [Ullm94], [Tayl94], [Bree93], [Hoov94]. Continuing research topics include:

- top-down design support
- methods to work in abstraction for conceptual design
- methods for abstraction in geometry
- means to specify function
- capture of functional relationships
- ability to work from multiple functional viewpoints
- methods for mapping function to form
- capture of product evolution information
- degree of knowledge formalization
- non-routine designs and first principle methods
- design intent inference methods
- how to evaluate the usefulness of support systems.

Current literature abounds with researchers working to address these topics. The discussion which follows will highlight some of these investigations. The reader is also referred to other worthy references in the areas of functional modeling and conceptual design including: [Salu94], [Sun94], [Rind88], [Yosh93], [Schu93], [Fave93], [Fauv94], [Welc92], [Kota92], [Mohd94], [Yao94], [Hund90], [Hund91], [Yama94], [Schm93], [Kusi93], [Osha92], [Bahr92], [Verm95], [Rind90].

Ullman [Ullm93] views design as the evolution of information. From this perspective, the intention is to provide a model of the information developed during the design process and define data types inherent in mechanical design. Several terms are defined to organize and

understand this evolution. Common definitions are provided for *function* - "what a device should do" or "how a device should perform." *Behavior* is initially defined as "what a device does" or "how a device performs." Behavior is based upon physical laws. Operations, objects and relations are terms used to categorize information and further describe the evolution of information and the relationship between functions and behaviors. Function and behavior are then redefined using these concepts. Form, function, and behavior are said to co-evolve during the design process. Ullman offers a simple mechanical design example for a securement system for mobility aids (wheelchairs). The observations from this research appear to be useful for mechanical design. However, the classification of design data would seem to be an after-thought following a completed design solution. No real measure of utility, complexity, or study of versatility has been provided.

Welch and Dixon [Welc94] suggest a model for conceptual design which uses *behavioral reasoning* to guide the design process. Again, the design process is viewed as the transformation of information from one state to another. Conceptual design is given as a two-step process where (1) *phenomenological design* transforms a functional requirement to a behavioral description, and (2) the *embodiment design* where the behavioral description is matched to an initial physical system. *Function* is defined as the relationship between functional parameters, inputs and outputs of the system. *Behavior* is how the relationships are met in terms of physical principles and phenomena. A graph-based approach, very similar to bond-graphs, is used to facilitate reasoning (synthesis) processes. While behavioral reasoning does appear useful as a mechanism to break the preconceived links between artifacts and functions, the authors point to many weaknesses and the need for future research. Reasoning procedures are combinatorial in nature using exhaustive

search. Only single primitive behaviors were investigated. Performance factors are assigned subjectively. The connectivity among embodiments has not been considered.

Gero, Tham, and Lee [Gero92] also distinguish behavior as an important link between function and structure in design. The authors submit that behavior serves as the *platform* for reasoning between the two. A framework based upon dependency networks is provided. While this does lend structure to the information content of designs, it also requires the complete enumeration of all variables and their relationships in the dependency structure.

Szykman and Cagan [Szyk92] present preliminary findings necessary for knowledge organization in an object-oriented approach to abstract model representation. Knowledge is separated and then related in a tree-like structure according to type. Knowledge types include: system-specific, domain-specific, graph management, connectivity, and non-connectivity knowledge. In addition to categorizing knowledge types, the authors propose design refinement through decomposition and the use of subgraphs. Capabilities and limitations have not yet been studied.

Colton and Ouellette [Colt94] combine two commercially available software tools (ICAD and GBB) for a knowledge based approach to what is described as the conceptual stage of mechanical design for an automobile. A pre-defined product model serves as the basis of this system. Users enter preference information either in explicit numerical form or through the selection of limited alternatives. The blackboard software (GBB) then configures the resultant vehicle based on these preferences and an encoded rule-set. The user is alerted to conflicts between feasibility and desired selections. The system suggests

alternatives based upon the relative importance given to aspects such as Transport, Please, and Protect. It appears that this approach is best suited to configuration processes and variant design types which can be described parametrically.

Chakrabarti and Bligh [Chak94] also work in the domain of mechanical conceptual design. A knowledge representation is offered for functional synthesis procedures. Emphasis is given to the transmission and transformation of mechanical forces and motion. The authors compare natural-language and mathematical representations of function. Strengths and weaknesses of each approach are discussed. Natural-language is given to be non-mathematical and therefore lacking precision and objectivity. However, natural-language does offer a representation which is close to the way designers express their ideas. Mathematical representations of function are expressed as the transformation between input and output and are formalizable. Further, the authors maintain that mathematical representations are better suited to a computational environment. However, it is also observed how for a man-machine environment, this type of representation would have to be mapped into a natural-language representation before any general functional reasoning support environment could be developed. Clearly, this speaks to the shortfall of many research investigations which never consider the end-user, i.e. the designer. For their approach, the authors use a vector-based representation to describe functional transformation.

Hundal and Langholtz [Hund92] place a greater importance on the user interface and organization of knowledge; presenting an implementation in an X Windows environment. An interactive modeler is proposed for the creation of functional descriptions. Functions are classified into five basic categories expressed as verbs: (1) store/supply, (2)

connect/branch, (3) channel, (4) change magnitude, and (5) convert. A functional structure is then built through a building-block approach using these verbs. A solution structure is then generated replacing the verbs in the structure with simplified representations of the general category of device required. From the designer's perspective, the potential benefit of such a pictorial editing system could be great. There are, however, significant needs that have not been addressed in this research. First, all possible solutions for each function are explicitly enumerated in the database; creating a maintenance burden and system which is not conducive to new applications for device and subassembly types. Second, there has been no provision for representing the important parameters which characterize the solution. And finally, the use and integration of the model with other stages of the design process are described as incompatible. Thus, there is a distinct separation between abstraction and detail.

In summary, varying success has been achieved in research endeavors for conceptual design. Challenging problems have been encountered as methods are sought to incorporate scientifically based engineering relationships, first principles, while still allowing for the ill-understood transformations so typical of early design. Much has been learned through the study of information and knowledge in design. To date, many of these investigations have been limited in both their breadth of application and their interconnectedness to an overall product realization process. Moreover, in a manner similar to systems for recurrent or variant design, systems have often not been generalizable, combinatorial in nature, or difficult to extend.

2.2.8 Mechatronics and Integrated Design

With the exception of VLSI, most research for computable methods to assist or automate engineering design has focused on mechanical design processes. Those who have suggested functional representations which include other domains, have most often only made mention of their consideration; still providing examples from the mechanical design domain. Further, there has been little done to address design problems where desired function is achievable from a variety of engineering perspectives. Consider how for technologically sophisticated products, it is possible to implement a required function with solutions in any one of the mechanical, electrical, or software domains. More likely, solutions are a combination of these disciplines. Relatively new, the area known as mechatronics is focused on addressing such issues.

Bradley *et al.* [Brad93] describe how individual specialists tend to seek solutions in their own particular domain of expertise. A computer-based support tool, Schemebuilder, is proposed as a mechanism to suggest possible solutions across a range of technologies. A representation similar to bond graphs is used to describe individual objects both by function and in terms of a series of "ports." In a manner similar to Hundal and Langholtz [Hund92], Schemebuilder provides the designer with a visual editor to place and connect objects reflective of function. An extensive library is maintained for the designer to access in the selection process. No provision is given for design evaluation such as costs, size, or other life-cycle issues.

At a much finer level of detail, Staton *et al.* [Stat93] report on their computer aided method for the design of permanent magnet DC motors. The fundamental equation for machine design is used to relate gross output torque of a motor to the armature volume,

specific magnetic loading, and specific electric loading. Four databases are used to store dimensional, winding and control parameters for design specification. This environment, like many others in mechatronics, focuses on detailed phases of design. Clearly, for the given domain of motor design, this parametric approach coupled with analysis and simulation would shorten product development cycle time.

In what is closer to an effective use of engineering analysis, Kott *et al.* [Kott94] examine design optimization of the power stage design of a power converter. Electrical, loss and thermal analyses are combined into a single system design problem. Following the results of analysis programs for each individual design subspace, a system level problem using a large matrix-based formulation is optimized for minimum weight.

2.2.9 Engineering Analysis Techniques

While not formally surveyed in this review, a discussion on computer-based design tools for engineering design would not be complete without some mention of engineering analysis techniques. Certainly, analysis techniques have played an important role as researchers have sought to improve quality, product performance and reduce design cycle times. The speed and accuracy of analysis methods have reduced the requirements for physical prototypes, have assisted the optimization of design parameters, and have permitted the consideration of numerous alternatives which would ordinarily have been prohibitive with short development deadlines. These benefits are particularly evident for industries such as aircraft manufacturers where material type, weight, and shape have a direct and measurable impact. With ongoing research, finite element analysis (FEM), thermal analysis, fluid dynamics, and other analysis methods are increasingly more capable,

accurate, and easy to use with better computing capabilities, graphic representation, and/or interfaces to CAD modeling systems.

2.3 Summary

The discussion overviewing engineering design and technology for design has covered a wide variety of topics. The intent in this broad coverage has been both for background and review purposes. It is hoped that in doing so, an appreciation has been gained for engineering design and the approaches that have been investigated to improve its methods.

Almost certainly, the debate among researchers over whether methods should focus on approaches that are purely scientific in nature or whether there is benefit to heuristic methods will likely continue indefinitely. In either case, it is clear that much is to be gained from advances pertaining to knowledge organization, model representation, and synthesis methods.

Weaknesses will always be present in engineering design support tools because, inherently, computers are limited to what they have been programmed to do, the information and knowledge sources they possess, and the sometimes awkward or cumbersome interfaces to which they are limited. Nonetheless, it is the belief of many, that the potential benefits of computer-based tools for engineering design far outweigh these limitations.

Designers need better methods to cope with complex systems. Research is still needed to assist modeling in the early stages of conceptual design. Beyond models for abstraction (functional model representations), methods are also needed to bridge the gap between

abstraction and detail, i.e., methods that support informed decision making and integrated design processes.

Clearly, the context in which a device exists affects its behavior [Brad91]. Hence, behavior must be coupled with knowledge of context. The application domain necessarily defines and restricts both the set of functions representative of intent as well as the set of relevant embodiments. Experienced designers understand the domain in which they operate. Computers, as well as inexperienced designers, must be explicitly informed and/or taught. The 'all knowing and good for every situation' design system is unlikely to ever exist. To be useful, design systems must be relevant and customizable to a given application domain.

Contrary to ideal models of design processes, complex problems are not 'solved' in abstraction and then passed on to a more detailed level of design. In addition to evaluating feasible functional alternatives, other factors such as cost, size, availability, quality, assembly, the age of the technology, and numerous other life-cycle factors require consideration early in decision processes.

Described with more detail in subsequent chapters, this research presents a new framework capable of bridging the gap between abstraction and detail; blending top-down and bottom-up approaches in an integrated methodology for conceptual design. Several features of this framework distinguish it from the work of others. First, rather than presenting a system that is specific to an application domain, the framework offers a design environment that is customizable. Although a prototype problem is included from the domain of power conversion systems, the framework provides the means to collect,

classify, and apply meaningful knowledge from any application domain. At the other end of the spectrum, the framework does not suffer from lost usefulness, as have many other systems, in the pursuit of *the* generic modeler based solely on first principles of science.

Second, the object-oriented architecture of the framework prevents the combinatoric nature of solution synthesis and knowledge collection that is often experienced by other systems. Expert knowledge is separated (logically and physically) from components knowledge and parts data bases.

Finally, and perhaps most significant, the proposed framework provides the versatility necessary for an integrated approach to the modeling tasks and information needs of early development processes. It overcomes the shortfalls that are often experienced with the exclusive practice of either a top-down or a bottom-up approach.

3. *Conceptual Modeling System - Functional View*

As described previously, the primary purpose of this research is to develop a framework blending top-down and bottom-up approaches to engineering design for an integrated design methodology. With an emphasis on the early stages of product development, three mechanisms have been identified that are integral to such a framework, including a:

- functional object modeling environment,
- component knowledge-base, and an
- integrated design domain.

Figure 9 offers a pictorial view of this integrated conceptual design environment. Central to the environment, the conceptual modeling system (CMS) supports design processes in four important ways. Listed here, these include capabilities to:

1. create and visualize a conceptual model representation,
2. configure solutions that meet functional requirements (i.e., synthesis),
3. collect and re-use knowledge, and
4. interface with tools and data necessary for analysis and evaluation.

Based upon an object-oriented paradigm and semantic reasoning, the framework has been designed in detail. A windows-based prototype has also been developed to demonstrate the conceptual modeling system. This chapter describes the framework and resultant system developed for conceptual design from a *functional viewpoint*. Having applied a *use case approach* to system development, a brief introduction to use case modeling is first offered. Next the use cases and accompanying scenarios that describe the conceptual

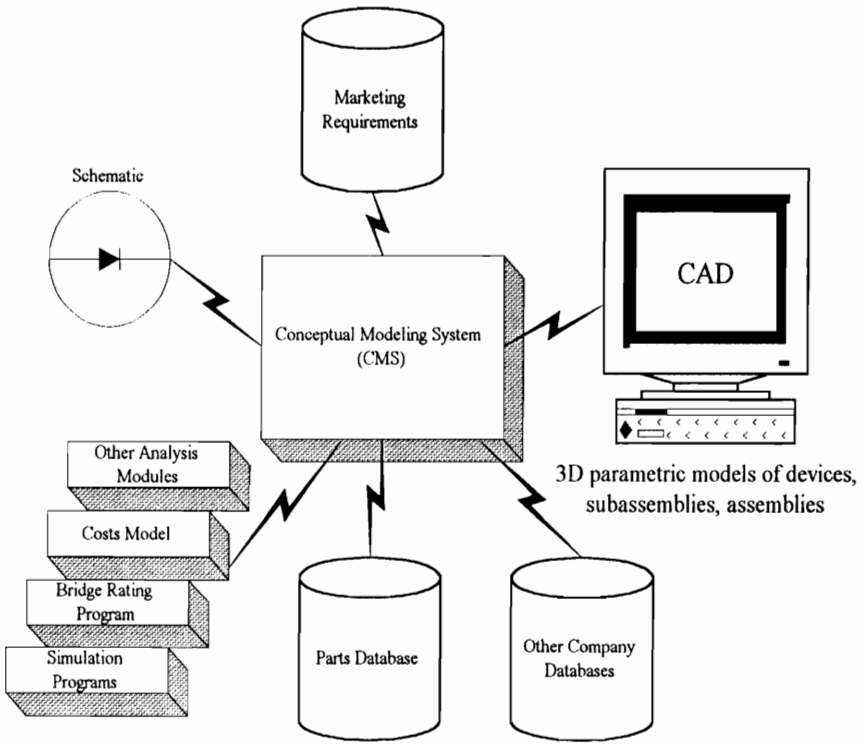


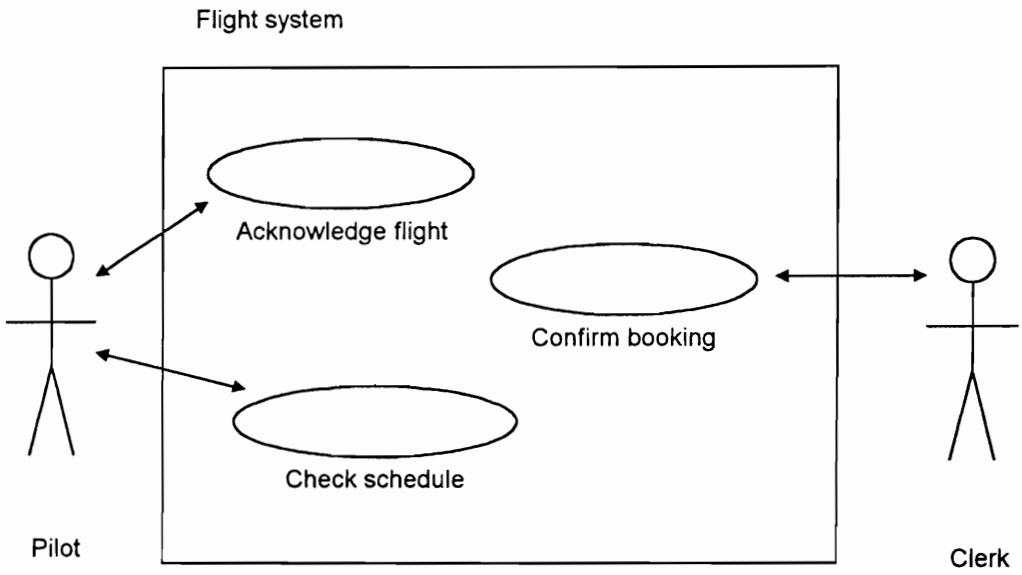
Figure 9. Conceptual Design Environment.

design framework are presented. Visuals (screen shots) of the prototyped system for conceptual design are included as figures accompanying use case scenarios. An architectural description of the framework follows in the next chapter.

3.1 Use Case Approach

The use case approach, as the name would imply, centers around the description of a system in terms of its intended uses. Several *use cases* and *actors* are typically needed to define a *use case model*. Actors define what exists outside the system and use cases define what should be performed by the system. Each use case focuses on a particular function that must be performed by the system. Actors represent particular roles that a user can assume when interacting with the system. A simple example of a use case model for a flight system is depicted in Figure 10. Using Jacobson's notation [Jaco94], the system is bounded by a box. Each actor is represented by a stick figure outside the box and use cases are represented as ellipses contained within the box. For the flight system example, the Pilot actor may use the system to acknowledge flights or to check schedules. Clerks may confirm bookings.

Each use case in a model can be described with one or more *scenarios*. A scenario is a single thread through the system (i.e., a particular instance of the use case) and describes the time-ordered sequence of events between actors and the system. The scenarios included in a use case are not exhaustive, but focus on the fundamental uses required to describe the boundaries of the system. Collectively, the set of all use cases specifies system functionality (a requirements model) and provides the basis for the remaining stages of the development process (analysis, design, implementation, and testing).



Actors:	Pilot, Clerk
Use Cases:	Acknowledge flight Confirm booking Check schedule
System:	Flight System

Figure 10. Example use case model [Jaco94].

In this research and development effort, it was found that the use case approach to object-oriented modeling and system development was intuitive and provided a good foundation for developing the system prototype and architectural model.

3.2 Conceptual Modeling System - Use Case Model

This section presents the functional view of the conceptual modeling system by way of a use case model description. As shown in Figure 11, system functionality can be described in terms of six primary use cases. Briefly, the first use case is concerned with modeling actions required to build a conceptual model. The second use case is concerned with actions required to maintain the building blocks of the system. The third use case is also concerned with maintaining the modeling environment; in this instance focusing on the relationships available for relating the function building blocks placed in a model. The fourth use case deals with the actions needed for providing requirements specification data to the system. The fifth use case addresses the actions related to solution synthesis (configuration and reporting). The sixth, and last use case, is concerned with interfacing from the modeling environment to tools and data that exist outside its definition.

A more detailed treatment of the six use cases is contained in the sections that follow. For each use case, an overview description is offered first followed by the scenarios required to specify system functionality. As appropriate, visuals from the conceptual modeling system are included for clarity within scenario descriptions.

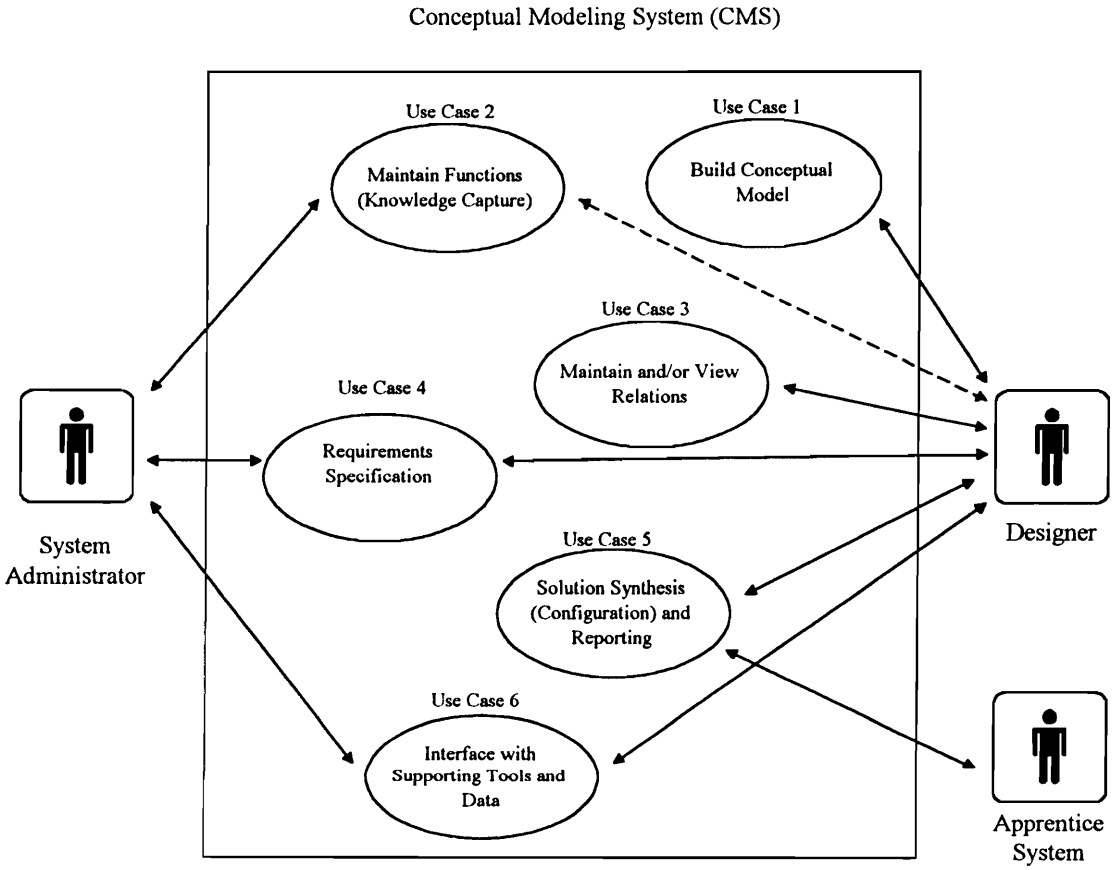


Figure 11. Use Case Model

3.2.1 Use Case 1: Build Conceptual Model

The scenarios of this use case describe the interactions required to build (compose) a conceptual model. A screen work-area provides the graphical palette for a building-block approach to conceptual modeling. Within the work-area, design objects (functional requirement building blocks) may be placed, connected, related, and constrained. Building blocks are available for selection from function category templates. At the discretion of the designer, building blocks placed in the model may be of a specific solution type available within the function category, or alternatively may be generic in nature (i.e., specify the function category name).

Two methods for locating functional building blocks within category templates are also described (keyword match and list selection). In addition, scenarios are included for the actions required to delete functions or relations from the conceptual model, and for file save actions (permitting the exit and return to an in-process model during a later design session).

3.2.1.1 Scenario 1: Locate desired functional object (keyword match) and open function group

- The user selects the Function Category button (shown as a hierarchical graph) from the toolbar with the mouse (Figure 12).
- The system displays the available function categories listed in alphabetical order. In addition, the system displays buttons labeled **OK**, **ADD**, **DELETE**, and **CANCEL,** and a type-in area to enter the function category name desired.
- The user does not use the alphabetized list of functions, but instead enters in a word in the type-in area describing the desired function.

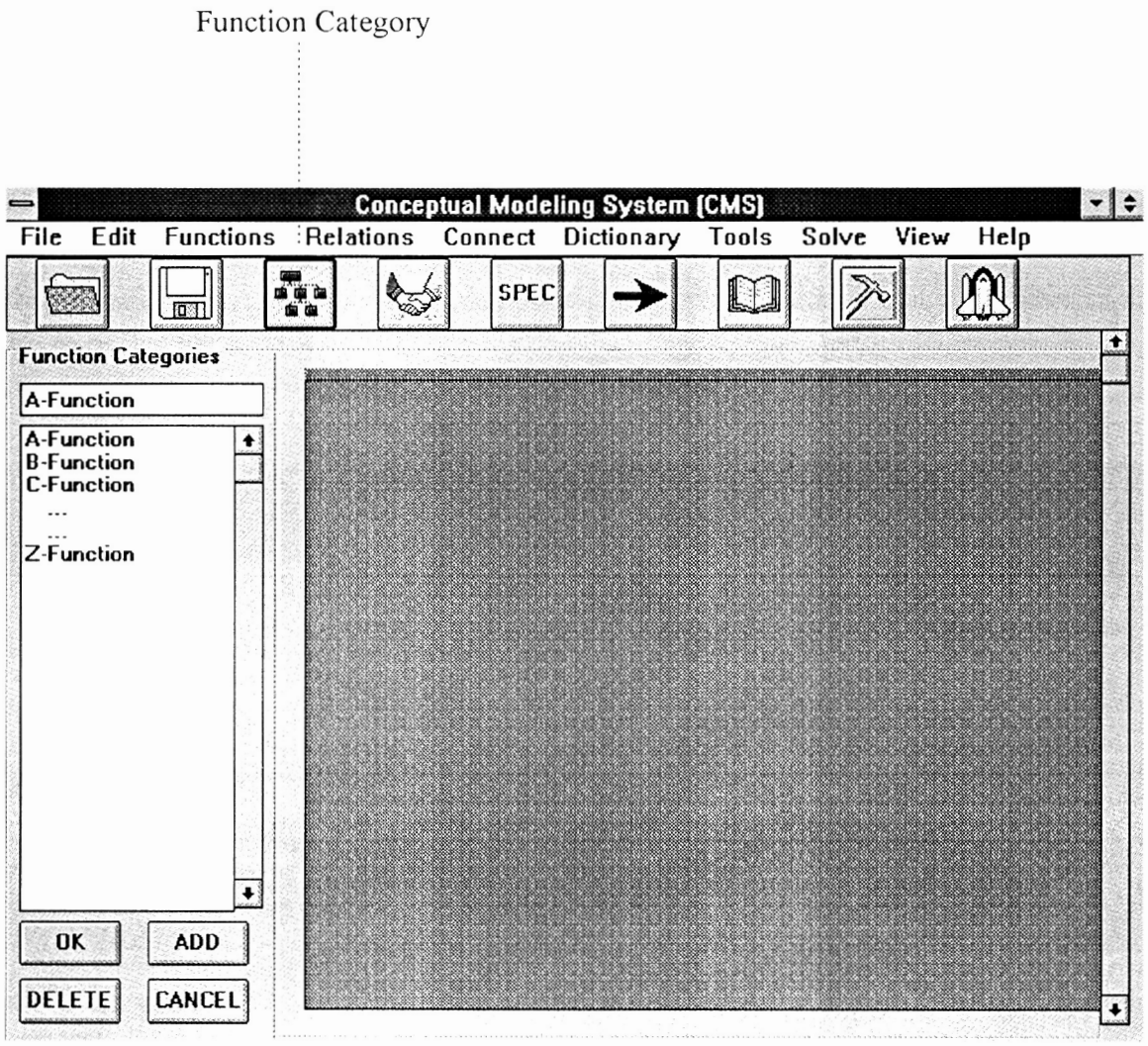


Figure 12. Function Categories

- The system updates the display with the list positioned at or following the function matching the type-in (still an alphabetically ordered list).
- The user highlights the selection and presses **OK.**
- The system opens and displays the function category template for the selected function (Figure 13).

3.2.1.2 Scenario 2: Locate desired functional object (selection from function group list) and open function group

- The user selects the Function Category button with the mouse.
- The system displays the available function categories listed in alphabetical order. In addition, the system displays buttons labeled **OK**, **ADD**, **DELETE**, and **CANCEL**, and a type-in area to enter the function category name that is desired.
- The user scrolls the list of functions until the desired function category is reached.
- The system updates the list as required by the user actions.
- The user selects the function category desired and presses the **OK** button.
- The system displays the function category template selected.

3.2.1.3 Scenario 3: Select and add a functional object to the model

- The user highlights the desired function from the Solution Options displayed and drags it to the desired location in the conceptual model. The selection may be either the general function category or a specific type of solution from the options template.
- The system displays the updated conceptual model to reflect the addition of the selected function to the model.

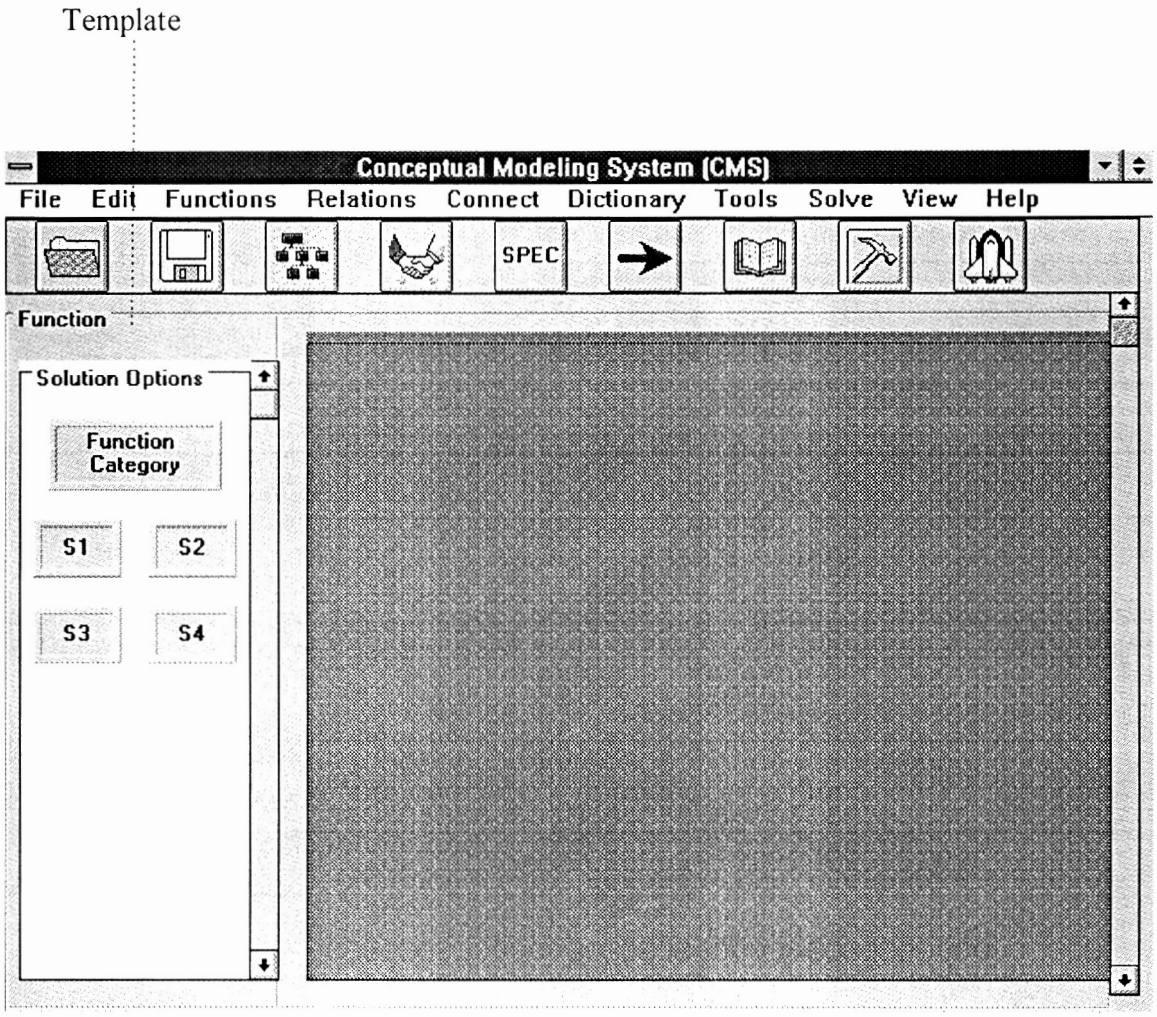


Figure 13. Function Solution Template

3.2.1.4 Scenario 4: View and/or constrain the parameters of a function in the model

- The user double clicks on the functional object of interest within the conceptual model work area.
- The system displays the current data related to the function (Figure 14).
  (Note that the user does not need to indicate whether the function is atomic. The system will display the correct data as appropriate.)
- The user specifies constraints on attributes in the unprotected screen fields and presses **Enter**.
- In the event that the user is unfamiliar with the naming and/or syntax of attributes as established within the data dictionary, the user selects **Dictionary**.
- The system displays a list (alphabetical) of attribute names currently in the dictionary.
- The user highlights attribute selections and presses **Enter**.
- The system adds the attributes to the current function data.
- The user enters values for attributes as desired and presses **Enter**.
- The system updates the model, closes the data screen and redisplays the conceptual model in the work area.

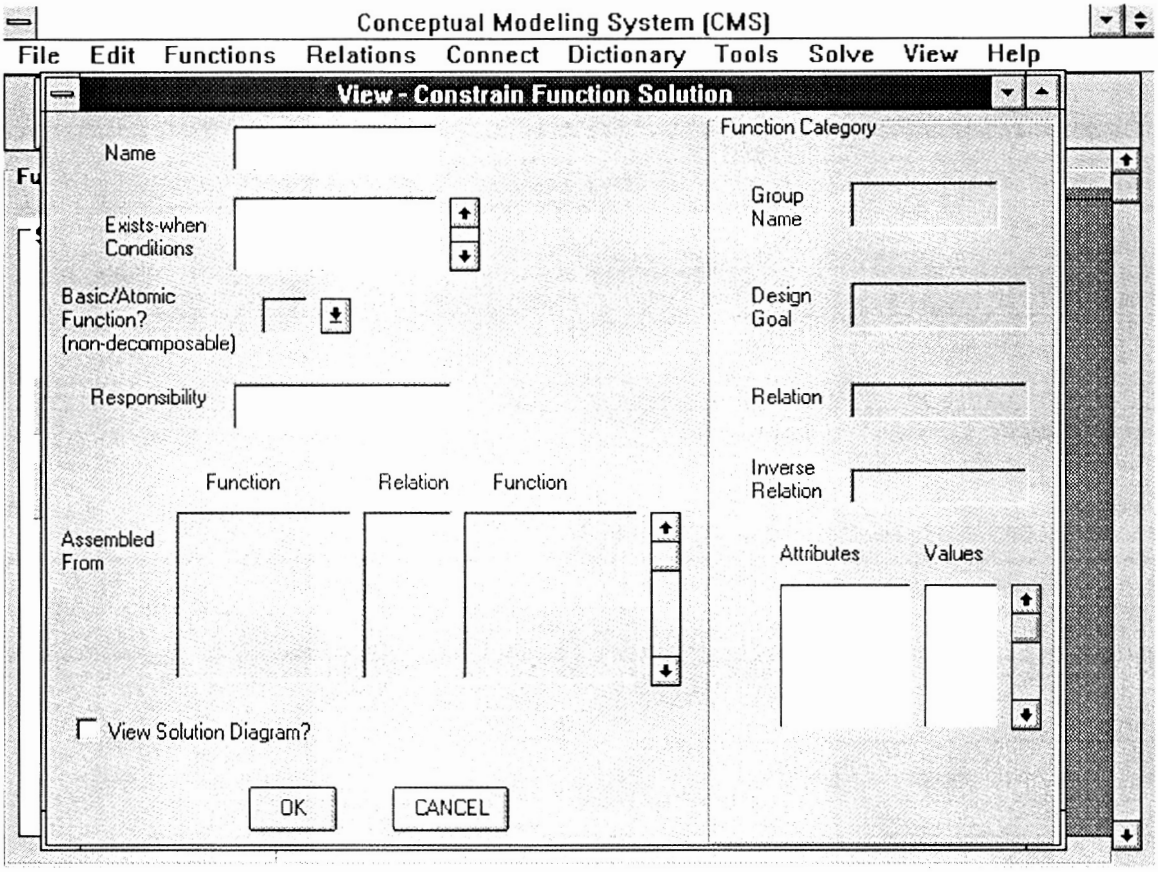


Figure 14. Function Data.

3.2.1.5 Scenario 5: Create a relationship (connection) between functional objects in the conceptual model

- User selects two functions in the conceptual model to connect; highlighting selections with the mouse.
- User clicks on the Connect button (shown as an arrow on the toolbar, Figure 15).
- The system displays a connecting line between the two functions with an endpoint arrow directed toward and placed at the second function selected. Additionally, the system displays the name of the relation over the line connecting the two functions. The relation name of the source function is the default name used to label the connecting relationship.
- The user may click on the Connect button a second time to toggle the direction of the connecting line.
- The system responds as required; updating the proper direction and relation name.
- The user double clicks on the connecting line to rename the relationship connecting the two objects (not accepting the default).
- The system displays the list of current relations available.
- The user highlights the relation desired and presses **Enter**.
- The system updates the conceptual model and displays the relation name along the connecting line.

3.2.1.6 Scenario 6: Delete function(s) and/or relationship(s) from conceptual model

- The user selects functional object(s) and connectors with the mouse, then presses the **Delete** key on the keyboard.
- The system updates and redisplays the conceptual model.

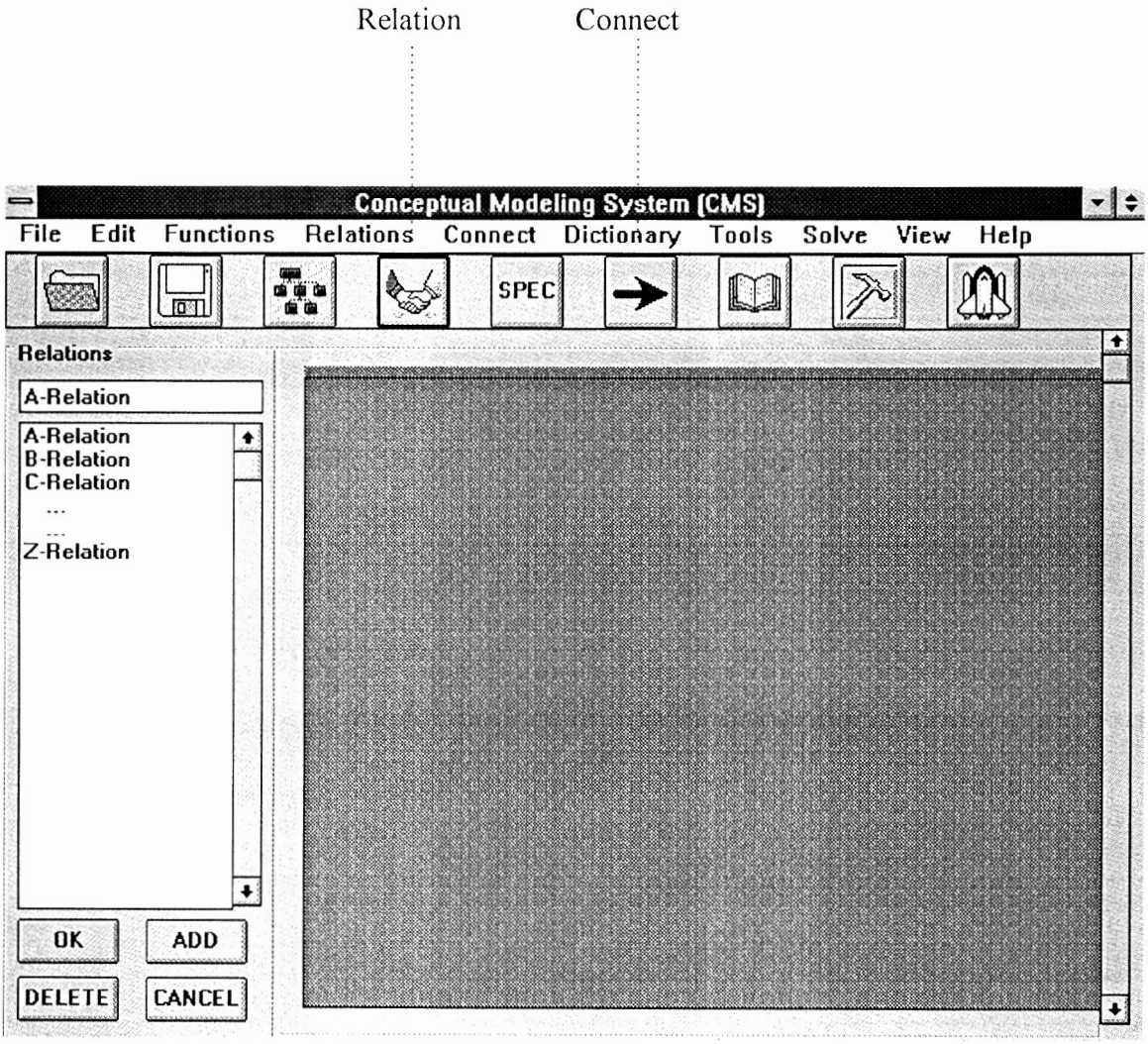


Figure 15. Relations.

3.2.1.7 Scenario 7: Save Conceptual Model (in-process model)

- The user clicks on the save button (shown as a disk) from the toolbar with the mouse (Figure 16).
- The system displays a screen for the user to enter file information for the save action.
- The user provides the file name (including directory specification) and presses **Enter**.
- The system saves the conceptual model (including accompanying data that is specific to the model under development).

3.2.2 Use Case 2: Maintain Functions (Knowledge Capture)

The scenarios of this use case describe the interactions related to the maintenance of function categories (i.e., the templates defining the building blocks available for modeling). This includes adding a new function solution to an existing function category as well as the creation of new categories. Scenarios are included that describe the addition of basic/atomic (i.e., non-decomposable) as well as higher level functions. Conceptual models built within the modeling work area serve as the source for the definition of higher level functions. Data requirements for function solutions and function categories are documented within the scenario descriptions and referenced figures included in this use case.

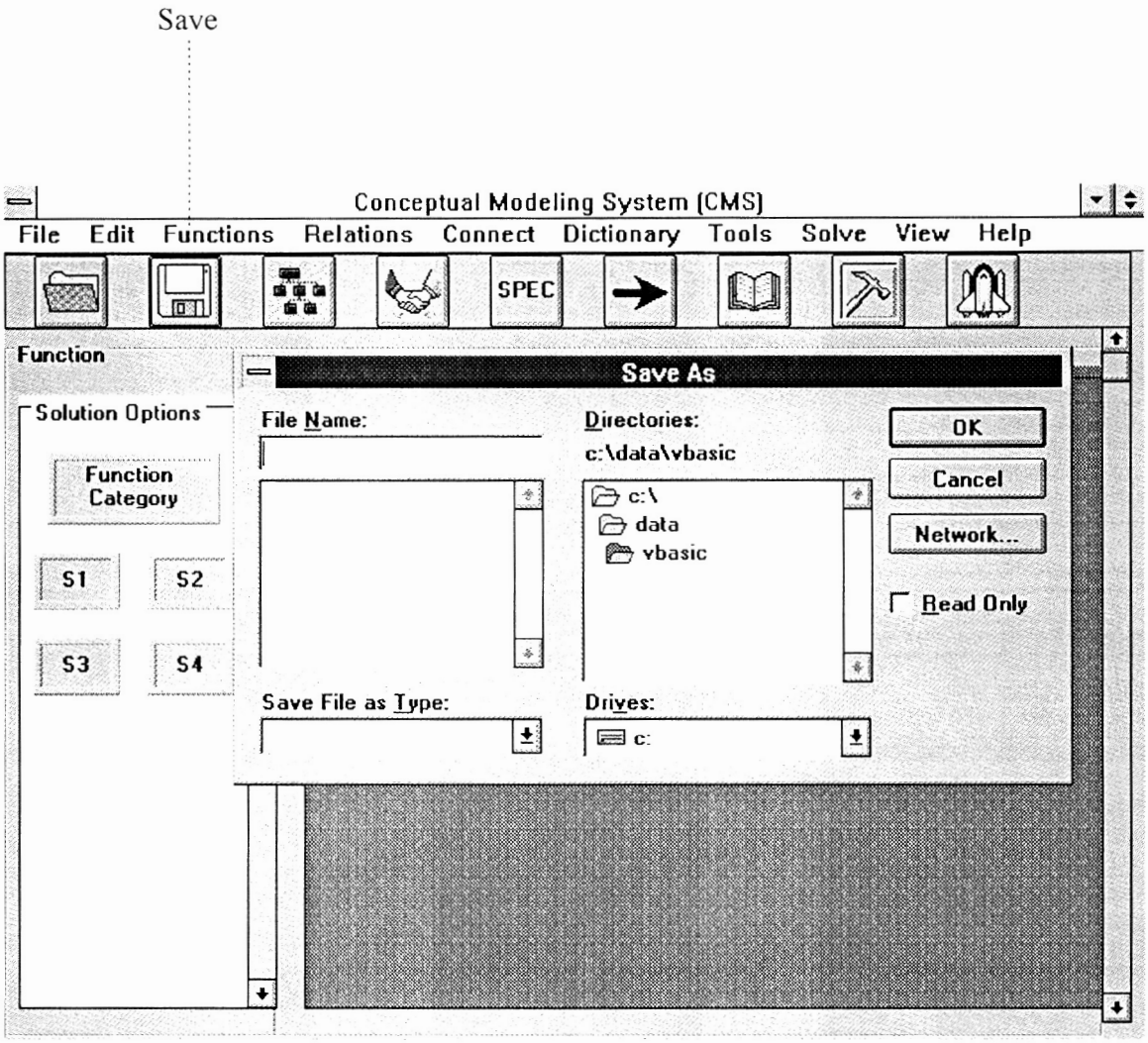


Figure 16. Save.

3.2.2.1 Scenario 1: Add a new functional solution to an existing function category (case when function is atomic, i.e., not decomposable)

- The user selects the Function Category button with the mouse.
- User locates and opens the function category where the new functional object (solution) is to be added (refer to actions of either Scenario 1 or Scenario 2 of Use Case 1 (i.e., keyword or list search)).
- After highlighting the desired function category, the user presses the **ADD** button. The system displays a prompt and selection boxes labeled **Function Category** and **Solution Option**, and buttons labeled **OK** and **CANCEL** (Figure 17).
- The user selects **Solution Option** with the mouse.
- The system displays a formatted screen (Figure 18) for entering functional object data including:

  Function name (name of function currently adding)

  Conditions for existence (exist-when parametric selection criteria)

  Attributes - displays default attributes associated with the function category (can override and/or add to, may be calculated or fixed)

  Basic/atomic function (non-decomposable) Y/N?

  Responsibility for function solution (name)

  Group data displayed in protected mode (for information only):

  Group name (current function group)

  Design goal (functional purpose) - goal of function group

  Relation name - relation exhibited by members of function group

  Inverse relation name - inverse relation for function group

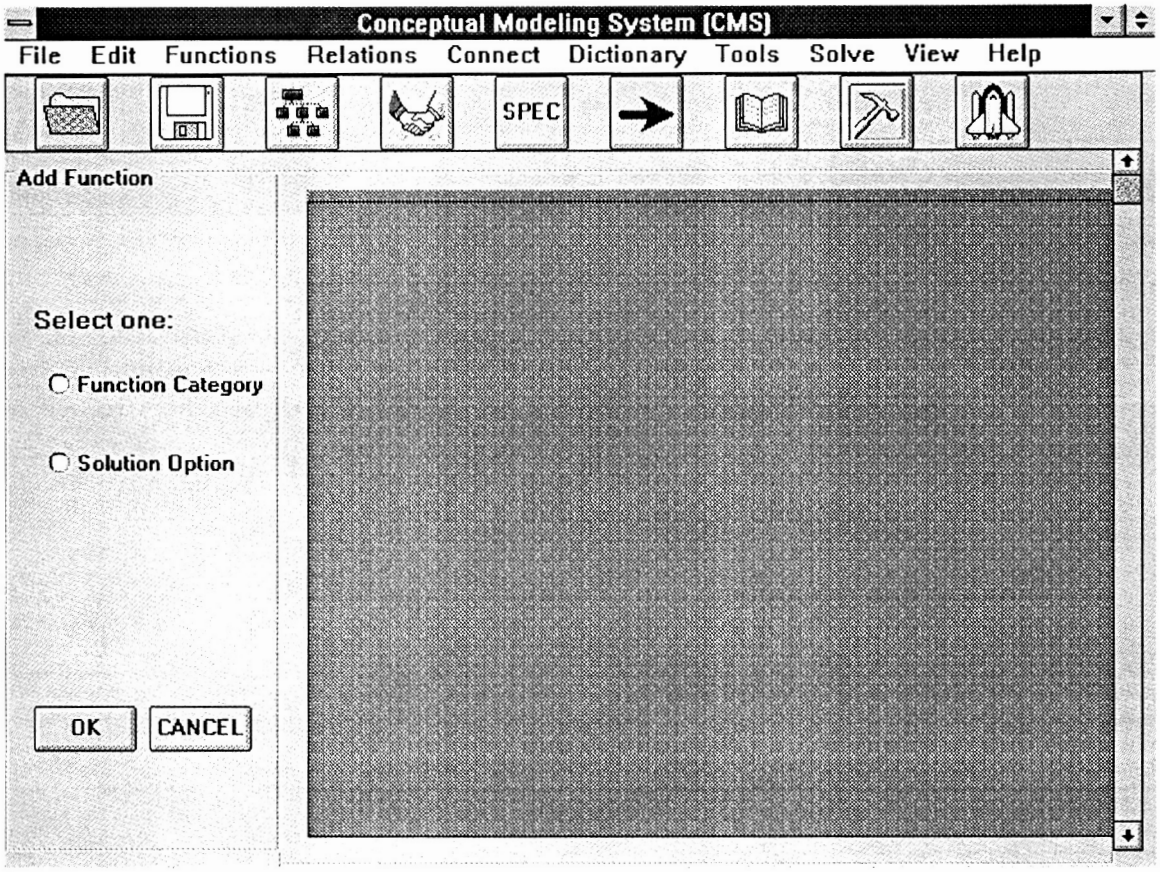


Figure 17. Add Function.

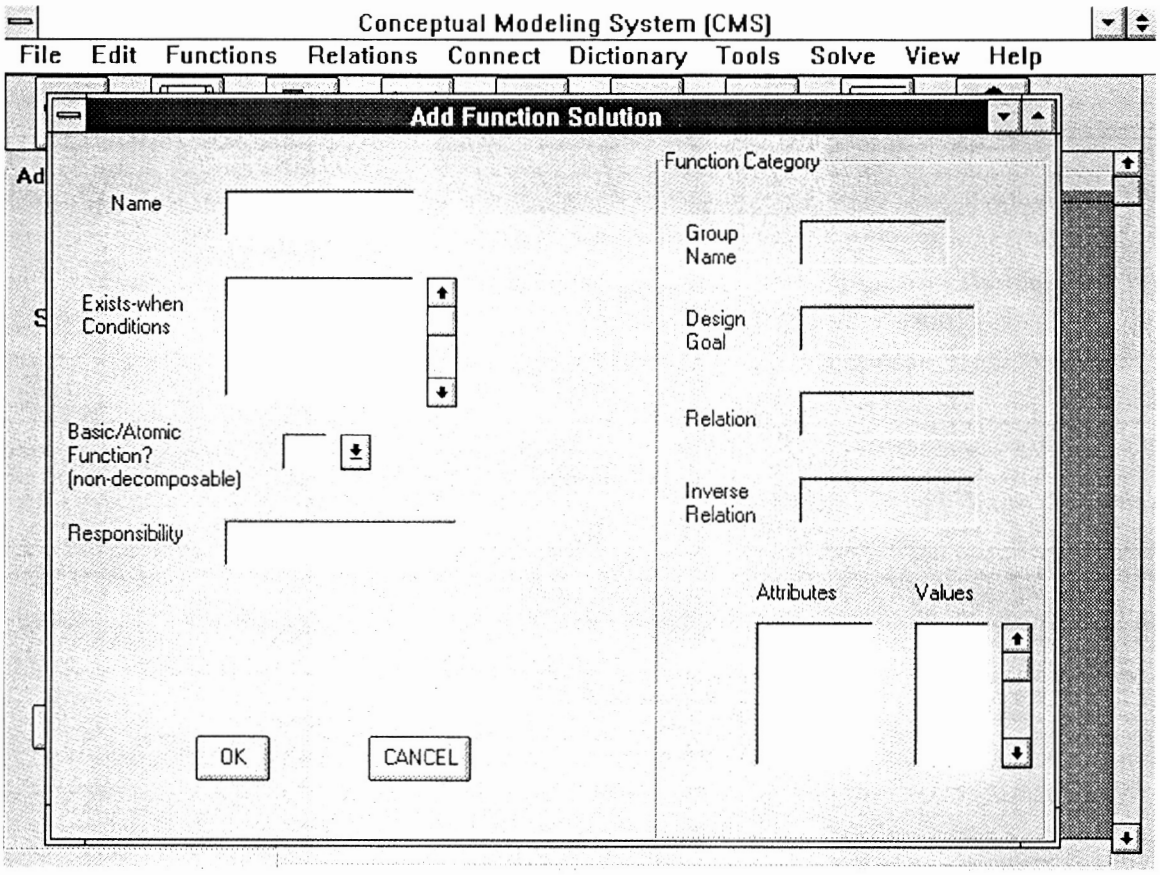


Figure 18. Add Function Solution (Atomic Case).

- The system also displays buttons labeled **OK** and **CANCEL**.
- The user fills in the template (for this scenario entering **Y** to indicate the addition of an atomic function solution) and presses **Enter**.
- The system adds a new icon to the function category template displaying the name of the new function solution and updates the data structure (Expert Model and Engineering Model).

3.2.2.2 Scenario 2: Add a new function solution to an existing Function Category (case when function is not atomic, i.e., decomposable)

- User has prepared a new conceptual model that is current in the screen work area (perhaps retrieved from a file saved from a previous conceptual model session).
- This scenario follows the same actions as in Use Case 2 for Scenario 1 through the display of the formatted screen for entering functional object data.
- The user fills in the template (for this scenario entering **N** to indicate the addition of a non-atomic function) and presses **Enter**.
- The system displays the message "Highlight desired functions and relations for new function, or press **Select All** to select the entire model" along with buttons labeled **Select All**, **OK** and **CANCEL** (Figure 19).
- The user presses **Select All**.
- The system highlights the entire conceptual model.
- The user presses **OK**.
- The system adds a new icon to the function category template displaying the name of the new function solution and updates the data structure (Expert Model).

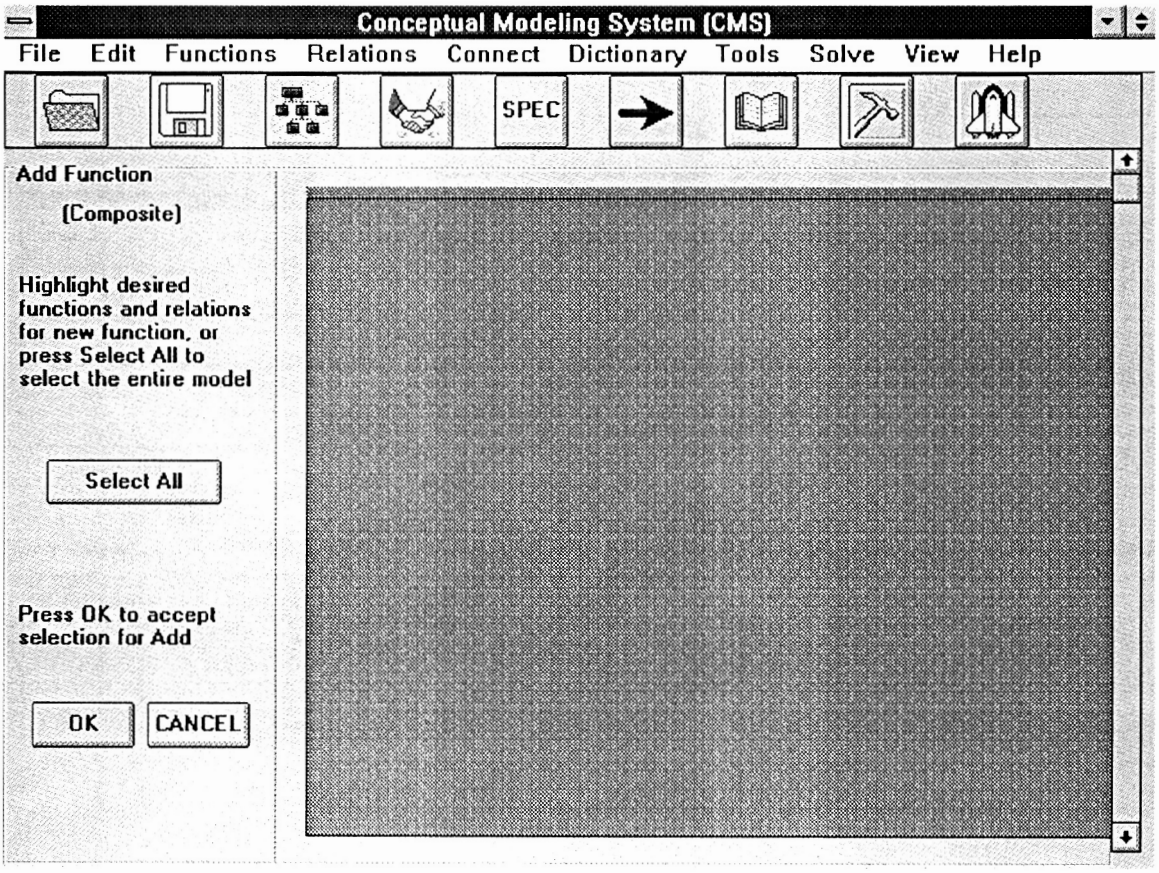


Figure 19. Add Function (Composite).

3.2.2.3 Scenario 3: Create a new Function Group Category

- The user selects the Function Category button with the mouse.
- The system displays the available function categories listed in alphabetical order. In addition, the system displays buttons labeled **OK**, **ADD**, **DELETE**, and **CANCEL**, and a type-in area to enter the function name (refer back to Figure 12).
- The user selects the **ADD** button.

  The system displays a prompt and selection boxes labeled **Function Category** and **Solution Option**, and buttons labeled **OK** and **CANCEL**.
- The user selects **Function Category** with the mouse.
- The system displays a formatted screen (Figure 20) for entering function group data including:

  Name (name of function category currently adding)

  Design goal (functional purpose) - goal of function

  Relation name - relation exhibited by members of function category

  Inverse relation name - inverse relation for function category (protected; automatically displayed following the selection of Relation)

  Attributes - parameters common to function solution members of this category (may be calculated or fixed)

  Responsibility for function category (name)
- The system also displays buttons labeled **OK** and **CANCEL**.
- The user fills in the data for unprotected fields.
- The user clicks on **Relation** to selected from relations available.
- The system displays the list of current relations available.
- The user highlights the relation desired and presses **Enter**.

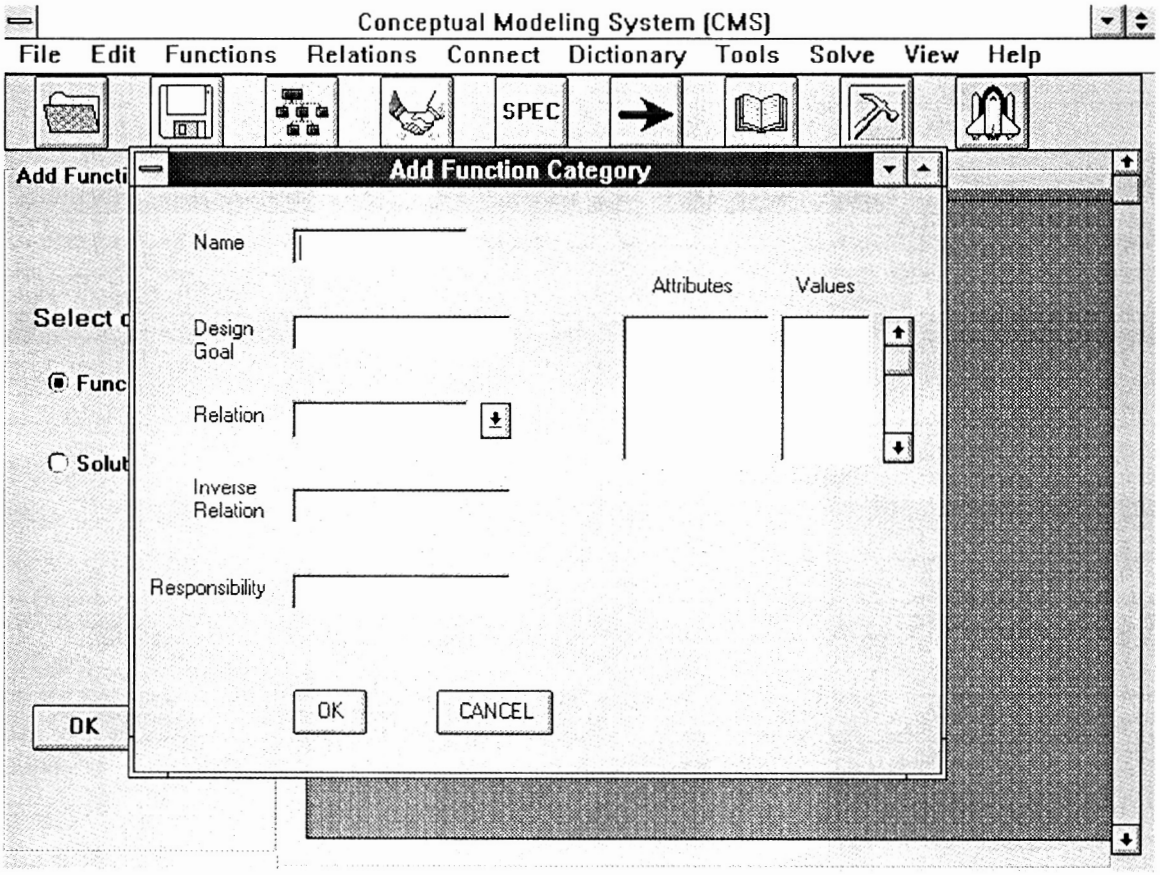


Figure 20. Add Function Category.

- The system refreshes the Relation and Inverse Relation fields.
- The user presses **OK**.
- The system adds the new function group category to the templates available and updates the data structure (Expert Model).

3.2.3 Use Case 3: Maintain and/or View Relations

The scenarios of this use case describe the interactions related to maintaining and viewing relations available for conceptual modeling. In their simplest form, relations are used to name the connection placed between functional objects in the conceptual model. A relation may also possess additional responsibilities to perform some action when placed in the conceptual model. Similar to Use Case 1 when searching for a function, locating the desired relation is accomplished either through a keyword match or through the selection from a list. Scenarios are also included for add and delete operations.

3.2.3.1 Scenario 1: Locate and view desired relation data (keyword match)

- The user selects the Relation button (shown as a hand-shake) from the toolbar with the mouse (refer back to Figure 15).
- The system displays the relations available listed in alphabetical order. In addition, the system displays buttons labeled **OK**, **ADD**, **DELETE**, and **CANCEL**, and a type-in area to enter the relation name desired.
- In this scenario, the user does not use the alphabetized list of functions, but instead enters in a word in the type-in area describing the desired relation.
- The system updates the display with the list positioned at or following the relation matching the type-in (still an alphabetically ordered list).
- The user highlights the selection and presses **OK**.

- The systems opens and displays the relation data (Figure 21).

3.2.3.2 Scenario 2: Locate and view desired relation data (selection from list)

- The user selects the Relation button from the toolbar with the mouse.
- The system displays the relations available listed in alphabetical order. In addition, the system displays buttons labeled **OK**, **ADD**, **DELETE**, and **CANCEL**, and a type-in area to enter the relation name desired.
- The user scrolls the list of relations until the desired relation is reached.
- The system updates the list as required by the user actions.
- The user selects the relation desired and presses the **OK** button.
- The system opens and displays the relation data (Figure 21).

3.2.3.3 Scenario 3: Add a new Relation

- The user selects the Relation button from the toolbar with the mouse.
- The system displays the relations available listed in alphabetical order. In addition, the system displays buttons labeled **OK**, **ADD**, **DELETE**, and **CANCEL**, and a type-in area to enter the relation name desired.
- The user selects the **ADD** button.
- The system displays a formatted screen (Figure 22) for entering relation data including:

  Relation name - relation exhibited by members of function group

  Inverse relation name - inverse relation for function group

  Description - more detailed description of relation

  Services - (future potential - user would name or select operations the relation can perform)

  Responsibility for relation (name)

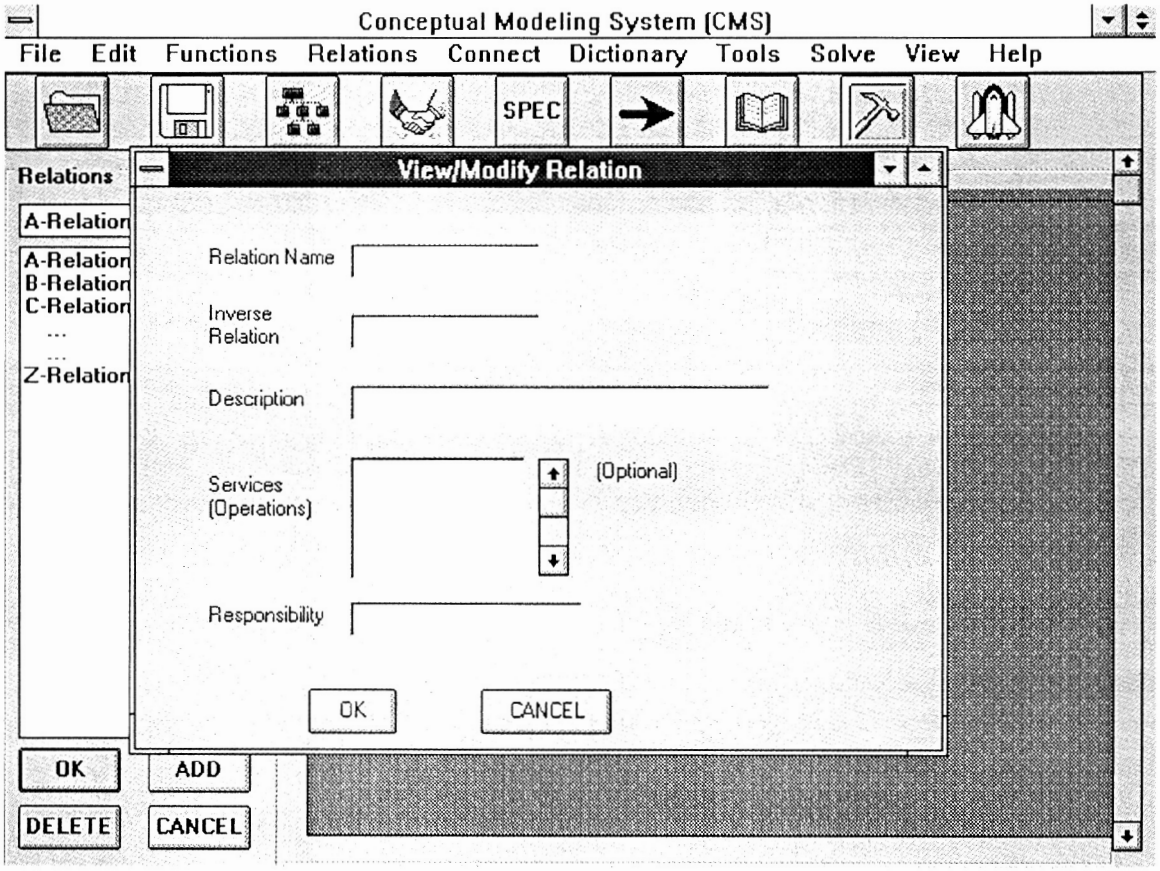


Figure 21. View/Modify Relation.

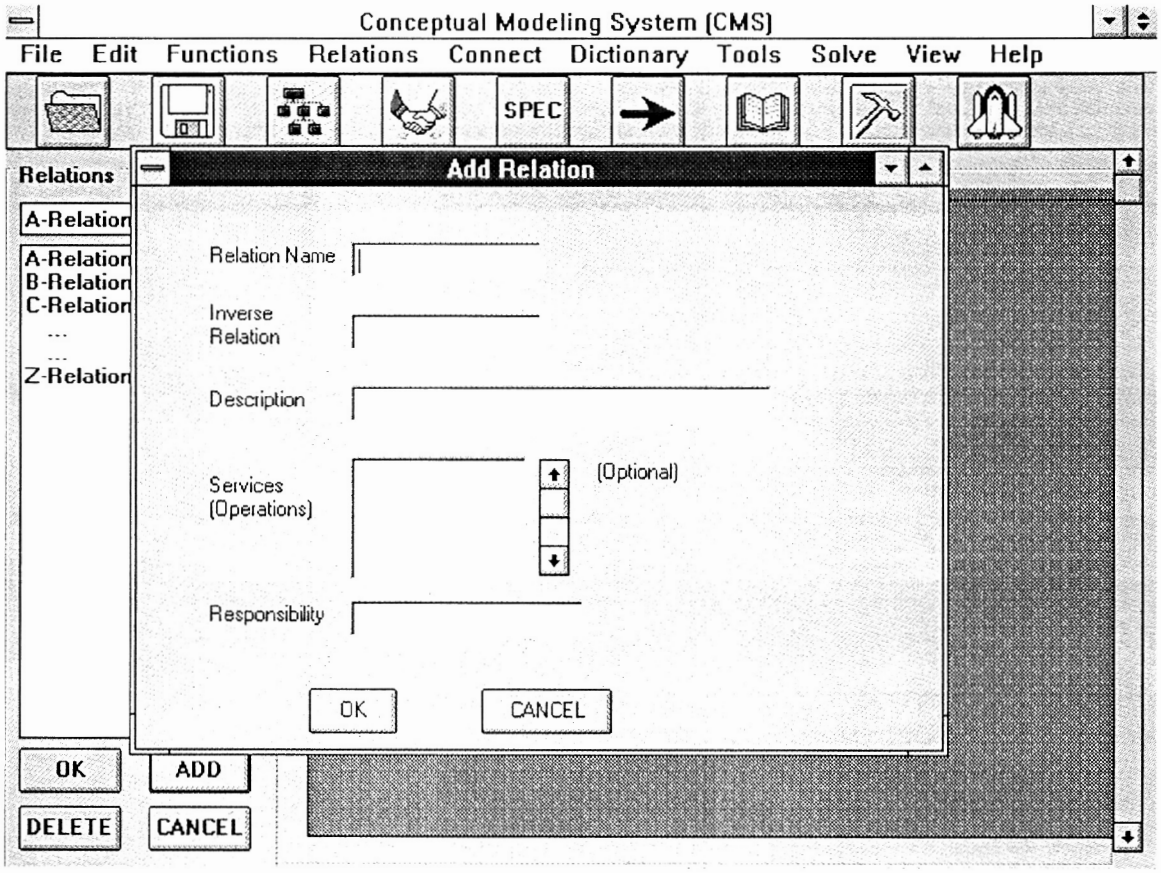


Figure 22. Add Relation.

- The system also displays buttons labeled **OK** and **CANCEL**.
- The user fills in the data and presses **OK**.
- The system adds the new relation to the list and updates the data structure (Expert Model).

3.2.3.4 Scenario 4: Delete relation(s)

- The user selects the Relation button from the toolbar with the mouse.
- The system displays the Relations available listed in alphabetical order. In addition, the system displays buttons labeled **OK**, **ADD**, **DELETE**, and **CANCEL**, and a type-in area to enter the relation name desired.
- User locates the Relation to be deleted (refer to actions of either Scenario 1 or Scenario 2 of Use Case 3 (i.e., keyword or list search)), highlights the selection using the mouse, and presses the **Delete** key on the keyboard.
- The system deletes the Relation from the list and updates the data structure (Expert Model).

3.2.4 Use Case 4: Requirements Specification

Requirements specification data provides the context for the configuration process (i.e., solution synthesis). Modeling tasks can precede the stipulation of this data, but it is imperative that the particular specification is provided prior to any requests for configuration solution. This use case includes two scenarios that describe the interactions related to defining, maintaining, and viewing requirements specification data. The first scenario describes the interactions used when specifying the particular *values* of attribute and parameter data. The actions for viewing the requirements specification data is handled in much the same manner, and is therefore not included as a separate scenario. The second scenario describes the actions required to initially define the requirements

specification template. When completed, the template (frame) includes all attribute variables required to describe the context of any modeling situation for the application domain. This is an administrative task and as such implies that once the template is established, there should be very little reason for modification. Regular users of the modeling system will make use of Scenario 1.

3.2.4.1 Scenario 1: Specify and/or view requirements data

- The user selects the Requirements Specification button (shown as **SPEC)** from the toolbar with the mouse (Figure 23).
- The system displays the formatted screen of requirements specification data and buttons labeled **OK, ADD, DELETE,** and **CANCEL**. (The template screen has been previously defined for the application domain with variables and slots alongside for values). Slots may contain data or display as empty depending on whether data has been previously specified for the current model.
- The user fills in the slots. Valid slot values may be a fixed number, calculated, specify a range, or require selections from available options.
- In the case of available options, the user clicks on the downward pointing arrow to display the list of available options.
- The system displays the list.
- The user highlights the selection.
- The system fills the slot value in with the selection.
- When all slots the user wishes to specify or restrict are filled-in, the user clicks on the **OK** button.
- The system updates the current model, closes the requirement specification screen, and redisplays the conceptual model.

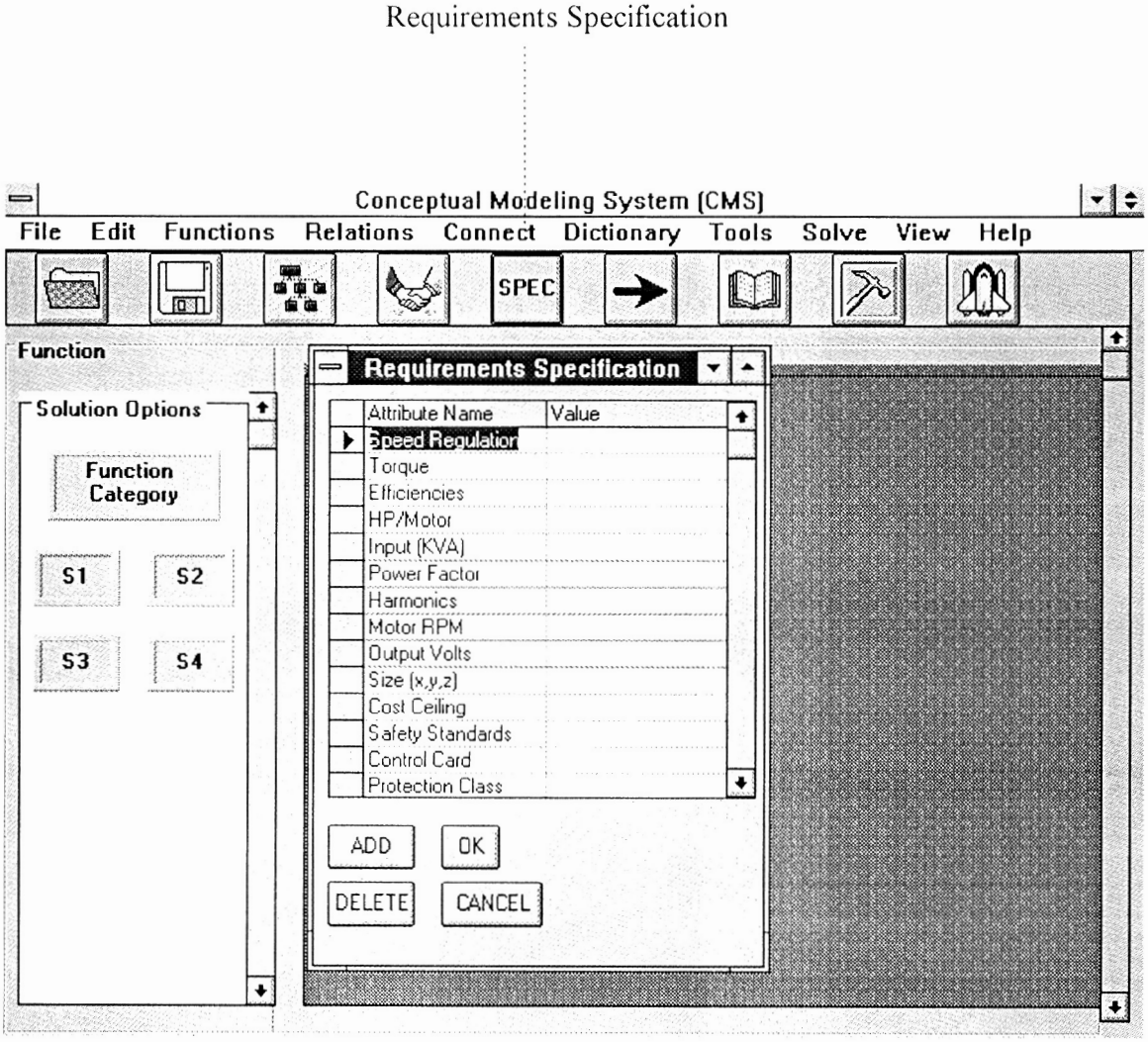


Figure 23. Requirements Specification.

3.2.4.2 Scenario 2: Define the requirements specification template

- The user selects the Requirements Specification button **SPEC** with the mouse.
- The system displays the formatted screen of requirements specification data and buttons labeled **OK**, **ADD**, **DELETE**, and **CANCEL**.
- 1 - The user selects the **ADD** button with the mouse.
- 2 - The system displays a submenu that shows a list of attributes available in the data dictionary along with buttons labeled **OK**, **ADD**, **DELETE**, and **CANCEL** (Figure 24).
- 3 - The user scrolls to and highlights the attribute to add to the requirements specification template and clicks **OK**.
- 4 - The system adds the selected attribute to the requirements template and closes the attribute submenu.
- The user repeats steps numbered 1 - 4 until all desired attributes have been added to the requirement specification template.
- When the user is not able to locate the attribute desired in the submenu, the user selects the **ADD** from the attributes submenu.
- The system displays an Add Attribute screen to add new attributes to the data dictionary (Figure 25).
- The user provides the information required: Attribute Name, Type (numeric, text, function), and Default Value (optional) and presses **OK**.
- The system adds the new attribute to the dictionary where it is now available for selection as described above.
- When all attributes for the template have been specified, the user presses **OK**.
- The system closes the Requirement Specification screen and re-displays the conceptual model screen.

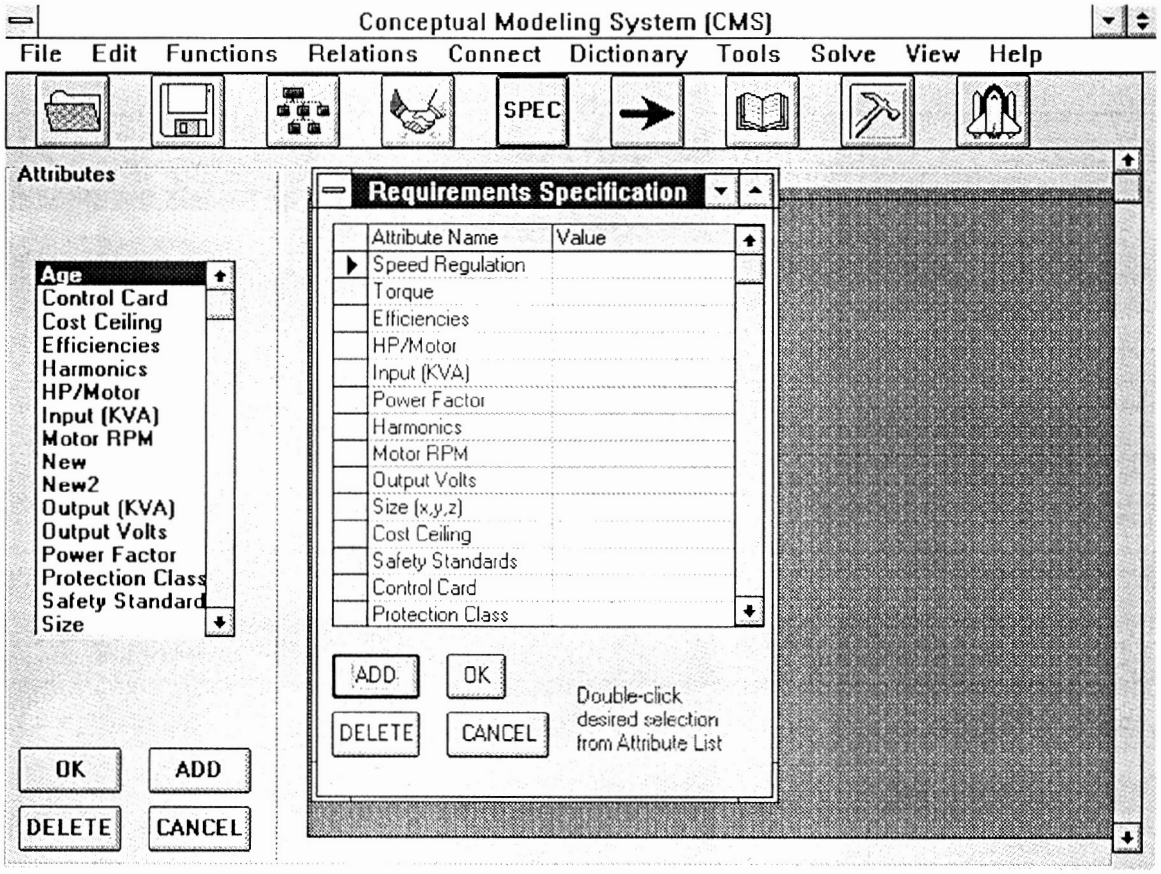


Figure 24. Display Attributes.

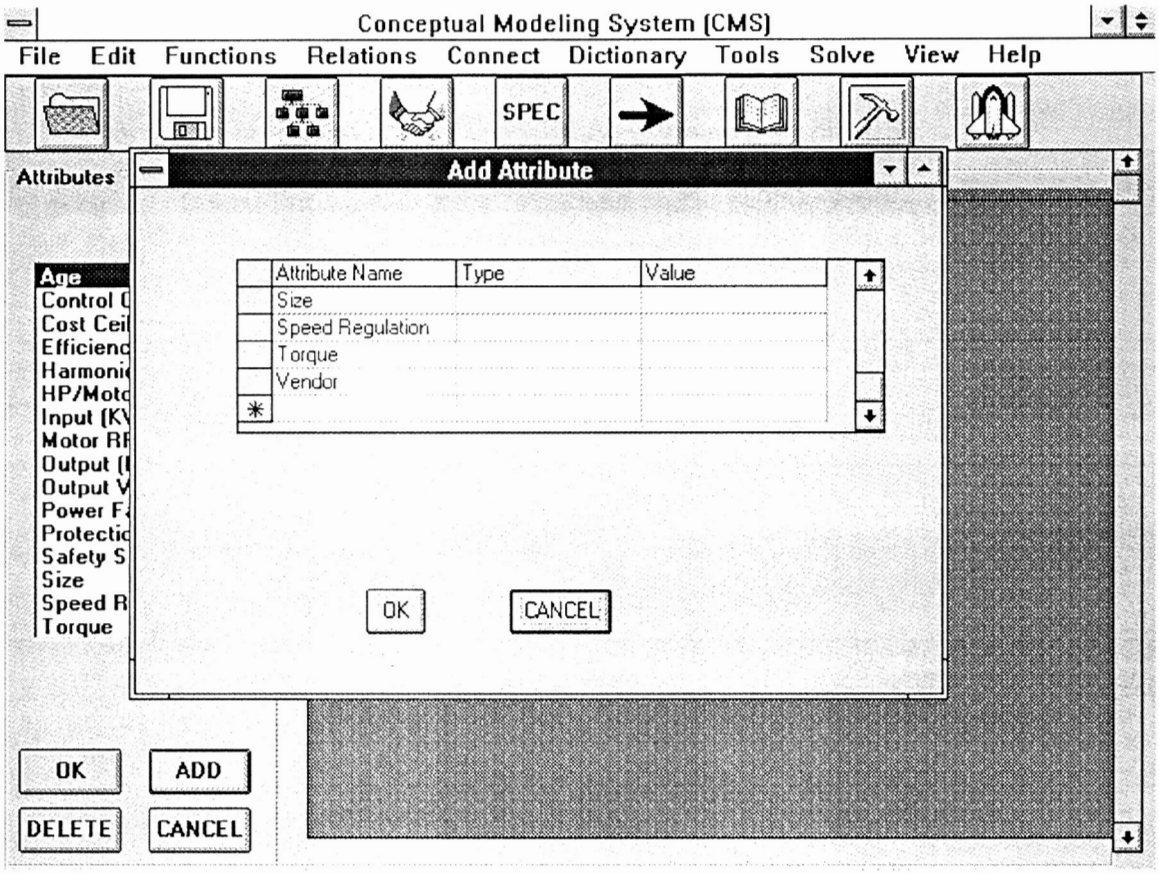


Figure 25. Add Attributes.

3.2.5 Use Case 5: Solution Synthesis (Configuration) and Reporting

Scenarios of this use case focus on the user actions and system responses associated with the solution of the conceptual model. Users of the system have the option to request a configuration solution over a subset of functions present in the conceptual model, or alternatively, choose to configure the complete model. Solution synthesis is constrained by requirements specification data, optional additional constraints specified by the user, and/or implicit constraints imposed by relationships with neighboring functions.

Dependent on the maturity of the system (completeness of functional knowledge and accompanying parts database), solution synthesis (configuration) can result in several possible conditions:

- components and parts fully specified
- components specified, no match found in parts database
- multiple solutions capable of satisfying the requirements, insufficient information available to distinguish the selection.

In the case of no match found in parts database, the designer is provided with valuable information; i.e., the specification of required components (atomic level functions). This information can then be used in the selection from vendor catalogs or used as the specifications for the design of a new part. In the case of multiple solutions, the designer is presented with alternatives and must make a selection. Solution continues following this selection. Alternatively, when faced with multiple solutions, the designer may update the exists_when conditions of function solutions to be more specific and resubmit the configuration request. The scenarios which follow provide a more detailed description of the user and system actions for configuration.

3.2.5.1 Scenario 1: Select subset of the conceptual model for configuration

- The user selects the Solve/Launch button (shown as a rocket) from the toolbar with the mouse (Figure 26).
- The system displays the solve submenu where the user is prompted with the message 'Highlight model functions and relations to configure or press Select All to configure the entire conceptual model." In addition, the system displays buttons for **Select All**, **Additional Constraints**, **Launch Configuration**, and **CANCEL**.
- The user highlights several of the functions and relations present in the conceptual model.

3.2.5.2 Scenario 2: Select complete model for configuration

- The user selects the Solve/Launch button with the mouse (Figure 26).
- The system displays the solve submenu where the user is prompted with the message 'Highlight model functions and relations to configure or press Select All to configure the entire conceptual model." In addition, the system displays buttons for **Select All**, **Additional Constraints**, **Launch Configuration**, and **CANCEL**.
- The user presses **Select All**.
- The system highlights all functions in the conceptual model.

3.2.5.3 Scenario 3: Additional constraints

- The user presses **Additional Constraints**.
- The system displays a formatted screen for entering additional constraints and buttons labeled **Add**, **Delete**, **Refresh**, **Update**, and **OK** (Figure 27).
- The user fills in the slots. Valid slot values may be a numeric range or specify selections from available options.

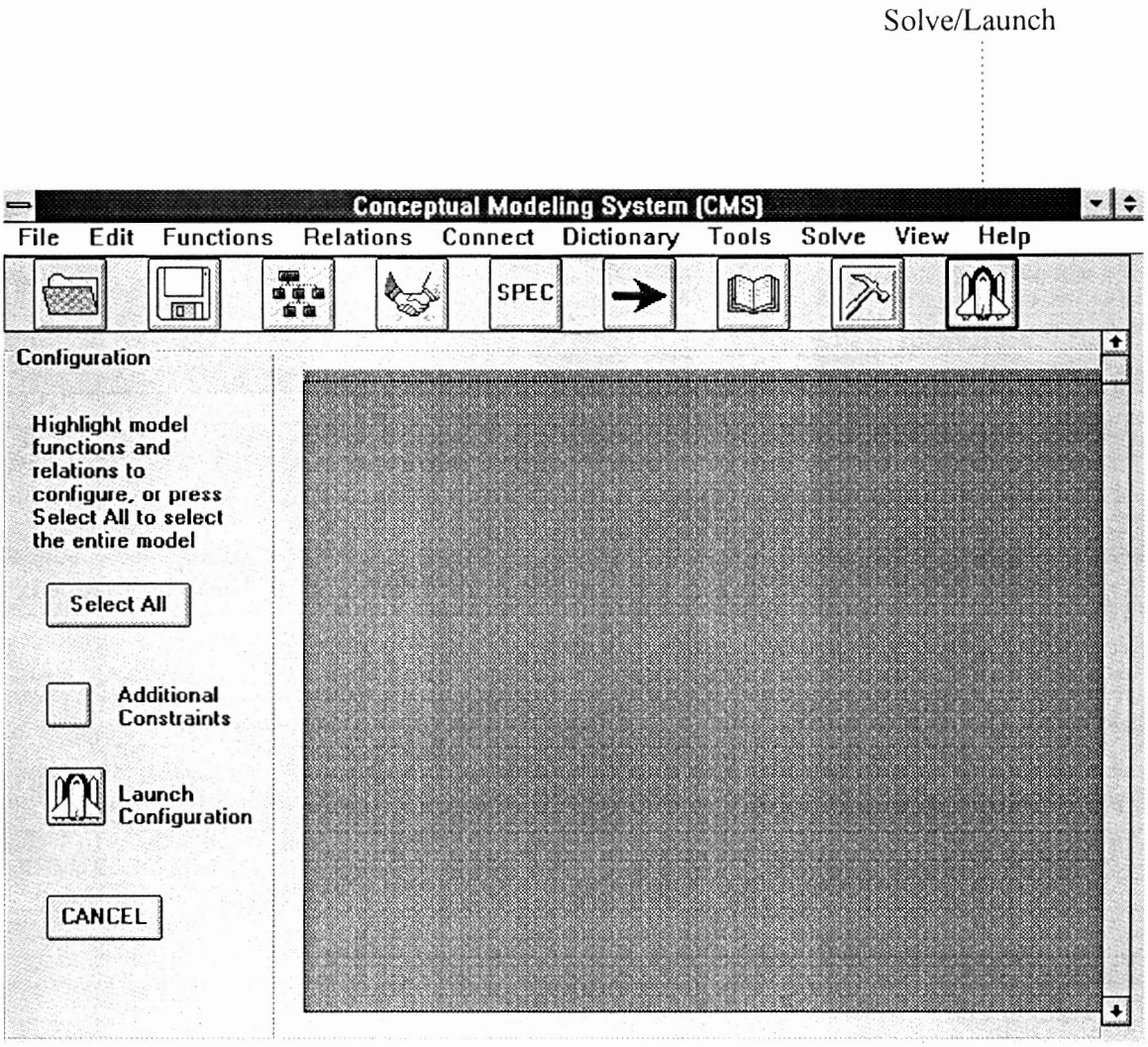


Figure 26. Solve Submenu.

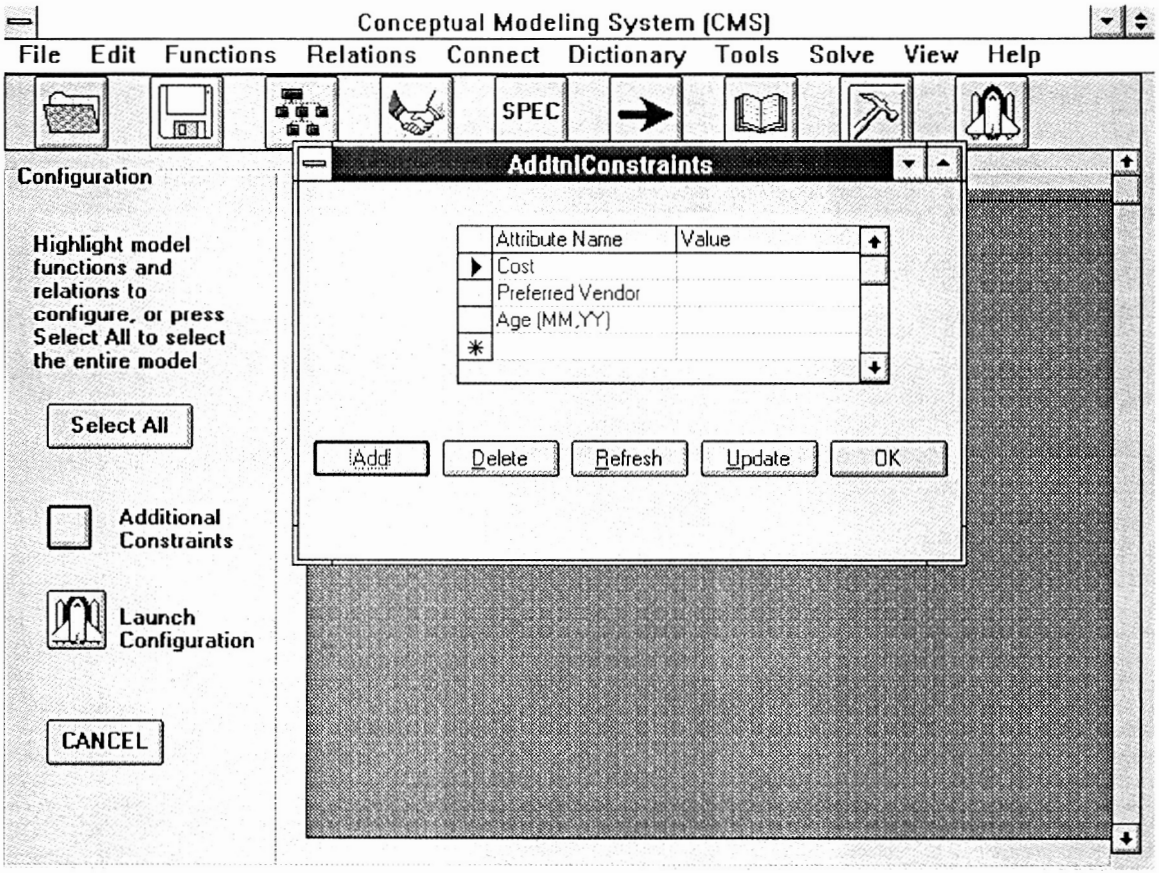


Figure 27. Additional Constraints.

- In the case of available options, the user clicks on the downward pointing arrow to display the list of available options.
- The system displays the list.
- The user highlights the selection.
- The system fills the slot value in with the selection.
- When all slots the user wishes to specify or restrict are filled-in, the user clicks on the **Update** button.
- The system updates the context for the current model, closes the additional constraints screen, and redisplays the solve submenu.

3.2.5.4 Scenario 4: Define the template for entering additional constraints

This scenario is almost identical to Scenario 2 of Use Case 4, where a description is provided for how an administrator would initially specify a template. In this case, rather than describing a requirements specification template, the user specifies the template for additional constraint parameters.

3.2.5.5 Scenario 5: Configuration request (when single function selected)

- The user presses **Launch Configuration** from the Solve submenu (selection, additional constraints, and requirements specification have been provided prior to this action).
- The system initiates the requested configuration process based upon the single functional object. Since this is a single function, its definition is known to the system and solution synthesis proceeds.

3.2.5.6 Scenario 6: Configuration request (when multiple functions/relation(s) selected)

- The user presses **Launch Configuration** (selections, additional constraints, and requirements specification have been provided prior to this action).

- The system builds a temporary function based upon the assemblage of the multiple functions selected and their accompanying relations.
- The system initiates the requested configuration process based upon the assembled functional object. Solution synthesis proceeds.

3.2.5.7 Scenario 7: Synthesis process

Apprentice Systems, a knowledge/object tool, provides the inferencing mechanism for solution synthesis. This scenario describes how the Conceptual Modeling System (CMS) and the Apprentice work together to accomplish configuration. As discussed previously, the context for configuration is set by functional objects and relations present in the model, requirements specifications, and any additional constraints that might have been requested. Listed below is an overview of the solution process.

- User has submitted a Launch Configuration request. The context and function name of this request provides the 'seed' for inferencing against the Expert Model (semantic knowledge model).
- Apprentice searches for solutions within the Expert Model capable of fulfilling the named function within the problem context. Exists_when conditions on potential solutions serve to distinguish solutions relative to constraining conditions.
- When exists_when conditions are not present on potential solutions, the user is presented with all solutions matching the named function (see Scenario 8).
- The Apprentice progresses through functions present in the conceptual model in the sequence determined by the dependencies of connecting relationships between functions.

- For each function the Apprentice uses the Expert Model to locate solutions. If the function is decomposable (i.e., not atomic), the Apprentice progresses through each member function of the containing function. Functional decomposition and solution using the Expert Model continues until all functions have been visited and have been decomposed to the component level.
- Once at the component level, the Engineering Model is used by the Apprentice to specify the component and locate potential parts capable of fulfilling the requirements from the company database.
- When all configuration processing is complete, the system alerts the user with a message "Processing Complete".
- The user then may view the configuration results (Scenario 9).

3.2.5.8 Scenario 8: Multiple solutions

- The Apprentice alerts CMS that multiple solutions exists. Configuration is interrupted, Apprentice passes the information to CMS, and waits for a reply.
- The system (CMS) displays a list of solutions for the unresolved function and a message prompting the user to make a selection amongst the choices and to press **OK** to continue or **CANCEL** to halt the configuration process (Figure 28).
- The user highlights the solution choice and presses **OK**.
- The system passes this selection on to the Apprentice and configuration continues.

3.2.5.9 Scenario 9: View (report on) configuration results

- The user highlights functions in the conceptual model to report and then clicks on the **View** menu bar selection.

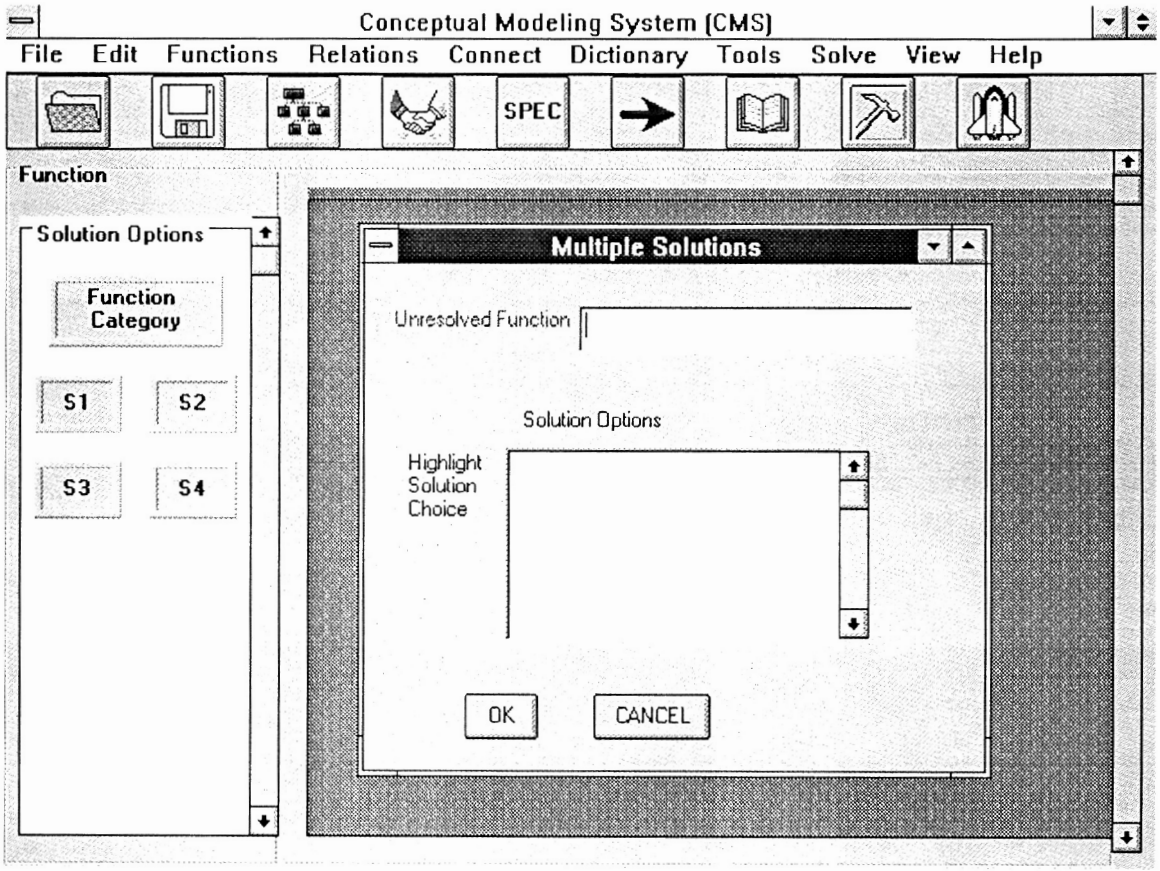


Figure 28. Multiple Solutions.

- The system displays a pull-down menu (Figure 29) where the user may choose the view desired (Bill of Materials, Conceptual Model, Cost Data, Mechanical Model, or Schematic).
- The user selects the view desired.
- The system displays the appropriate view based on the user request. An overview description of what the system displays for each selection option follows:

Bill of Materials	Tabular summary of configuration results broken down (hierarchically if necessary) and summarized by function, component category, and part number.
Conceptual Model	Conceptual model (function/relation view) current in modeling screen area.
Cost Data	Tabular summary of costs. Totals provided by function and grand total.
Mechanical Model	Image in CAD database associated with selection.
Schematic	2-D schematic view based on components results (including relations). When available, part references included on components.

- When finished with a view, the user clicks **OK** to close the viewing screen.

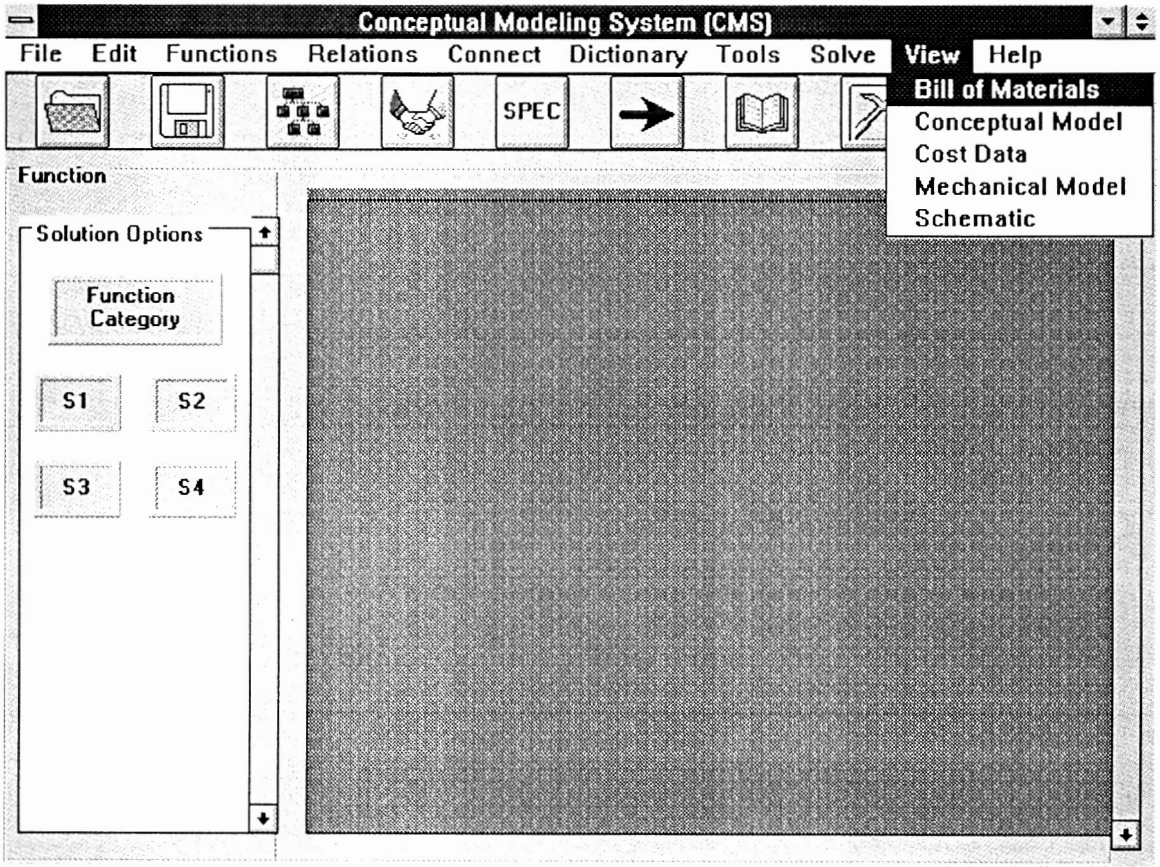


Figure 29. View Pull-down.

3.2.6 Use Case 6: Interface with Supporting Tools and Data

The focus of this use case is on the temporary exit from the modeling environment to access other tools and/or data that come to bear on the conceptual design process. As described relative to the current state of the conceptual modeling system, the actions of this use case are relatively trivial. For now, actions are limited to minimizing the application window; thereby allowing access to systems and data existing apart from the conceptual modeling environment. The provide the means to minimize the conceptual modeling application so that access is visible to other application windows that have been hidden from view while in CMS. In future extensions of the system, **Tools** menu selection or the tools button (displayed as a hammer on the toolbar) will list available applications and data sources that support conceptual design existing outside of CMS. Examples of these selections might include: MRP, vendor records, analysis programs, marketing/sales data, service records, manufacturing data, etc. (see Figure 30). Issues related to the exchange of parameters between CMS and outside applications will also be addressed.

3.2.6.1 Scenario 1: Temporarily exit from the conceptual modeling environment

- The user selects the tools button (shown as a hammer) from the toolbar (Figure 31).
- The system prompts the user to confirm the request to minimize CMS or **CANCEL** to exit.
- The user confirms the minimize request.
- CMS is minimized and other windows become visible.

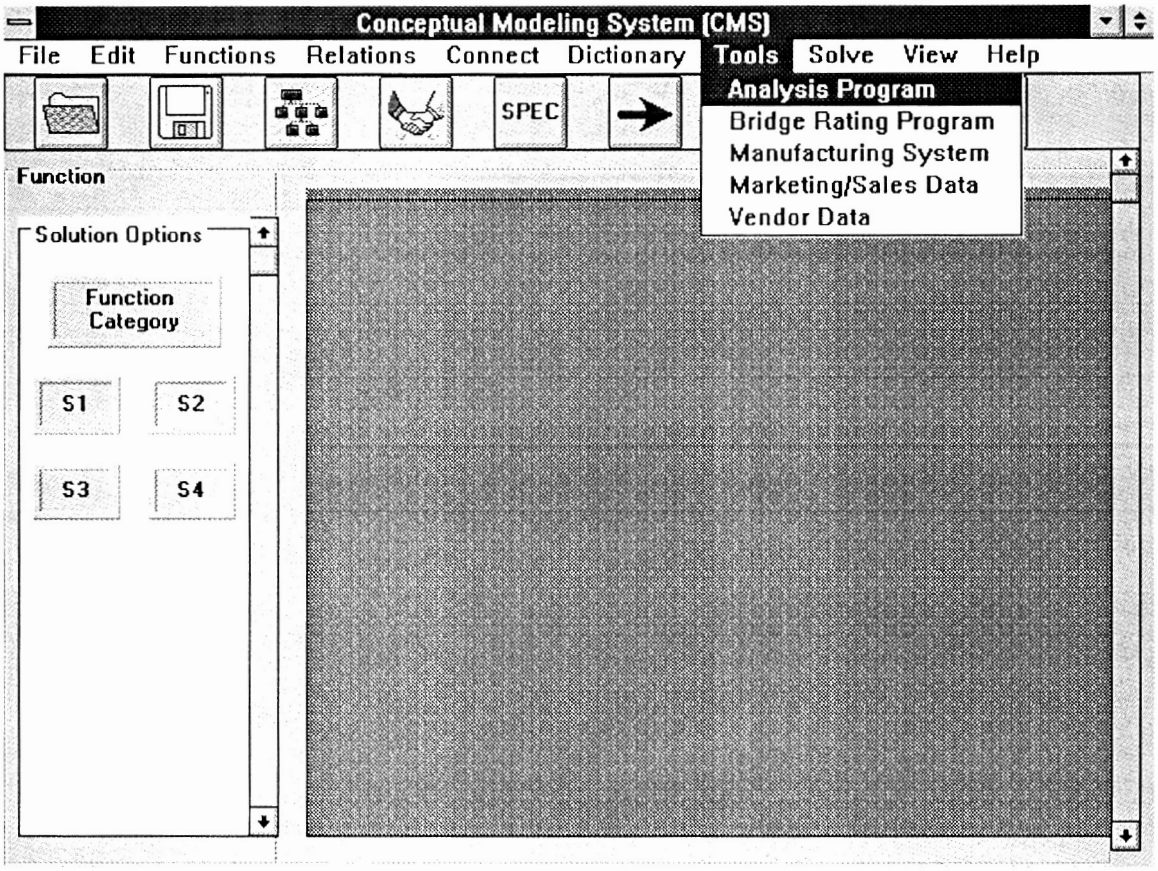


Figure 30. Tools Pull-down.

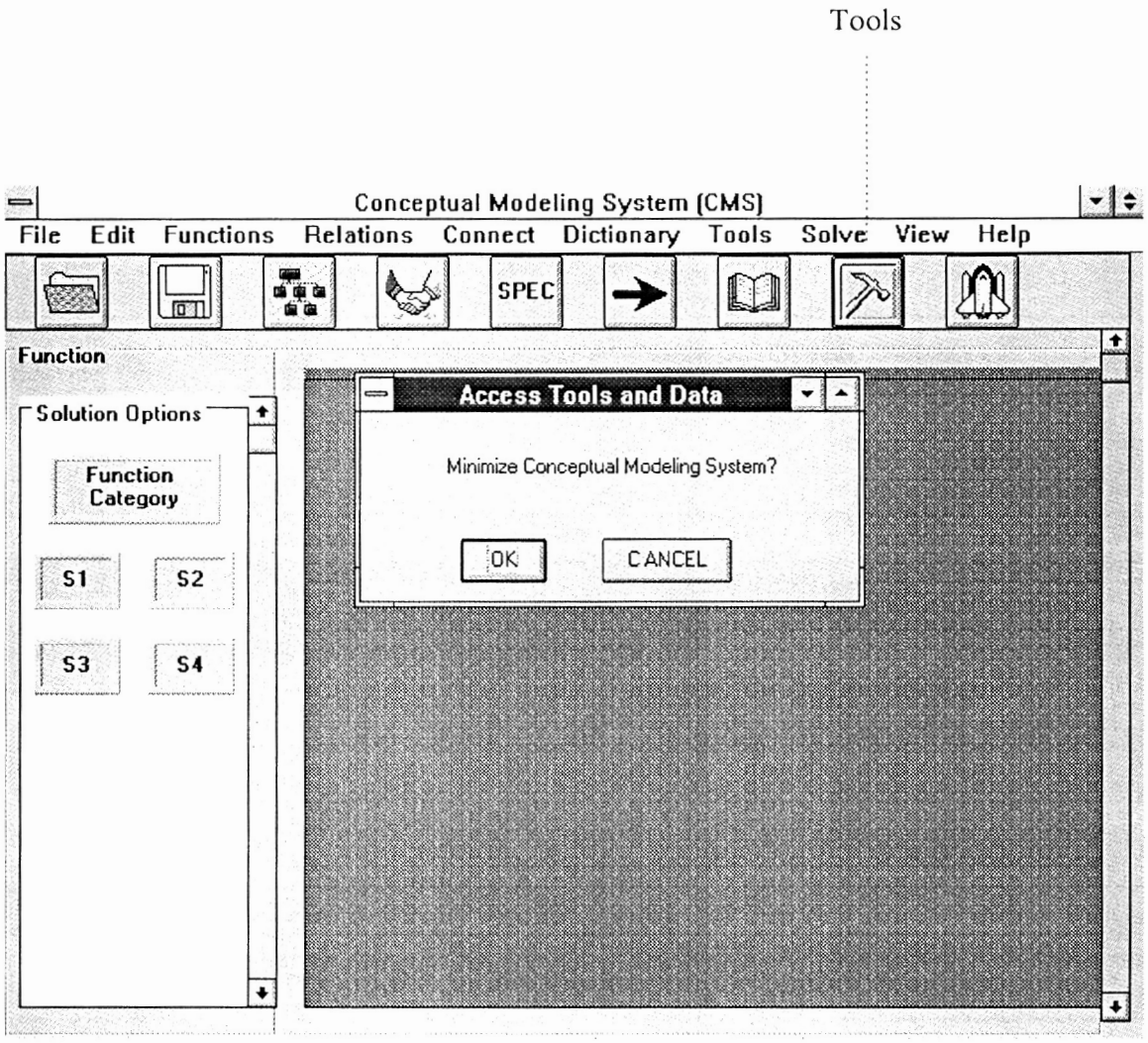


Figure 31. Minimize CMS.

3.2.6.2 Scenario 2: Run analysis program(s) or access data not currently integrated into the conceptual modeling environment

3.2.6.3 Scenario 3: Return to the conceptual modeling environment

- The user double clicks on the minimized icon for the conceptual modeling system.
- The application window is re-opened.
- The user continues with the modeling actions (Use Case 1) based on the added information from the temporary exit (now informing the next modeling action).

4. Conceptual Modeling System - Architectural View

The previous chapter provided a functional view of the conceptual modeling system by way of a use case model description. Utilizing and extending this model, this chapter provides an architectural view of the framework for integrated design. Serving to clarify this presentation, a brief overview of the object-oriented analysis and design (OOAD) methodology applied in the development process is first offered. Based on this methodology, several diagrams (intermediate and final) are then presented detailing the resultant architecture of the framework.

4.1 Object-Oriented Analysis and Design Methodology

It is not the purpose of this section to offer detailed instruction in OOAD methods. An abundance of texts, articles, and courses are available for those desiring an in-depth treatment of the subject. The interested reader is referred to any of the following (highly recommended) resources: 1) the public course in computer aided software engineering (CASE) offered by Rational Software Corporation, "Object-Oriented Analysis and Design using Booch" [ACC96], 2) textbooks by Ivar Jacobson [Jaco94] and Grady Booch [Booc94], and 3) Rational Rose (a CASE tool offered by the Rational Software Corporation [Rati95]). The inclusion of this section is motivated by the numerous OOAD methods currently available; each defining the OOAD model with a different set of diagrams, vocabulary, syntax, and/or techniques. The intent here is to provide an *overview* of the particular methodology applied in developing the integrated conceptual modeling framework.

Guided largely by the "recipe for success" prescribed in OOAD using Booch [ACC96], the outline provided below summarizes the highly iterative steps that were applied in developing the framework. As is common to OOAD methods, the process of transforming need into system realization centered around the progressive refinement and ultimate definition of a class diagram. In the end, the class diagram and other diagrams generated as a result of the OOAD process document the design (system behavior and architecture), and provide the basis for computer code generation, implementation, and testing. Given the research nature of this development project, some of the team oriented and business review steps have been omitted from the original outline. The primary tasks applied include:

1) Gather/develop information and begin conceptualization
    - write problem statement
    - outline and briefly describe use cases
    - investigate similar or legacy systems
    - brain-storm preliminary concepts and prototypes
2) Identify potential classes/objects
    - underline nouns in written documents (problem statement, etc.)
    - object "blitz" (invented, domain experts)
    - scenario analysis
3) Filter objects to select classes
    - eliminate potential objects that are vague, redundant, outside of the problem domain, or those that are more appropriately attributes or operations
4) Define preliminary class diagram
    - group classes according to: hardware/software, whole/part (aggregation), generalization (superclass/subclass), identify class categories

- define associations
- define attributes/operations from a high-level

5) Create scenarios (defining sequence of events for use cases)
6) Create a message trace diagram for scenarios (pictorial view of all classes participating in a scenario and messaging requirements)
7) Create object message diagrams from message trace diagrams (closer examination of how objects work together to accomplish useful work found in use cases)
8) Update class diagram based on information gained from steps 6 and 7 (allocate operations, define new classes as required)
9) Review application model
10) Refine model and test against use cases
11) Implement (code templates, etc.)
12) Test, review, refine.

4.2 Architectural View

The use case model provides the initial focus of discussion; demonstrating diagramming methods and providing samples of the intermediate steps in the OOAD process. Next, a top-level view summarizes the major components (class categories) of the architecture. These categories are then discussed in greater detail by way of class diagrams. Assisting this presentation, Figure 32 provides an initial summary of diagramming symbols and notation used in this chapter.

4.2.1 Use Case Model Analysis

The following three diagrams (Figures 33 - 35) provide a pictorial representation of the use case model. Shown using the notation of Grady Booch [Booc94], classes are

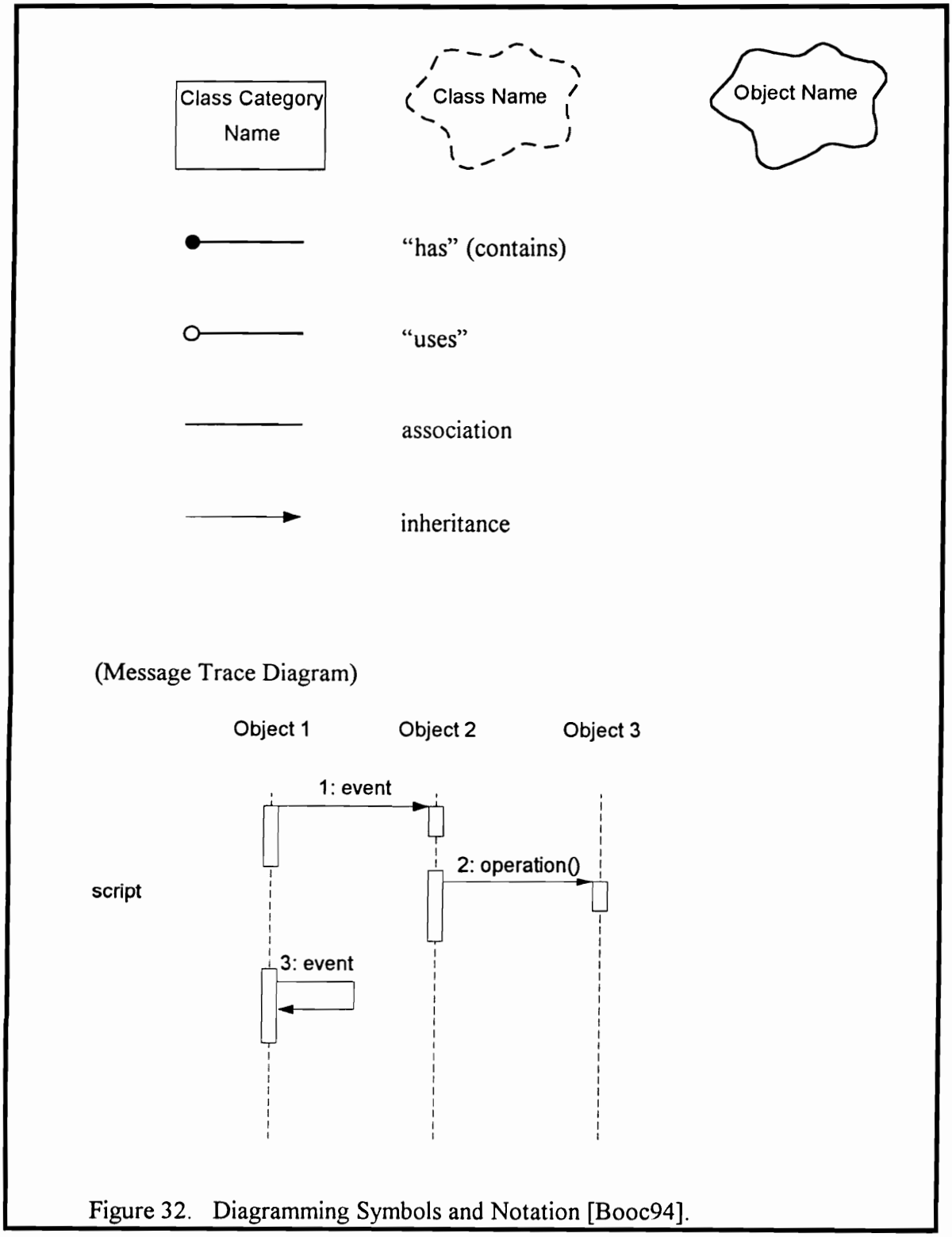


Figure 32. Diagramming Symbols and Notation [Booc94].

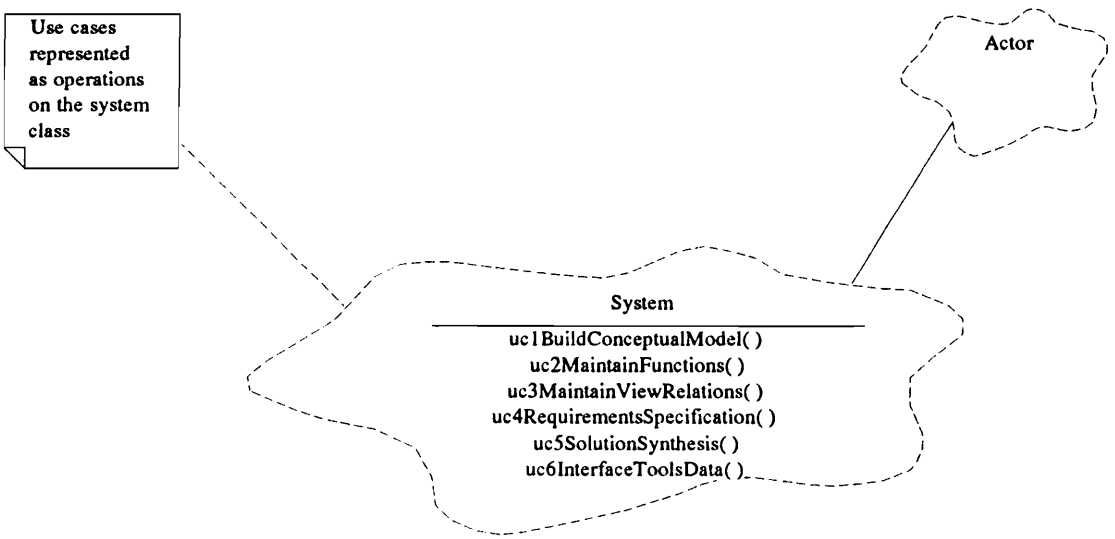


Figure 33. Class Category - Use Case Model.

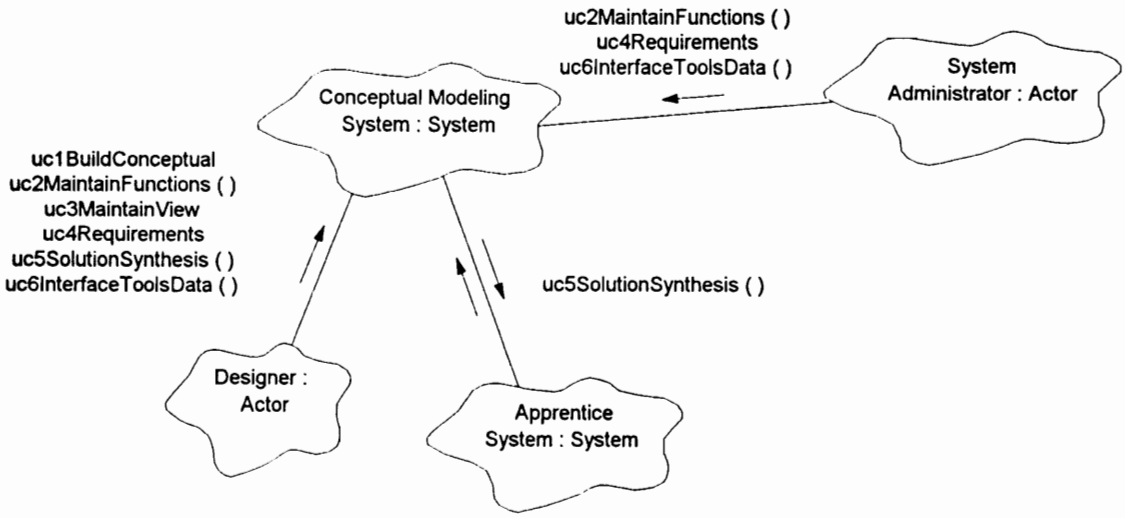


Figure 34. Actors and Use Cases.

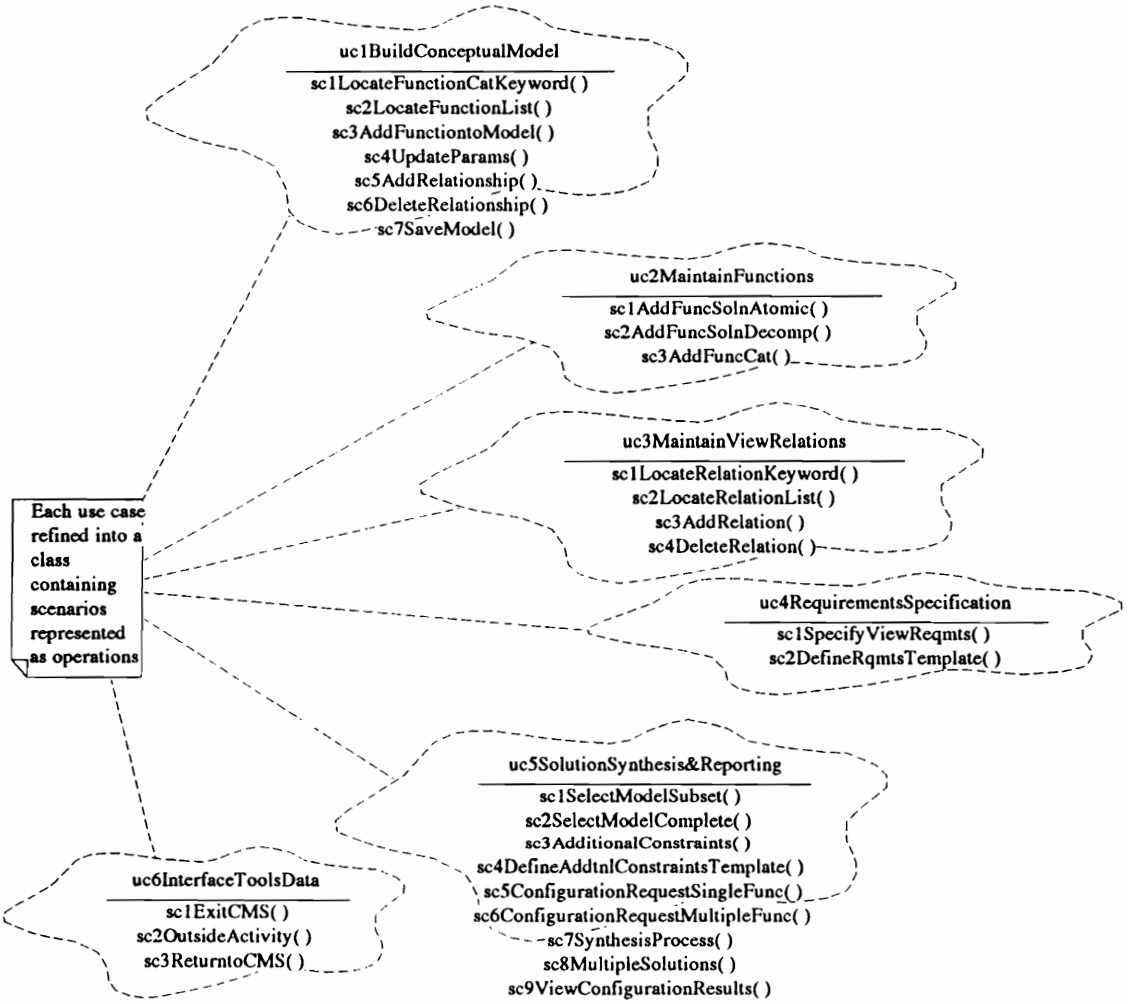


Figure 35. Use Cases as Classes.

represented with dashed-line clouds and object instances of classes with solid-line clouds. Connecting lines between clouds indicate an association between classes.

The first diagram, Figure 33, represents the use case model as a class category. Here, "system" is a class in the category. Initially, use cases are operations on this class. In addition, "actor" is a class in the model. Not explicitly shown in this diagram, all actors on the system are members (object instances) of the actor class. The next diagram (Figure 34) shows actors relating to the conceptual modeling system through use case operations. This type of object diagram is known as a context diagram and serves to clarify the links between actors and use cases. Later, each use case is refined into a class with scenarios as operations (Figure 35).

Continuing the process, scenarios are then analyzed to identify classes along with associated attributes and operations necessary for system definition. Six diagrams follow, providing examples of this refinement process. Figures 36 and 37 display the message trace diagram and object message diagram for the first scenario of Use Case 1. Recall from the previous chapter that Use Case 1 is concerned with the interactions required to build a conceptual model. Scenario 1 relates to the actions involved in locating a desired functional object using a keyword match and then opening the associated function category template. As shown, the message trace diagram is built with a dashed vertical line for each domain object participating in the scenario. In this case, object instances are required of classes Actor, System, Use Case 1, Main Toolbar, Function Category, and Function Solution. Horizontal lines represent the sending of messages between objects. As shown, the message trace diagram is read from top to bottom and left to right; reflecting the single time-ordered sequence of events through the system as defined in the scenario (described in the previous chapter). The name assigned to a horizontal line in the

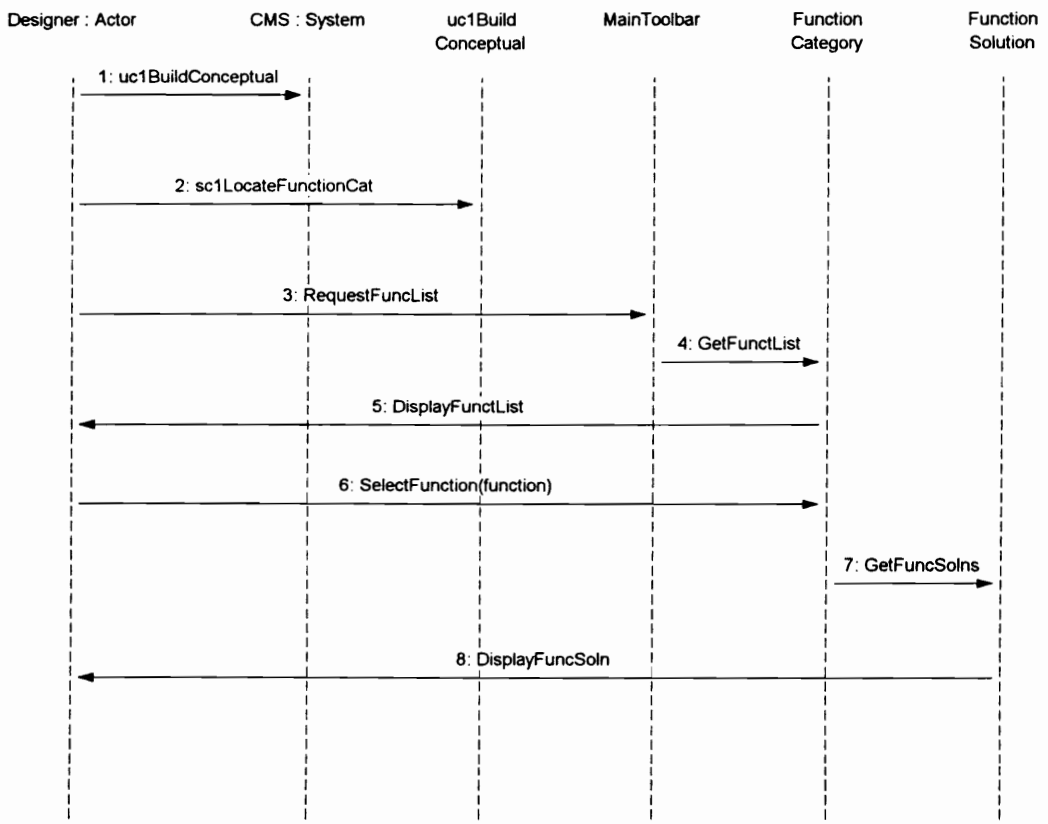


Figure 36. Message Trace Diagram - Use Case 1, Scenario 1.

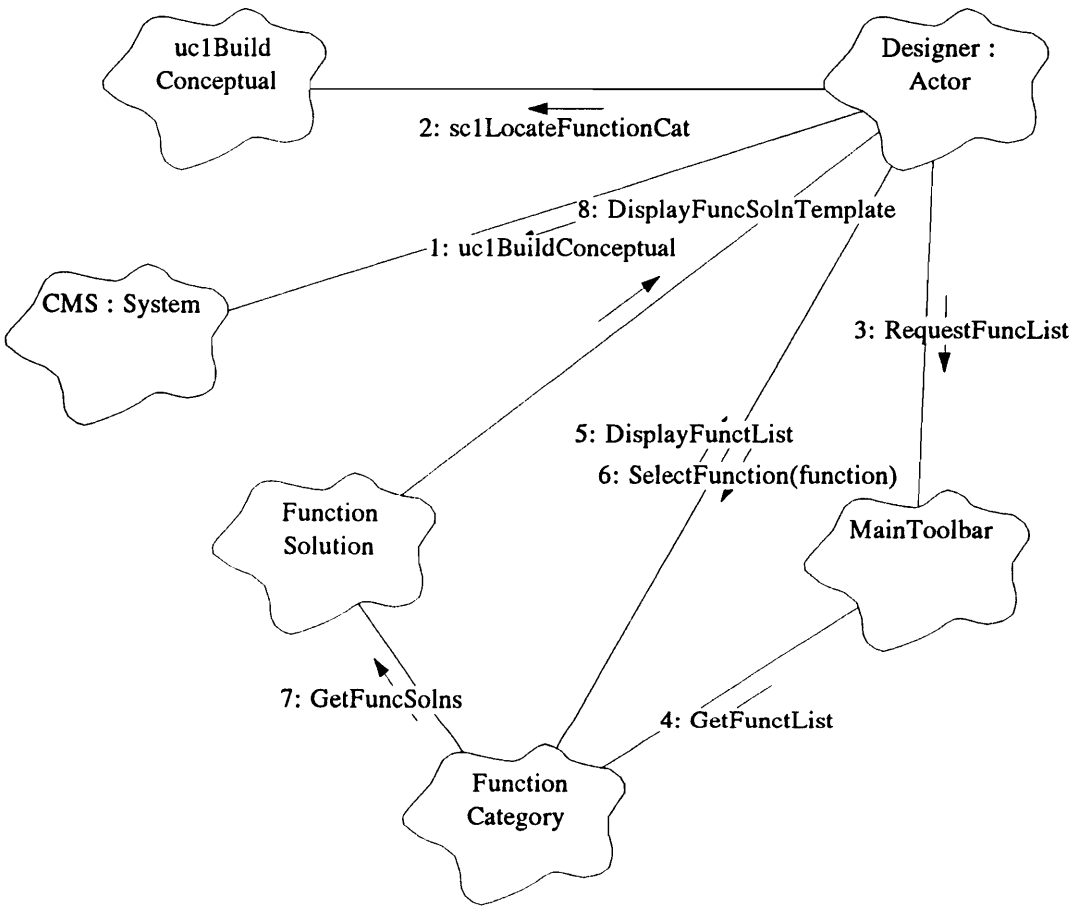


Figure 37. Object Message Diagram - Use Case 1, Scenario 1.

message trace diagram indicates the purpose of the message between objects. At first, the naming of each connection is informal. In time, these messages are refined (name and attributes) and become the operations that are defined in the class definition.

Examining the message trace diagram for Scenario 1 of Use Case 1 (Figure 36), the Designer (an instance of the Actor class) initiates the interaction with CMS (an instance of the System class). A particular instance of Use Case 1 is initiated by Designer. A request for a Function List is made by the Designer from the Main Toolbar. The Main Toolbar responds by requesting a list from the Function Category class object. The Function Category responds with the list displayed to the Designer. The Designer selects the particular function from the Function Category. The Function Category responds by making a request to the Function Solution class. Function Solution responds by displaying the Function Solution category selected to the Designer.

The object message diagram (Figure 37) is similar in information content to the message trace diagram. Here, objects are represented as class instances (clouds) with operations between indicating messaging link requirements. Using Rational Rose [Rati95], an object message diagram can be generated directly from the message trace diagram. Likewise, the message trace diagram can be generated from an object message diagram (depending on which diagram the user chooses to work with first).

In a similar manner, Figures 38 and 39 provide the message trace and object message diagrams for Use Case 1, Scenario 3. Again related to building a conceptual model, Scenario 3 involves the selection of a function from an open template of solutions and the

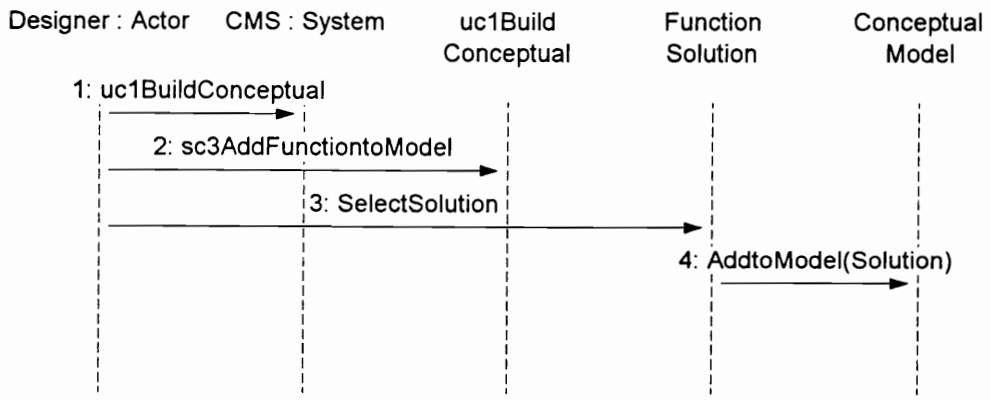


Figure 38. Message Trace Diagram - Use Case 1, Scenario 3.

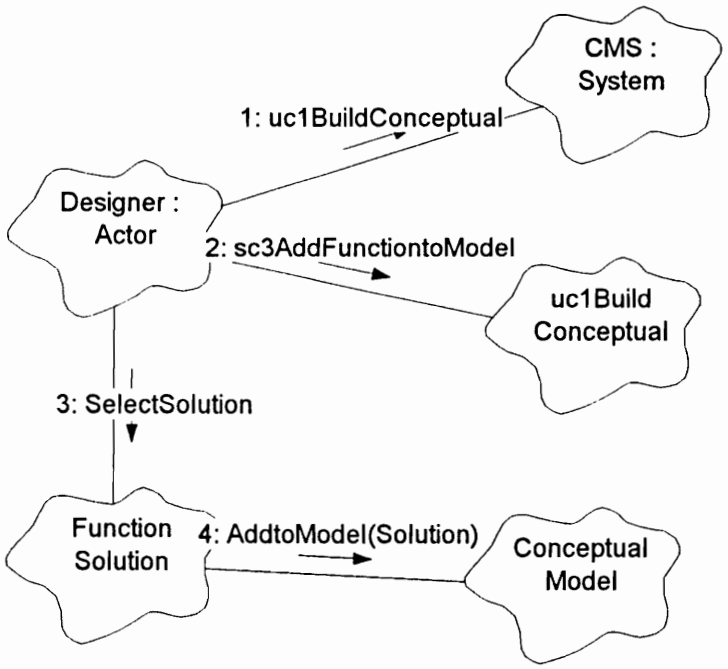


Figure 39. Object Message Diagram - Use Case 1, Scenario 3.

placement of the function in the conceptual model under construction in the workspace modeling area of CMS. As in the previous example, the message trace diagram displays domain objects participating in this scenario with vertical lines. Horizontal lines represent messages between objects. Offering a more involved example, Figures 40 and 41 depict the message trace and object message diagrams for the fourth scenario of Use Case 1. Again, referring to Chapter 3, Scenario 4 describes the time ordered sequence of events related to viewing and/or constraining the parameters of a function in the conceptual model. As shown in the diagrams, eight class objects and fourteen message requests (operations) are required to accomplish the actions of this modeling objective.

Use cases are very helpful in documenting requirements as well as defining and relating primary classes. Alone, however, use cases are not sufficient to describe system architecture. The next section provides a top-level view of the architecture by way of class categories. Subordinate sections report on the class diagrams describing each of these categories.

4.2.2 Class Categories

Booch [Booc94] recommends, particularly for systems containing over a dozen classes, that the architectural view include a top-level diagram where classes are grouped together logically into *class categories*. Each class category is a subset, or collection, of classes collaborating to provide a set of services.

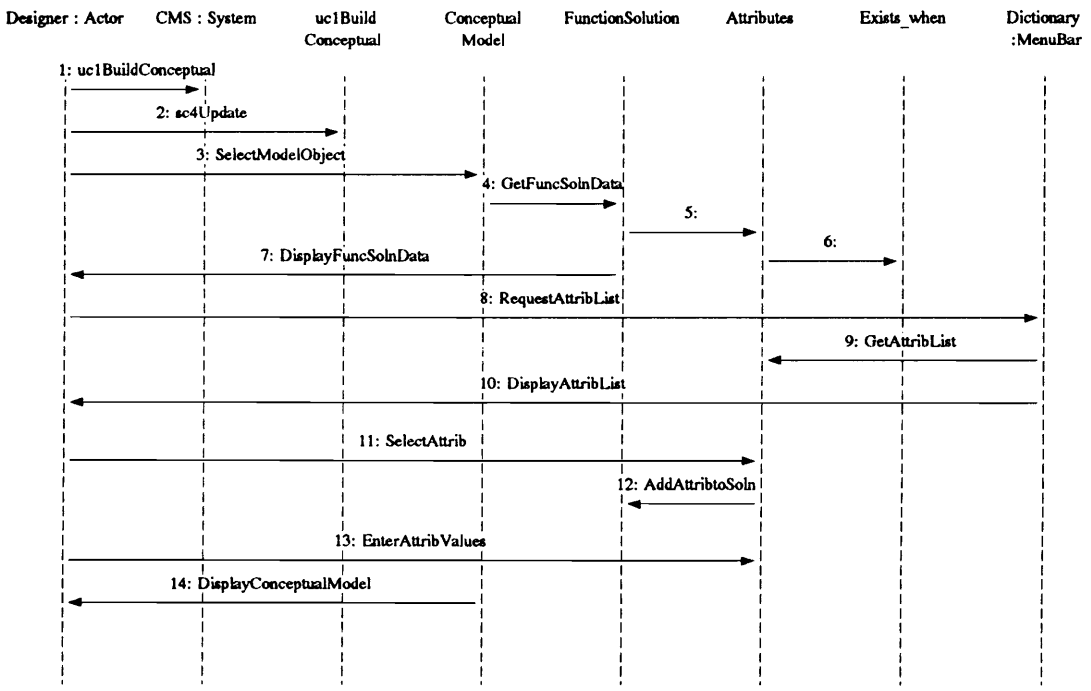


Figure 40. Message Trace Diagram - Use Case 1, Scenario 4.

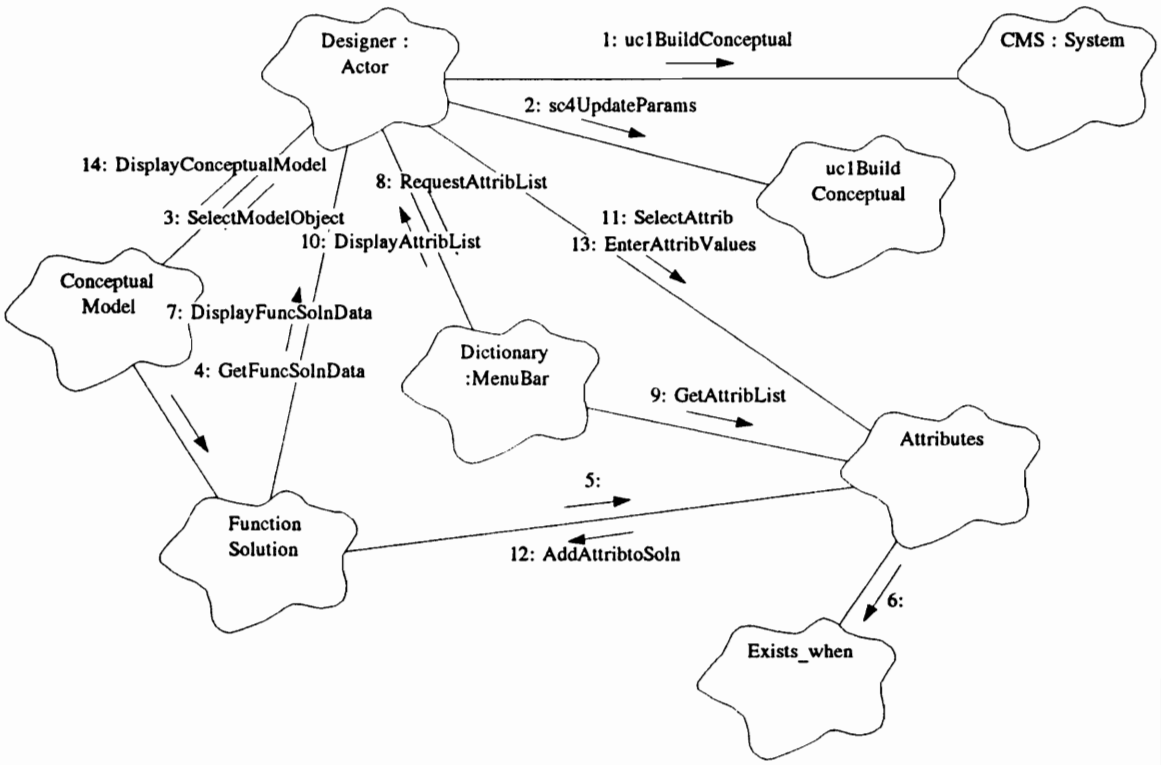


Figure 41. Object Message Diagram - Use Case 1, Scenario 4.

The top-level view of the architecture for integrated conceptual design is shown in Figure 42. The primary class categories include:

- Graphical User Interface (GUI)
- Conceptual Modeling System (CMS)
- Apprentice System
- Conceptual Model
- Expert Model (Functional Knowledge-Base)
- Engineering Model (Components Knowledge-Base)
- Presentation Model
- Company Parts Database

As shown, the categories are arranged in layers with several connections indicating "uses" relationships between categories. The sections that follow examine these class categories with greater detail. The role, or service, that the category provides to the overall system is discussed. When appropriate, the class diagram for the category is included; offering a pictorial description and additional details of the architecture.

4.2.2.1 Graphical User Interface

The graphical user interface (GUI) is at the highest level of abstraction in the top-level diagram. The role of the GUI is to provide the interface between the user and the conceptual modeling system. The architecture view, class diagram, for this category is depicted in Figure 43. The discussion that follows offers explanation of the class diagram. As necessary, class names used in the discussion are highlighted in **bold** typeface.

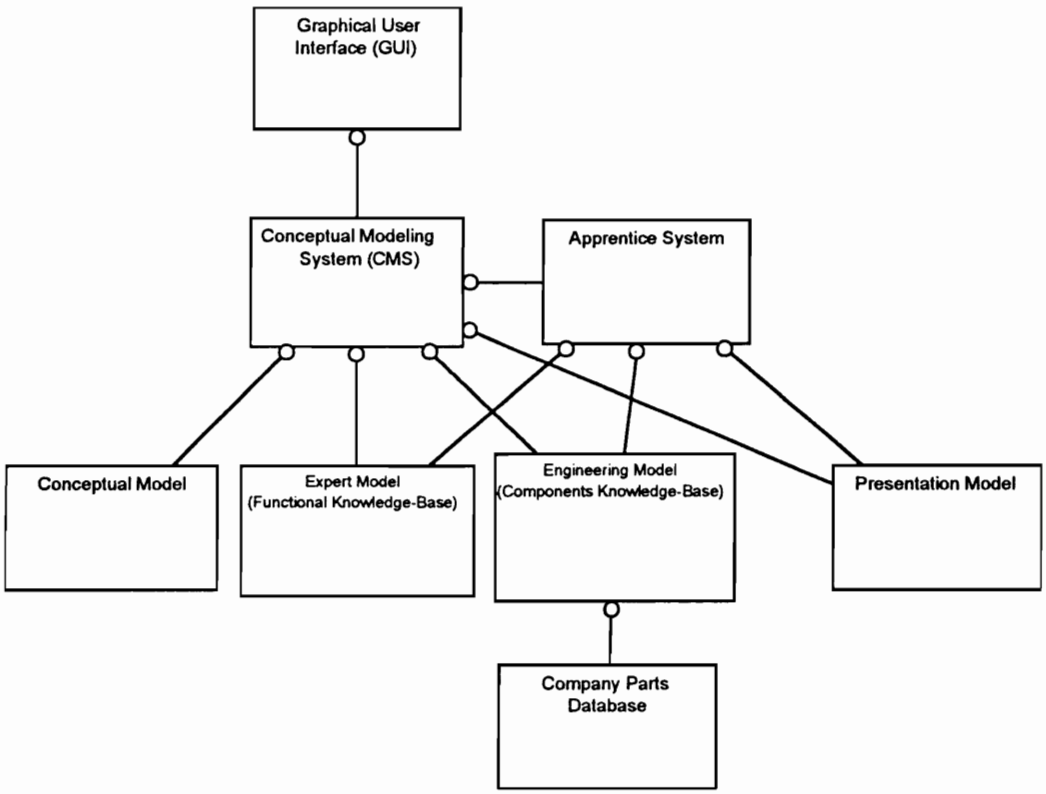


Figure 42. Top-level Class Category Diagram.

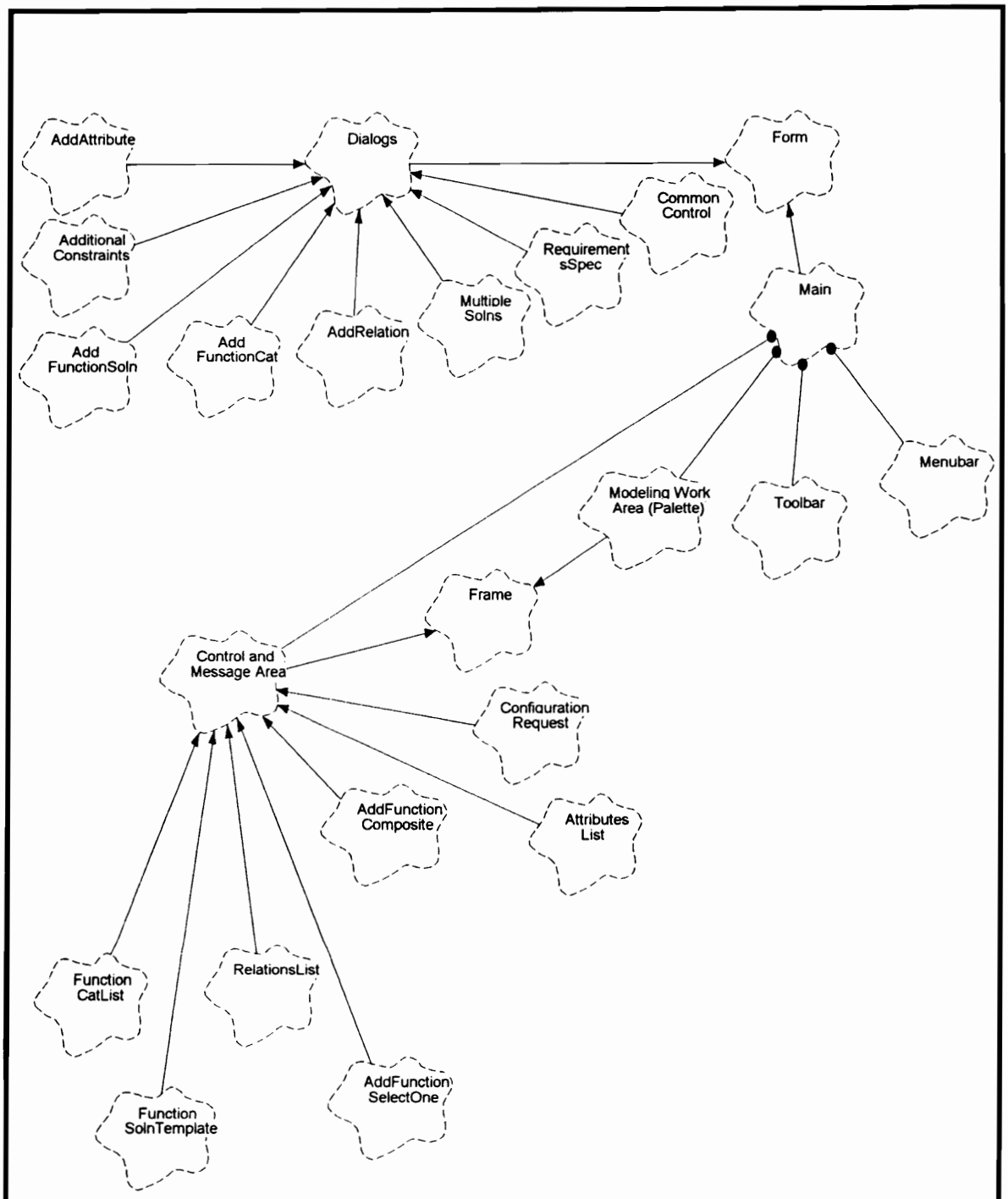


Figure 43. Graphical User Interface (GUI).

Reflected in the diagram, interface screens are defined using two **Form** types. Briefly, in Microsoft Visual Basic [Micr95], a form is the window that the developer customizes in building an interface to the application. In the first type of form, **Main** provides the primary interface for the conceptual modeling system. It is made from (contains) four distinct components:

- **Control and Message Area**
- **Modeling Work Area**
- **Toolbar**
- **Menubar**

Toolbar and **Menubar** are provided within the software development tool set of Visual Basic (VB). Each was customized with the control buttons and pull-down menu selections required in CMS. **Control and Message Area** and **Modeling Work Area** are types of a **Frame**. Frames allow the form to be partitioned into distinct areas. As shown in the class diagram, the **Control and Message Area** class is specialized into several subclasses. The **Control and Message Area** frame of **Main** is located to the left of the **Modeling Work Area**. Depending on the actions of the user and the state of the system, subclass frames are swapped in and out of view. The **Modeling Work Area** frame provides the graphical palette where the user builds the conceptual model.

Dialogs are the second type of form defined in CMS. In contrast to the **Main** form, **Dialogs** are temporary. **Dialogs** overlay the **Main** form when displayed to the user and are visible only until the appropriate action has been taken. As shown in the class diagram, several dialog types have been defined allowing the user to update and view system data.

4.2.2.2 Conceptual Modeling System

The conceptual modeling system, as shown in the class category diagram (refer back to Figure 42), is central to the architecture for integrated conceptual design. The purpose of this category in the diagram is to represent the functional partitioning of CMS and relationships with other systems. As implemented, the graphical user interface manages the interactions with class categories shown in "uses" relationships with the conceptual modeling system. Detail of the participating components in CMS follows.

4.2.2.3 Conceptual Model

The conceptual model class category is concerned with the model that is current in the work area. The primary role of this category is to manage the conceptual model as it is constructed. Discussed in later sections, the conceptual model also provides input for configuration processes and input for updates to the Expert Model (Functional Knowledge-Base).

Illustrated in the class diagram for this category (Figure 44), several classes are required to build and constrain a model. As shown (with cardinality numbers on associations), the **Conceptual Model** is composed of exactly 1 **Requirement Specification**, 0 to many **Additional Constraints**, 0 to many **Function Solutions**, and either 0 or 1 **Assembly**. If there exists more than one **Function Solution** in the model, an **Assembly** is used to bind the model together. In this case, and depending upon the number of **Function Solutions** in an **Assembly**, there are 1 to many **Relations** in the **Assembly**. In addition, **Function Solutions** can be constrained with **Attributes** and/or **Exists_when** conditions. Inherently, **Function Solutions** are of a particular **Function Category** that is characterized with a given set of **Attributes**.

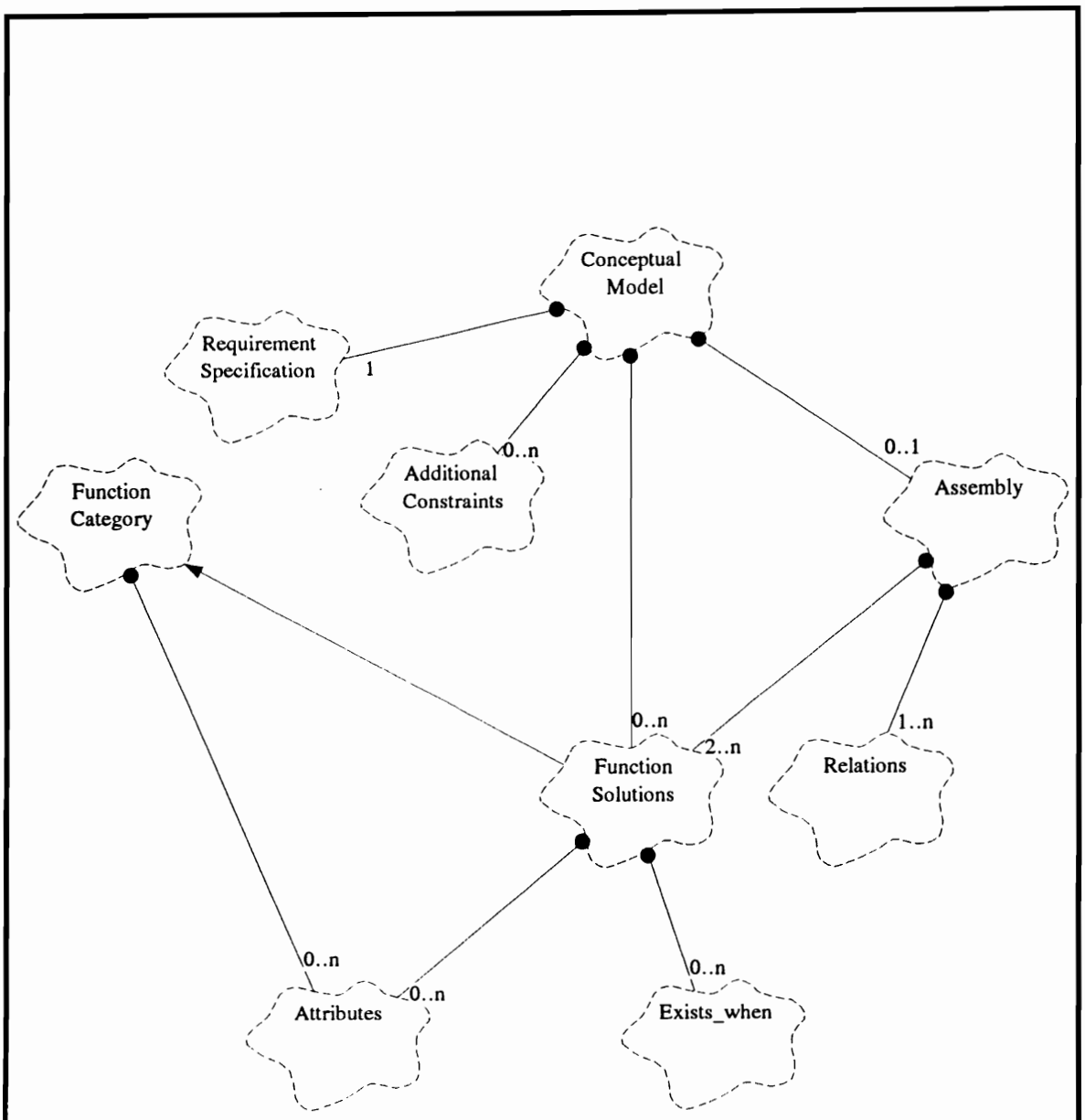


Figure 44. Conceptual Model.

4.2.2.4 Expert Model (Functional Knowledge-Base)

The expert model provides two primary services to the conceptual modeling system. First, it provides the foundation for knowledge capture and reuse. As shown in the class diagram displayed in Figure 45, **Function Solutions** are collected and categorized into meaningful groups. Later, when retrieved for reuse, a given **Function Category** provides a wealth of knowledge with proven solutions available for selection. Each **Function Solution** may be described (optionally) in terms of a set of specific **Exists_when** conditions and/or **Attributes**. Each **Function Solution** may be a simple (non-decomposable) solution for the **Function Category**, or it may be an **Assembly** containing multiple **Function Solutions**. In the case of an **Assembly**, the solution contains **Relations** between member solutions.

Discussed previously in the use case model offered in Chapter 3, the expert model is also an integral component in functional decomposition and solution synthesis procedures.

4.2.2.5 Engineering Model (Components Knowledge-Base) and Company Parts Database

The engineering model category is concerned with the low-level functions (components) of design. The primary role of this class category is to support configuration processes of CMS. The engineering model is used to specify components and interface with company databases in the part selection process. As shown in Figure 46, component knowledge is described in terms of component categories. Each **Component Category** represents a particular class of component. Potentially, sub-class **Types** can also be defined for a given component class. Each **Component Category** is characterized with attributes as well as

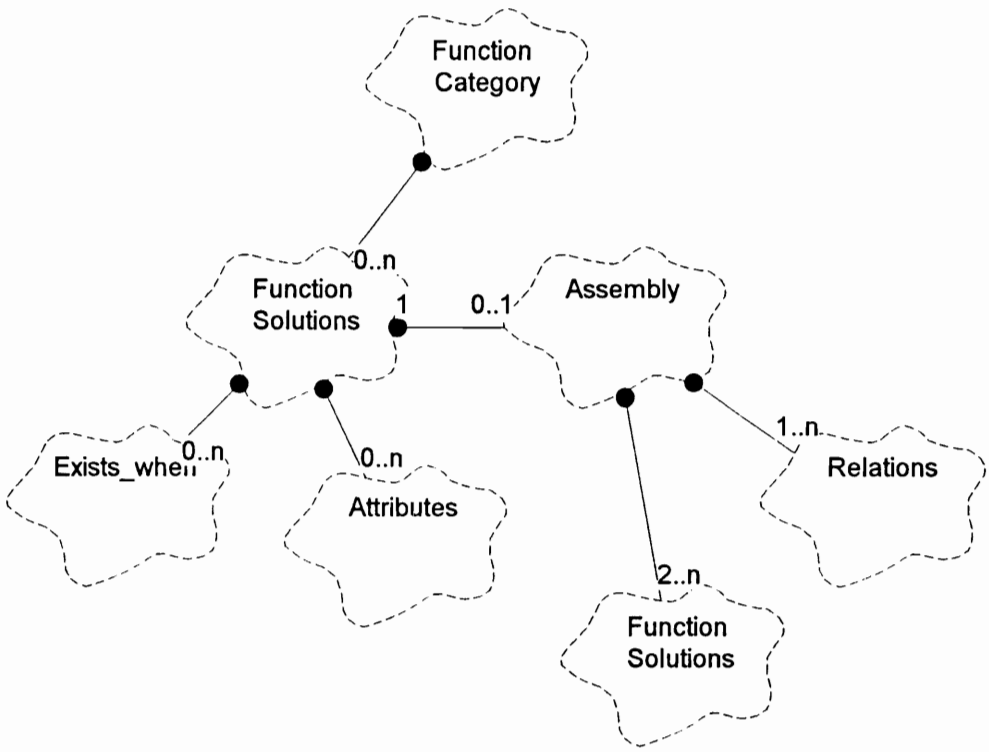


Figure 45. Expert Model (Functional Knowledge-Base).

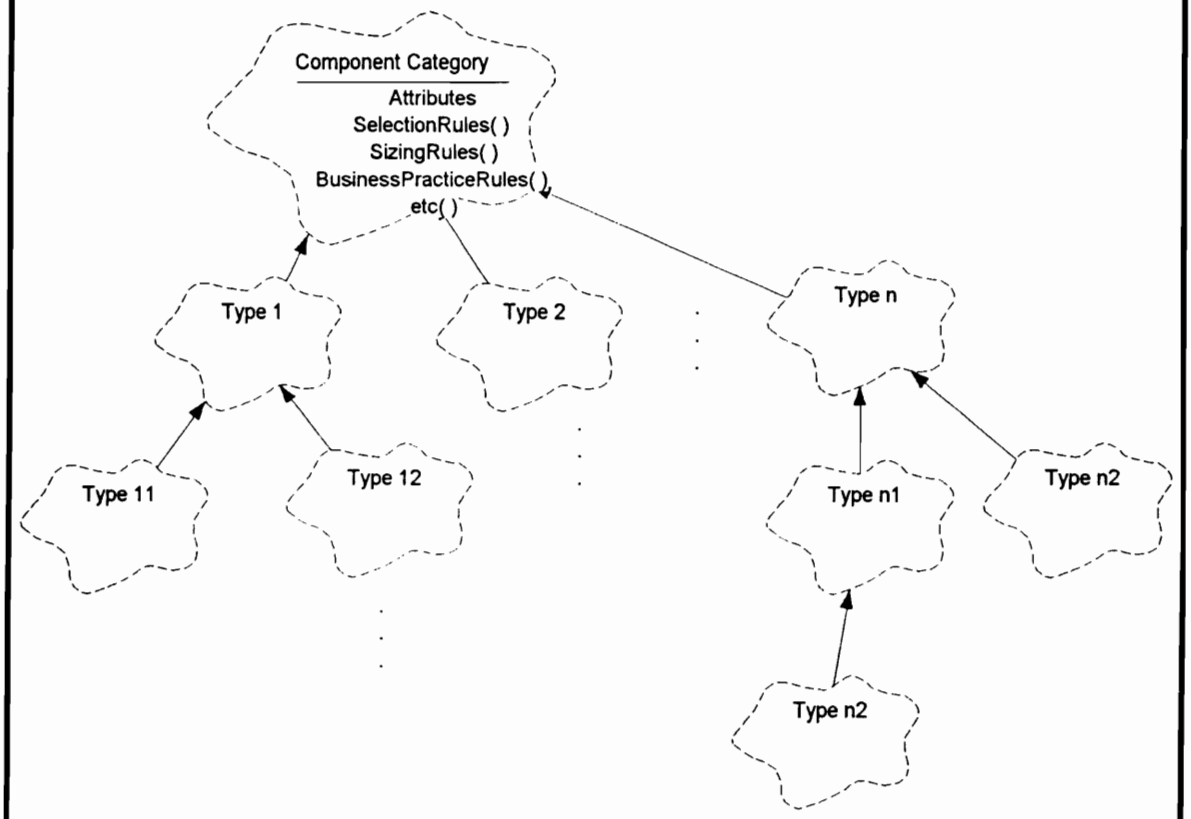


Figure 46. Engineering Model (Components Knowledge-Base).

operations reflecting the rules applied in the engineering process. Among others, rules may be defined for sizing, selection, material specification, or to enforce company standards. Referring again to Figure 46, the class category diagram illustrates the distinct separation of the engineering model from the company parts database. This decoupling of knowledge from data provides for greater generalization as well as reduced system maintenance.

4.2.2.6 Apprentice System and Presentation Model

The Apprentice System is a commercially available knowledge-based engineering tool. As shown in the class category diagram, Apprentice System works directly with CMS. Its purpose is to provide the reasoning mechanism for solution synthesis (configuration). A functional view of this relationship was offered within the use case model of Chapter 3 (Use Case 5 - Solution Synthesis and Reporting). Reviewing this discussion briefly, CMS provides the Apprentice a Conceptual Model for configuration when requested by the user. The Apprentice reasons over the Conceptual Model, decomposing the functional model based on the function/solutions of the Expert Model until all functions are at an atomic level. The Engineering Model is then used to size, select, or otherwise specify components against the Company Parts Database. Viewing and reporting configuration results is handled by the Presentation Model. Functionally, this model has been specified in the Use Case model of Chapter 3. Further consideration of human interface and resultant architectural issues will be addressed in future extensions. Currently, the Apprentice System does offer its own form of a Presentation Model that can be customized for a given application domain. For example, within piping applications, the Presentation Model of Apprentice is concerned with the schematic (2-D diagram layout) representation of configuration results.

5. *Discussion and Examples*

The framework for an integrated conceptual design environment has been described in the previous two chapters. First, the conceptual modeling system (CMS) was described from a functional perspective. Next, an architectural view was provided. This chapter evaluates the framework against the original research premise that called for the blending of top-down and bottom-up approaches to conceptual design. As stated in Chapter 1, problem questions to be addressed in this evaluation include:

- How can top-down and bottom-up approaches be blended?
- Will the blended approach improve the quality of design?
- Will the blended approach reduce human resistance to the rigor of top-down design?
- Will the blended approach reduce the combinatorial explosion that is often present in bottom-up design?

The sections that follow respond to these questions in discussion and with examples. Power conversion systems provide the context for much of this discussion. The reader is encouraged to first review Appendix A for background and conceptual design considerations in power conversion, and the initial statement of an example problem.

In keeping with the original intent as well as grammatical correctness of the questions, the response to the first question will address "how" (i.e., the manner or method) top-down and bottom-up approaches can be blended. Only limited evaluation of the blended approach is offered in this section. The remaining three questions (each suggesting a possible merit or limitation) will address the evaluation of the blended methodology.

5.1 How can top-down and bottom-up approaches be blended?

Several features of the framework enable a blended conceptual design methodology. Summarized from previous discussion, the main features of this new framework include:

- ***functional modeling*** using a building-block approach; enabling knowledge capture, reuse, and functional decomposition for solution synthesis,
- a ***components knowledge-base*** supporting configuration; sizing and/or selecting components based on the conceptual model and company parts database, and
- an ***integrated design domain*** accessible from the functional modeling environment interconnecting tools, analysis routines, and data sources necessary for design synthesis, analysis, and evaluation.

The role and significance of each of these features in the blended methodology are highlighted in the examples that follow. The first example demonstrates the framework using a top-down approach. The second example addresses a bottom-up approach. Finally, these examples are considered when blending top-down and bottom-up approaches is indicated.

5.1.1 Example - Top-Down Approach

Referring to Appendix A, it is clear that the design of power conversion systems requires the consideration of numerous issues. Application, environmental conditions, packaging (size) constraints, power requirements, safety features, thermal and electrical protection, and control are just some of the obvious functional issues that must be addressed.

If not already specified in the requirements documentation from marketing, a top-down approach to designing a power conversion system begins with the determination of bridge topology (i.e., the type of bridge). The designer uses the function categories defined in CMS to locate the design function desired. Figure 47 shows an alphabetized list of function categories for the domain of power conversion systems. In this case, the designer selects "Power-bridge". CMS opens and displays the template of previously defined solutions (functional knowledge in the Expert Model, Section 4.2.2.4) associated with the power bridge function category. (A small portion of twenty-four feasible types of bridge topologies previously defined to the system is shown in Figure 48.)

The designer chooses to let the system advise which type of bridge is appropriate based solely on the design problem requirements rather than choosing a specific type. In this case, the designer selects and places the general building block associated with the power bridge function category in the modeling work area, fills in the requirement specification information, any additional selection criteria, and launches the configuration request. CMS then invokes inferencing (Apprentice System, Section 4.2.2.6) based on the Conceptual Model description (Section 4.2.2.3).

For instance, based on a requirements specification where a high system performance is needed (i.e., greater than 3000 horse power (hp) and greater than 4160 volts) and an application requiring the control of a synchronous motor, inferencing determines (using exists_when conditions of functional solutions in the Expert Model) that the bridge type should be an LCI (Load-Commutated Inverter). From this result, solution synthesis proceeds. With each pass, functional building-blocks are decomposed. Ultimately, functions are reduced to the atomic (component) level.

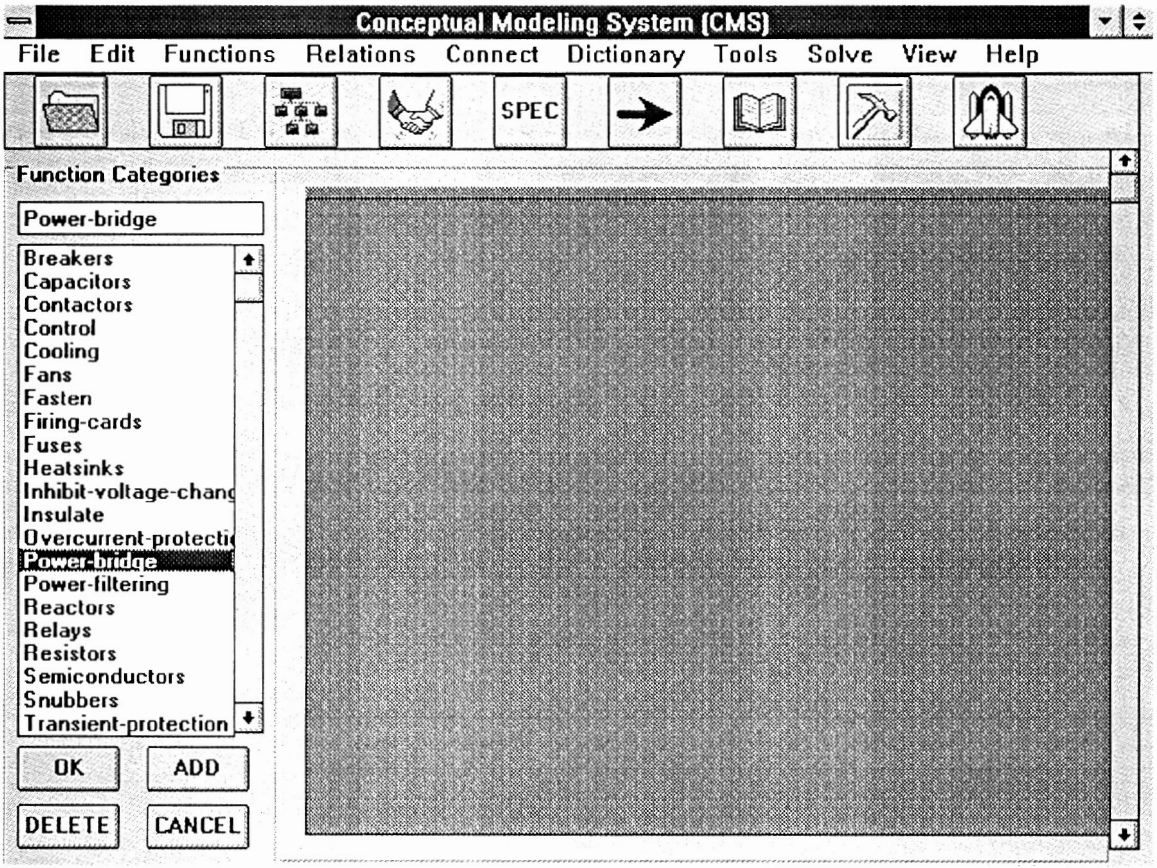


Figure 47. Locate and Select Function Category.

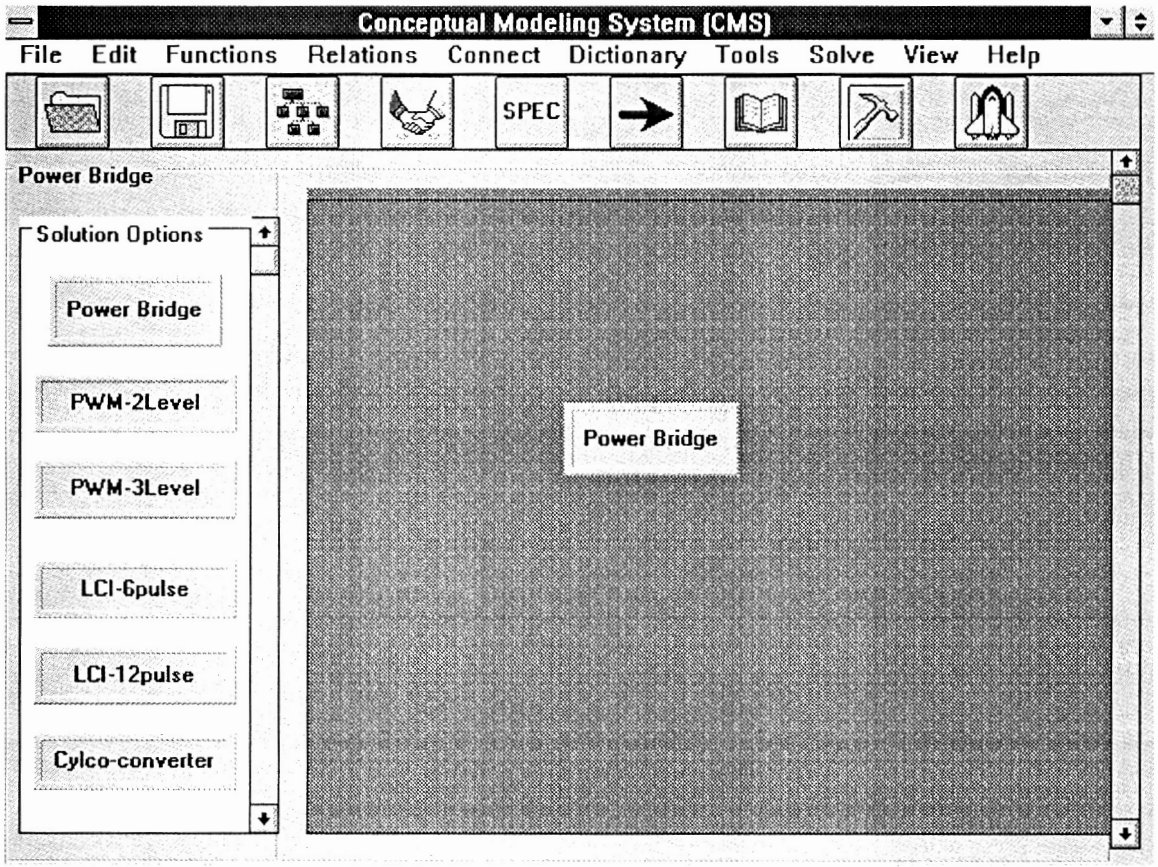


Figure 48. Top-down Functional Modeling.

Figure 49 illustrates how each step in the functional decomposition process is an independent mapping between the function space and the solution space. With each mapping, the specific solution is a result of exists_when conditions and function-solution relationships present in the Expert Model. As depicted in the first mapping of Figure 49, from all known solutions for Power Bridge, the LCI is selected based on exists_when conditions (e.g., performance and motor requirements).

Continuing with solution synthesis, the LCI is itself assembled from six functional building-blocks (transient protection, overcurrent protection, feedback, power conversion, power filtering, and control). Each of these building-blocks must then be considered. Figure 49 illustrates the continued decomposition of the power conversion building-block. As shown, it is assembled from power converter, control, cooling, and transient protection. Each of these functions is then considered. In the case of power converter, it is not decomposable (i.e., at the atomic/component level). Based on exists_when conditions (e.g., costs, application, rating) the thyristor solution is recommended as the solution for power converter. Still with a focus on the left side of the figure and a top-down approach, the decomposition process would continue through other sections, assembled functions, and components. In each case, decomposition continues until functional building-blocks are fully decomposed to the component level (atomic functions).

Turning attention to the right side of Figure 49, the Engineering Model (Section 4.2.2.5) refines the specification of component level functions based on sizing rules, selection rules, and/or rules defined for business practice standards. In this case, the results are dependent

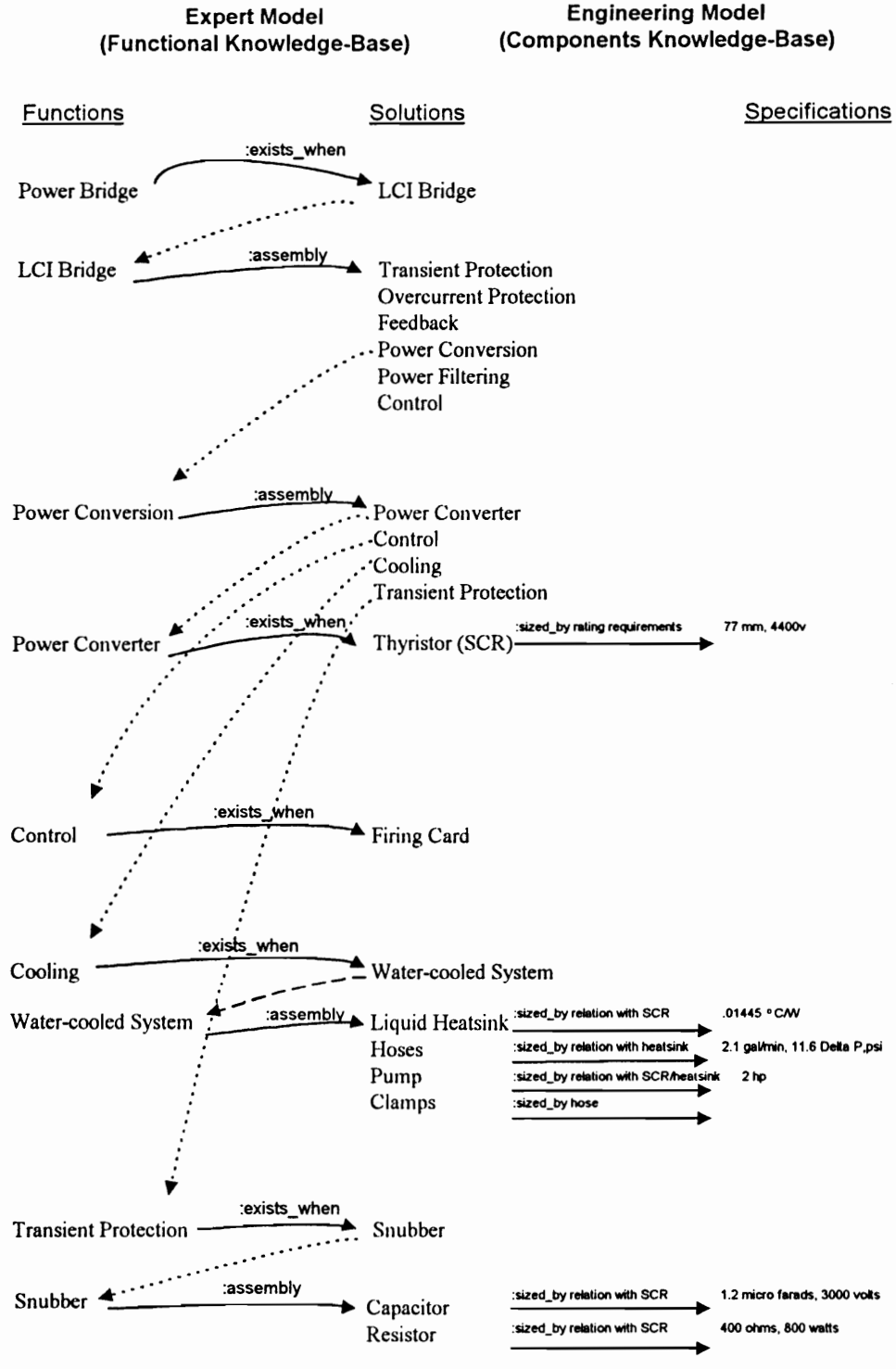


Figure 49. Synthesis and Functional Decomposition.

on the constraints prescribed by the requirements specification and any additional constraints provided before the solution request. Further, relationships with other building-blocks also affect the results of the specification process. Again referring to Figure 49, it can be seen that atomic level function solutions are mapped to specifications by component rules present in the Engineering Model (e.g., :sized_by).

This section has illustrated how the framework accommodates a top-down approach using a power bridge design example. This type of an approach provides the greatest degree of assistance to the design process, assures functional feasibility of solution instances, and poses the highest potential for automated processes. However, this approach does not accommodate solution concepts that deviate significantly from knowledge accumulated in the system. The next section examines how the framework accommodates a bottom-up approach to conceptual design.

5.1.2 Example - Bottom-Up Approach

CMS cannot "invent" knowledge (functional capabilities that are not known) for complex systems such as power converters. It can, however, provide the building-block and integrated environment to aid the design process for such cases.

For instance, consider the situation when requirements come from marketing for a power converter to fill a range of powers where there is no current product capable of covering this range. Unfortunately, it is clear there is a market demand for this range of powers as evident in the increased market share of a competitor following the introduction of a new product offering for this range of powers. In response, a new topology (bridge type) must be developed quickly.

Described in Appendix A, designing a *new* bridge topology requires the expertise of experienced designers, analysis and simulation programs, and a variety of information sources and company databases. In addition, outside vendors may participate in this type of development effort if requirements indicate the need for new components. This is common in power conversion applications where feasibility constraints may require custom heatsinks for cooling, or lead to requirements for new semiconductors to achieve new performance or reliability standards. A description of how CMS accommodates this need for a bottom-up approach follows. Clarifying this discussion, Figure 54 of Appendix A provides a pictorial representation of this highly iterative process.

As in the previous top-down example, the designer uses the function categories defined in CMS to locate design functions desired. In this case, there is no power bridge solution capable of fulfilling the power range. A bottom-up approach must be taken. Figure 50 shows an in-process model with one leg of a power bridge that is under development. As shown, several lower-level function building-blocks have been selected, placed, and interconnected in the modeling work area. To the left of the work area, Figure 50 shows the function category template for semiconductors. Again, rather than choosing a specific type of semiconductor, the general building-block ("Semiconductor" in this case) has been selected and placed in the model.

The designer fills in requirement specification information, additional selection criteria, highlights the semiconductor function in the model, and requests a configuration. The system responds as described previously; using exists_when conditions and rules for sizing and selection. Suppose that for this example, several semiconductor solution types are capable of fulfilling the rating requirements. The designer selects the thyristor (SCR)

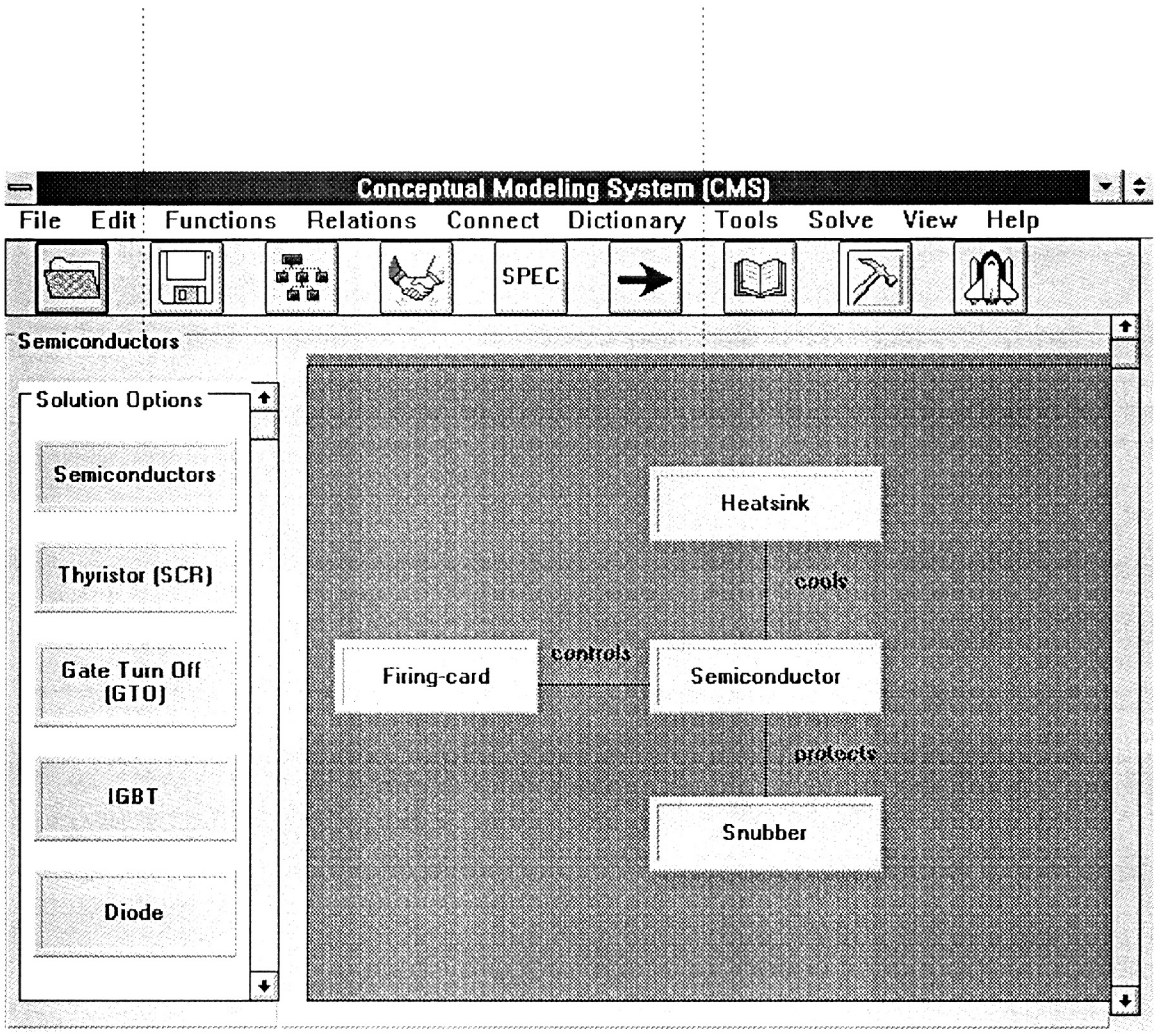


Figure 50. Bottom-up Functional Modeling.

and then resubmits the configuration request. This time, inferencing invokes the rules of the Engineering Model that size and select the appropriate SCR from the company database.

Next, the designer selects and requests configuration for the heatsink function building-block of the conceptual model. CMS returns with the specifications required for the heatsink, but finds no parts available in the company database capable of satisfying this requirement. Continuing to investigate, the designer submits the semiconductor and heatsink component specifications derived in CMS to thermal analysis programs. Following several iterations in analysis, the designer is satisfied that specifications for the heatsink will satisfy cooling requirements. Vendors of heatsinks can now be contacted with specifications.

Next, transient protection for the bridge leg (shown as "Snubber" in Figure 50) is considered. In a similar manner to previous iterations, the building-block is selected, constrained, submitted for configuration, and then analyzed. In this case however, snubber is a building-block one level above the component (atomic) level. Thus, CMS first determines the appropriate solution for snubber using exists_when conditions. Next, CMS considers the capacitor and resistor blocks defining the snubber function. Again, configuration using CMS provides the initial specifications of these components. Electrical simulations (circuit analysis) performed outside of CMS are then used to validate and refine component specifications. In the end, this iterative bottom-up process of specification, analysis, and evaluation results in a verified solution. The designer then uses CMS to capture this model as a functional building-block for reuse in future applications.

This section has illustrated how the framework accommodates a bottom-up approach. As in the top-down example, the example for bottom-up design was related to the design of a power bridge. As shown, the bottom-up approach is most appropriate when new designs are required that deviate significantly from past experience and knowledge. Although this type of approach is limited in the assistance provided from the Expert Model (functional knowledge), much of the highly iterative process of conceptualization is aided by the integrated framework. CMS assists the look-up of lower-level functions (components), sizing and selection tasks (Engineering Model), and access to supporting tools and data from the modeling environment. The next section examines blending top-down and bottom-up approaches to conceptual design.

5.1.3 Blending Top-Down and Bottom-Up Approaches

The previous two examples have addressed the design of power conversion systems from what would appear to be opposite directions. In truth, these examples are limited views of a more complete blended methodology.

Consider the bottom-up example. It is likely that a top-down approach was first attempted. When no solution was available for the power bridge based on the rating requirements, a bottom-up approach was indicated. Lower-level function building-blocks were combined as a new solution to the higher-level design problem was sought. CMS assisted modeling by coupling the low-level functional building-blocks placed in the model with a components knowledge-base (Engineering Model) to aid the configuration process. Designers were not picking parts from a database using a manual trial-and-error method and blindly experimenting; hoping to meet functional requirements. The modeling and synthesis process was assisted by component specifications that were generated as a function of problem requirements and constraints contained within the model description.

Thus, although building-blocks placed in the model were low-level (implying a bottom-up approach), the framework supported solution synthesis with a top-down approach (i.e., a systems approach driving requirements to solution).

The bottom-up example also demonstrated how a variety of functions and levels of abstraction can be blended in a single modeling situation. Heatsink, semiconductor, and firing-card were from differing functions, but all at the lowest level of abstraction. Snubber was one level higher and required decomposition prior to specification. This is but a small example of the variety permitted by the modeling system. The designer could select and combine functional building-blocks in a single model from a wide range of abstraction (high and low levels). In the case of power conversion systems, there is significant benefit to this capability when the design task moves beyond the power conversion section of the power bridge.

As discussed in the top-down example and in Appendix A, bridge design must also consider high-level functions such as transient protection, overcurrent protection, feedback, power filtering, and control. In many cases, these are functions where all or a part of the solution is common to other bridge types (topologies). It is therefore likely that functional building-blocks and solutions would already exist in CMS for these functions. The blended approach allows the designer to couple high-level functions (promoting reuse) with low-level functions (supporting new design) into a single integrated model.

The next three sections address the evaluation of the blended methodology. Quality of design is considered first. This is followed by discussion that focuses on evaluating the blended methodology against the limitations common to methods that are purely top-down

or bottom-up in approach. In particular, the human resistance to the rigor of top-down design and the combinatorial explosion often present in bottom-up design are considered.

5.2 Will the blended approach improve the quality of design?

Prior to responding to this question directly, one must qualify what is meant by "quality" in the context of engineering design. Without delving into to the wealth and variety of definitions currently available, the following terms/phrases typically associated with quality are offered:

- reduced defects
- reduced costs
- reduced time
- reliable
- satisfies requirements
- added features
- customer satisfaction

In the context of this research effort, quality of design is considered relevant to both the design process *and* the results of the design effort (i.e., products, systems, plans, processes, etc.). Defined in this manner, the response to this question would likely be "yes". The blended approach should improve the quality of design. Subsequent research will help to validate this conjecture. Several features of the blended approach lead to this conjecture. The following points highlight these features:

- The **building-block approach** that is central to the blended methodology provides for knowledge capture and reuse. As illustrated in the examples of this chapter, designers can create, save, and use functional knowledge in the form of modeling building-blocks. Modeling with functional building-blocks results in reduced cycle times with faster construction of applications and faster modification of existing applications. Building-blocks also promote greater consistency between designers in construction and use. This leads to reductions in time and costs for the initial modeling effort as well as the savings that consistency in design practices provides for manufacturing and long-term maintenance of products. Further, the blended approach allows quality building-blocks (verified with analysis and approved by veteran designers) to be saved as a corporate resource, reused, and shared with less experienced designers.

- The **integrated design environment** supports informed decisions early in the development process with access to analysis and simulation programs and company information resources. This type of environment is essential to quality; enabling not only the validation and evaluation of design alternatives based on feasibility, but also incorporating information directly related to reliability (product service records, vendor records, etc.).

- The **object-oriented framework** allows a systems approach to solution synthesis. Independent of the starting level of abstraction (low-level or high-level function), the system will always seek to find a solution using a top-down approach. Functional decomposition and component specification is driven by requirements and problem constraints toward solution.

At all levels of decomposition in the synthesis process, knowledge available in the framework provides more than a stagnant set of solutions that *have been* designed. Rather, the object-oriented structure of the framework supports functional decomposition with a knowledge structure that embodies the potential for what *can be* designed. Discussed with some detail in the examples of this chapter, each mapping between functional requirements (embodied in the model description) and design solutions (in the Expert Model) is independent. In each case, the system seeks to satisfy requirements and find the best solution based on current conditions. All solutions in the Expert Model are semantically equivalent (i.e., each solution is pushed into existence based on the particular modeling situation).

- Finally, the **decoupling of engineering knowledge from data** allows for continuous quality improvement. Component solutions and company parts databases are not "hard-wired". Rather, parts are selected from the database at the time of solution request. The Engineering Model determines component specifications and follows this specification with an independent retrieval from the company parts database. Without any changes to the functional model, future solutions of the model will likely improve as parts with improved performance and quality are added to the database.

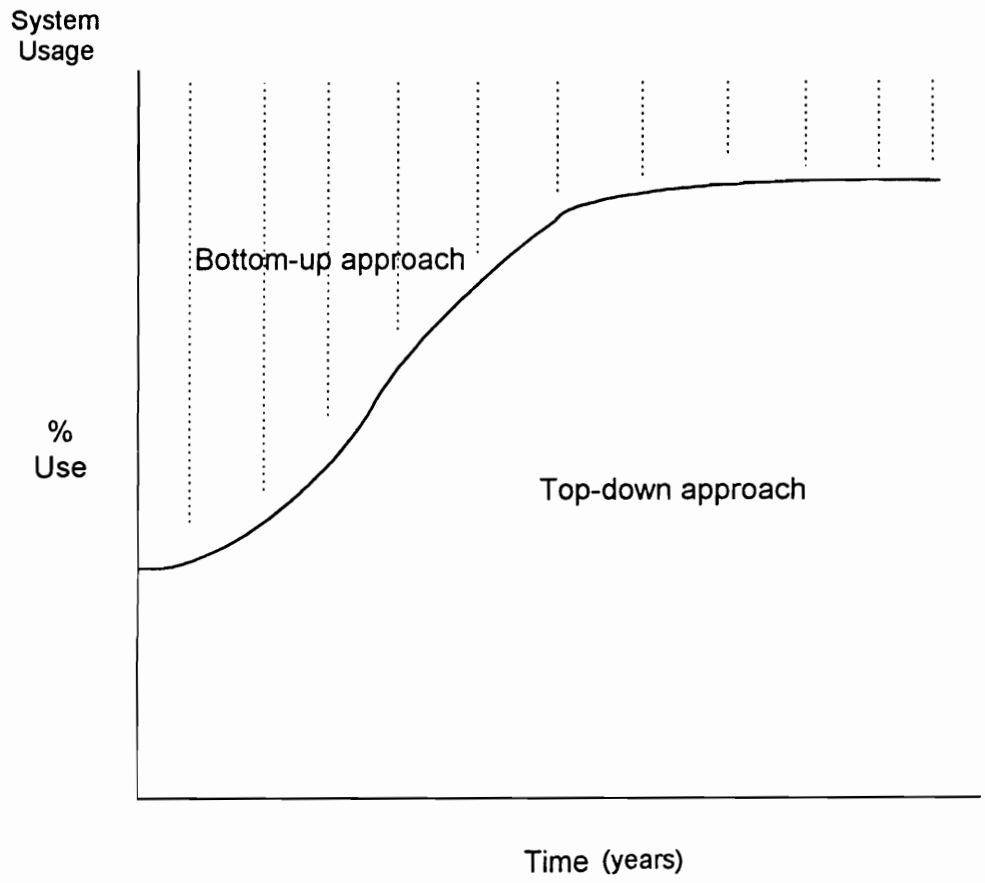
5.3 Will the blended approach reduce human resistance to the rigor of top-down design?

In a manner similar to the previous question, the meaning and context of terminology ("rigor" in this case) are considered prior to responding to the question directly. Referring to *Webster's* dictionary [Wool76], definitions of rigor include phrases such as: strict precision, exactness, rigid, and the quality of being unyielding or inflexible. In the context of top-down design, rigor implies a solution synthesis approach that is strictly

driven from the top through successive levels of refinement, ultimately to the low-level specifications of solutions capable of satisfying functional requirements. Discussed previously, although systems that have been developed to support top-down design assure adherence to functional requirements, there is no guarantee that solutions are realizable. As a result, designers abandon the top-down approach preferring instead to approach the problem from the bottom; configuring solutions from devices and then testing in hopes of meeting functional requirements.

Now responding to the question directly, the answer would likely be "yes". The blended approach should reduce human resistance to the rigor of top-down design. How? Primarily, the blended approach counters this resistance to top-down design through knowledge capture and reuse. Designers are encouraged by solution results drawn from a system that contains not only the expertise of respected designers, but solutions that are verified and known to be realizable. Figure 51 shows a graph predicting the percent use of top-down versus bottom-up approaches as knowledge accumulates over time. As shown, there will likely be a greater percentage of bottom-up design when the knowledge in the system is immature. Over time however, as knowledge builds and higher-level solutions are more plentiful, a top-down approach should predominate.

Posing a related question: Will the top-down approach eventually eliminate the need for bottom-up design? The answer would have to be that this is unlikely. There will always be a need for some amount of bottom-up design. Discovery and invention are not without experimentation. Again in the context of power conversion systems, consider as an example a gate turn-off (GTO) semiconductor. As one would expect, its normal application would be in the power conversion section of the bridge. Knowing how a GTO



** The usage of each approach is shown as a percentage of overall system usage

Figure 51. Percent Usage of Top-down versus Bottom-Up.

operates however, a savvy design engineer might consider its application as a motor disconnect in the overcurrent protection section of the bridge. This is certainly outside of the normal use of such a device, but one that might warrant investigation. The point to emphasize here is that no matter how automated or sophisticated, the design system must allow for this type of experimentation as new solutions to design problems are discovered and retained for reuse in the future. Again, this points to the need for a system, such as CMS, that accommodates a blending of top-down and bottom-up approach to design.

5.4 Will the blended approach reduce the combinatorial explosion that is often present in bottom-up design?

It is expected that the blended methodology will reduce the combinatorial explosion that is characteristic of a bottom-up approach to design. Three primary reasons account for this reduction. First, as discussed in the previous section, as the knowledge in the Expert Model matures, a larger percentage of the design process will be approached from the top. Inherently, the greater this top-down percentage, the lower the incidence of bottom-up practices. Second, when designing at the component level, often the component in question is only a small portion of a larger design task. Although a purely bottom-up approach is permissible, it is unlikely that a designer would choose such a process when quality solutions are available for other portions of the model. Third, as illustrated in the example problems and discussed in the response to the previous two questions, when a bottom-up approach is taken, component selection and sizing are assisted. Once the designer specifies a low-level function in terms of a component type, the rules defined in the Engineering Model perform the sizing and selection tasks associated with parts specification. The designer is relieved of the trial-and-error and often combinatorial tasks of manually picking parts.

6. CONCLUSIONS AND RECOMMENDATIONS

Remaining competitive in today's marketplace requires design processes that are conducive to creativity, yet have incorporated the consideration of other life-cycle issues. Design environments which support these objectives must overcome organizational and process boundaries. Notably, there is significant motivation for research which focuses on improving design processes from the *earliest* stages.

To date, advances have most often been for tasks that are well into the latter stages of product design. Developments for early design processes remain largely investigational, specialized, or rarely consider the requirements for abstraction *and* detail necessary in a concurrently engineered development process. In general, methodologies have taken either a top-down or a bottom-up approach to design, and as such, have virtually guaranteed the continued separation of abstraction and detail during conceptualization.

6.1 Contributions

As distinguished from other research in conceptual design, this dissertation work represents a novel approach that will serve to define and improve overall product realization processes. In particular, an integrated design environment capable of enabling a blending of top-down and bottom-up approaches has been proposed. Three primary mechanisms define the integrated design environment including: functional modeling, a components knowledge-base, and an integrated design domain.

Functional modeling is supported by the Expert Model where functional knowledge is categorized and stored for reuse. The designer applies this functional knowledge using a building-block approach in the construction of concept models. Following functional decomposition in solution synthesis, the Engineering Model performs sizing, selection,

and/or company standards rules associated with specification at the component level based upon problem requirements. In addition, informed design decisions are supported with a window environment permitting access to tools, analysis routines, and data sources necessary for design synthesis, analysis, and evaluation.

The benefits of the integrated design environment are numerous. Summarized here, features and associated benefits include:

- A building-block approach provides for knowledge capture and reuse, consistency in design practices, and the ability to visually model with functional objects during conceptualization. Building-blocks embody refined and verified engineering expertise and encourage a systems approach to design.

- The framework is customizable. Though demonstrated in the context of power conversion, the framework holds the potential for defining a rich object vocabulary for modeling in any application domain. With each new application domain explored, lower-level functions provide for freedom and flexibility, while higher-level functions provide the greatest benefit of reuse.

- An Engineering Model that automates the specification and selection of parts eliminates the need for manual trial-and-error practices. This greatly reduces the combinatoric effects often present in bottom-up design.

- An integrated modeling environment permits access to valuable sources of tools and data outside of the system. This supports not only reductions in design cycle-time, but improved performance for products that have considered valuable company data related to other life-cycle processes.

- The decoupling of knowledge and data sources provides for greater generalization, reduced system maintenance, and continuous quality improvement.

- And most significant, the framework enables blending top-down and bottom-up approaches to engineering design. This caters to both the need for abstraction and detail that is so important to early design practices; overcoming the difficulties experienced with the exclusive practice of either a top-down or bottom-up approach.

6.2 Recommendations

Several recommendations for future work and extensions of this research are possible, including:

- Automating the launch of analysis and simulation programs from the modeling environment. This would eliminate the need for redescribing problem attributes to outside programs.

- Spatial considerations of solution alternatives. This is always a concern as feasibility is often restricted due to geometric constraints. Integration with CAD is a most promising extension.

- Ability to work from multiple viewpoints. This would provide the ability to switch between views such as functional objects, electrical (schematics), mechanical (graphical representations), and cost models.

- Integration with cost models. This would provide for generalized evaluation of design based upon economic measures of merit.

- Refine the functional building-blocks for power conversion. Thus far, this has been limited to the prototype model. The benefits to completing this task and implementing CMS for power conversion could be great.

- Automate the generation of electrical schematics based on configuration results and model relationships. For power conversion, automating the generation of electrical schematics should be a readily obtainable extension. When decomposed, the functional object model contains the basic components to generate a schematic. Symbols and layout standards would need to be developed.

- Demonstrate the framework for additional application domains. Much is to be gained by investigating other application domains. This would reinforce the concepts of the framework and perhaps lend insight into additional requirements that have yet to be discovered. Inherently, the study of any application domain would lend insight into knowledge collection and knowledge organization.

Many of the above extensions and recommendations are broad problem areas yet to be resolved. Others, however, can be accomplished in the more immediate future. As an example, consider knowledge collection and organization. Though the benefits of a components and building block architecture are significant, its creation for a given application area can be a challenging problem. Doing so requires discovering, categorizing, parameterizing, and understanding the interrelationships of functions characteristic of the application domain. Additional work is required to understand, generalize, and assist this process.

In conclusion, the future for enabling technologies for engineering design remains a bright one, and one for which the possibilities are numerous. Technologies that are capable of supporting top-down and bottom-up approaches in an integrated problem solving environment are crucial to improved design processes. This is particularly important for the earliest stages of conceptual design. The methods presented in this dissertation represent one promising basis for these future advancements.

REFERENCES

[Aase92] Aase, Jan H. "Standards for Model and Feature Libraries". *Proceedings - CRD Symposium on the Revolution of Knowledge-Based Engineering in Mechanical Design*, GE Corporate Research and Development, Schenectady, NY, May 1992.

[ACC96] Advanced Concepts Center of Lockheed Martin (for Rational Software Corporation), "Object-Oriented Analysis and Design Using the Booch Method", *Course Lecture Notes*, Atlanta, Georgia, March 12-15, 1996.

[Agog89] Agogino, A.M., Cagan, J., and Molezzi, M.J. "Meta-Design: Reflections on a Graduate Course in Design Theory and Methodology". *Proceedings of the 1988 NSF Grantee Workshop on Design Theory and Methodology*. Springer-Verlang, New York, 1989, pp. 18-28.

[Bahr92] Bahrami, A. "Computer-assisted conceptual design by utilizing fuzzy logic and quality function deployment". *Journal of Design and Manufacturing*, v 2, n 3, 1992, pp. 155-165.

[Balk93] Balkany, A., Birmingham, W.P., and Tommelein, I.D. "An Analysis of Several Configuration Design Systems". *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AI EDAM)*, v 7, n 1, 1993, pp. 1-17

[Banc92] Bancroft C.E., et al. General Product Design, in Bakerjian R. (eds): *Tool and Manufacturing Engineers Handbook*, v 6, ed 4. Dearborn, Society of Manufacturing, 1992, chap 10.

[Bard93] Bardasz, T. and Zeid, I. "DEJAVU: Case-Based Reasoning for Mechanical Design". *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AI EDAM)*, v 7, n 2, 1993, pp. 111-124.

[Booc94] Booch, Grady, *Object-Oriented Analysis and Design with Applications, 2nd Edition*, Benjamin/Cummings Publishing Company, Inc., Redwood City, California, 1994.

[Boot83] Boothroyd, G. and Dewhurst, P., *Design for Assembly - A Designer's Handbook*, Department of Mechanical Engineering, University of Massachusetts at Amherst, 1983.

[Bowe91] Bowen, J. and Bahler, D. "Supporting cooperation between multiple perspectives in a constraint-based approach to concurrent engineering". *Journal of Design and Manufacturing*, v 1, n 2, 1991, pp. 89-105.

[Boyl93] Boyle, J.M. "Interactive engineering systems design: a study for artificial intelligence applications" *Artificial Intelligence in Engineering Design*, ed. R.A. Adey, Computational Mechanics Publications, Great Britain, v 12, pp. 50-61.

[Brad93] Bradley, D.A., Bracewell, R.H., and Chaplin, R.V. "Engineering Design and Mechatronics: The Schemebuilder Project". *Research in Engineering Design*, v. 4, n. 4, 1993, pp. 241-248.

[Brad91] Bradshaw, J.A. and Young, R.M. "Evaluating Design Using Knowledge of Purpose and Knowledge of Structure". *IEEE Expert*, v 6, n 2, 1991, pp. 33-40.

[Bree93] Breedveld, P.C. "Computer-aids for modeling and conceptual design". *Proceedings of the 1993 International Conference on Systems, Man and Cybernetics*, v 2, pp. 209-215.

[Caga91] Cagan, J. and Agogino, A.M. "Inducing Constraint Activity in Innovative Design". *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AI EDAM)*, v 5, n 1, 1991, pp. 47-61.

[Cata91] Catania, G. "Form-features for Mechanical Design and Manufacturing". *Journal of Engineering Design*, 2(1), 1991, pp. 21-43.

[Chak94] Chakrabarti, A. and Bligh, T.P. "An Approach to Functional Synthesis of Solutions in Mechanical Conceptual Design. Part I: Introduction and Knowledge Representation". *Research in Engineering Design*, v 6, n 3, 1994, pp. 127-141.

[Chan88] Chang T.C., Anderson D.C., and Mitchell O.R. "QTC - an Integrated Design/Manufacturing/Vision Inspection System for Prismatic Part", *Proceedings of the ASME 1988 Computers in Engineering Conference*, v 1, 1988, pp. 417-426.

[Chan90] Chang, Tien-Chien. *Expert Process Planning for Manufacturing*. Reading, Mass: Addison-Wesley, 1990.

[Chan92] Chang, Tien-Chien, and Anderson, D.C. "Quick Turnaround Cell - An Integration of Feature Based Design and Process Planning", *Proceedings - 1992 First Industrial Engineering Research Conference*, Chicago, 1992, pp. 83-85.

[Colt90] Colton, J.S., and Dascanio, J.L. "Integrated, intelligent design environment". *Proceedings of ASME International Computers in Engineering Conference and Exposition*, Boston, Aug. 1990, pp. 9-15.

[Colt91] Colton, J.S. and Dascanio, J.L. II "An Integrated, Intelligent Design Environment". *Engineering with Computers*, v 7, n 1, 1991, pp. 11-22.

[Colt94a] Colton, J.S., and Pun, R.C. "Information Frameworks for Conceptual Engineering Design". *Engineering with Computers*, v 10, n 1, 1994, pp. 22-32.

[Colt94b] Colton, J.S., and Ouellette, M.P. "A Form Verification System for the Conceptual Design of Complex Mechanical Systems". *Engineering with Computers*, v 10, n 1, 1994, pp. 33-44.

[Cond92] Condoor, S.S., Shankar, S.R., Brock, H.R., Burger, C.P., and Jansson, D.G. "A Cognitive Framework for the Design Process". *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 277-281.

[Culv93] Culverhouse, P.F. "Four design routes in electronics engineering product development". *Journal of Design and Manufacturing*, v 3, n 2, 1993, pp. 147-158.

[Cunn88] Cunningham J.J. and Dixon J.R. "Designing with features: the origin of features", *Proceedings of the ASME 1988 Computers in Engineering Conference*, v 1, 1988, pp. 237-243.

[Dice93] Dicesare, F.P. "Re-engineering to achieve a concurrent engineering environment". *Journal of Design and Manufacturing*, v 3, n 2, 1993, pp. 75-89.

[Dima93] Dimarogonas, A.D. "On the Axiomatic Foundation of Design". *Proceedings - 5th International Conference on Design Theory and Methodology*, 1993 ASME Design Technical Conferences, Albuquerque, New Mexico, Sept. 1993, DE v 53, pp. 253-258.

[Dixi89] Dixit, Shrikant. *Expert System for Machinability Data Integrated with a CAD System*, Master of Science Thesis, Mechanical Engineering, VPI&SU, August 1989.

[Dixo88] Dixon, J.R., Duffey, M.R., Irani, R.K., Meunier, K.L., and Orelup, M.F. "A Proposed Taxonomy of Mechanical Design Problem". *Proceedings of the ASME 1988 Computers in Engineering Conference*, v 1, 1988, pp. 41-46.

[Dixo89] Dixon, J.R. "On Research Methodology Towards a Scientific Theory of Engineering Design". *Proceedings of the 1988 NSF Grantee Workshop on Design Theory and Methodology*. Springer-Verlang, New York, 1989, pp. 316-337.

[Dixo91] Dixon, J.R., Nielsen, E.H., Orelup, M.F., and Welch, R.V. "Computer-Based Design Process Models and Feature-Based Design Object Representations: A Research Progress Report". *Proceedings of the 1991 NSF Design and Manufacturing Systems Conference*, Univ. of Texas at Austin, Jan. 1991, pp.995-1000.

[Dixo92] Dixon, J.R. "Why We Need Doctoral Programs in Design". *Mechanical Engineering*, February 1992, pp. 75-79.

[Dome94] Domeshek, E.A., Herndon, M.F., Bennett, A.W., and Kolodner, J.L. "A Case-Based Design Aid for Conceptual Design of Aircraft Subsystems". *Proceedings - Tenth Conference on Artificial Intelligence Applications*, IEEE Computer Society, San Antonio, March 1994, pp. 63-69.

[Duff93] Duffy, A. "Bibliography - Artificial Intelligence in Design". *Artificial Intelligence in Engineering Design*, v 12, pp. 30-36.

[Elli92] Elliott, M. "Application of Wisdom Systems to Mechanical Design Automation of Transformers", *Proceedings - CRD Symposium on the Revolution of Knowledge-Based Engineering in Mechanical Design*, GE Corporate Research and Development, Schenectady, NY, May 1992.

[Eppi94] Eppinger, S.D., Whitney, D.E., Smith, R.P. and Gebala, D.A. "A Model-Based Method for Organizing Tasks in Product Development". *Research in Engineering Design*, v 6, n 1, 1994, pp. 1-13.

[Ever82] Eversheim W., Abolins G. and Buchholz G. *Computer Aided Generation of Workshop Drawings of Sheet Metal Parts*, Paris, MICAD; Aachen, RWTH Aachen, 1982.

[Fabr92] Fabrycky, W.J. "Indirect Experimentation for System Optimization: A Paradigm Based on Design Dependent Parameters", *Proceedings, Second Annual International Symposium, National Council on Systems Engineering*, Seattle, Washington, July, 1992.

[Fauv94] Fauvel, O.R. "On the Distinction Between Design Parameters and Functional Requirements". *Proceedings - 6th International Conference on Design Theory and Methodology*, 1994 ASME Design Technical Conferences, Minneapolis, Minnesota, Sept. 1994, DE v 68, pp. 371-372.

[Fave93] Favela, J., Wong, A., and Chakravarthy, A. "Supporting Collaborative Engineering Design". *Engineering with Computers*, v 9, n3, 1993, pp. 125-132.

[Fing90] Finger, S. and Safier, S.A. "Representing and Recognizing Features in Mechanical Designs". *Proceedings - Design Theory and Methodology Conference*, ASME, Chicago, Sept. 1990, DE v 27, pp. 19-25.

[Gadh89] Gadh R., Hall M.A., Gursoz E.L., Prinz F.B. "Knowledge driven manufacturability analysis from feature based representations", in *Winter Annual Meeting of ASME*, DE v 21, PDE v 36, San Francisco, ASME, 1989, pp 21-34.

[GEDS96] General Electric Company, GE Motors and Industrial Systems, Salem, VA.

[Gero92] Gero, J.S., Tham, K.W., and Lee, H.S. "Behaviour: A Link Between Function and Structure in Design," *Intelligent Computer Aided Design*, D.C. Brown, M. Waldron and H. Yoshikawa (Editors), Elsevier Science Publishers B.V. (North-Holland), pp. 193-225, 1992.

[Glic91] Glicksman, J., Hitson, B.L., Pan, Y.-C., and Tenenbaum, J.M. "MKS: A conceptually centralized knowledge service for distributed CIM environments". *Journal of Intelligent Manufacturing*, v 2, n 1, 1991, pp. 27-42.

[Grew92] Grewal, Sukhminder. "ICAD, Wisdom, & CAD/CAM/CAE - a Corporate Perspective". *Proceedings CRD Symposium on The Revolution of Knowledge-Based Engineering in Mechanical Design*. Schenectady, NY. May 14-15, 1992.

[Grig93] Grigely, L.J. and Billatos, S.B. "Case Based Reasoning Approach to Concurrent Engineering". *Intelligent Concurrent Design: Fundamentals, Methodology, Modeling and Practice. Winter Annual Meeting of ASME*, DE v 66, New Orleans, ASME, Nov. 1993, pp 83-95.

[Gu89] Gu, P.H., ElMaraghy, H.A., and Hamid, L. "FDDL: A Feature Based Design Description Language". *Proceedings of the Design Theory and Methodology Conference*, ASME, Quebec, Sept. 1989.

[Hern91] Hernandez, J.A., Luby, S.C., Hutchins, P.M., Leung, H.W., Gustavson, R.E., De Fazio, T.L., Whitney, D.E., Nevins, J.L., Edsall, A.C., Metzinger, R.W., Tung, K.K., and Peters, T.J. "An Integrated System for Concurrent Design Engineering". *Proceedings - Seventh Conference on Artificial Intelligence Applications*, IEEE Computer Society, Miami, Feb. 1991, pp. 205-211.

[Heum86] Heumann, K. *Basic Principles of Power Electronics*, Berlin, Springer-Verlag, 1986.

[Hoov94] Hoover, S.P. and Rinderle, J.R. "Abstractions, Design Views and Focusing". *Proceedings - 6th International Conference on Design Theory and Methodology*, 1994 ASME Design Technical Conferences, Minneapolis, Minnesota, Sept. 1994, DE v 68, pp. 115-129.

[Hubk82] Hubka, V. *Principles of Engineering Design*, Butterworth Scientific, London, 1982.

[Hund90] Hundal, M.S. and Byrne, J.F. "Computer-Aided Generation of Function Block Diagrams in a Methodical Design Procedure". *Proceedings - Design Theory and Methodology Conference*, ASME, Chicago, Sept. 1990, DE v 27, pp. 251-257.

[Hund91] Hundal, M.S. "Use of Functional Variants in Product Development", *Proceedings - 3rd International Conference on Design Theory and Methodology*, 1991 ASME Design Technical Conferences, Miami, Sept. 1991, DE v 31, pp. 159-164.

[Hund92] Hundal, M.S., and Langholtz, L.D. "Computer-Aided Conceptual Design: An application of X Windows, with C". *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 1-9.

[Huss89] Husslage, Robin. "Putting conceptual design to work". *CAE, Computer-Aided Engineering*, 8(10), 1989, p. 3.

[Huss90] Husslage, Robin. 'Parametrics. A software link". *Tooling & Production*, 55(12), 1990, p.4.

[Ishi93] Ishii, K. 'Modeling of Concurrent Engineering Design". *Concurrent Engineering: Automation, Tools, and Techniques*, (ed) Kusiak, A., New York, John Wiley & Sons, Inc., 1993, pp. 19-39.

[Ishi92] Ishii, K. and Miller, R.A. 'Design Representation for Manufacturability Evaluation in CAD: Beyond Feature-Based Design". *Proceedings of ASME International Computers in Engineering Conference and Exposition*, v-1, 1992, pp. 37-44.

[Iyen92] Iyengar, G., Lee, C.-L., and Kota, S. "Towards an Objective Evaluation of Alternative Designs". *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 19-25.

[Jaco94] Jacobson, I. *Object-Oriented Software Engineering, A Use Case Driven Approach*. New York, Addison-Wesley, 1994.

[Jaki89] Jakiela, M.J. and Papalambros, P.Y. "Concurrent Engineering with Suggestion-Making CAD Systems: Results of Initial User Tests", *Advances in Design Automation*, ASME, DE v. 19-1, 1989, pp. 223-230.

[Jone91] Joneja A., Chang T.C. 'Search anatomy in feature-based automated process planning", *Journal of Design and Manufacturing*, v 1, 1991, pp. 7-15.

[Kapu94] Kapur, K.C. "Robust design, manufacturing and concurrent engineering". *Journal of Design and Manufacturing*, v 4, n 1, 1994, pp. 31-39.

[Keir90] Keirouz, W., Pabon, J., and Young, R. "Integrating Parametric Geometry, Features, and Variational Modeling for Conceptual Design," *Proceedings - Design Theory and Methodology Conference*, ASME, Chicago, Sept. 1990, DE v 27, pp. 1-9.

[Kim90] Kim, Won. *Introduction to object-oriented databases.* Cambridge, Mass: The MIT Press, 1990.

[Kim91] Kim, Joo-Yong, O'Grady, P., and Young, R.E. "A Feature Taxonomy for Rotational Parts". *Technical Report 91-01*, Department of Industrial Engineering, North Carolina State University, 1991.

[Knig91] Knight, T.P, and Kim, S.H. "A Knowledge System for Integrated Product Design". *Journal of Intelligent Manufacturing*, v 2, n 1, 1991, pp. 17-25.

[Kolb93] Kolb, Mark A. "FRODO: Constraint-Based Object-Modeling for Preliminary Design". *Advances in Design Automation*, ASME, DE v. 65-1, 1993, pp. 307-318.

[Kota90] Kota, S. and Ward, A.C. "Functions, Structures, and Constraints in Conceptual Design". *Proceedings - Design Theory and Methodology Conference*, ASME, Chicago, Sept. 1990, DE v 27, pp. 239-250.

[Kota92] Kota, S. and Chiou, S.-J. "Design Representation and Computational Synthesis of Mechanical Motions". *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 365-372.

[Kota93] Kota, S. and Lee, C.-L. "Genreal Framework for Configuration Design: Part 1 - Methodology". *Journal of Engineering Design*, v 4, n 4, 1993, pp. 277-289.

[Kott94] Kott, G., Gabriele, G.A., and Korngold, J. "Power Converter Design with Coupled Thermal Analysis using Multidisciplinary Design Optimization". *Advances in Design Automation*, ASME, DE v. 69-2, 1994, pp. 187-194.

[Kusi93] Kusiak, A. and Szczerbicki, E. "Transformation from Conceptual to Embodiment Design". *IIE Transactions*, v 25, n 4, 1993, pp. 6-12.

[Kutt93] Kuttig, D. "Potential and Limits of Functional Modeling in the CAD Process". *Research in Engineering Design*, v 5, n 1, 1993, pp. 40-48.

[Lawl89] Lawlor-Wright, T. and Hannam, R.G. "A feature-based design for manufacture: CADCAM package". *Computer-Aided Engineering Journal*, v 6, n 6, 1989, pp. 215-220.

[Lee91] Lee, H.H. and Arora, J.S. "Object-Oriented Programming for Engineering Applications". *Engineering with Computers*, v 7, n 4, 1991, pp. 225-235.

[Lee87] Lee Y.C. and Fu K.S. "Machine understanding of CSG: extraction and unification of manufacturing features". *IEEE Computer Graphics and Applications*, v 7, n 1, 1987, pp. 20-32.

[Liba88] Libardi, E.C. Jr., Dixon, J.R., and Simmons, M.K. "Computer Environments for the Design of Mechanical Assemblies: A Research Review". *Engineering with Computers*, v 3, n 3, 1988, pp. 121-136.

[Lin93] Lin, W. and Myklebust, A. "A Constraint Driven Solid Modeling Open Environment". *Proceedings of the Second ACM/IEEE Symposium on Solid Modeling and Applications*, Montreal, Canada, May 19-21, 1993..

[Lu88] Lu, S. C-Y., and Subramanyam, S. "A Computer-Based Environment for Simultaneous Product and Process Design". *Advances in Manufacturing Systems Engineering. Winter Annual Meeting of ASME*, v 31, Chicago, ASME, Nov. 1988, pp 35-46.

[Mahe90] Maher, M.L., "Process Models for Design Synthesis". *AI Magazine*, v 11, n 4, 1990, pp.49-58.

[Mahe93] Maher, M.L. and Zhang, D.M. "CADSYN: A Case-Based Design Process Model". *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AI EDAM)*, v 7, n 2 1993, pp. 97-110..

[Mart93] Martin, J. *Principles of Object-Oriented Analysis and Design*, Englewood Cliffs, Prentice-Hall, Inc. 1993.

[Maye94] Mayer, R.J., Su, C.J., Sun, T.-L., and Wysk, R.A. "ECTOF: a feature representation technique for concurrent engineering applications". *Journal of Design and Manufacturing*, v 4, n 1, 1994, pp. 49-65.

[Micr95] Microsoft Corporation. *Microsoft Visual Basic Programming System for Windows*, Version 4.0, United States, 1995.

[Mile93] Mileham, A.R., Currie, G.C., Miles, A.W., and Bradford, D.T. "A parametric Approach to Cost Estimating at the Conceptual Stage of Design". *Journal of Engineering Design*, v 4, n 2, 1993, pp. 117-125.

[Mile94] Miles, J. and Moore, C. *Practical Knowledge-Based Systems in Conceptual Design*. London, Springer-Verlag, 1994.

[Mill92] Miller, G.S. and Colton, J.S. "The Complementary Roles of Expert Systems and Database Management Systems in a Design for Manufacture Environment". *Engineering with Computers*, v 8, n 3, 1992, pp. 139-149.

[Mohd94] Mohd-Hashim, F., Juster, N.P., and de Pennington, A. "A Functional Approach to Redesign". *Engineering with Computers*, v 10, n 3, 1994, pp. 125-139.

[Mull91] Mullins, S.H. and Anderson, D.C. "Feature-based tolerance representation for design and analysis". *Journal of Design and Manufacturing*, 1(2), 1991, pp. 107-118.

[Mura92] Murakami, T. and Gossard, D.C. "Mechanism Concept Retrieval by Behavioral Specification Using Configuration Space", *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 343-350.

[Navi91] Navinchandra, D., Sycara, K.P., and Narasimhan, S. "A Transformational Approach to Case-Based Synthesis", *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AI EDAM)*, v 5, n 1, 1991, pp. 31-45.

[Nevi89] Nevins, J.L. and Whitney, D.E., et al., *Concurrent Design of Products and Processes*, McGraw-Hill Publishing Company, New York, 1989, pp. 1-24.

[News91] Newsom, S.L. and Spillers, W.R. "Computer Tools for Conceptual Design: What Do Expert Designers Need?" *Proceedings of the 1991 NSF Design and Manufacturing Systems Conference*, Univ. of Texas at Austin, Jan. 1991, pp. 1123-1125.

[Niel91] Nielsen, E.H., Dixon, J.R. and Zinsmeister, G.E. "Capturing and using designer intent in a design-with-features system". *Proceedings - 3rd International Conference on Design Theory and Methodology*, 1991 ASME Design Technical Conferences, Miami, Sept. 1991, DE v 31, pp. 95-102.

[Nnaj90] Nnaji B.O. and Kang T. "Interpretation of CAD models through neutral geometric knowledge", *AI EDAM*, v 4, n 1, 1990, pp. 15-45.

[Nnaj91] Nnaji B.O., Kang T., Yeh S., and Chen J., "Feature reasoning for sheet metal components", *International Journal of Production Research*, v 29, n 9, 1991, pp.1867-1896.

[Nobl93] Noble, J.S. and Tanchoco, J.M.A. "Design for Economics". *Concurrent Engineering: Automation, Tools, and Techniques*, (ed) Kusiak, A., New York, John Wiley & Sons, Inc., 1993, pp. 401-435.

[NRC91] National Research Council. *Improving Engineering Design, Designing for Competitive Advantage*, Committee on Engineering Design Theory and Methodology Manufacturing Studies Board, National Academy Press, 1991.

[OGra91a] O'Grady P., and Young, R.E. "Issues in concurrent engineering systems". *Journal of Design and Manufacturing*, v 1, n 1, 1991, pp. 27-34.

[OGra91b] O'Grady P., Young R.E., Greef A., Smith L. "An advice system for concurrent engineering", *International Journal of Computer Integrated Manufacturing*, v 4 (2), 1991, pp. 63-70.

[OSha92] O'Shaughnessy, K. and Sturges, R.H. Jr. "A Systematic Approach to Conceptual Engineering Design". *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 283-291.

[Padh90] Padhy, S.K., Sharan, R., and Dwivedi, S.N. "Feature based approach for casting design". *Proceedings - Design Theory and Methodology Conference*, ASME, Chicago, Sept. 1990, DE v 27, pp. 113-118.

[Pahl88] Pahl, G., and Beitz, W. *Engineering Design - A Systematic Approach*, Springer-Verlag, New York, 1988.

[Park94] Park, H., Cutkosky, M.R., Conru, A.B., and Lee, S. "An Agent-Based Approach to Concurrent Cable Harness Design". *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AI EDAM)*, v 8, n 1, 1994, pp. 45-61

[Pars88] Parsaye, Kamran, and Chignell, Mark. *Expert Systems for Experts*. New York: John Wiley & Sons, 1988.

[Pars89] Parsaye, Kamran, Chignell, Mark, Khoshafian, Setrag, and Wong, Harry. *Intelligent Databases*. New York: John Wiley & Sons, 1989.

[Papa91] Papanikolopoulos, N.P. "FORS: A Software Tool for Flexible Design". *Journal of Intelligent Manufacturing*, v 2, n 1, 1991, pp. 5-15.

[PazS89] Paz-Soldan, J.P. and Rinderle, J.R. "The Alternate Use of Abstraction and Refinement in Conceptual Mechanical Design", *Proceedings of the ASME Winter Annual Meeting*, ASME, San Francisco, CA, December 1989.

[Pete92] Peters, T.J. "Combinatorial Analysis of Feature Recognition", *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 91-97.

[Pu93] Pu, Pearl. "Introduction: Issues in Case-Based Design Systems". *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AI EDAM)*, v 7, n 2 1993, pp. 79-85.

[Rati95] Rational Software Corporation, *Rational Rose 3.0*, Santa Clara, California, 1995.

[Rind89] Rinderle, J.R., Colburn, E.R., Hoover, S.P., Paz-Soldan, J.P., and Watton, J.D. "Form-Function Characteristics of Electro-Mechanical Designs". . *Proceedings of the 1988 NSF Grantee Workshop on Design Theory and Methodology*. Springer-Verlang, New York, 1989, pp. 132-147.

[Rind90] Rinderle, J.R. and Balasubramaniam, L. "Automated Modeling to Support Design". *Proceedings - Design Theory and Methodology Conference*, ASME, Chicago, Sept. 1990, DE v 27, pp. 281-290.

[Rizz93] Rizzoni, G. *Principles and Applications of Electrical Engineering.* Illinois, Richard D. Irwin, Inc., 1993.

[Rode84] Rodenacker, W.G. *Methodisches Konstruieren (Methodical Designing)*, Springer-Verlag, Berlin, New York, III ed., 1984.

[Rode93] Roderman, S. and Tsatsoulis, C." PANDA: A Case-Based System to Aid Novice Designers". *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing (AI EDAM)*, v 7, n 2 1993, pp. 125-133.

[Rose91] Rosen, D.W., Dixon, J.R. and Dong, X. "Methodology for conversions of feature-based representations". *Proceedings - 3rd International Conference on Design Theory and Methodology*, 1991 ASME Design Technical Conferences, Miami, Sept. 1991, DE v 31, pp. 45-51.

[Rose92a] Rosen, D.W., Dixon, J.R., and Finger, S. "Conversions of Feature-Based Representations via Graph Grammar Parsing". *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 83-90.

[Rose92b] Rosen, D.W., Dixon, J.R., and Poli, C. "Features and Algorithms for Tooling Cost Evaluation in Injection Molding and Die Casting". *Proceedings of ASME International Computers in Engineering Conference and Exposition*, v 1, 1992, pp. 45-52.

[Sakt93] Sakthivel, T.S. and Kalyanaraman, V. "A KBES for Integrated Engineering". *Engineering with Computers*, v 9, n 1, 1993, pp. 1-16.

[Salu94] Salustri, F.A., and Venter, R.D. "A New Programming Paradigm for Engineering Design Software". *Engineering with Computers*, v 10, n 2, 1994, pp. 95-111.

[Schm89] Schmekel, H. *A System for Conceptual Design of Mechanical Products.* Licentiate Thesis, Computer Systems for Design and Manufacturing, Dept. of Manufacturing Systems, The Royal Institute of Technology, 1989.

[Schm93] Schmidt, L.C. and Cagan, J. "Recursive Annealing: A Computational Model for Machine Design". *Proceedings - 5th International Conference on Design Theory and Methodology*, 1993 ASME Design Technical Conferences, Albuquerque, New Mexico, Sept. 1993, DE v 53, pp. 243-251.

[Schu92] Schulte, R.M., Padmanabhan, S., and Devgun, M.S. "Feature-driven, process-based approach to the integration of CAD/CAM in wireframe models". *International Journal of Production Research*, 30(5), 1992, pp. 1005-1028.

[Schu93] Schulte, M., Weber, C. and Stark, R. "Functional features for design in mechanical engineering", *Computers in Industry*, v 23, 1993, pp. 15-24.

[Seki94] Sekimoto, S. and Makoto, U. "A Study of Creative Design Based on the Axiomatic Design Theory". *Proceedings - 6th International Conference on Design Theory and Methodology*, 1994 ASME Design Technical Conferences, Minneapolis, Minnesota, Sept. 1994, DE v 68, pp. 71-77.

[Shaw86] Shaw, M.C., "Creative Design", *CIRP Annals*, v 35, n 2, 1986, pp. 461-465.

[Shig77] Shigley, J.E. *Mechanical Engineering Design*, McGraw-Hill, 3rd Edition, 1977.

[Simo91] Simoudis, E. Guest Editorial, "Frameworks for Integrating Design and Manufacturing Knowledge". *Journal of Intelligent Manufacturing*, v 2, n 1, 1991, pp. 1-4.

[Snav93] Snavely, G.L. and Papalambros, P.Y. "Abstraction as a Configuration Design Methodology". *Advances in Design Automation*, ASME, DE v. 65-1, 1993, pp. 297-305.

[Stat93] Staton, D.A., McGilp, M.I., and Miller, T.J.E. "Interactive Computer Aided Design of Permanent Magnet DC Motors". *IAS '93 - Conference Record of the 1993 IEEE Industry Applications Conference Twenty-Eighth IAS Annual Meeting*, Part I, pp. 217-225.

[Stev95] Stevenson, A. "Power Conversion Course Class Notes". Lecture Notes in Power Conversion. General Electric Company, Drive Systems. Salem, Virginia. January 1995.

[Subr92] Subramaniam, B. and Ulrich, K. "Representation of Producibility Constraints". *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 65-72.

[Subr89] Subramanyam S. and Lu S.C. "Computer-aided simultaneous engineering for components manufactured in small and medium lot sizes", in *Winter Annual Meeting of ASME*, DE v 21, PDE v 36, San Francisco, ASME, 1989, pp 175-184.

[Suh90] Suh, Nam P. *The Principles of Design*. New York, New York: Oxford University Press, Inc., 1990.

[Sun94] Sun, K. and Faltings, B. "Supporting Creative Mechanical Design". *Proceedings of ASME International Computers in Engineering Conference and Exposition*, v 1, 1994, pp. 343-352.

[Szyk92] Szykman, S., and Cagan, J. "A Computational Framework to Support Design Abstraction", *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 27-39.

[Take92] Takeda, H., Tomiyama, T. and Yoshikawa, H. "A Logical and Computable Framework for Reasoning in Design". *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 167-174.

[Tayl94] Taylor, L.E. and Henderson, M.R. 'The Roles of Features and Abstraction in Mechanical Design," Proceedings - 6th International Conference on Design Theory and Methodology, 1994 ASME Design Technical Conferences, Minneapolis, Minnesota, Sept. 1994, DE v 68, pp. 131-140.

[Terp93] Terpenny, J.P. and Nnaji, B.O. 'Feature Based Design Evaluation for Machine/Tool Selection for Sheet Metal", *Proceedings of the Second Industrial Engineering Research Conference*, Los Angeles, CA, May 1993, pp. 26-30.

[Terp92] Terpenny, J.P. and Deisenroth, M.P. "A Concurrent Engineering Framework: Three Basic Components", *Proceedings of the Second International Flexible Automation and Information Management Conference*, Falls Church, VA, June 1992, pp. 237-247.

[Tong93] Tong, C. 'Toward an engineering science of knowledge-based design", *Artificial Intelligence in Engineering Design*, ed. R.A. Adey, Computational Mechanics Publications, Great Britain, v 12, pp. 146-179.

[Tong92] Tong S. 'Engineous: A generic shell for design optimization, automation, and integration", *Proceedings GE CR&D Symposium on The Revolution of Knowledge-Based Engineering in Mechanical Design*. Schenectady, NY. May 14-15, 1992.

[Ullm92] Ullman, D.G. *The Mechanical Design Process.* McGraw-Hill, Inc., USA, 1992.

[Ullm93] Ullman, D.G. "The Evolution of Function and Behavior During Mechanical Design". *Proceedings - 5th International Conference on Design Theory and Methodology*, 1993 ASME Design Technical Conferences, Albuquerque, New Mexico, Sept. 1993, DE v 53, pp. 91-103.

[Ullm94] Ullman, D.G. "Issues Critical to the Development of Design History, Design Rationale and Design Intent Systems". *Proceedings - 6th International Conference on Design Theory and Methodology*, 1994 ASME Design Technical Conferences, Minneapolis, Minnesota, Sept. 1994, DE v 68, pp. 249-258.

[vanE90] van Emmerik, Maarten J.G.M. "An Interactive Graphical Approach to Feature Modeling Using Halfspaces and Geometric Constraints". *Proceedings - 16th Design Automation Conference*, 1990 ASME Design Technical Conferences, Chicago, Sept. 1990, v 1, pp. 97-106.

[VDI87] VDI-GKE, *VDI Guideline 2221, Systematic approach to the design of technical systems and products*, Dusseldorf: VDI Verlag, 1987.

[Verm95] Verma, D. and Fabrycky, W. "Development of a Fuzzy Requirements Matrix to Support Conceptual System Design". *International Conference on Engineering Design - ICED 1995*, August, Praha.

[Wald88] Waldron, M.B., Waldron, K.J., and Owen, D.H. "Use of Systemic Theory to Represent the Conceptual Mechanical Design Process", *Proceedings of the 1988 NSF Grantee Workshop on Design Theory and Methodology*. Springer-Verlang, New York, 1989, pp. 36-48.

[Ward94] Ward, A.C., Liker, J.K., Sobek, D.K. II, and Cristiano, J.J. "Set-Based Concurrent Engineering and Toyota". *Proceedings - 6th International Conference on Design Theory and Methodology*, 1994 ASME Design Technical Conferences, Minneapolis, Minnesota, Sept. 1994, DE v 68, pp. 79-90.

[West93] Westerberg, A., Grossmann, I., Talukdar, S., Prinz, F., Fenves, S. and Maher, M.L. "Applications of AI in design research at Carnegie Mellon University's EDRC", *Artificial Intelligence in Engineering Design*, ed. R.A. Adey, Computational Mechanics Publications, Great Britain, v 12, pp. 131-145.

[Wool76] Woolf, H.B. (ed): *Webster's New Collegiate Dictionary*, G. & C. Merriam Co., Springfield, Massachusetts, 1976.

[Welc92] Welch, R.V., and Dixon, J.R. "Representing Function, Behavior and Structure During Conceptual Design". *Proceedings - 4th International Conference on Design Theory and Methodology*, 1992 ASME Design Technical Conferences, Scottsdale, Arizona, Sept. 1992, DE v 42, pp. 11-18.

[Welc94] Welch, R.V., and Dixon, J.R. "Guiding Conceptual Design Through Behavioral Reasoning". *Research in Engineering Design*. v 6, n 3, 1994, pp. 169-188.

[West91] Weston R.H., Hodgson A., Coutts I.A., Murgatroyd I.S. "Soft' integration and its importance in design to manufacture", *Journal of Design and Manufacturing*, v 1, n 1, 1991, pp. 47-56.

[Wier91] Wierda L.S. "Linking design, process planning and cost information by feature-based modeling", *Journal of Engineering Design*, v. 2, n. 1, 1991, pp. 3-19.

[Will90] Williams, B.C. "Interaction-based Invention: Designing Novel Devices from First Principles". *Proceedings - Eighth National Conference on Artificial Intelligence (AAAI-90)*, American Association for Artificial Intelligence, July 1991, v 1, pp. 349-356.

[Wils88] Wilson, P.R. and Pratt, M.J. "A Taxonomy of Features for Solid Modeling". *Geometric Modeling for CAD Applications*, M.J. Wozny, H.W. McLaughlin, and J.L. Encarnacao (eds.), Elsevier Science Publishers B.V., North-Holland, 1988, pp. 125-136.

[Winn88] Winner, R. I., Pennell, J.P., Bertrand, H.E. and Slusarczuk, M.G. *The Role of Concurrent Engineering in Weapons Systems Acquisition*. Institute for Defense Analysis. Report R-338.December, 1988.

[Wisd92] Wisdom Systems, Inc., *The Concept Modeler*, 1992.

[Woya95] Woyak, S. A., Malone, B., and Myklebust, A. "An Architecture for Creating Engineering Applications: The Dynamic Integration System". *Proceedings of the ASME Computers in Engineering Conference*, September 1995.

[Yama94] Yamakawa, H. "A Unified Multidisciplinary Optimum Design Method Using Genetic Algorithms". *Advances in Design Automation*, ASME, DE v. 69-2, 1994, pp. 329-334.

[Yann91] Yannoulakis, N.J., Joshi, S.B., and Wysk, R.A. "Manufacturability evaluation and improvement system". *Proceedings - 3rd International Conference on Design Theory and Methodology*, 1991 ASME Design Technical Conferences, Miami, Sept. 1991, DE v 31, pp. 217-226.

[Yao94] Yao, Z. and Johnson, A.L. "Towards Fewer Design Failures". *Proceedings of ASME International Computers in Engineering Conference and Exposition*, v 1, 1994, pp. 335-341.

[Yosh93] Yoshioka, M., Nakamura, M., Tomiyama, T., and Yoshikawa, H. "A Design Process Model with Multiple Design Object Models". *Proceedings - 5th International Conference on Design Theory and Methodology*, 1993 ASME Design Technical Conferences, Albuquerque, New Mexico, Sept. 1993, DE v 53, pp. 7-14.

[Youn91] Young, R.E., Greef, A. and O'Grady, P. "An Artificial Intelligence-Based Constraint Network System for Concurrent Engineering". *Project Report PR-91-01, Group for Intelligent Systems in Design and Manufacturing*, Industrial Engineering Dept, NC State Univ., 1991.

APPENDIX A: POWER CONVERSION SYSTEMS

Power conversion, referred to by some as power electronics, is a field within electrical engineering with a history dating back to the middle of the last century. Since that time, there have been many textbooks, research related publications, and students who have studied the subject. It is therefore infeasible in this presentation to attempt to fully describe the subject or create an expert in the field. The intention here is to provide sufficient detail to understand the application area well enough to appreciate the blended conceptual modeling approach within the context of a problem example. This appendix introduces power conversion systems with a general background description and an overview of conceptual design considerations. Supporting the discussion of Chapter 5, the statement of an example problem is also included.

A.1 Background

Power conversion systems play an important role in numerous industrial and commercial applications. Papermill and metal-rolling industries are good examples of applications where power conversion systems are used to control *and* synchronize the motors of a continuous manufacturing process-line. Control and/or conditioning of power is also prevalent in other industries such as electric power generation (turbine control) and in large material handling systems found on cargo ships or off-shore oil drilling rigs.

Described by Heumann [Heum86], power electronics encompasses the switching, control, and conversion of electrical energy using semiconductor devices, including the associated measuring and open- and closed-loop control. Figure 52 depicts a very simple block diagram representation of a power electronics system. As illustrated in the figure, the

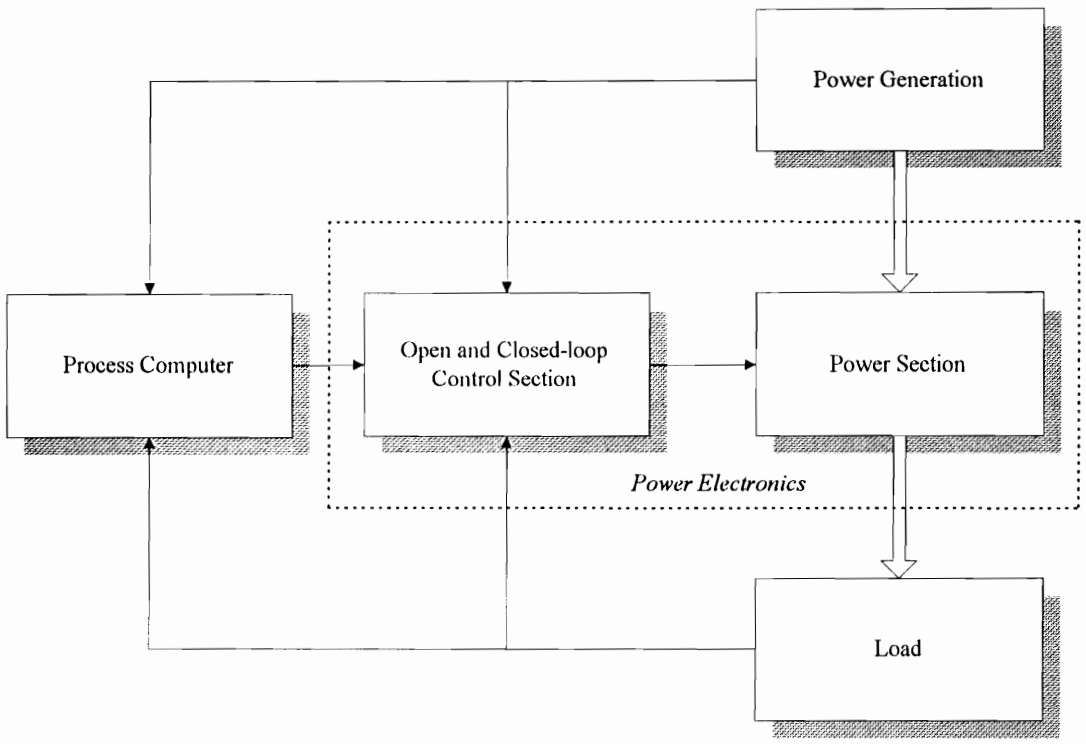


Figure 52. Power Electronics [Heum86].

power electronics system (encircled with a dotted line) provides an important link between power generation and the load. It is through this link that electrical energy is converted or controlled thereby enabling the energy flow between different systems.

When alternating current (AC) and direct current (DC) systems are coupled, four basic functions are possible [Heum86]:

- ***Rectification*** - the conversion of AC into DC whereby energy flows from the AC source system into the DC load system,
- ***Inversion*** - the conversion of DC into AC whereby energy flows from the DC load system into the AC source system,
- ***DC Conversion*** - the conversion of DC of a given voltage and polarity into that of another voltage and where applicable reversed polarity whereby energy flows from one DC system into the other, and
- ***AC Conversion*** - the conversion of AC of a given voltage, frequency, and number of phases into that of another voltage, frequency, and where applicable number of phases whereby energy flows from one AC system into the other.

The next section will elaborate on this background discussion with a focus on conceptual design considerations. A problem example is then introduced.

A.2 Conceptual Design Considerations

With numerous interdependencies between system elements, designing power conversion systems can be quite challenging; requiring engineering input from electrical, mechanical, and software areas.

Assuming ideal conditions, system elements of a converter include:

- voltage sources,
- transformers,
- resistors,
- magnetic and electrical energy storage, and
- power semiconductors.

In addition, in order to operate properly, the power conversion system must contain elements to accommodate:

- gate (turn on) and base (running) currents,
- transient voltage and current stresses,
- faults (internal and external),
- harmonics, and
- heat losses.

Broadly, the definition of a power conversion system is driven by these ten elements and marketing/customer requirements. Conceptual design begins with a request from marketing driven by either a direct customer requirement, not already satisfied within current product offerings, or from informed market speculation indicating the need for new product development. Ideally, though rarely the case, the request comes to design with a complete set of requirements in the form of a detailed specification. (For illustration purposes, a sample requirements specification document [GEDS96] is provided in Appendix B.)

Parameters commonly used to characterize requirements include:

- Performance Specifications
    1. Speed regulation
    2. Torque (duty cycles)
    3. Efficiencies
    4. Power
        a. HP/motor
        b. input (KVA)
        c. output (KVA)
        d. power factor
        e. harmonics
        f. motor RPM
        g. output volts
- Size
- Costs (customer total $ limit)
- Safety Standards Codes (UL/CSA/IEC)
- Control Card Architecture
- Time (customer promise date)
- Protection Class

Further, the initial requirements also specify the topology (type of bridge control), ratings, external interfaces, cooling system type, and control capabilities. Figure 53 shows an example block diagram indicating standard and optional high level interface requirements.

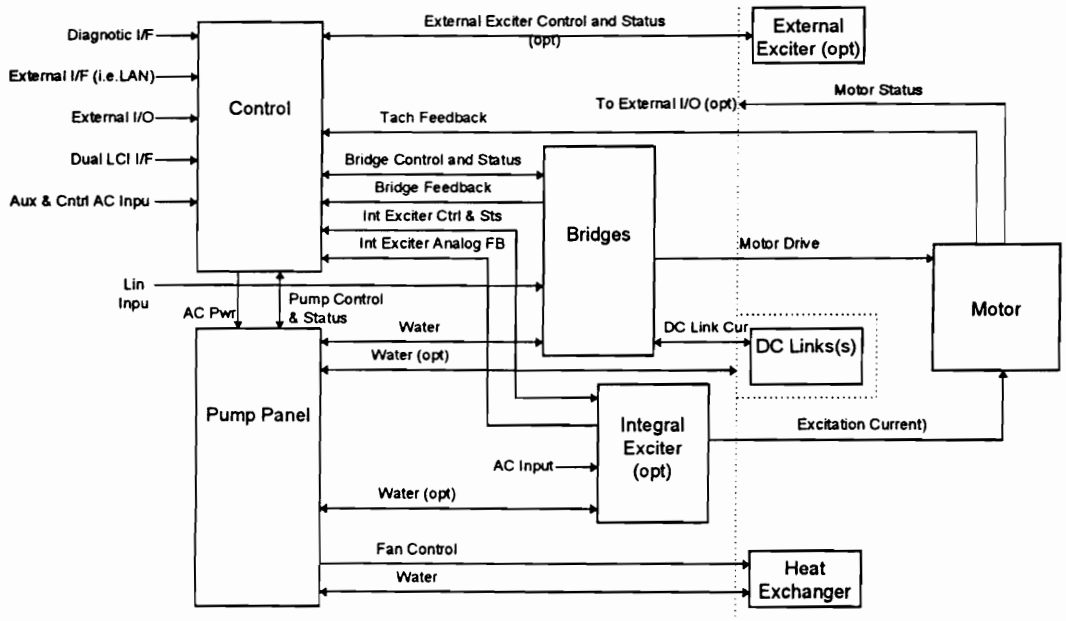


Figure 53. Top Level Block Diagram [GEDS96].

As indicated in the diagram, system requirements appear to be decomposable into distinct functional subsystems. This observation has only limited validity early in design however, as many alternatives are sought and considered. In truth, the subsystems are highly dependent upon one another. For example, consider the core of the power conversion system (bridge section). Here, semiconductors and associated electro/mechanical devices generate heat during operation. Prone to failure at high temperatures, semiconductors must be accommodated with adequate cooling. If, however, cooling alternatives are costly, violate size constraints, or are incapable of dissipating the heat sufficiently, the choices of semiconductors and related devices must be reconsidered. Alternatively, if the choice of semiconductors is rethought for some other reason (performance, cost, etc.), then the cooling system must be reconsidered. A small sample of the very early and highly iterative nature of bridge conceptual design is illustrated in Figure 54.

As shown, system concepts progressively evolve as design decisions are made, analyzed, evaluated, and rethought. Guiding this process the designer calls upon basic principles of physics, power electronics, company practices, and experience. Though highly iterative, the conceptual process is methodical, alternating between the search for solutions capable of satisfying functional need and analysis/evaluation (i.e., abstraction *and* detail). Chapter 5 takes a closer look at these needs/issues while examining the framework for functional modeling and configuration in conceptual design. Supporting this discussion, the next section introduces an example problem in the domain of power conversion systems.

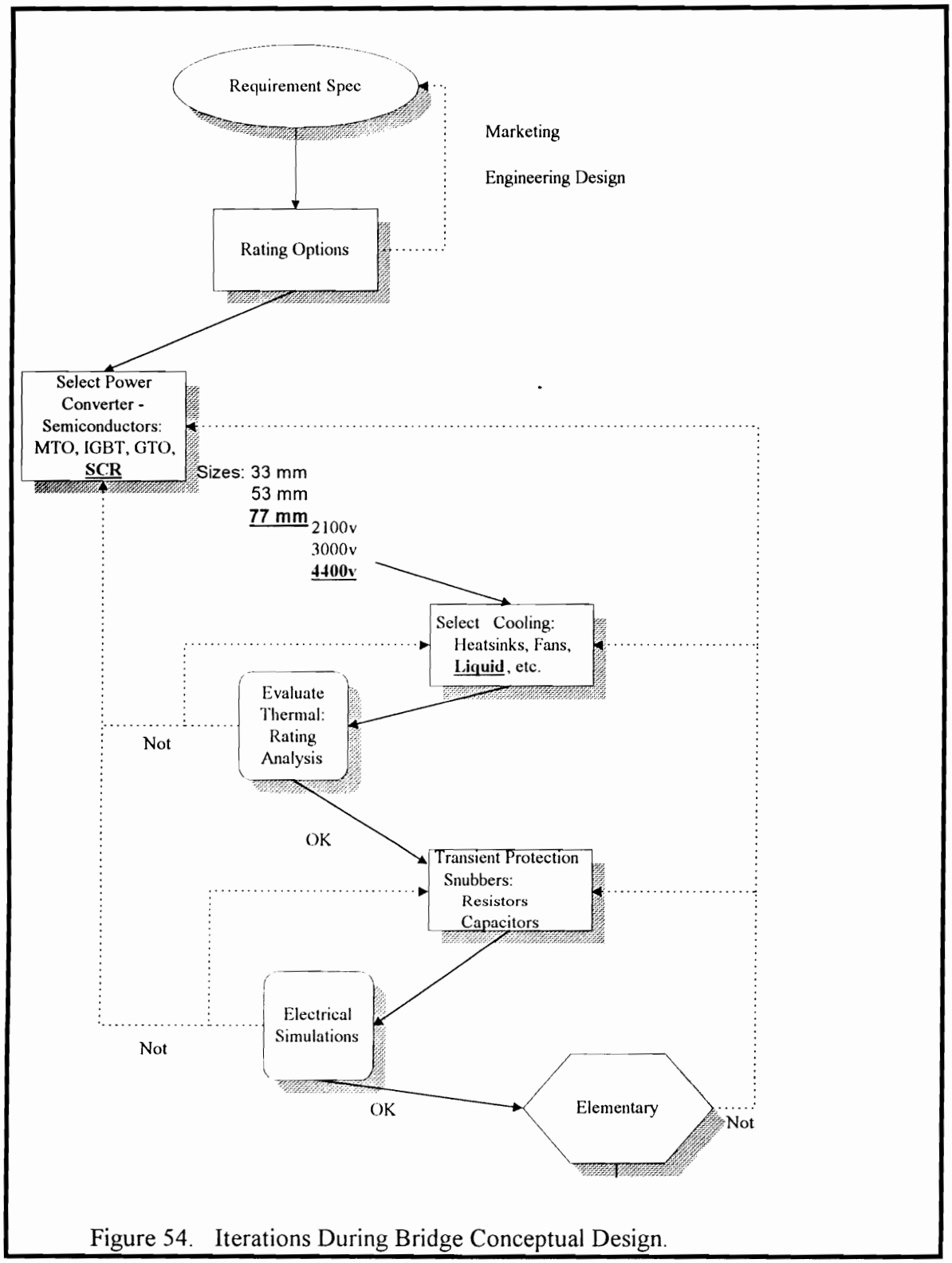


Figure 54. Iterations During Bridge Conceptual Design.

A.3 Example Problem

For purposes of illustration, the proposed research is demonstrated and evaluated in the context of an example problem in power conversion. Previously represented in Figure 53 in terms of a top level block and in the requirements specification of Appendix B, the LCI, or Load-Commutated Inverter, provides the basis for this example. This section describes the LCI in general terms followed by a closer look at the system core (power bridge section).

Technically stated, the LCI is a variable output frequency current source power amplifier consisting of a source bridge connected through a DC link inductor to a load bridge and its associated controls [GEDS96]. It is intended for use in high horsepower synchronous machine applications such as turbine starters and variable speed AC drives for utility, industrial, and marine applications. The synchronous machines used in these applications must be designed for service with an LCI type of power converter.

At the core of the system, the bridge section is where semiconductors and accompanying electromechanical devices perform the main functions related to power conversion. Including sections for fault conditions and control, Table 1 lists and summarizes the primary sections of the power bridge by function.

Table 1. Power Bridge Sections

FUNCTION	DESCRIPTION	SOLUTIONS
Transient Protection	Prevent damage from spikes (incoming and outgoing)	Filters, MOVs
Overcurrent Protection	Prevent damage when load current is above rated output and during faults	Fuses
Feedback	Sense parameters for protection and control purposes	Current Transformer (CT), Tachometers
Power Conversion	Heatsink assembly where power is converted/conditioned.	6 Legs, each containing: Thyristors (SCRs), Firing Cards, Snubbers (Capacitors, Resistors), Heatsinks
Power Filtering	Reduce harmonics	Reactors (inductors), Capacitors
Control	Control of bridge	Software, Firmware

A 3-D CAD solid model depicting a small portion of the bridge is shown in Figure 55. In particular, this subassembly is one of six legs of the water cooled heatsink assembly found in the power bridge. It contains a variety of power electronic devices and mechanical components including: semiconductors, firing cards, snubbers (capacitors, resistors), bus bar, heatsinks, hoses, clamps, and electrical connectors.

Further detail of the issues and design considerations of the bridge are included as needed in the discussion and examples of Chapter 5.

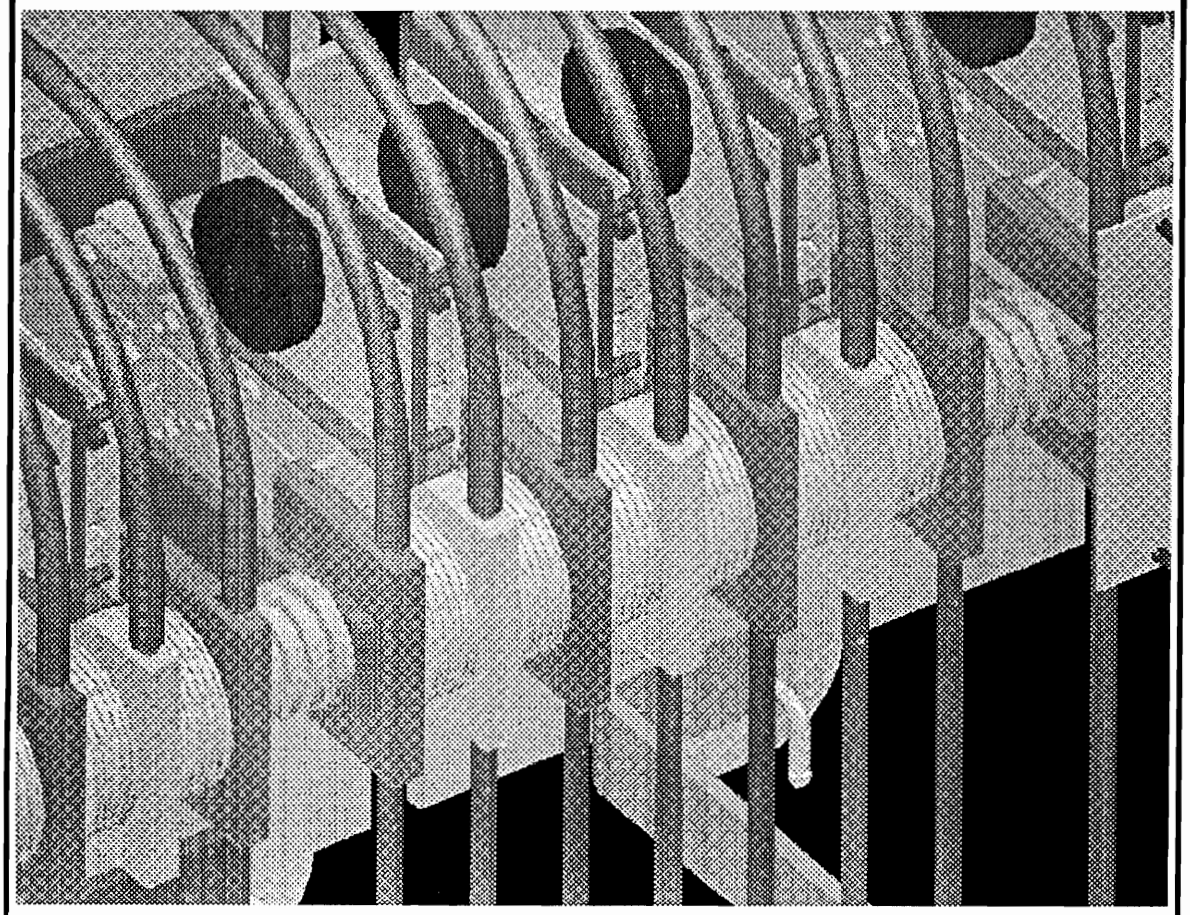


Figure 55. LCI Heatsink Assembly

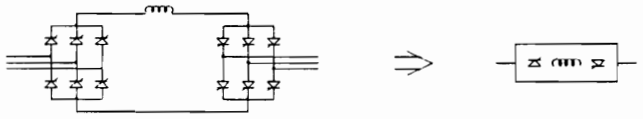
Appendix B: Sample Requirements Specification

This appendix provides an example of the requirements specification for the power conversion example used in this research. The text which follows is an except from a portion of the active document for the LCI power converter design. Many proprietary details have been excluded. The inclusion here is to illustrate the types of requirements typical of a specification and is for demonstration purposes only.

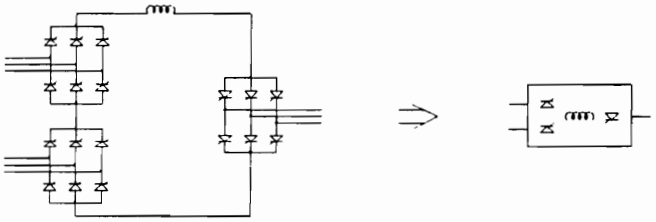
LCI POWER CONVERTER
Configurations

a) The following LCI bridge configurations shall be supported:

- 6 Pulse Source, 6 Pulse Load



- Series 12 Pulse Source, 6 Pulse Load



b) Main power fuses shall NOT be provided as part of the power converter design (*73). In order to prevent cell rupture, the fault current will be limited to 30000 amps rms symmetrical and the peak fault current shall be limited to 60 kA (for the 77mm version)by external means. Typically, this is done by operating the converter from a dedicated transformer which is matched to the converter rating with a 0.08 to 0.12 PU impedance. If this is not the case, external fuses or other reactors must be provided.

c) Load bridge AC line filters with neutral ground shall be used. Any other intentional grounds shall be evaluated on a per application basis.

d) There shall be no power isolating means in the lineup. However, the LCI shall be able to control and monitor an external one (*75, 76).

e) The DC Link Reactor (DCLR) may be split, (50% in each DC bus) based on design considerations (*77).

f) AC and DC filters (if used) shall be fused with blown fuse detectors.

g) The converter shall be designed to operate with a maximum input/output capacitance to ground of 0.25 mf (0.125 mf from shielded cable and a breaker surge capacitance of 0.125 mf) (*37). Additional capacitance will require extra external reactance.

Bridge Ratings

a) The 6 pulse source, 6 pulse load configuration shall be available with 2300, 4160, and 6600 Vrms output voltage ratings. The series 12 pulse source configuration shall be available with 4160 and 6600 V rms output voltage ratings (*71, 72).

b) Series thyristors shall be used to achieve the required voltage ratings. The design shall not require the use of individually selected cells for the bridge legs, but will require a specification with several controlled elements.

c) N + 1 and N capability are both required. The number of thyristors per leg for the "N" version shall be consistent with good engineering practice for seriesed thyristor configurations. Failure of a single thyristor in the "N" version shall not immediately result in the failure of the entire bridge leg but WILL require shutdown of the converter (*1, 2).

d) The ratings for each voltage are summarized below ((*71, 72, 93, 68):

Nominal Output Voltage, V RMS			
Nominal Input Voltage, V RMS +/- 10%			
Continuous Output Current, A RMS			
Continuous Output KVA			
Max. Ambient Air Temperature, °C (note 1)			
Nominal Input Frequency, Hz			
Minimum Output Frequency, Hz (note 2,3)			
Maximum Output Frequency, Hz (*18a,b)			
Efficiency (note 4)			

Notes:

1) The first number is the max with NO derating. The second number is the absolute maximum.

2) Derating is required for steady state operation below 2 Hz.

3) Refer to section 2.5.m for low frequency control performance capabilities.

4) Minimum at full load operating at 60 Hz input and output, excluding excitation losses and DC Link Reactor

5) The ratings in the table are nominal expected maximums - equipment design must support the maximum utilization of the semiconductors

Mechanical Design

a) The mechanical design shall utilize a single stack per bridge configuration, providing the same mechanical configuration for each of the voltage ratings (*79). The intent is that a maximum commonality is to be provided so that there are as few different components as possible to simplify inventory and manufacturing. The bridge should also be completely testable (both electricaly and hydraulically (*80)) in the "clean room" where it is assembled so that no basic testing needs to be done at final assembly.

b) A single basic configuration shall allow the use of either 77 or 53 MM thyristors (*81).

c) The 77mm design will be configured for either 4160 or 6600 Vac, with the only difference being the number of thyristors per leg and the associated gate drivers, snubbers, etc. The 53mm thyristor version will be based on the 77mm design and will be rated for 2300, 4160 and 6600 vac.

d) The design shall be braced for 30,000 Arms symmetrical short circuit current.

e) The converter shall use de-ionized water in a closed loop system to remove heat from all heat producing elements in the cabinet (*82). The base product shall include the complete pump, filter, de-ionizer resin bed and connections for an external heat exchanger. Redundant pumps (*14) and temperature regulating valve shall be provided. Moisture condensation in the power converter cubicle shall not be permitted. A goal will be to have the deionzer and filter cartridges last at least one year per set

f) Bridge rating based on the maximum bridge incoming water temperatures of 30, 40 (*13) and 50 degrees C. (*84) The use of antifreeze shall be optional.

g) The standard shall be for the DC Link Reactor (DCLR) to be external from the lineup, but for certain sizes it will be available in the line up. The application will dictate whether it is air or water cooled (*43, 85).

h) A goal shall be to minimize the footprint and customer installation cost (*12, 87).

i) The design shall use the metric system only to the following extent (*88):
   - all fasteners to be English
   - enclosure outline dimensions shall be made to metric dimensions

j) All openings in the cabinet shall be designed in a manner which will reduce the emission of electro-magnetic interference to the level required to satisfy competent body's recommendation for "CE" stamp.~~.~~

k) The LCI layout shall be compatible with the required mounting in a power control room. (*27)

l) Cubicle size compatibility with the present style is not required (*29)

Applicable Standards

a) The LCI shall be designed to comply with the latest revision of the proposed UL347A (*91).

b) The LCI shall be designed with the intent to achieve UL/CSA/CE certification.

Converter Control

a) The control shall use a new version of control architecture which is based on the C31 DSP main control card.

b) The control shall have expandable, customizable I/O with "rugged" Terminal blocks that will be acceptable for customer use (*4)

c) The design shall be documented in a manner that is still to be determined. ~~The design shall adopt the DCP functionality "a~~

d) The control shall be capable of supporting source bridge operation at 50/60 Hz nominal input frequency (*93). Only forward phase sequence shall be supported (*94a) Reverse phase sequence is a goal. For series 12 pulse source configurations, the control shall support the follower source bridge 30 degrees leading or lagging from the master source bridge (*95).

e) The control shall be capable of supporting load bridge operation at 90 Hz maximum nominal output frequency. Forward and reverse phase sequence shall be supported (*97).

f) The control shall be capable of maintaining normal operation at reduced load with 70% main, auxiliary, and control voltages for up to 30 seconds (*98).

g) The control shall provide full regenerative torque capability but not down to or through zero speed in all cases.

h) The control shall provide bus transfer ride through capability. Momentary loss of main power shall result in the control shutting off drive current and allowing the motor to coast. Upon restoration of main power the control shall restart on the fly assuming the run command is still true (*99). The control shall be able to ride through a complete loss of control power for 1/2 cycle (8 msec). _When the control power is supplied by a UPS, the control drive must be able to tolerate auxiliary power interruption for up to ~~2~~500 msec without tripping.

i) The control shall be able to run the motor without a tachometer. Starting and low speed performance will be limited (refer to performance table) (*100). The tachless control shall be able to start the motor from a standstill without transient reverse rotation (*101).

j) The control shall be able to take advantage of having a tachometer on the motor. Enhanced starting and low speed performance shall result (refer to performance table). Tachometers with and without markers shall be supported. Full starting performance will not be available on the first start after a hard reset on the control (*102). For systems with a tach, the loss of the tach will (optional) cause the control to revert to the "tachless" regulation spec. (*15)

k) The control shall provide support for dual channel applications which parallel 2 LCI converters through either dual windings in an output transformer or dual windings in the motor. Each channel (LCI converter) shall have its own control (*103).

l) The control shall provide support for applications which place multiple motors on the same shaft, each driven by a separate LCI/Exciter (*104).

m) Below approximately 10% of base speed (10% motor voltage), the load bridge shall be force commutated by transient extinction of the link current (link commutation). Above this speed, the load bridge shall be load commutated(*105).

n) The control shall interface to and control a variety of excitation systems, including:
   - AC/DC2000 family exciters via an analog reference (*106)
   - EX2000 family exciter via analog reference (0-10 Vdc) (*107)
   - 3rd party exciters via analog reference (0-10 Vdc, 4-20 mA) (*108)
   - fixed excitation systems will not be interfaced (*109)

o) Performance (*110):

	NO TACH BRUSHLESS/SLIP RING	TACH BRUSHLESS/SLIP RING
Speed Range		
Speed Regulator Response regulation		
Flux Regulator Response		
Voltage Limit Reg Response		
Dynamic Reversing availability		

torque thru 0 speed		
Starting max torque from standstill max speed from forward max speed from reverse		
Constant Torque speed range max overload		
Constant Horsepower speed range max overload		
Impact Loading below base speed above base speed		
Max Accel/Decel Rate below base speed above base speed around corner point		

Notes:

1) see dynamic reversing
2) provided there is sufficient excitation

Regulation

The control shall regulate motor current (magnitude and angle) and motor voltage magnitude to produce the torque requested by the outer loop regulators. This task shall be coordinated with the capabilities of the LCI and exciter power converters. The motor/converter control shall provide the following features(*111):

a) Primary regulation of motor terminal voltage by commanding field current/voltage to the exciter with the following features (*112):
   - reference defined by piecewise linear volts/hertz profile as a function of speed (3 segments) (*113) with an optional 1/speed taper above base speed (*114)
   - transient regulation thru stator current magnitude and phase (*115)
   - an exciter current limit function (*116)
   - a flux reduction function to limit load bridge snubber losses at light load(*117)
   - a feedforward function to improve impact load response (*118)

b) Primary regulation of motor current magnitude (dc link current magnitude) via source bridge phase control (*119)with the following features:
   - up to 300 radian response with feedback acquisition strategies to minimize beat interaction between load and source bridges (*120)
   - a current lower limit function to avoid operation in discontinuous current (*121)
   - different current maximum limit for starting (*122)
   - different current maximum limit for load bridge commutation modes (*123)
   - current rate limit function (typically set to 20 pu/sec) (*124)
   - transient regulation thru load bridge phase control when the source bridge saturates (*125)
   - a load dc link voltage feedforward function to improve response to load bridge angle changes(*126)
   - a Fire As Soon As Possible (FASAP) function to maximize source p.f. (over limited speed range) (*127)

c) Primary regulation of motor current phase angle via load bridge phase control with the following features (*128):
   - a Fire As Late As Possible (FALAP) control strategy to maximize motor p.f (*129).
   - transient deviations from FALAP to satisfy overriding regulation demands such as current regulator crosstie, dc link current lower limit, etc.

Bridge Gate Control

a) Source Bridge
   - shall support phase lock loop based gate scheduling over the operating range
   - shall provide an adjustable advance limit (0 degrees minimum)
     .shall provide an adaptive retard limit as a function of commutating reactance, current, voltage, frequency plus an adjustable margin angle.

b) Load Bridge
   - shall support phase lock loop based gate scheduling above 2 Hz
   - gate scheduling to minimize disturbance from unbalanced load ac voltages (up to 10%)
   - shall provide an adjustable advance limit (0 degrees minimum)
     .shall provide an adaptive retard limit as a function of commutating reactance, current, voltage, frequency with an adjustable margin angle
   - shall support unity power factor during link commutation

Protectives (*131)

a) Hardwired Backup Overcurrent (*132)
   - indicates converter current has exceeded normal operating range for a long enough period of time that it is assumed that the firmware implemented protectives have failed
   - individual monitors for source and load bridge currents, adjustable magnitude threshold
   - Fixed time delay to coordinate with firmware implemented overcurrent
   - a HARDWIRED circuit is used to initiate a disconnect of the LCI converter from the incoming ac mains and/or motor without microprocessor intervention
   - the intent of this protective may be met using redundant CPUs, the bottom line requirement is that "the bridge shall not blow up" during any fault that may occur.

b) Overcurrent (*133)
   - indicates converter current has exceeded normal operating range
   - individual monitors for source and load bridge currents, adjustable magnitude threshold
   - extinguish link current when detected, resume normal operation when link current zero
   - trip fault when frequency of occurrence exceeds an adjustable threshold

c) Commutation Failure (*134)
   - indicates a misfiring of the source/load bridge which will result in a diametric conduction pattern on the source/load bridge
   - individual monitors for source and load bridge, including a difference current detector with an adjustable threshold - capable of detecting a shoot through in a maximum of 15 degrees of a 60 Hz cycle.
   - extinguish link current when detected, resume normal operation when link current zero
   - trip fault when frequency of occurrence exceeds adjustable threshold

d) Master/Follower Current Imbalance (series 12 pulse source) (*135)
   - indicates an excessive difference between the source master and source follower bridge currents

- adjustable magnitude threshold
- extinguish link current when detected, resume normal operation when link current zero
- trip fault when frequency of occurrence exceeds adjustable threshold

e) SCR Failure (*136)
- indicates a shorted scr has been detected in one of the converter bridges
- trip fault if "N" or alarm if "N+1"
- ability for fault message to identify bridge/leg/cell # of the failed cell

f) Phase Lock Loop (*137)
- indicates the gating phase locked loop has lost lock
- individual monitors for source and load bridge, adjustable magnitude threshold
- extinguish link current when detected, resume normal operation when lock achieved
- trip fault when frequency of occurrence exceeds adjustable threshold

g) AC Overvoltage (*138)
- indicates bridge input voltage has exceeded normal operating range
- individual monitors for source and load bridges, adjustable magnitude threshold
- trip fault when magnitude exceeds adjustable time delay threshold

h) Source Undervoltage (*139)
- indicates source bridge input voltage below normal operating range, also detects single phasing
- adjustable magnitude threshold (typically 70%)
- extinguish link current immediately when detected
- alarm for temporary dip, trip fault for persistent (adjustable time)

i) Gate Driver Supply Undervoltage (*140)
- indicates loss of gating power supply
- extinguish link current immediately when detected
- alarm for temporary loss, trip for persistent (adjustable time)

j) Bridge Filter Fuse (*141)
- indicates that protective fuses for ac or dc bus R-C filters are blown
- alarm only

k) Failure To Start (tach drive) (*142)
- indicates failure of drive to rotate at initial startup
- trip fault

l) Excessive Force Commutation Time (static starter) (*143)
- indicates the drive has taken an excessive time to transition from link to line commutated mode
- trip fault with adjustable time limit, detected only during staring

m) High Speed Restart (static starter) (*144)
- indicates the drive has been restarted "on the fly" above base speed (this can result in a severe overvoltage on the inverter if undetected)
- trip fault

n) Rotor Position Start (*145)
- indicates the control has failed to detect machine stator flux after the exciter energizes the field
- alarm only, detected only during starting

o) Microprocessor Misoperation (*146)
- family of faults which detect misoperation of the microprocessor control such as stack fault, EE Prom transfer failure, I/O Addressing failure, etc.
- alarm or trip as appropriate

p) Ground Fault (*147)

- indicates a ground somewhere between (and including) the source isolation transformer and the ac output buses including the motor
- trip fault

q) Shorted Motor Winding (*148)
- indicates unbalanced machine voltage, adjustable threshold
- trip fault

Diagnostics (*149)

a) Simulator Mode (*150)
- test mode in which embedded simulations for the synchronous motor and power converters synthesize feedback signals for the motor control algorithms

b) Crowbar Test Mode (*151)
- test mode in which source bridge regulates link current through a diametric on load bridge
- the load bridge diametric is "walked" through each of the three possible diametric pairs at a low frequency using link commutation mode
- dc link current level is controllable open loop

c) Gate Test Mode (*152)
- test mode in which bridge gating is enabled as long as the converter is off line (no voltage
- will gate all legs in sequence or one leg at a time

d) Exciter Only Test Mode (*153)
- test mode in which only the exciter is run
- exciter voltage/current reference is controllable open loop

e) Control Self-Test (*154)
- automated test in which comprehensive self test run on the control electronics
- failures are identified to the customer field replaceable level
- tests are run off-line upon power up or hard reset

f) Cell Test / Torque Prove(*155)
- automated test mode in which the ability of LCI converter to control motor current if verified
- failures are identified to the individual gate driver/scr
- tests are run off-line and can be configured to execute every time the drive is started

g) Circular List(*156)
- provide the ability to perform advanced diagnostics with event triggerable circular lists
- data can be taken and displayed on-line

h) Auto-Tune (*157)
- speed regulator auto-tune shall be supported

i) Test Points (*158)
- all of the following test points are goals with a full review at the preliminary design phase
  . all bridge currents and bridge voltages
  . provision for measuring each individual SCR gate current pulse with a clip on current probe
  . motor speed (frequency) signal
  . all power supply buses, all setpoint trimpot wipers
  . two assignable d/a converters, 8 or more bits resolution

- an absolute minimum requirement is 2 line voltage and 2 line current signals per bridge which must be reproduced (at the control voltage potential) with a bandwidth of at least 5 KHz to allow seeing voltage commutation notches, while the drive is in operation at rated voltage.

Direct Connect (Bypass) (*278)

The LCI shall support direct connect (bypass) applications (when using an external exciter)which use the LCI to soft start the synchronous motor and transfer the motor to the ac line. This control shall work through motor speed reference and motor flux reference to match motor voltage magnitude, frequency, and phase to the incoming ac line prior to transfer. This function must accommodate isolation transformer voltage ratio and phase shift. The LCI shall control the bypass contactor. The external exciter (EX2000) must be able to switch to a VAR, PF, or manual field regulation mode once the motor has been transferred to the ac line.

VITA

Janis P. Terpenny was born on April 2, 1957, in Richmond, Virginia to Herbert W. and Betty C. Pinchefsky. She attended high school at Thomas Jefferson High School in Richmond, Virginia where she completed her studies in three years as a member of the National Honor Society. She then obtained her Bachelor of Science in Applied Mathematics at Virginia Commonwealth University in Richmond, Virginia where she graduated with honors distinction. Next, she obtained a Master of Science degree at Virginia Polytechnic Institute and State University in Industrial Engineering and Operations Research with an emphasis in Operations Research. The year following the completion of this degree, she was an Instructor for the Department of Industrial Engineering and Operations Research. This experience was followed by a career with the General Electric Company. While with General Electric, she completed a corporate training program in Information Systems and gained significant experience as a Systems Analyst developing solutions and managing projects for applications in functional areas of the business such as engineering, manufacturing, transportation systems, employee relations, cost accounting, and communication.

Since returning to Virginia Polytechnic Institute and State University in pursuit of a doctorate in Industrial and Systems Engineering, Dr. Terpenny has had a wealth of experiences. With an emphasis in Manufacturing Systems, her studies have been supported by the Charles Minor Fellowship (awarded by the College of Engineering), and research grants from Virginia's Center for Innovative Technology (CIT), and by the General Electric Company in Salem, Virginia. She was privileged to conduct her doctoral research at General Electric where she was directly involved in numerous projects related to the design and evaluation of advanced systems to support process re-engineering and

engineering design. She also served the Department of Industrial and Systems Engineering during her doctoral studies both as a Graduate Assistant and as an Instructor. In addition, she assisted the Governor's School of Excellence in Robotics and was funded by the General Electric Foundation to research and document a case study in design for manufacture. She has published, presented, and been a moderator at several international conferences. Her current research interests are focused on enabling technology and its application in the development of intelligent engineering design systems for Concurrent Engineering. Currently, Dr. Terpenny is an Instructor in the Department of Industrial and Systems Engineering at Virginia Polytechnic Institute and State University. She is a member of the Institute of Industrial Engineers (IIE), the Society of Manufacturing Engineers (SME), the American Society of Mechanical Engineers (ASME), and Alpha Pi Mu, the honor society for Industrial Engineers.