This is to certify that the

dissertation entitled

DESIGNING COMPOSITE MATERIAL SYSTEMS USING
GENERIC TASKS AND CASE-BASED REASONING

presented by

TIMOTHY JOEL LENZ

has been accepted towards fulfillment
of the requirements for

Ph.D. degree in Chemical Engr

Martin C Hawley
Major professor

Date May 2, 1997

DESIGNING COMPOSITE MATERIAL SYSTEMS USING GENERIC TASKS AND CASE-BASED REASONING

VOLUME ONE

By

Timothy Joel Lenz

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Chemical Engineering

1997

ABSTRACT

DESIGNING COMPOSITE MATERIAL SYSTEMS USING GENERIC TASKS AND CASE-BASED REASONING

By

Timothy Joel Lenz

The evolution and application of engineering design techniques for polymer composite material systems provide the focus of this dissertation. Paradigms that extend and expand the intellectual capabilities of composite designers are developed. In the quest for these paradigms, the design of polymer composites is considered at varying levels of complexity. This consideration addresses the accumulation and organization of design knowledge, and the philosophical nature of the design process. The enabling emphases are knowledge-based systems and the meticulous engineering of design information interactions.

A cohesive approach is developed for efficaciously designing composite material systems. This includes both design guidelines and software implementations of these guidelines with which to design composite material systems. The computational tools presented in this dissertation can assist in the design of composite material systems

through both strategically organized heuristics (knowledge-based systems) and previously detailed designs (case-based reasoning).

Not only do these computational tools embody a new understanding of a complicated and multifaceted design process, but they are also exemplars of structured knowledge reuse. The use of knowledge- and case-based systems in polymer composites design simplifies the use and reuse of engineering design knowledge, thus enabling the transfer of expertise. By streamlining the tedious design processes, these systems can also free design engineers for more creative design activities.

For MLO

ACKNOWLEDGEMENTS

I would like to thank all of those who have inspired me to push the limits of my understanding, forced me to think, taught me how to question everything (and then to go find the answers), and generally demanded more of me than I did of myself: Mrs. Grabow, Joanne Kelly, Paul Willems, Jim Bauer, Prof. Cynthia Weberg, Prof. Sigurd Lee, Prof. L.E. Scriven, Prof. Matthew Tirrell, Mark Stewart, Prof. Martin C. Hawley, Prof. Jon Sticklen, and Dr. James K. McDowell.

I would also like to specifically thank those comprising my personal infrastructure, both near and far: Julie, Mark & Esther, Jeremy, Pam, Tom, Bill, Tony, Rick, & Jim. You have all helped in ways you do not even know, and I thank each one of you.

The professional infrastructure here at Michigan State has made my life easier (most of the time): Rob Hawkins, Ahmed Kamel, Chris Penney, Jason Greanya, Iliana Martinez, and the wonderful ChE superwomen (you know who you are!). Thank you.

And finally, I would like to offer a special thanks to my committee members, especially Prof. Alec Scranton and Prof. Gary Blanchard. You have stayed with me through all of the weird twists over the past several years, and have always provided an enthusiastic perspective. Thank you.

I would also like to acknowledge the financial support of DARPA (via the MADE program), the U.S. DOD (via AASERT), the Department of Chemical Engineering at Michigan State University, and General Electric Aircraft Engines.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 5

Chapter 6

Chapter 7

Chapter 8

Chapter 9

Chapter 10

Chapter 11

Chapter 12

Appendix B

Appendix E

Part I

Prelude

CHAPTER 1

Introduction

When a chemical engineer approaches the design of a petrochemical process, say, the hydrodealkylation of toluene to produce benzene, he or she does not begin with a detailed economic analysis of the energy-integrated flowsheet of the process, attempting to achieve the maximum possible energy recovery. Starting off at that level of complexity is a ludicrous proposition. To apply an energy integration analysis, all the heat loads and all the stream temperatures throughout the process must be known. But these cannot be figured out unless the flow rates and compositions of every process stream are also known. Obviously, this is the wrong level of detail at which to begin.

Such an attempt brings to mind the following quote:

> *"I have yet to see any problem, however complicated, which, when you looked at it in the right way, did not become still more complicated."* Poul Anderson

What is the appropriate level of detail to consider for the design of the hydrodealkylation process? There are most assuredly less complex ways of addressing the design of this process than above (although probably not many more complicated).

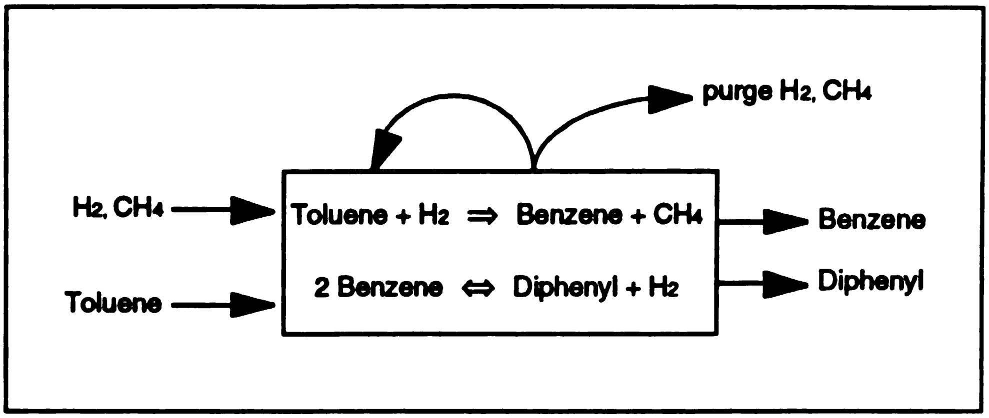

Figure 1. Hydrodealkylation reaction and input-output behavior.

There are several less-detailed levels at which the design can be considered with some utility. A general hierarchy of the levels of decisions for designing just such a chemical process has been established [Douglas, 1988]. The simplest is whether the chemical process under consideration is to be a batch or continuous process. Slightly more detailed than that is an input-output flowsheet of the process. Figure 1 shows the input-output behavior of the hydrodealkylation process, along with the reactions in question [Douglas, 1988]. Additional levels of detail include the consideration of the recycle flowsheet, the general structure of the separation system, and finally, the heat-exchanger network.

In order for the designer to be able to do the detailed economic analysis on the energy-integrated (i.e., up to and including the heat-exchanger network) process flowsheet, the entire process must be designed. Many decisions must be made throughout this design process, but as the design of chemical processes is well understood, specific plans for making these decisions exist and can be used linearly (i.e., one after the other).

These plans are ways of approaching applied engineering design. However, there is more to engineering design than just the linear execution of specific plans. The following section presents some of these additional issues.

1.1 Engineering Design Issues

Say a chemist has discovered a reaction to make a new catalyst for an existing, commercial reaction. A chemical engineer is called in to transfer this discovery to a new process. The engineer starts with only knowledge of the reaction conditions and perhaps some information about available raw materials and products. The engineer must supply all the other information needed to define the design problem. Then, once the design problem has been completely defined, it must be solved.

This typical example depicts the two main difficulties with engineering design. The first is the fact that most engineering design problems are ill-structured. The open-ended and underdefined nature of engineering design problems is the other difficulty.

Design problems are open-ended because many acceptable solutions exist, even for those designs which are relatively well defined. For the new chemical process above, decisions must be made about what unit operations to use, how to connect them, and the conditions at which they will operate. The challenge is to effectively limit the design alternatives, as there are often a very large number of possible solutions.

Design problems are considered to be ill-structured because their solutions can not normally be found through the simple use of mathematical formulas [Dym, 1994]. Most engineering design problems are inherently nonlinear, and require much more sophisticated approaches to problem solving than can be represented mathematically.

The fundamental approaches to engineering design can be summarized as [Glegg, 1969]: domain-specific techniques, general rules, and universal principles. General rules and universal principles are more often than not implicitly embedded within the details of domain-specific techniques. The two most important universal approaches to engineering design are assumption and reuse.

The most commonly used type of engineering design approach is that of estimation and assumption. The first thing any engineer does when approaching a new problem is to make a rough estimate, or a "back-of-the-envelope" calculation of the design. This rough estimate generally focuses on only a portion of the final design, leaving out those portions of the design with which the designer is unconcerned. The wing box sketch in Figure 2 on page 24 is an example of just such a rough design. The utility of back-of-the-envelope calculations is the quick determination of the approximate worth of a design. If a back-of-the-envelope design appears feasible, then the engineer proceeds to a more rigorous design; if it is not feasible, that design alternative is rejected.

An example of the utility of the back-of-the-envelope calculation can be seen in the following anecdote related by the head of an architectural consulting firm [Petroski, 1992].

> *"When my engineers come to me with millions of numbers on a high rise, I know there is one number that tells me a lot of things - how much the top of the building will sway in the wind. If the computer says seven inches, and my formula, which takes thirty seconds to do on the back of an envelope, says six or eight, I say fine. If my formula says two, I know the computer results are wrong."*

The practice of reuse in engineering design is expressed by [Glegg, 1969]: "Now one of the [best] ways of designing something is not to design it at all. Use [a part] that is designed already by someone else." The design practice of modifying past designs is utilized within nearly all engineering fields.[1]

1. Of course, domain-specific techniques are of prime importance in determining the specific methods actually used to perform the modification of previous designs.

Many definitions of engineering design exist, and each definition is more or less influenced by the specific domain in which it is applied. A recent definition attempts to avoid this pitfall by defining engineering design outside of any particular domain context [Dym & Levitt, 1991; Dym, 1994]:

> *Engineering design is the systematic, intelligent generation and evaluation of specifications for artifacts whose form and function achieve stated objectives and satisfy specified constraints.*

This definition is almost general enough to be completely useless without extensive interpretation. Not surprisingly, there are many implicit assumptions behind this definition [Dym, 1994]:

- design is not a mindless process, but is replete with significance and is, at least to some extent, comprehensible;
- both form and function can be represented, although not necessarily individually or facilely; nonetheless a correlation between the two can be expressed;
- the representation must be able to interpret not only the original design problem specification, but also its stated design objectives and constraints;
- problem-solving techniques must exist that can utilize this representation to develop solutions for the design;
- the generated design solutions must be translatable, irregardless of the representation utilized, into appropriate fabrication specifications.

The key issue within these assumptions is that without the appropriate knowledge representation, there can be no design [Dym, 1994]. There must exist a way of describing a designed artifact at all stages of design, from the original problem specification through to the evaluation and fabrication of the solution. The problem solving and evaluation methods utilized in the design process are enmeshed within and dictated by the representation utilized, and must be expressed and implemented at the appropriate level of abstraction.

1.2 The Composites Design Problem

The design of chemical processes presented earlier is an example of established and linear engineering design. The specific design tasks have been identified and many heuristics have been developed. A result of this rich design understanding is the linear nature of chemical process design: a definite order for determining portions of the design exists. Conversely, composites design is not currently understood completely enough to be considered linear.

Similarities do exist between the problem posed for the hydrodealkylation process and composites design. When a composites designer is charged with designing, say, a wing box for an aircraft, he or she does not jump right into a detailed economic cost analysis of the design. To do that, the designer must know the details of the materials of construction, the processing and the specific architecture of the wing box. Those details are unknown when the design problem is specified, and must be decided. Figure 2 gives a comparison of the initial wing box concept with the detailed wing box design.

Obviously, the composites designer must consider the design at some higher level of abstraction. A simple input-output level of description gives the design specifications as input, but does not give much for an output other than the rough wing box shown in Figure 2. The designer (or more often, a design team) must consider the part configurations, material specifications, and manufacturing process simultaneously.

The overall design process for composites is typically split into three phases [Principe *et al.*, 1987]. At each of these phases, part configurations, material specifications, and manufacturing processes are considered simultaneously.

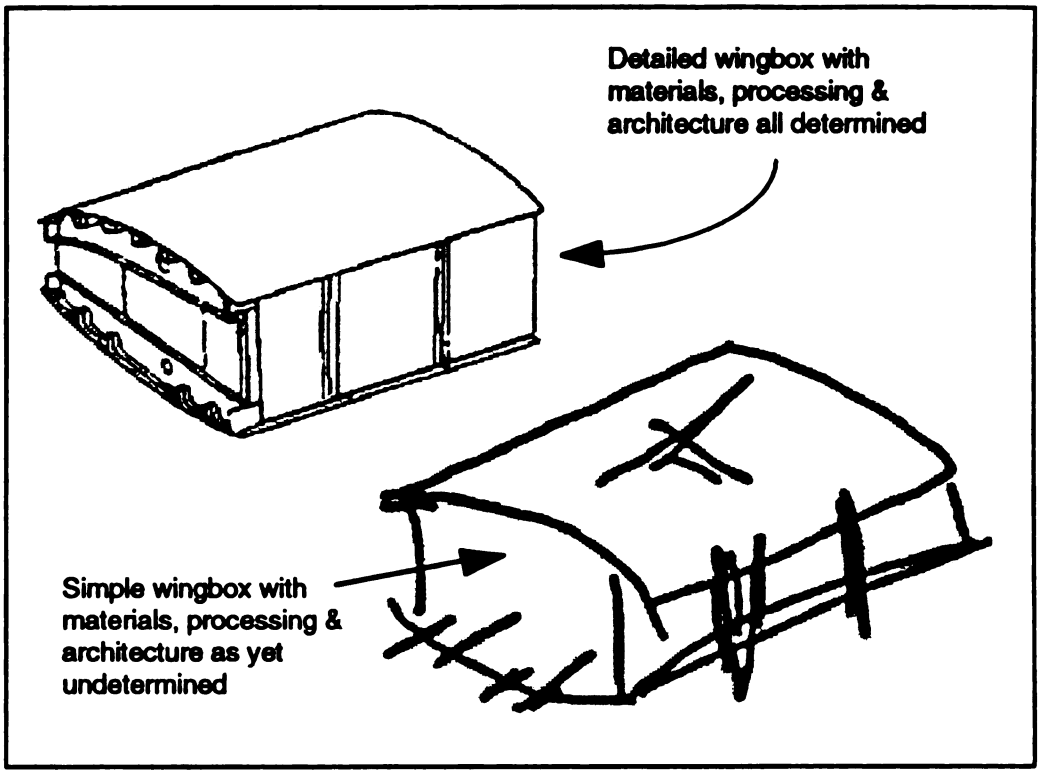

Figure 2. Rough and detailed composite wing boxes.

1. During the initial phase, requirements such as cost, size, weight, performance, loading specifications, and environment are defined, and preliminary designs are developed for the material, process and part.

2. The second phase involves evaluating and verifying the new materials, joint design, and manufacturing techniques specified in the first phase. Detailed finite element analyses of critical areas may also be conducted.

3. In the third phase, a full-scale prototype is fabricated and evaluated for cost, performance capabilities, and effects from the environment.

If all specifications have been met after the third phase, production of the part may commence. However, if the part fails to meet the specifications, it must be redesigned. Often, design processes for composites involve significant amounts of costly redesign.

As for the design of the chemical process, many decisions must be made throughout the composites design process. Unlike the design process for chemical processes, the design process for composites is not linear. Many iterative design cycles are required before the final design is specified. Trial-and-error also requires iteration. Specific linear design plans for making these decisions for composites do not exist. The onus for the redesign common to composites can be clearly placed on the immaturity of composites technology and the less than reliable standards for materials, design, manufacturing, and inspection. The composites design process is not straightforward, and in fact, is rife with pitfalls for the inexperienced or incautious designer.

1.3 Goals and Strategies of this Research

The problems inherent in engineering design were exemplified by the preceding examples of two very different design processes. The strategies for addressing these problems can be summed up by the following quote from Henry Petroski [Petroski, 1994]:

> *"If persuasive paradigms and instructive case histories can be multiplied and disseminated in an effective way, there is reason to believe that they can become as important a part of the designer's intellectual tool kit as are laws of mechanics, rules of thumb, and computer models."*

Numerous heuristics, design equations, and case histories exist for the design of chemical processes. Chemical engineers use this guiding material whenever they begin the design of a new process, and the use of this material greatly facilitates the design process. Such understanding and guiding material is not ubiquitous for composites design.

The research presented in this dissertation focuses on the application and development of engineering design techniques for designing polymer composite material systems. Principles of engineering design in general, and chemical engineering design in particular, are prevelant throughout this domain. In the spirit of the quote from Petroski, the goal is to develop 'persuasive paradigms' for the design of composite material systems. These paradigms are intended to extend and expand the intellectual capabilities of composite designers by giving them additional 'tools' (both abstract design guidelines and actual software implementations of these guidelines) with which to perform the design of composite materials.

Specific strategies for achieving the increased understanding of engineering design for composites are given in the remainder of this discussion. Key issues of polymer composite material design, including particular design considerations and nomenclature, set the scope of the composite material design problem. The applied computational techniques to be used for enabling the use of the expanded comprehension of this design are also presented.

1.3.1 Polymer Composite Material Design

Polymer composite materials are typically very complex and nonintuitive constructs. Careful consideration must be given to the multiple interactions between materials, processing conditions, and structural features; these interactions are often difficult to assess *a priori*.

The study of polymer composites focuses on the properties and fabrication of materials and is based largely on polymer science and chemical engineering. This area is inherently multidisciplinary. A key to realizing the potential of polymer composite materials is to enable a rapid transition from setting material specifications to the successful commercial manufacturing of a material meeting those specifications. Addressing this challenge becomes increasingly important as polymer composite materials penetrate the durable goods markets. This importance is magnified in performance driven markets, such as aerospace, as these markets become more cost conscious.

There are two main points about polymer composites important enough to draw mention separately. The first reiterates the urgent need for and potential impact of design processes specifically addressing composite materials. The second is the distinction between polymer composite materials and polymer composite material *systems*.

Conceptual Design Considerations. The standard procedures utilized in the generation of a design for a new composite artifact are often heuristic in nature or even worse[1], the result of tedious and expensive trial-and-error testing. Although the composites designer has an overwhelming amount of information available, this information typically consists of experimental data and collections of unorganized heuristics, neither of which is immediately conducive for use in design. Consequently, the design of polymer composites can be viewed essentially as an artform guided by experience and scientific principles.

It has been well documented that the early conceptual design phase for polymer composites determines many of the critical cost drivers for the detailed design of the final composite. Providing support for the conceptual design of polymer composites therefore has great potential for providing substantial downstream cost control. A comprehensive approach to the design of polymer composite materials can provide a way to reduce design and manufacturing time with the goal of getting the design right the first time. This can eliminate or at least reduce the costly redesign process.

1. This does not necessarily imply that heuristics are bad, however. The main problem with using heuristics for design is that designers occasionally use them without regard for their limitations.

Composite Material Systems. Material design for polymer composites involves the mapping from environmental and performance requirements (mechanical, thermal and chemical) to choices for fibers and matrices. Depending on the type of matrix, a detailed recipe of reactive diluents, inhibitors, fillers, initiators, and other additives may be specified as well.

The distinction to be made for the research in this dissertation is between polymer composite materials and polymer composite material *systems*. This distinction is especially important when considering the implications of the design of either. *Design of polymer composite materials* implies a creative aspect to the design, with new classes of materials potentially being invented. As that is not the intent in this dissertation, another term must be used to explicitly avoid this misinterpretation. Hence, polymer composite material systems.

As used within this dissertation (particularly in Parts IV & V), polymer composite material systems specify combinations of polymer matrix materials, chemical agents (curing, cocuring, reactive diluents), fiber materials and fiber lengths. The design of polymer composite material systems specifies nothing other than determining valid combinations of material system constituents.

1.3.2 Applied Computational Techniques

There are two main computational techniques utilized for the design of composite material systems in this dissertation. The representation and reuse of knowledge serve as the foundations for both knowledge-based systems and case-based reasoning. These applied computational techniques are eminently suited for implementing systems that perform engineering design. The stated constraints of a design-centric knowledge representation and an implementation at the proper level of abstraction are in fact strengths of these techniques. Knowledge-based systems are particularly useful for performing estimation and assumption. The entire premise of case-based reasoning is the reuse of previous design solutions.

Knowledge-based Systems. Knowledge-based systems function as aids to the determination of complex (engineering) problems through the use of precompiled domain knowledge and specific inferencing techniques. These systems include representation and inferencing techniques specific to the type of knowledge considered.

The inferencing techniques and knowledge representation upon which the work in this dissertation is based are Task Specific Architectures and Generic Tasks. These techniques entail an engineering science style methodology for designing and developing intelligent decision support systems. This facilitates the knowledge acquisition process by focusing on the high level descriptions of problem solving and not on specific implementation languages. The advantage of a representation framework that can capture the concepts supporting design processes is that it separates the domain knowledge from the computational processes [Brown & Chandrasekaran, 1989], easily permitting knowledge adaptation without a complete reconstruction of the underlying inference procedures.

Case-based Reasoning. A case-based reasoning (CBR) approach is capable of utilizing the specific knowledge of previous concrete design problems. This is an experience-based method which attempts to deal with current problems through a comparison to past solutions. A new problem is solved by finding a similar past case and reusing it as determined by the new situation. As new problems are solved, the solutions are stored as cases. This allows incremental learning, which in turn allows the reasoner to become more efficient. This incremental learning occurs as a case-based reasoning system accumulates both specific and general knowledge. The mixture of knowledge abstractions facilitates the solution of problems requiring the simultaneous handling of multiple interacting considerations.

1.4 Anticipated Outcomes of this Work

An overall vision for the conceptual design of composites has been generated by researchers in the Intelligent Systems Laboratory [McDowell *et al.*, 1993]. This conceptual design vision encompasses every facet of composites design, including material design. While inspired by and fitting under the auspices of this vision, the research in this dissertation departs from the conceptual design vision. Of specific interest in this dissertation is the engineering design of composite material systems, but not necessarily at a predetermined conceptual level of detail.

The primary focus of this dissertation is to examine the ideas behind the design of polymer composite material systems at varying levels of complexity and to develop a cohesive approach for efficaciously performing design. This is to be done with an emphasis on knowledge-based systems, carefully engineering the possible interactions of information and structuring the problem of material design in such a way as to provide not only substantive research results, but genuinely usable answers as well.

1.4.1 Composite Material System Design Knowledge

"Science is built up with facts, as a house is with stones. But a collection of facts is no more a science than a heap of stones is a house." Jules Henri Poincare

The contributions of this dissertation to design knowledge for composite material systems focus on two main issues: the accumulation and organization of design knowledge, and the philosophical nature of the design process.

Not only is design knowledge about composite material systems accumulated, but it is organized in such a manner as to be useful for design. This accumulation and organization addresses both the ill-structured and open-ended nature of the composite material system design process. The open-ended nature of the design process mandates the determination of abstracted heuristic knowledge. Such knowledge is useful for limiting the scope of the design space. Any problems due to ill-structure are eliminated (or at least significantly reduced) by the establishment of a detailed structure of the polymer composite material system design problem.

The applicability of the rigid phase level design process for composites as a way of thinking about the design and as a guideline to implementation of the design process is addressed. An examination of the design tasks comprising the problem to be solved guides this deliberation. A decision is made whether or not the conceptual phase of the design should be exclusively compartmentalized when implemented.

1.4.2 Engineering Design Tool Implementation

"Each problem I solved became a pattern, that I used later to solve other problems." Rene Descartes.

The computational tools presented in this dissertation will be capable of designing composite material systems through the use of both compiled heuristics (knowledge-based systems) and previously detailed designs (case-based reasoning). The archival nature of the CBR system will allow users of the system to access previous cases, speeding up the entire design process. Ideally, the composite material system designer will be used in conjunction with other computational tools which address the design of the processing and structural aspects of the composite material.

The use of knowledge- and case-based systems in the context of polymer composites design facilitates the use and reuse of engineering design knowledge, thus enabling the transfer of expertise. By streamlining the mundane design processes, these systems can also free design engineers for more creative design activities.

1.5 Walkthrough

The main issues involved in and the targeted areas for additional research have been presented in this first chapter, which serves as a Prelude to this dissertation. The two major sections that follow contain the foundations of the research discussed in this dissertation. These foundations are the enabling computational techniques utilized in this research (Part II) and the methodologies & ideologies of various related computational efforts for designing composites (Part III). The heart of the research in this dissertation is presented immediately following, in Part IV. Discussions which extrapolate on these research results and an overall summary of the issues raised by and the accomplishments of this research are presented in Part V.

Part II leads off with an introduction to the basic ideas of artificial intelligence in Chapter Two. The concepts of performing reasoning at both the knowledge level and the task level are also presented in this chapter. A presentation and discussion of the types and specific implementations of task- and knowledge-level approaches used in this dissertation concludes Chapter Two. An in-depth presentation of the theoretical and practical issues of case-based reasoning comprises Chapter Three.

The summary of previous approaches to designing composites that is presented in Part III is twofold. The first part is a wide spectrum of computational approaches to various areas of composites design and a discussion of trends evident in these approaches. This is given in Chapter Four. The immediate legacy of the research in this dissertation, namely, the evolution of the knowledge-based system approach to composites design in the Intelligent Systems Laboratory (ISL) at Michigan State University, makes up Chapter Five.

Part IV discusses how the composites design legacy in the ISL has been specifically augmented. A reevaluation of the previously utilized approach to composites design and the consequent refocusing of the overall design vision is presented in Chapter Six. Also presented in Chapter Six is the specific plan for the research in this dissertation. Chapter Seven sets the groundwork for the development of the next generation composite material designer by closely scrutinizing the previous system, CMatDesign. The lessons learned from this analysis, in addition to the directives presented in the specific research plan, provide the guidance for the implementation of a fourth-generation composite material designer, COMADE. The details of COMADE's implementation and examples of its use are given in Chapter Eight (and in Appendices A through D). Chapter Nine details a new approach to performing case-based reasoning within a generic task environment, and presents two computational tools developed for this approach:

CaseFinder and DesignMod. A specific application of these tools used in conjunction with COMADE is shown in Chapter Ten (and in Appendix E).

A summary of various uses of COMADE, ranging from simple demonstrations to serving as a foundation for the development of a dedicated commercial decision support system, leads off Part V in Chapter Eleven. The lessons learned throughout the development of COMADE, including emphasizing the proper levels of abstraction for design and utilizing the appropriate computational techniques for the design task at hand, are presented in Chapter Twelve. The current design vision, a detailed summary of the specific contributions presented in this dissertation, and suggestions for additional work are listed in the concluding Chapter Thirteen.

Part II

Computational Infrastructure

CHAPTER 2

Applied Artificial Intelligence Techniques

Throughout the history of AI, researchers have been attempting to test their hypotheses of new computational approaches by building test systems. Some of the earliest of these were called expert systems. These systems were a combination of a substantial amount of domain knowledge and one or more reasoning mechanisms for applying this knowledge to problems.

Many of these first systems were in the medical domain, where a great deal of pre-compiled knowledge was available in forms conducive to inclusion in such systems. MYCIN, one the most influential of the early medical expert systems, used production rules for the representation of knowledge [Davis *et al.*, 1977].

Other expert systems such as R1 (also called XCON) [McDermott, 1982; McDermott, 1984] and PROSPECTOR [Hart *et al.*, 1978; Duda *et al.*, 1979] also used the production rule approach to solve nonmedical domain problems. These two systems addressed the domains of configuration of DEC VAX systems and mineral exploration advice generation, respectively. DENDRAL was another early expert system that, while not strictly a rule-based system, inferred the structure of organic compounds from empirical data using production rules with constraint satisfaction [Buchanan and Lederberg, 1971]. Figure 3 shows examples of typical production rules used in these systems.

Many knowledge representation schemes and reasoning mechanisms have been developed besides production rules (e.g., frames, semantic nets, predicate logic, statistical reasoning), but they all can be characterized along a syntactic-semantic representation spectrum. At one extreme are the syntactic systems, which do not bother with the meaning of the represented knowledge. These systems have simple procedures for manipulating the representation. As a result, a shortcoming of the purely syntactic systems was the lack of control for the order of execution of, say, production rules. At the other end of the spectrum are the semantic systems, in which every aspect of the representation corresponds to a different piece of information; the inferencing procedures are correspondingly complicated.

Other approaches besides expert systems have arisen over the years, including blackboard systems (which have a general workspace with competing systems all modifying the problem solution), model-based systems, knowledge-based systems and decision support systems. These approaches have combined mixtures of syntactic and semantic representations, something that the early expert systems generally did not do. Many early expert systems were purely syntactic in nature, and were thus brittle and difficult to modify or validate. That later approaches have moved away from the extremes of this spectrum has helped to decrease brittleness and has also simplified the modification and validation of these systems.

A typical MYCIN rule:

```
If: (1)  the stain of the organism is gram-positive, and
    (2)  the morphology of the organism is coccus, and
    (3)  the growth conformation of the organism is clumps,
then there is suggestive evidence (0.7) that the identity of the
       organism is staphylococcus.
```

each MYCIN rule included a certainty factor

A typical R1/XCON rule:

```
If:  the most current active context is distributing massbus
        devices, and
     there is a single-port disk drive that has not been assigned
        to a massbus, and
     there are no unassigned dual-port disk drives and the number
        of devices that each massbus should support is known, and
     there is a massbus that has been assigned at least one disk
        drive and that should support additional disk drives,
     and the type of cable needed to connect the disk drive to the
        previous device on the massbus is known
then:  assign the disk drive to the massbus.
```

no probabilities in R1's rules

A typical PROSPECTOR rule:

```
If:  magnetite or pyrite in disseminated or veinlet form is
        present
then:  (2,-4) there is favorable mineralization and texture for
        the propylitic stage.
```

there were two confidence values for PROSPECTOR:

the 1st suggested the evidence's validity,

the 2nd indicated the necessity of the evidence.

Figure 3. Examples of typical production rules used in early expert systems.

In this dissertation, knowledge-based or decision support systems are of particular interest. These systems are aids to the solution of a complex (engineering) problem that use precompiled domain knowledge and specific inferencing techniques. These systems include representation and inferencing techniques that are specific to the type of knowledge considered.

The inferencing techniques upon which the work in this dissertation is based can be understood at two levels of abstraction: knowledge level, and task level. This chapter continues with a discussion of each of these levels, and concludes with a presentation of the tools implemented in the ISL to perform these inferencing techniques.

2.1 Knowledge Level Reasoning

At the root of all artificial intelligence research efforts lies a communal need to understand knowledge intensive problem solving behavior. There is an implicit need to do so independently of any implementational specifics. Whether the problem solving has been encoded with Lisp or Cobol or C++, a common mode of understanding the problem solving is needed. The Knowledge Level as proposed by Newell [Newell, 1982] provides such a means for obtaining this understanding of intelligent problem solving:

> Knowledge Level Hypothesis: *There exists a distinct computer system level lying immediately above the symbol level, which is characterized by knowledge as the medium and the principle of rationality as the law of behavior.*

The key insight that this hypothesis embodies is that knowledge can be used to effectively abstract away from the specifics of problem solving behavior, and still be used to define the problem solving process. Endowing knowledge with a problem solving capability facilitates the understanding and explanation of problem solving behavior.

When considered at the Knowledge Level, problem solving becomes a knowledge process [Smithers, 1996].

Newell's Knowledge Level provides a way of understanding a problem solving agent apart from the implementation details. Although this allows a deeper understanding of problem solving than that possible through a symbolic level evaluation, it does not always permit the prediction of the behavior of an agent. This deficiency is due to the absence of problem solving control.

An organizational approach intended to facilitate interaction between different problem solving tasks has been proposed by Sticklen in the Knowledge Level Architecture (KLA) [Sticklen, 1989]. The underlying hypothesis here builds on Newell's proposal by allowing the discussion of the control issue in terms of knowledge organization and control at the Knowledge Level. This organization and control are ensconced in the KLA and the explicit hypothesis of it [Sticklen, 1989]:

> Knowledge Level Architecture Hypothesis: *If a problem solving agent may be decomposed into the cooperative efforts of a number of subagents, the larger agent can be understood at the Knowledge Level by giving a Knowledge Level description of the subagents and specifying the architecture the composition follows.*

This hypothesis enables the specification of the behavior of a multi-task system by explicitly representing the interactions between its agents. To facilitate this interaction, distinct message protocols and fixed communication channels between the agents are used. These provide both a vocabulary for agents to request work (and to respond to such requests for work) and a way of organizing the knowledge of the agents, respectively. Since control is only passed to an agent when another agent sends a request, the Knowledge Level Architecture also provides a means of understanding the problem solving activity occurring among the agents of a multi-task system.

Other researchers have presented alternatives for the integration of multiple problem solving types with the Knowledge Level as a motivating factor. Goel has developed a technique for the combination of case-based reasoning and model-based reasoning in KRITIK [Goel, 1989]. Principles of Punch's TIPS architecture [Punch, 1989] allowed for the flexible integration of different problem solving types. The SOAR system [Laird *et al.*, 1987; Steier *et al.*, 1993] also used a flexible integration architecture for its problem solving operators.

2.2 Task Level Reasoning

Although the Knowledge Level allows an understanding of problem solving at a deeper level than is possible with a symbolic evaluation, it does not always permit the prediction of the behavior of an agent. The lack of control knowledge during problem solving contributes to this deficiency. By considering problem solving at a task level, this control knowledge can be delineated, and the behavior of an agent can be predicted. Task Specific Architectures have been developed for this task level reasoning.

Task Specific Architectures were founded on the idea that modeling of problem solving behavior at the Knowledge Level is not accidental. There are commonalities that can be employed when describing a system's problem solving. Such commonalities are typically general enough to be widely applicable, yet specific enough to allow focussed analysis [Allemang, 1994].

Generic Tasks accomplish these two criteria by delineating task descriptions. A task corresponds to a problem solving goal. Tasks are either decomposable into subordinate tasks or contain applicable problem solving knowledge. Generic tasks are based on the hypothesis that the two antithetical criteria of simultaneous generality and specificity can be embodied within a single task structure.

Task specific architectures and generic tasks both support a long range goal of an engineering science style methodology for designing and developing intelligent decision support systems. These foundations facilitate the knowledge acquisition process by focusing on the high level descriptions of problem solving and not on the specific low level implementation languages.

Generic tasks support the development of specialized role limited languages for the coding of intelligent decision support software systems. The main advantage for using a representation framework that can capture the ideas supporting design processes is that it separates the knowledge from the computational processes that operate on it [Brown & Chandrasekaran, 1989], easily permitting adaptation of the knowledge without a complete reconstruction of the underlying inference procedures.

The approach to design employed in this dissertation follows the generic task methodology, which restricts itself to the high level problem solving goals, knowledge structuring and inference strategies of a problem solver and does not concern itself with the implementation details. A complete description of a generic task should give a listing (and explanation) of: information processing, representation of knowledge, and inference strategy [Bylander & Chandrasekaran, 1987].

Solutions to real world problems require some sort of problem solving technique. Generic tasks are techniques that have utility in solving these problems. Which technique can solve the problem can be determined by looking at the types of input and output and the specific intended function. This input/output consideration of a problem is the

information processing task. The **representation of knowledge** deals with the organization and structure of knowledge necessary to accomplish the GT's function. Directly associated with the representation of knowledge are the ontological commitments made by the GT and how knowledge is organized in terms of the utilized ontology. The **inference strategy**, or the problem solving technique used by the GT, can be defined by how it operates on the knowledge.

It has been acknowledged by researchers working with GTs that the term "generic task" is misleading [Brown & Chandrasekaran, 1989]. The important idea to keep in mind is that what is labelled to be a "generic task" is really a fundamental generic combination of information processing, representation, and inference strategy. It should come as no surprise that this confusion over what is or is not a generic task has led to debate over the number of actual generic tasks that exist.

2.3 Employed ISL Generic Task Tools

There are three commonly accepted GTs that are of interest in this dissertation: Structured Pattern Matching [Bylander *et al.*, 1991], Routine Design [Brown, 1987; Brown & Chandrasekaran, 1989] and Hierarchical Classification [Bylander & Mittal, 1986]. Each of these generic tasks exists within the ISL's GT toolset and is encoded using ParcPlace-Digitalk's VisualWorks version of Smalltalk. Beyond these three generic tasks, the toolset includes another item that is of interest to this dissertation. This final item is a preliminary integration architecture for combining different generic task problem solvers.

2.3.1 Structured Pattern Matching

This inferencing technique was derived from the techniques used in an early game-playing program, Samuel's checkers program [Samuel, 1963]. This program eventually played checkers well enough to beat its creator. Its "skill" was due in part to two capabilities that inspired structured pattern matching: organized storage of information, and generalization. These two capabilities imply a type of data caching called *rote learning*.

Structured pattern matching is a simple inferencing mechanism for performing inferences of the form: "If conditions A, B, C,.... hold, then result X is valid." Rules of this form are combined and organized into patterns corresponding to a particular portion of the problem domain (e.g., malfunction diagnosis for a particular subsystem of a chemical reactor). It is this domain-driven organization and structure that distinguishes these rules from the production rules discussed earlier.

Figure 4 shows an example of a typical structured pattern matcher. In this figure three rules related to a condensate withdrawal system in a chemical reactor are combined into a structured pattern matcher. This matcher considers the variables **CondensateFlowRate** and **IncorrectSolventRatio** to assess if there is a potential problem with the condensate withdrawal portion of a chemical reactor.

Each of the three rows represents a rule related to this portion of the reactor. The first row translates as: "*If the condensate flow rate is not normal, then there is a strong indication of a problem with the condensate withdrawal system.*" In the second row, the matching pattern reads, "*If the solvent ratio is incorrect, irregardless of the condensate flow rate, then there is a strong indication of a problem with the condensate withdrawal system.*" Finally, the third matching pattern reads, "*If the condensate flow rate is normal*

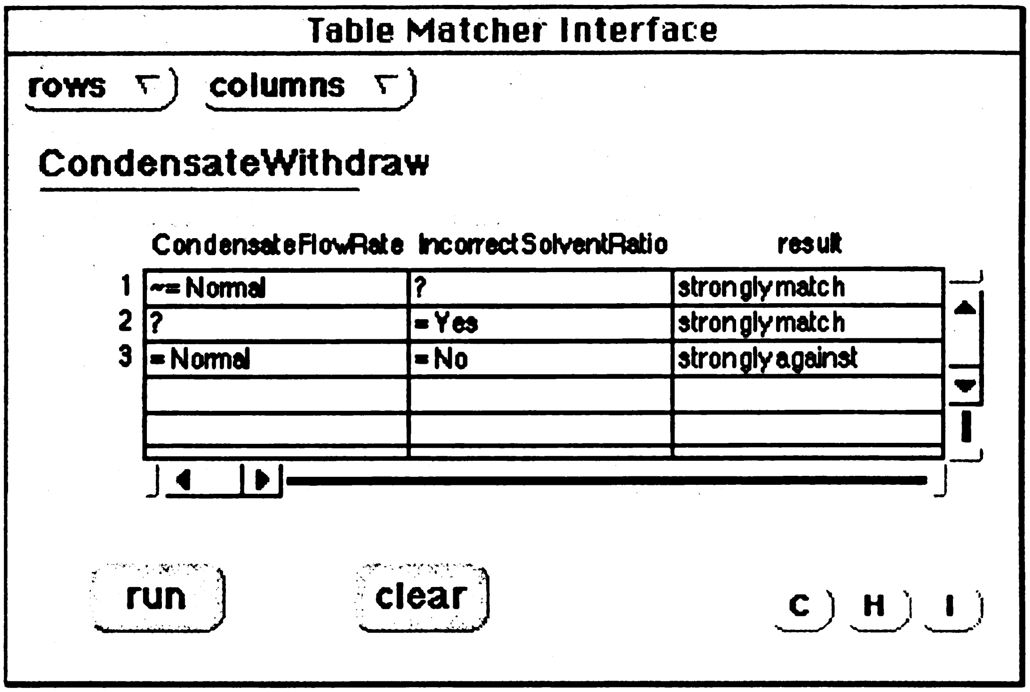

Figure 4. SPM for malfunction diagnosis of a condensate withdrawal system.

and the solvent ratio is correct, then there is a strong indication that there is no problem with the condensate withdrawal system." The question marks in rows one and two show that the value of that variable is unimportant for the pattern of interest.

The structured matching task involves hierarchical symbolic abstraction. An abstraction of the data is computed as a degree of fit. Structured pattern matching is symbolic because the abstraction is presented as a discrete qualitative measure of fit (e.g., strongly match, weakly match, against, etc.). It is hierarchical because the final abstraction is evaluated from intermediate abstractions, which in turn can be the culmination of nested sub-abstractions.

There are seven arbitrary qualitative levels to describe the fit, but not all are necessary for each matcher. Tables that incorporate structured patterns are typically sparse and computationally very manageable, as only a few of these levels ever need be considered; this is due to the abstracted nature of the patterns. Notice that in the example for the condensate withdrawal presented above, only *strongly match* and *strongly against* were used. This task of matching hypotheses against data is a general subtype of reasoning useful in many contexts, and is used with the other generic tasks.

2.3.2 Routine Design

The Routine Design architecture was proposed by Brown for performing design and planning tasks in areas for which substantial experience is available. Brown defined a method for Routine Design, and an accompanying representation language, DSPL (Design Structures and Plans Language) [Brown, 1987; Brown & Chandrasekaran, 1989].

As an engineer designs similar artifacts time and again, the design process becomes routine. With this repetition, the engineer discovers effective ways of decomposing the artifact design into smaller design problems and detailed designs for each of these subproblems. Through this accumulation of experience the engineer now knows at each stage of the design what the options are and in what order to choose them. However, this does not necessarily mean that the designer knows the exact sequence of steps for the overall artifact design in advance.

Finding an overall plan for the artifact design is non-trivial problem that can be quite tedious; knowledge-based problem solving is especially suited for handling this. Note that Routine Design does not organize knowledge for novel or exploratory designs. It depends instead on available expertise and offers templates for capturing and facilitating the use of this design expertise.

Design problem solving in Routine Design is based on a hierarchical structure of design **specialists**, each of which offers a collection of expertise or knowledge on how to complete part of the design problem. This hierarchical organization implies that the design problem is decomposable or nearly decomposable into appropriate subproblems. The specific arrangement of the specialists depends on the regularities that exist in a particular domain. The organization of design subproblems in routine design requires a linear design process. A particular portion of the design problem may affect later portions, but it must also be completely independent of any later portions. In typical control terminology, feedforward is possible (but not necessary) while feedback is not allowed. Besides specialists, Routine Design includes several other knowledge constructs (selectors, sponsors, plans, constraints, tasks and steps).

In Routine Design each specialist follows one of several predetermined plans. These **plans** dictate the problem solving actions to be followed and are defined when the system is built. Each plan has an associated **sponsor** that contains knowledge indicating when the plan is appropriate for use. For a given specialist, the **selector** associated with it examines the results of the sponsors from the subordinate plans and chooses the best plan for execution. A plan may contain other specialists, specific design tasks, checks of design **constraints**, or any combination of these. **Tasks** serve to group related primitive portions of the design process. These primitive portions of the design process are called steps. A **step** is responsible for determining the value of a design attribute. This value may be the result of a computation or it may be selected using a pattern matcher.

Figure 5 shows part of a design decomposition. This figure illustrates several knowledge constructs mentioned above. Thermoplastic Specialist is (obviously) a specialist with three subordinate plans among which its selector may choose. Each of the three plans (Semicrystalline TP, Amorphous TP, and Blend TP) has a sponsor to determine when each is appropriate for further consideration. If, say, Blend TP Plan was selected, its subordinate task and steps would be executed and a specific thermoplastic blend and chemical agent would be determined.

An extension to the standard routine design approach available within the ISL toolset is that of Multiple Design [Kamel, 1994; Kamel *et al.*, 1994]. Whereas standard single routine design specifies only one answer to a design problem, Multiple Design (as its name suggests) can provide more than one answer. The main difference between the two is the inferencing procedure used. Multiple design considers all applicable plans, and

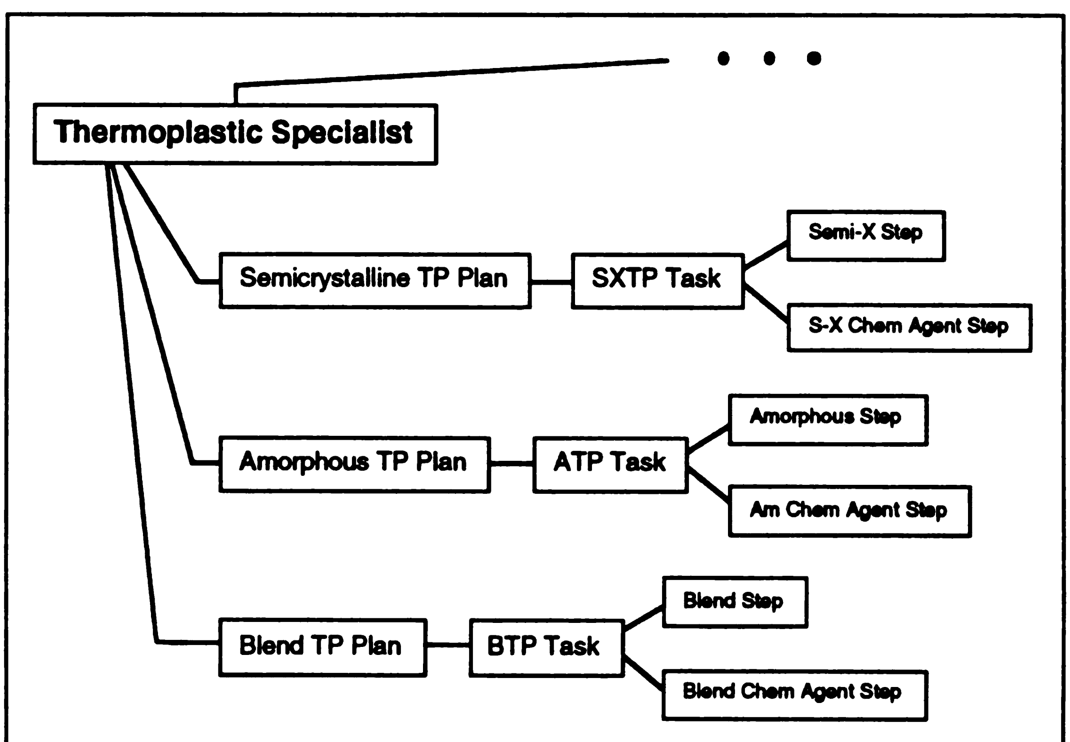

Figure 5. A portion of a design hierarchy for the design of thermoplastics.

not just the best plan as does single design. Therefore, in the thermoplastic example where just the *Blend TP* plan was selected previously, now the *Semicrystalline TP* plan may be selected as well. In this way, more than one design for the thermoplastic may be generated.

Figure 6 shows a direct comparison between the results generated from single versus multiple design. The top output window shows the single design output; that material system design is highlighted in the multiple design output below. Notice that the multiple design approach generated an additional 18 designs from the same input and same knowledge. Each of these additional designs met the use requirements as specified by the user.

2.3.3 Hierarchical Classification

This is intuitively a knowledge organization and control technique for selecting among many hierarchically organized options. The abstract engine used for hierarchical classification (and structured matching), known as CSRL, was the first TSA shell and is described in [Bylander & Mittal, 1986].

A classification problem-solving process is based on the decomposition of knowledge into a classification hierarchy of cooperating specialists. The specialists at the top of the hierarchy represent the most general classification knowledge, with more specific subclassifications distributed in layers beneath. The control regime is a top-down establish-refine mechanism in which each specialist, when invoked, decides if the evidence of the current input supports the classification knowledge the specialist represents. Pruning the hierarchy at high levels of generality eliminates some computational complexity inherent in the classification problem. In this manner, the specialists in a classification hierarchy

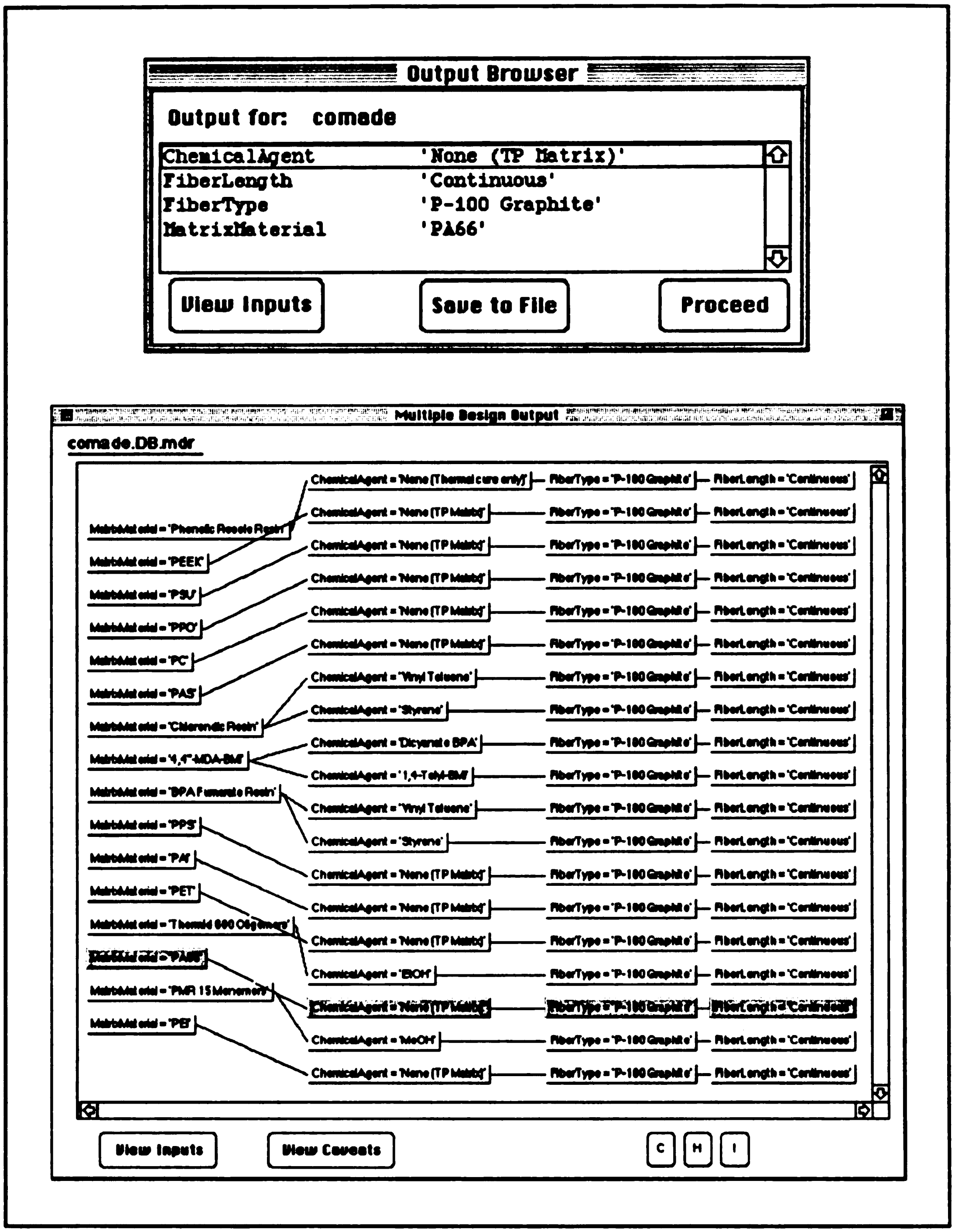

Figure 6. Comparison between single and multiple routine design output.

interact with other specialists as components, sending and receiving messages in a fixed vocabulary along preestablished pathways.

See Figure 7 for an example of a hierarchical classification system. This shows part of a classification hierarchy for the malfunction diagnosis of a chemical reactor. PressureControl will first establish and then refine itself. To establish itself, PressureControl uses locally available domain knowledge to decide if there is any kind of problem with the pressure control in the chemical reactor. If a malfunction is suggested, then PressureControl will refine itself by sending its subordinate specialists (PCtrlReliefValve, NitrogenFeedValves, PCtrlManualValve, and PressureCtrlValve) a message to establish and refine. Consider a malfunction where the pressure control valve setpoint has been incorrectly specified and enough corroborating evidence has been provided. In this instance, only PressureCtrlValve would establish. It would then send a message to its

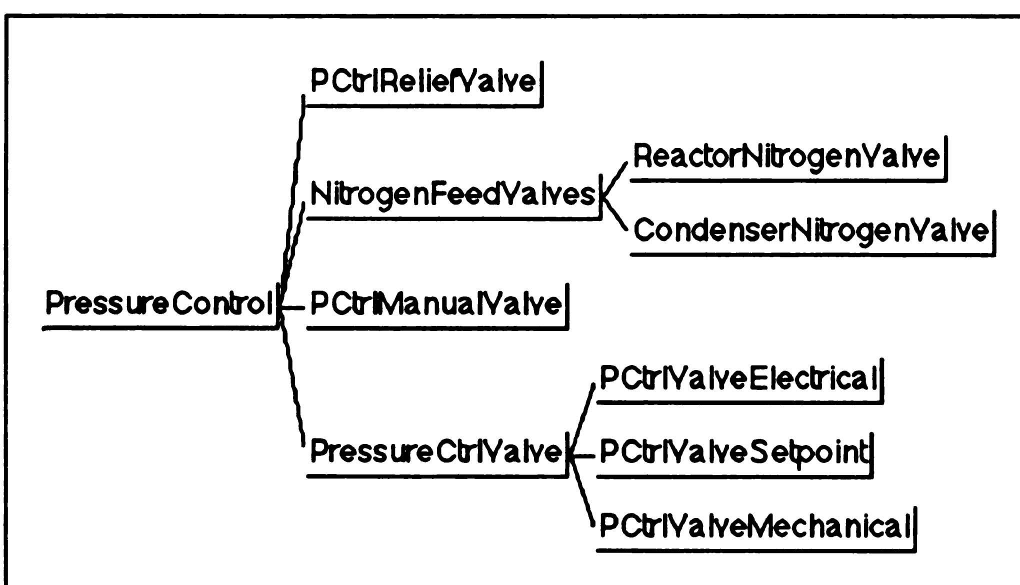

Figure 7. An HC for pressure control diagnosis of a chemical reactor.

subordinate specialists (PCtrlValveElectrical, PCtrlValveSetpoint, and PCtrlValveMechanical) to establish and refine. Only PCtrlValveSetpoint would establish, and as it is a tip node and therefore shows the malfunction, no further refinement would be necessary.

In retrospect, the behavior of portions of both MYCIN and PROSPECTOR can be considered hierarchical classification. The diagnostic portion of MYCIN can be considered to classify a patient description into a hierarchy of infectious agents [Davis *et al.*, 1977], while the behavior of PROSPECTOR can be interpreted as classifying a geological description into a type of mineral formation [Hart *et al.*, 1978; Duda *et al.*, 1979].

2.3.4 Generic Task Integration Framework

Recent exploratory research in the ISL has resulted in a preliminary methodology for integrating two problem solvers [Schroeder, 1995]. This follows in the spirit of Sticklen's Knowledge Level Architecture and is based on establishing a communication channel between a problem solver requesting work and other cooperating problem solving agents.

Possible points for the existence of communication channels between problem solvers are defined explicitly by the developer as the integrated generic task problem solver is built. Explicit mappings between the variables in the requesting and cooperating problem solving agents' databases must also be specified.

The current capabilities of the GT integration framework restrict the requesting problem solver to a routine design problem solver. The cooperating problem solving agents can be hierarchical classification problem solvers, other routine design problem solvers, or algorithmic problem solvers (another component in the ISL GT toolbox). In routine design, instead of setting a parameter at the step level with a structured pattern matcher, a cooperating HC can replace that step. A cooperating algorithmic problem solver can interact with the requesting RD in the same way as a cooperating HC. Cooperating RDs can replace specialists within the requesting RD.

This integration framework has enabled the construction of a case-based reasoning system within a generic task environment, the details of which are presented in Chapters Nine and Ten of this dissertation. However, before the details of such an approach can be presented the requirements of a CBR system must be elucidated. The next chapter does so with a detailed discussion of case-based reasoning.

CHAPTER 3

Case-Based Reasoning

Case-based reasoning (CBR) is a problem solving paradigm that is arguably different in essence from other major AI approaches. Instead of relying entirely on general knowledge of a problem domain, or establishing abstracted relationships between problems and solutions, CBR can use the specific knowledge of previous concrete problems. This is an experience-based method that attempts to deal with current problems through a comparison to past solutions. A new problem is solved by finding a similar past case and reusing it as indicated by the new situation.

The development of CBR was motivated in part by a realization that humans often rely on previous experiences to solve new problems [Schank, 1982]. Other researchers have also contributed to the development of this technique. Work on memory organization gave a framework for a dynamic memory [Kolodner, 1984]. Work on transformational and derivational analogies examined the mapping, adaptation, and replay processes [Carbonell, 1983; Carbonell, 1986]. Other work on analogy has highlighted the importance of the mapping process [Gentner, 1983].

CBR has been demonstrated to be conducive to many types of problem solving applications: planning (e.g. MEDIC [Turner, 1988; Turner, 1989], ROENTGEN [Berger, 1989; Berger, 1994]), design (e.g. PANDA [Roderman & Tsatsoulis, 1993], DEJAVU [Bardasz & Zeid, 1993]), and explanation & diagnosis (e.g. PROTOS [Dvorak, 1988; Bareiss, 1989], CASEY [Koton, 1989]), among others. Those applications that focus on design problem solving are emphasized in the discussion and presented examples throughout this chapter.

For each problem solving application, experiences cached as cases allow incremental learning, which in turn allows the reasoner to become more efficient. This incremental learning occurs as a case-based reasoning system accumulates both specific and general knowledge. The mixture of knowledge level abstractions enables the solution of problems requiring the simultaneous handling of multiple interacting considerations.

At the highest level of generality, the CBR cycle may be described by the following four processes [Aamodt & Plaza, 1994]: retrieve, reuse, revise, and retain. A new problem is solved by retrieving one or more previously experienced cases that are similar along some important dimensions. The case is reused in one way or another and the solution given in the retrieved previous case is revised. Finally, the new solution and the problem it solves is retained by incorporating it into the existing case-base & updating any indexing. These four processes each involve several more specific steps, the details of which are hotly debated within the CBR community. See Figure 8 for a framework of these processes and the interactions among them (after Figure 1, [Aamodt & Plaza, 1994]).

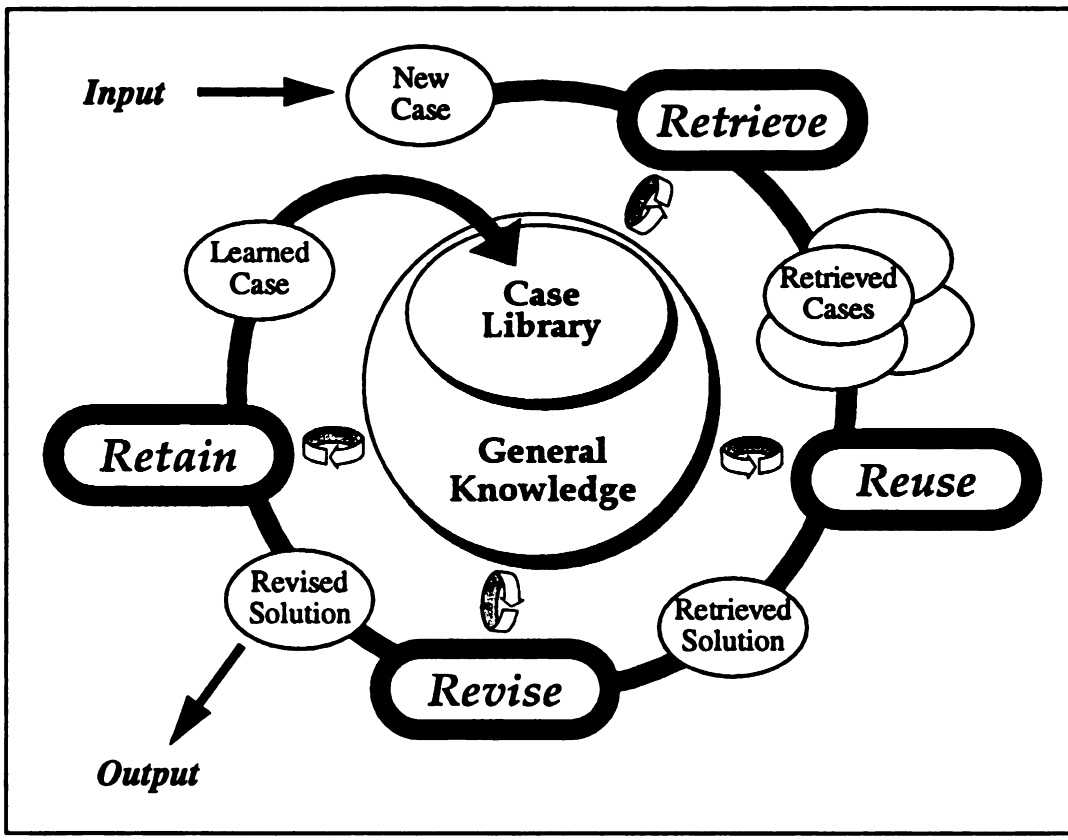

Figure 8. A framework of the CBR process.

To emphasize the utility of case-based reasoning, several CBR characteristics are contrasted with those of other computational approaches in the following section. The remainder of this chapter looks at two broad categories within which the major issues of CBR can be summarized. The first of these is that of knowledge representation and the organization of both cases and libraries of cases. The second category addresses the problem solving tasks comprising each of the four main processes in the CBR cycle (Figure 8), and gives specific examples of how the CBR community has implemented methods for performing these tasks.

3.1 Contrasting CBR with Other Computational Approaches

To understand what makes CBR a useful reasoning technique, and also to point out its unique features, CBR must be compared to other reasoning techniques. Other computational approaches to which CBR can be compared include: rule-based systems, databases, generalization approaches, model-based reasoning systems, and task-specific architectures. The discussion that follows succinctly addresses the differences between CBR and each of these approaches, and is drawn in part from the discussion in [Leake, 1995].

3.1.1 Rule-based Systems

There is sharp disagreement within the CBR community over how comparable rule-based systems are to CBR systems. One school of thought insists that CBR can be thought of as a type of rule-based reasoning in which the rules are very large [Kolodner, 1993], while another explicitly refutes this viewpoint by pointing to the noticeable differences in knowledge use [Leake, 1995].

There are three areas in which CBR can be contrasted with a rule-based approach [Leake, 1995]: the type of knowledge used; how the knowledge is used; and the fundamental role of learning.

The type of knowledge used. Rule-based approaches use if-then rule representations of knowledge that are usually identified by a domain expert. A CBR approach reasons from examples (cases) rather than rules. This presentation of knowledge as cases keeps information fluid and dynamic, whereas when represented as rules, the same knowledge becomes rigid and unchangeable. Due to a mixture of different knowledge abstractions, cases can integrate information; rules by their very nature striate information at different levels of abstraction. However, the main advantage of CBR over a rule-based approach is the relatively easy knowledge acquisition through case addition (as opposed to the generation and inclusion of new domain rules).

How the knowledge is used. A CBR approach replaces a rule-based technique of *generate and test* with *retrieve and adapt*. Whereas the *generate and test* requires the exhaustive generation of a complete solution before it can be checked for accuracy, *retrieve and adapt* allows a much more ad hoc approach to finding an accurate solution. The major advantages of this different technique are increased efficiency and flexibility. Adaptation in case-based reasoning intrinsically implies partial matching. Therefore, knowledge within a case library can be used even when it is not directly applicable to the current situation. This is not an option in a rule-based approach, the rigid structure of which prevents indirectly applicable knowledge from being utilized.

The fundamental role of learning. CBR systems learn by adding cases after a previous case has been successfully retrieved, modified, and identified as significant. CBR systems can also learn from failures. Typical lessons learned from failures include: cases, problems to anticipate, and failure recovery strategies. A major advantage for case-based reasoning is that the CBR system automatically refines its knowledge as it is run. A rule-based system, however, can only learn or refine its knowledge through explicit modification of its rules by an expert.

3.1.2 Databases

CBR is perhaps most closely related to databases; so closely related are the two that a case library can be thought of as a special kind of database [Kolodner, 1993]. Like a database, a case library stores many records. The retrieval algorithms for both must efficiently find appropriate records when requested. The comparison fades, however, when the organization of records and the specific retrieval methods used are considered closely.

Organizational Structures. Design descriptions as expressed within a database are driven strictly by the syntax of the design. The representation of the knowledge in the database's organization is often decided by ease of access and is not necessarily intended to directly contribute to a problem solution. Case libraries, on the other hand, are influenced by the semantics and the syntax of the design descriptions comprising them. In this way, case libraries are structured to facilitate the determination of a problem solution. This organization has a direct impact on the retrieval algorithms that can efficiently obtain records.

Retrieval Algorithms. The main distinction between retrieval methods used for databases and CBR systems is the capability to do partial matching. Databases cannot retrieve according to semantics because of their syntactical organization, and therefore cannot retrieve a close partial match. On this point hinges the utility of CBR. By enabling partial matching (and subsequent adaptation of these partial matches to fit the problem), CBR permits increased robustness and flexibility.

An important point to be made is that the organizational structure of the library dictates the type of retrieval algorithm to be used. Different database retrieval algorithms can search structures such as B-trees or heaps, while different CBR retrieval algorithms exist for searching very broad (breadth-first) or very deep (depth-first) case libraries.

3.1.3 Generalization Approaches

If a system can apply old solutions to new situations, it often uses some sort of generalization. As CBR does indeed apply old solutions to new situations, it can be considered to use generalization techniques. A CBR system with a wide-ranging domain coverage must be capable of doing generalization. As the diversity of the domain increases, so must the capability for generalization. For example, a CBR system that includes representations of both planetary orbits and the migratory patterns of birds must be able to find the similarities that exist between these seemingly disparate types of cases. Two generalization approaches are considered here [Leake, 1995]: inductive generalization, and explanation-based generalization.

Inductive Generalization. CBR learns incrementally, generalizing cases over time as a by-product of the retrieval and adaptation process. A typical inductive generalization approach groups together concepts and then generates a class definition representative of those concepts. Figure 9 shows a simple example of this. Several different flies and bees are shown and a generalization of *flying insects* is given. Aggressive generalization facilitates rapid solutions to new problems, but can be inefficient and even incorrect in domains

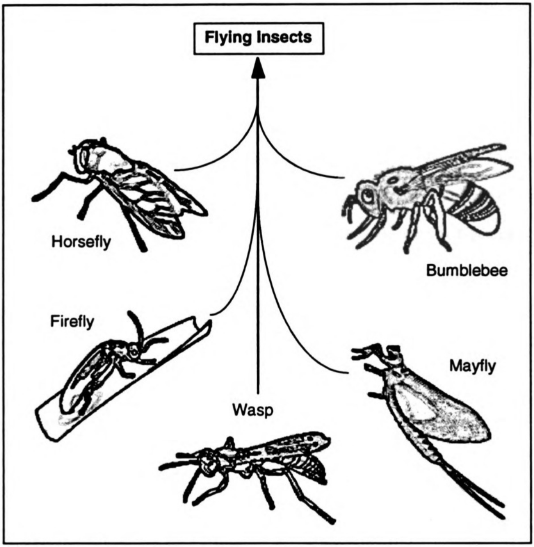

Figure 9. A simple example of inductive generalization.

where proper generalizations are unknown [Rich & Knight, 1991]. Besides incremental generalization, CBR exhibits single-example learning, a natural integration of inductive and analytic learning (through case adaptation), and an emphasis on content (rather than interpretation).

Explanation-based Generalization. A CBR approach does not require a complete domain theory, a prerequisite for explanation-based generalization. The domain theory is used to trim away the unimportant portions of an example with respect to a stated goal. What remains is an explanation of why the example is an instance of the goal. The explanation is then abstracted as far as possible while still describing the goal idea [Mitchell *et al.*, 1986]. As complete and consistent domain theories are seldom realized, getting explanation-based generalization to work efficiently is difficult. CBR is much more flexible, adapting a solution only when needed and allowing lazy generalization & operational cases without a complete domain theory.

3.1.4 Model-based Reasoning

There are several intriguing parallels between CBR and model-based reasoning. Neither performs *de novo* reasoning. The knowledge within each approach is compiled, and the reasoning process within each uses aggregated knowledge. The differences between the two, however, are what are of interest [Kolodner, 1993]: knowledge content, domain utility, and problem-solving relevance.

Knowledge Content. Model-based reasoning fashions solutions by interpolating from the first principles describing a problem domain. For example, the operation of a terephthalic acid reactor may be modeled by comparing the flow rates of the input paraxylene stream, the reactor temperature, and the pressure of the gas exhaust stream. Particular combinations of these three features can indicate a system malfunction. The main distinction to be made here is that CBR eliminates the extensive precompilation required by model-based reasoning. A primary advantage to CBR is that the knowledge upon which its solutions are based is gathered incrementally as the system is used. In this way, a CBR approach can be considered to learn the domain knowledge without a need for knowledge precompilation by an expert.

Domain Utility. When a domain is understood well enough for there to be a suitably robust causal model, model-based reasoning is applicable. CBR is also applicable for such domains. Where CBR distinguishes itself from model-based reasoning is in domains for which there is incomplete knowledge. In such instances, the case library serves as an abstract model of the domain.

Problem-solving Relevance. Solutions can be verified using model-based reasoning, but the initial generation of those solutions is not engineered at all. Conversely, case-based reasoning facilitates efficient generation of solutions. Evaluation of these solutions is based on a relative comparison with the best available cases.

3.1.5 Task Specific Architectures

The representation and reuse of knowledge serve as the foundations for both task specific architectures and case-based reasoning. Task specific architectures are particularly useful for performing estimation and assumption and do so with precompiled domain knowledge and specific inferencing techniques. The entire premise of case-based reasoning is the reuse of previous design solutions. Specific points of comparison between

TSA's and CBR include problem solving approaches, knowledge content, and structures for organizing knowledge.

Problem Solving Approach. A CBR approach uses specific knowledge of previous concrete design problems. This experience-based method solves current problems through a comparison to past solutions. A new problem is solved by finding a similar past case and reusing it as determined by the new situation. TSA also uses specific knowledge of, say, design problems. The specific knowledge in a TSA is represented in an abstracted form, rather than monolithically, as for CBR.

Knowledge Content. A case library consists of both specific and general knowledge in forms of cases and indices, respectively. This mixture of knowledge abstractions makes solving complex problems easier. Heuristics that capture the essence of various facets of a particular problem solving episode are used to find solutions to new problems. These heuristics can be simple or complex (or both), depending upon the problem solving task and the domain in which the TSA is used.

Organizational Structures. Both CBR and TSA's organize the knowledge within them in a way that separates the specific knowledge from the computational approaches used. In a CBR system, case structures are not dependent upon the navigation of the case library or the indices used to organize that library. TSA's also separate the knowledge from the computational processes that operate on it, allowing adaptation of the knowledge without a complete reconstruction of the underlying inference procedures.

3.1.6 Potential Capitalization on Differences

In spite of the preceding emphasis on the contrasts between CBR and other computational approaches, CBR can benefit from interacting with rules, databases, generalizations, models and task specific architectures [Leake, 1995]. Rule-based systems can generate solutions when cases are not available. Databases can manage case libraries. Inductive or explanation-based generalization can be used for indexing and retrieval of cases in a library. Model-based systems can be used to validate retrieved solutions. TSA's provide standard organizational structures and problem solving approaches for use in organizing and retrieving cases.

Each of the computational approaches discussed can also benefit from CBR. Rule-based systems, when used together with CBR, can exhibit increased efficiency. For instance, rules can guide the adaptation of cases to fit a problem different from that to which they are a solution. Rules can also assist in the adaptation of a closely matching case. CLAVIER used rules to limit configurations of parts in the autoclave when it presented the user with possible modifications to the current autoclave part layout [Hennessy & Hinckel, 1992]. The organizational and retrieval techniques of CBR can make databases more flexible. CBR can also supplement generalizations (e.g. with exceptions) or replace them entirely.

There are many instances of integrated systems that combine CBR and other computational approaches. CASEY integrated model-based and case-based reasoning for diagnosis [Koton, 1989], while KRITIK combined the two reasoning techniques for design [Goel, 1989]. JULIA combined CBR with constraint posting and a TMS (truth maintenance system) to keep track of recipe interactions [Hinrichs, 1992]. Anapron uses cases to identify exceptions to pronunciation rules, and the combination of the two performs better than either does alone [Golding & Rosenbloom, 1994]. This list of fielded systems implies that CBR is most effective when part of a problem-solving repertoire.

3.2 Knowledge Representation & Organization

As mentioned in "Databases" on page 59, the organization of a case library can dictate the method(s) which must be used for retrieving cases. This holds for not only the organization of the cases within the library, but also for the representation of the cases and the indices used to summarize the cases.

3.2.1 Case Representation

The domain in which CBR is being used indicates the problem solving task(s) and allows the definition of case composition to be established. A typical task for which CBR can be used is generic problem-solving (e.g., reusing solutions to generate designs (meals: JULIA; devices: KRITIK; buildings: CADRE) or plans (recipes: CHEF; robot control: SIMMS)). Other domains that define the task and case composition include classification/interpretation and teaching/aiding [Leake, 1995]. Depending upon the domain and task for which the CBR approach is used, the exact definition of a case may change, varying in shape and size, temporal coverage, and associational structure (e.g., solutions with problems, outcomes with situations).

Although case definitions are domain- and task-dependent, insight can be gained by abstractly discussing the representation of a case. A distinction can be made between input and stored cases [Leake, 1995]. Input cases can be considered descriptions of specific problem situations. Previous specific problem situations with corresponding solutions and outcomes comprise the cases stored in a case library. Alternatively, stored cases can be said to contain a lesson and a specific context in which the lesson was applicable. The context is used to decide when the lesson may apply again.

Case representations used in fielded CBR systems show possible alternatives for case representation. Simple flat feature-attribute pairs, domain-specific structured descriptions and graphical models are the case representations most commonly found. The case representations of three systems are presented here: DEJAVU, a mechanical artifact design system; CLAVIER, an autoclave layout determination system; and CADRE, a building design system.

The cases within DEJAVU use a design plan representing a mechanical artifact's structure, properties, part & property relationships and abstracted design rationale [Bardasz & Zeid, 1993]; this design plan is an example of a domain-specific structured representation. Design plans within DEJAVU are explicitly encoded as *define-part* constructs. These describe an object by specifying its general properties and constituent parts. For example, a chair can be defined as having certain dimensions (width, depth, height) and materials. It can also be described as having a seat, back and legs.

The cases within CLAVIER contain knowledge about the layout of polymer composites in an autoclave [Hennessy & Hinckel, 1992]. This knowledge includes not only specific details, but abstracted classifications and supporting information as well:

- the parts and their relative positions on a table;
- the tables and their relative positions in the autoclave;
- the classification of the layout (e.g., large-part loads, fiberglass loads, steel-mold loads); and

- production statistics (e.g., start & finish time of the run, ΔT between temperature extremes during the run).

Also implicitly stored within CLAVIER's cases are each part's heat-up characteristics and thermal interactions between part positions. This implicit information also includes the effect of the part's location in the autoclave and its relative position to the other parts in the layout.

CADRE incorporates only shallow cases in its case library. Shallow cases are those cases that contain only a design solution, and no associated design history (as would be present in a deep case). These shallow cases include [Hua & Faltings, 1993]:

- the actual structure needed to satisfy a set of design requirements;
- the trade-offs made between functional requirements; and
- tacit considerations, such as the style of the building.

The actual representation of the cases is an annotated graphic. The structures within CADRE are represented as CAD models. Each structure's functional features are modeled symbolically as mapped constraints on the CAD model. The tacit considerations are not encoded explicitly, but exist within the mapped functional feature constraints.

3.2.2 Indexing Vocabulary

Finding appropriate indices is an ongoing research issue within the CBR community. It is very important to have indices that accurately represent the information embedded within the case library. Without proper indices, the information in the cases is inaccessible and is almost completely useless. The major goal of indexing work is a generally applicable indexing vocabulary, or a cross-domain theory of the representational content of indices [Kolodner, 1993].

Closely tied to case representation is the vocabulary used to index the cases. This index vocabulary is in turn closely related to the overall organizational structure of the case library. This confluence makes it difficult to separate issues as strictly vocabulary- or structure-related. These issues are more easily addressed when considered to be related to the organizational structure. Without going into a great deal of detail, some general principles for "good" indices are presented here. The details are left to an organizational structure discussion relegated to the next section.

Indices should be predictive of important case features and should be easy to recognize [Leake, 1995]. For example, the goal to be achieved by a plan is a good index for retrieving useful plans. This example follows a basic principle of CBR: "Generalize the indices, not the cases!" [Hammond, 1989]. Besides predictive features and a certain degree of abstractness (making an index easy to recognize), indices should incorporate a degree of concreteness and usefulness [Kolodner, 1993].

3.2.3 Organizational Structure

Retrieval methods are related to the methods used for building organizational structures just as those structures are closely related to retrieval methods. Therefore, each organizational structure mentioned here has associated retrieval and building methods. The particular organizational structure that is appropriate for an application is largely dependent upon four issues. These issues include: the number of cases in the case library, the complexity of the indices, the number of different ways the case library may be used, and the variability of the indices with respect to those various uses [Kolodner, 1993].

Organizational structures can be divided into two main categories: flat, and hierarchical. In a flat memory, cases are simply stored in a sequential list, with optional augmentation by shallow indexing or partitioning of those cases. Essentially, there is no "organizational structure" of which to speak for flat memories. Hierarchical organizational structures allow much more variability, including variations such as shared feature networks, category-exemplar networks, and redundant discrimination networks.

The three variations of hierarchical structures are examined by considering characteristics of each structure. Included in this examination are: a basic description, retrieval and structure-building methods, pros and cons of the structure, and an implementation in a fielded CBR system.

Shared Feature Networks. This organizational structure provides a means of clustering cases so that similar cases (i.e., those sharing many features) are grouped together. Each internal node within a shared feature network contains features shared by the cases below it. This clustering of similar nodes and cases continues down to the leaf nodes of the hierarchy, which contain the cases themselves.

Retrieval of a case from a shared feature network involves a type of breadth-first search [Kolodner, 1993]. The input is matched against each node at the highest level of the network. The node with the best match is chosen. This continues until the node chosen is a case. A shared feature network can be built similarly, substituting a clustering method for the node-level matching and the entire library of cases for the input. A shared feature network is then built from the top level down to the individual cases.

The main advantage of this organizational structure is that it partitions cases better than a sequential list, making retrieval more efficient than a serial search. A disadvantage is that adding cases to an existing shared feature network is a complex operation. It is difficult to add new cases while preserving the optimality of the network. When more than one reasoning goal is to be accommodated within the case library, several differently prioritized networks may be needed.

CADSYN's case memory [Maher & Zhang, 1993] is an example of a nested shared feature network. This organizational structure contains two components: case hierarchies and case indexing representations. A design case consists of a "supercase" and multiple levels of subcases. The supercase of a case provides an general design episode context and general description. Each subcase describes the local context and the solution of a design subsystem. Figure 10 shows an abstract shared feature network for a CADSYN design case. Every design case is stored declaratively in a case indexing hierarchy. The subcases can be used independently of the entire case, and are individually indexed along with the links used to retrieve the whole case. The case memory of

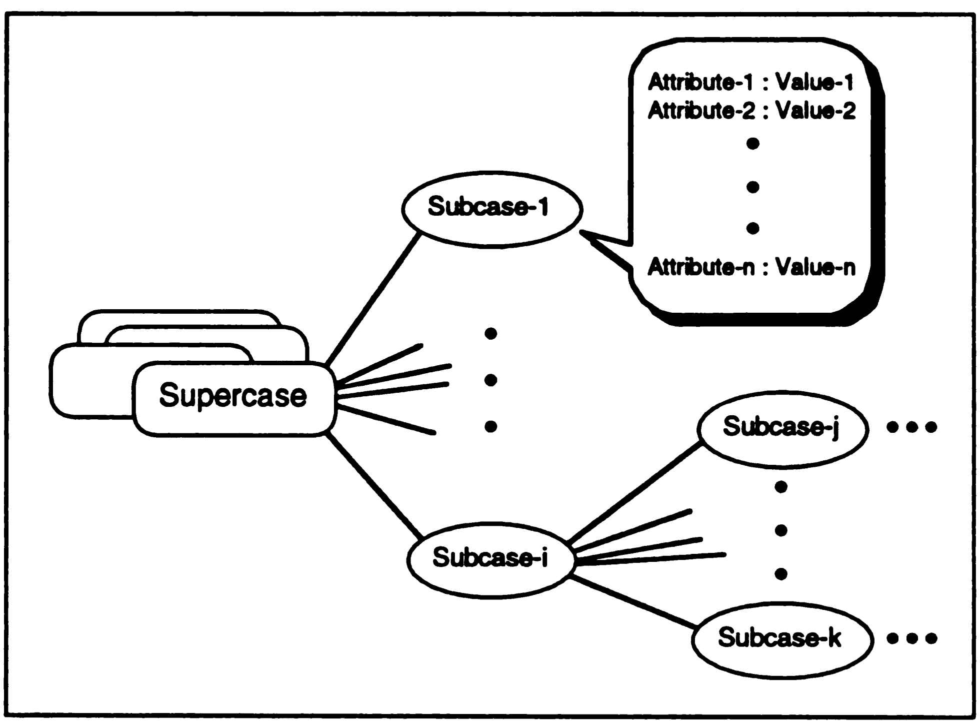

Figure 10. A CADSYN design case as a shared feature network.

CADSYN can be thought of as a shared feature network of *n* nested shared feature networks, where *n* is the number of design cases in the library.

Category-Exemplar Networks. The case memory is represented as a networked structure of categories, cases (or exemplars), and index pointers. Each case is associated with a category. An index may point to either a case or a category. There are three kinds of links: descriptive links, called remindings, which point from features to cases or categories; case links, or exemplar links, which point from categories to related cases; and difference links between cases whose features only vary slightly. A feature is generally described by a name and a value. Exemplars within a category are ranked according to how representative the exemplars are of that category. The linking of features and exemplars to and from categories within this memory structure can be seen in Figure 11. The arrows between the

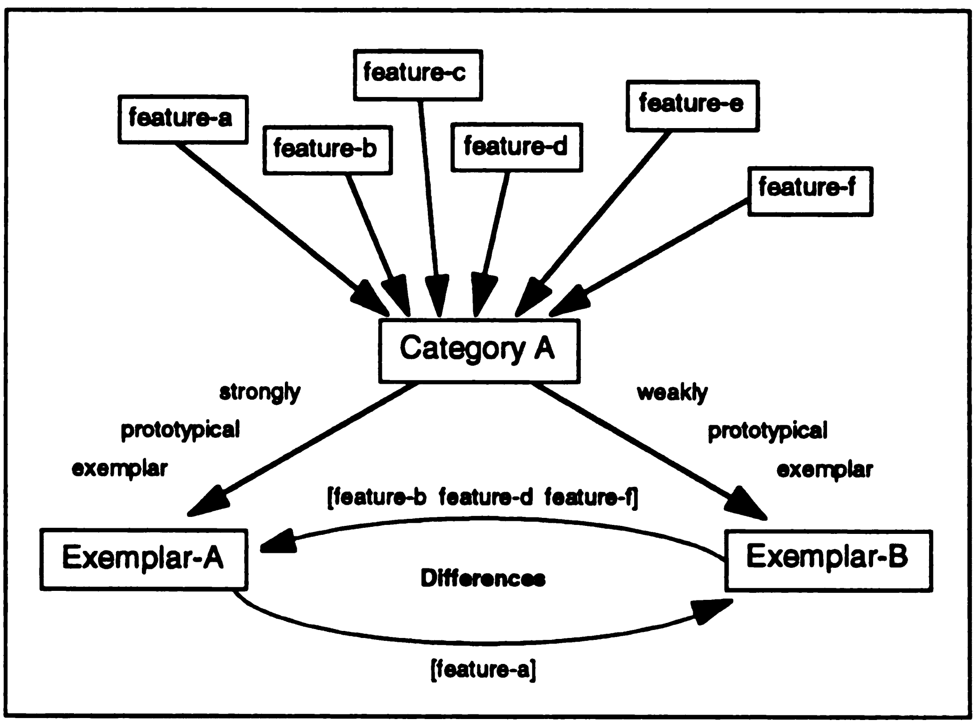

Figure 11. A Category-Exemplar case memory structure.

features and the category within Figure 11 are remindings. A semantic network that contains features and intermediate states (e.g. subclasses of goal concepts) links categories within the case memory. This network represents general domain knowledge and enables explanatory support for some CBR tasks.

Based on the surface features of an input case and in conjunction with the remindings, relevant categories are identified and the most likely is selected. The most prototypical case from the identified category is selected using exemplar links and its semantic similarity to the input is assessed using the general domain knowledge. If the two are similar enough, the category is valid. It the match is not close enough (by whatever measure used), difference links are used to choose another similar case (exemplar). This process continues until a close match is found. A new case is stored by establishing the appropriate feature indices and by searching for an appropriate category and a matching case. If a case is found with only minor differences, the new case may not be retained. The two cases may also be merged by following taxonomic links in the semantic network. Otherwise, the new case is placed within the category.

A category-exemplar organizational structure works well due to its combination of knowledge-based matching, rich semantic matching between exemplars, and a failure-driven knowledge acquisition process. The existence of a failure-driven knowledge acquisition process points out a potential flaw with this organizational structure, however. In order to find good matches for new cases, difference links must exist between cases. These links are added to the case memory by an expert only when the existing structure fails. A learning curve does indeed exist for any system constructed using this organizational structure.

The PROTOS system for diagnosing hearing disorders [Dvorak, 1988; Bareiss, 1989] used category-exemplar networks to organize cases in its case memory. The aim of PROTOS was to use the category-exemplar structure to find the best category, or disorder, for a set of input. When given a set of patient complaints and descriptions (e.g., speech:normal, age:greater-than-65, o-ac-reflex-u:elevated, and so on), PROTOS found the categories with which these features were associated and chose the best match. In this example, PROTOS was reminded of cochlear-noise, cochlear-age-and-noise, otitis-media, etc. Among these remindings, cochlear-noise was the strongest. The most prototypical exemplar of cochlear-noise was compared with the input set and found to not match well. By following a difference link labeled age:greater-than-65, PROTOS found another case under the cochlear-age-and-noise category. The match to this case was found to very close, and the patient was diagnosed as having cochlear-age-and-noise.

Redundant Discrimination Networks. The organizational structure in a redundant discrimination network is comprised of 'episodic memory organization packets' (E-MOPs [Kolodner, 1983a; Kolodner, 1983b]), or generalized episodes [Koton, 1989]. This model was developed from Schank's more general MOP theory and dynamic memory model [Schank, 1982]. The basic idea is to organize similar cases under a more general structure (a generalized episode). A generalized episode (GE) contains three different types of objects: norms, cases and indices. Norms are features common to all cases indexed under a GE. Indices are features which discriminate between a GE's cases. An index, which is comprised of a name and a value, may point to a more specific GE, or directly to a case.

Figure 12 shows a complex GE with its underlying cases and a subordinate GE [Aamodt & Plaza, 1994]. The entire case memory is embodied in a discrimination

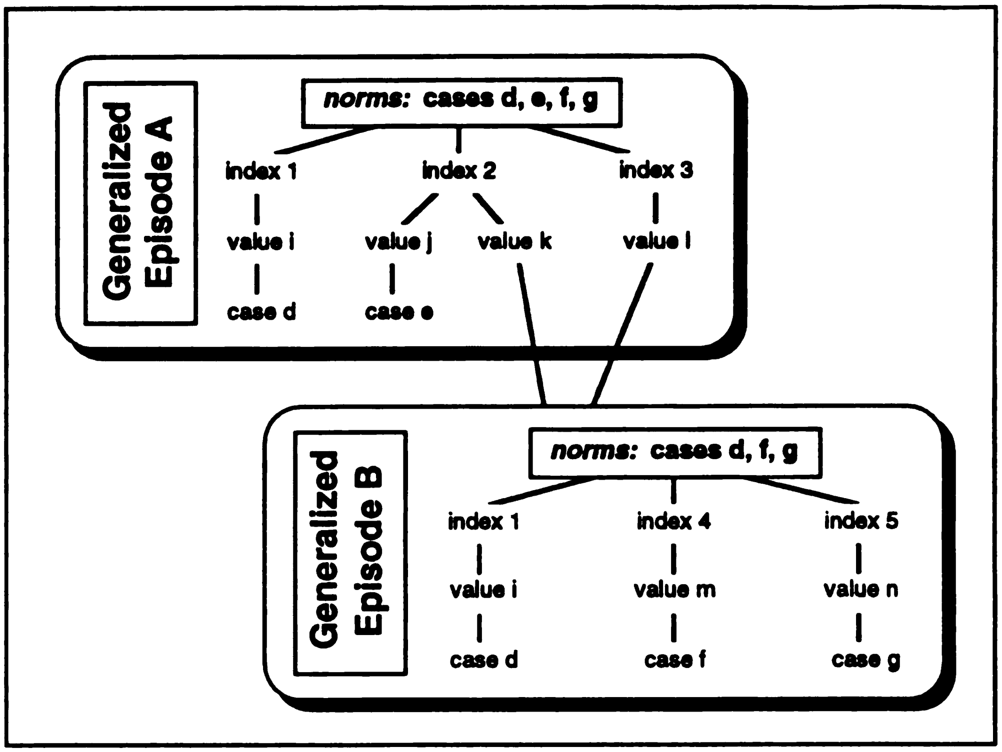

Figure 12. Generalized Episodes within a case memory structure.

network in which nodes are GEs (containing the norms), indices (names or values), or cases. Indices point from a GE to another GE or to a case. The indexing scheme is redundant, as there are multiple possible paths to a particular case or GE. This is shown in Figure 12 by the indexing of *case d*, which appears in both GEs.

As implied by its name, there is more than one network within a redundant discrimination case memory. Each network organizes items with a different ordering of questions. To retrieve a case, the networks are all searched in parallel. A case is retrieved by finding the GE with the most similar problem description norms. Indices under that GE are then traversed in order to find the case which contains most of the additional problem features. If the answer to a question within one network is missing, further search in that network is discontinued. Storing a new case is performed in the same way, with the additional process of creating generalized episodes. This memory structure is dynamic in the sense that similar parts of two case descriptions can be generalized into a GE, under which the cases are then indexed.

The main advantage for representing case libraries in redundant discrimination networks is the many different paths to each item in the network. Multiple paths allow retrieval even when features are missing in an input set [Kolodner, 1993]. That generalizations are formed as cases are incorporated is an added bonus. Since the index structure is a discrimination network, cases are stored under each index to differentiate them from other cases. This can lead to an explosive growth of indices as the number of cases (not to mention case complexity) increases. Most systems using a redundant discrimination network therefore enforce limits on the choice of indices for the cases by, say, permitting only a small vocabulary of indices [Aamodt & Plaza, 1994]. An additional disadvantage to this organizational structure is that the indexing within the networks may not discriminate well enough, resulting in the retrieval of barely-matching cases.

CASEY was a system for diagnosing heart failure [Koton, 1989]. It combined CBR with an accurate model-based reasoning system. As with most medical systems, CASEY stored a large amount of information in its cases. In addition to all observed features (i.e. description of and symptoms in the patient), it retained the causal explanation for the diagnosis found and the list of indicated states in the heart failure model. These states, referred to as generalized causal states, were also the primary indices to the cases. CASEY used redundant discrimination networks to organize its case memory around these generalized causal states. It also included prioritization of the discrimination scheme in order to eliminate the retrieval of barely-matching cases. This was an important addition, as the same heart disorder could manifest itself in many different ways, and different disorders could be symptomatically very similar.

3.3 Problem Solving Tasks

As mentioned in the introduction to this chapter, a general description of case-based reasoning decomposes it into four main processes: retrieve, reuse, revise, and retain. Each of these processes can involve a number of more specific steps. A summary of the important issues within each of these four processes follows. This discussion draws on the framework presented in [Aamodt & Plaza, 1994].

The four-process view in Figure 8 on page 56 emphasizes CBR as a cycle of sequential steps. A task-oriented perspective can describe the details of the four processes. In such a description, each step or subprocess is viewed as a small-grain problem solving task the CBR reasoner can perform. The idea of a task here is analogous to the discussion of Generic Tasks and Knowledge Level reasoning in the previous chapter.

A knowledge level view of a system is that of an agent which has goals and means to achieve them[1]. Three facets of a system can describe it: tasks, methods, and domain knowledge models. Tasks are defined by the goals of the system, and a task is performed using one or more methods. To accomplish its task, a method needs knowledge about the overall domain, the current problem, and the context of the current problem. A method

1. This perspective differs slightly from the KLA of Sticklen that was presented in the previous chapter [Sticklen, 1989]. The main distinction is that the KLA considers things more abstractly than does the interpretation used here.

specifies the algorithm that identifies and controls the execution of subtasks, and accesses and utilizes the knowledge and information needed to do this [Aamodt & Plaza, 1994].

It is important to keep in mind that not every CBR system incorporates every step, not to mention the four main processes, covered in the following discussion. Generally, the detailed steps involved in a CBR implementation are determined by the overall problem solving task and domain for which the system is intended, and not by any precompiled recipe for CBR. A system developed to use CBR for supervised learning has a different emphasis than a system for planning or design, or for that matter, a system for explanatory diagnosis or argument construction. This has resulted in wide variations in the definition of what is or is not CBR. The four step process description of CBR is universally accepted within the CBR community; it is the details which are hotly debated.

The task-method structure referred to in the following discussion is shown in Figure 13 (after [Aamodt & Plaza, 1994]). Tasks are indicated by solid letters and borders, while methods are indicated by italics and dashed borders. Task decompositions are shown by solid lines and proceed outward, with the superior tasks more heavily emphasized. The four major CBR tasks shown correspond to the four processes of Figure 8. This figure flows from the center outward and clockwise (at every level: task, subtask & method) starting from the RETRIEVE task. For example, RETRIEVE is partitioned into **identify features**, **search**, and **initially match**, in that order of execution. Similarly, **identify features** is partitioned (in order of execution) into **collect descriptors**, **interpret problem**, and **infer descriptions**.

Every task partition in the figure is intended to be complete, i.e. the set of subtasks of a task is sufficient to accomplish the task. The figure does not show any control

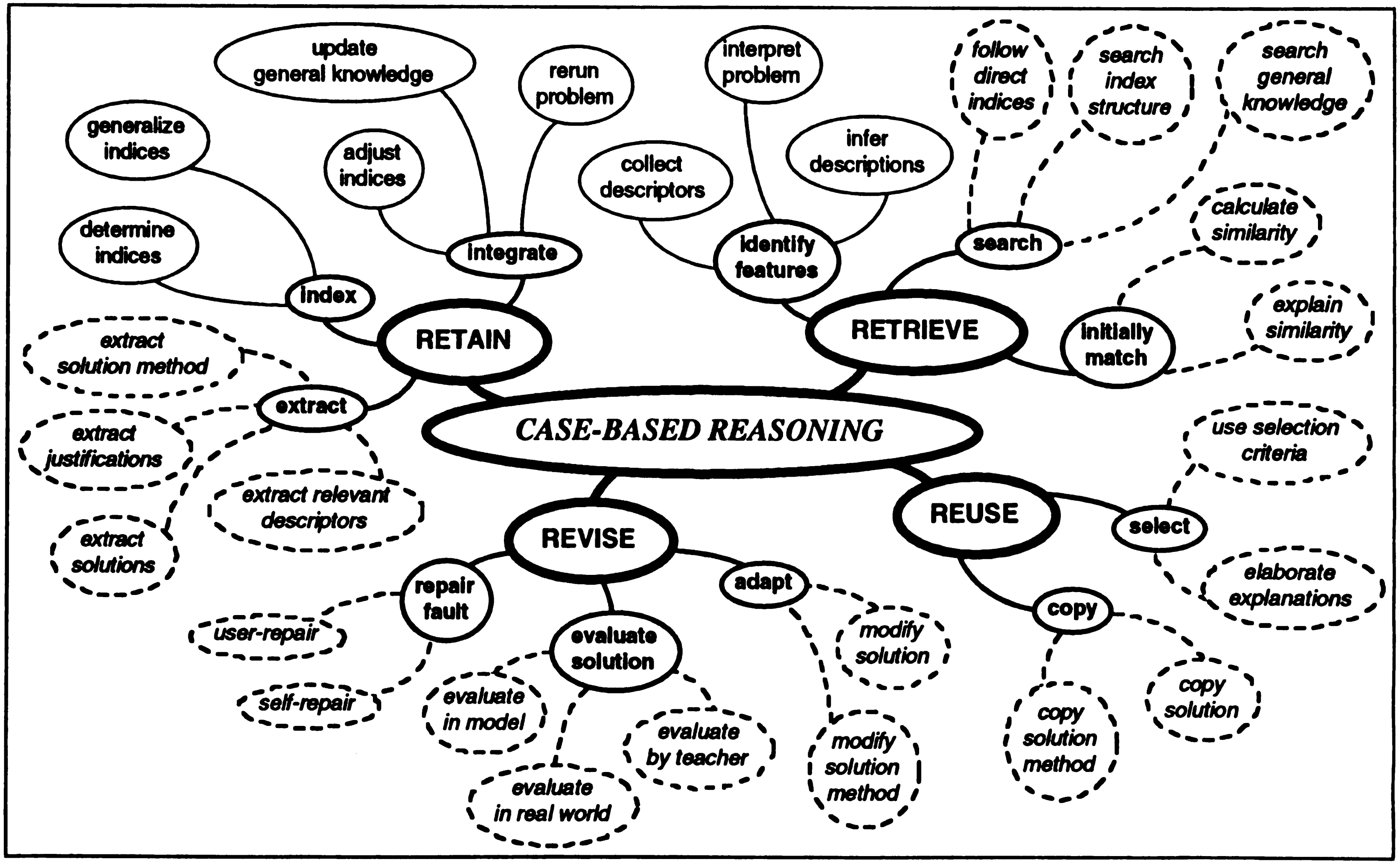

Figure 13. A task-method decomposition of CBR.

structure over the subtasks, although a rough sequencing of them is indicated by the layout flow as mentioned above. Multiple methods for a task identify alternatives for solving that task. Any one of the methods indicated may be sufficient to solve the task, several methods may be combined, or other suitable methods may exist.

3.3.1 Remembering the Appropriate Case(s)

The process of case retrieval, or remembering, is a complicated one which involves searching a potentially massive memory. As a new case (used as the retrieval probe) is unlikely to match exactly with a stored case, some sort of partial matching must be performed [Rissland *et al.*, 1989]. The retrieval of relevant cases becomes quite important in order to operate efficiently. Determination of the relevancy of cases can be done by a comparison of indexes, or abstracted comparators. Retrieval can also be thought of as looking for syntactic similarity.

Supervised learning systems use Euclidean distance and other syntactic similarity functions to retrieve cases[1]. Planning & design systems retrieve those cases which have the most shared goals with the input case. Diagnosis systems follow prototypicality links (e.g. PROTOS) and use causal feature indices. Argument construction systems retrieve cases that best support pro & con arguments as well as cases supporting counter-arguments.

The retrieve task starts with a problem description and ends when the best matching previous cases have been found. Its subtasks are **identify features**, **search**, and **initially match**. The identification task comes up with a set of relevant problem descriptors.

1. Syntactic similarity approaches such as Euclidean distance work well only when quantitative information is available within the case. Cases which are comprised mostly of qualitative information cannot be easily retrieved (if at all) by an assessment of their syntactic similarities.

The search task uses these descriptors to look through the indices of the case memory. The matching task returns a (small) set of cases that are sufficiently similar to the new case.

Identify Features. To identify a problem may involve simply noticing its input descriptors, but a more elaborate approach can be taken in which an attempt is made to understand the problem within its context. Unknown descriptors may be ignored or the system may request an explanation from the user. When an input feature was unknown to PROTOS, for example, it asked the user to supply an explanation linking the feature into the existing semantic network (category structure) [Koton, 1989]. Understanding a problem may involve eliminating extraneous problem descriptors, inferring other relevant problem features, checking whether the feature values make sense given the current context, or generating expectations of other features. Descriptors other than those given as input may be inferred by using a general knowledge model. Another way to infer descriptors is to retrieve a similar problem description from the case base and use the features of that case as expected features.

Search. The search, or retrieval, method is dictated to a great degree by the organizational structure of the case library. Flat case memories can be searched using a simple serial search in which every feature in every case is evaluated sequentially. Shared feature and redundant discrimination networks imply breadth-first searches, while prioritized discrimination networks imply depth-first searches. Parallel search methods can be used for nearly any organizational structure.

Irregardless of what type of search is used, there are three ways of retrieving a case or a set of cases: by following direct index pointers from problem features, by searching an index structure, or by searching a model of general domain knowledge. PROTOS combined the first and third of these, as it used direct pointers to hypothesize a candidate set and then justified that set as a plausible match using general knowledge.

Initially Match. An initial matching process retrieves a set of plausible candidates for the input case. Input features may directly guide the retrieval of cases or cases may be retrieved using features inferred from the input. Cases matching every input feature are occasionally found, but more often cases matching only a fraction of the problem features are retrieved. The relevance of a retrieved case is evaluated, particularly if it was retrieved based on a subset of features. A simple relevance test may check if a retrieved solution conforms with the expected solution type of the new problem. The degree of similarity must be assessed, generally by using methods which compare the surface similarities of both problem and case features.

3.3.2 Using the Retrieved Case to Determine a New Solution

The retrieved case is combined with the new case (the user input) in this step to form a solved case, i.e. a proposed solution to the initial problem. This gives a rough first pass answer about how to solve the new problem based on the solution method existing within the retrieved case. Often times, this will involve the identification of specific features which must be considered explicitly in order to solve the problem. For instance, a new case would not necessarily have a complete representation of the problem, let alone the solution. The retrieved case contains similarities to the specified portion of the new case, as well as detailed plans and procedures which describe a possible solution to the problem. Reuse can also be thought of as looking for semantic similarity.

The reuse process in supervised learning systems is trivial. Planning & design systems either use only the relevant portions of solutions or merge previous solutions to form a new solution. Diagnosis systems do not perform reuse per se. In argument construction systems, an improved match strengthens an argument, while other cases are used to explain unmatched facts.

The reuse of a retrieved case solution(s) for the input case addresses two issues: the differences between the past and the current case, and what part of a retrieved case is relevant to the new case.

Select. Choosing the best case is generally necessary after the set of partially matched cases has been determined. As it is often the retrieved cases that determine which of the new features of the new case are the most important for matching, contextual derivation of the importance of features must be utilized instead of a generic count or weighted count of matching features. The cases are ranked according to a metric or ranking criteria. Knowledge-intensive selection methods can generate explanations to support this ranking, and the case that has the strongest similarity explanation is chosen. Other properties of a case that are considered in some CBR systems include: relative importance and discriminatory strengths of features (e.g. CLAVIER [Hennessy & Hinckel, 1992]), prototypicality of a case within its assigned class (e.g. CASEY), and difference links to related cases (e.g. PROTOS).

Though the methods used may vary, a commonality is that the set of items retrieved each contributes general knowledge about what is important in solving them. Using this knowledge, the case selector can decide what to take into account when determining the best cases. The best cases are those which address the reasoner's current problem in the best way (the definition of which is dependent upon the method used).

Copy. In simple classification tasks the differences between the input case and the best retrieved case are considered non-relevant and are abstracted away. The solution structure of the retrieved case is then transferred to the new case as its solution structure. This is a very simple type of reuse. Other systems take into account differences between the past and the current case, preventing the direct transferal of the relevant part of the retrieved case. In such systems an adaptation process takes those differences into account.

3.3.3 Adapting and Evaluating the New Solution

As new situations rarely match the old ones exactly, the appropriate old solutions must be adjusted to fit the current situation. The main issue here is that of adaptation. After the old case has been modified to match the new situation more closely, an evaluation of this new solution must be performed. If any inconsistencies or errors in the new solution are found, that solution must undergo additional adaptation, or repair.

It is commonly thought that there are no universal adaptation methods, as the idiosyncrasies of every domain (and therefore, case structure) demand varying approaches for modification. Supervised learning systems do not include any revision. Planning & design systems either make repairs to reused cases or indicate which repairs need to be made. Diagnosis systems reevaluate cases by justifying and potentially modifying the indicated diagnosis. Argument construction systems examine and look for ways to increase the strength of matching arguments.

Adapt. Initial adaptation is comprised of two main steps: figuring out what needs to be done and actually doing the adaptation [Kolodner, 1993]. One particular approach to adaptation is fairly ubiquitous: deleting extraneous information. If a secondary component in a case performs no necessary function for the new situation, then removing it harmlessly streamlines the solution.

The past case solution can also be modified through transformation or by derivation [Riesbeck & Schank, 1989]. In transformational adaptation, the past case solution does not directly provide a solution for the new case. It can, however, be modified into a solution if the system contains knowledge appropriate for doing so. System knowledge for performing this transformation can be indexed around the differences detected among the retrieved and current cases. In CASEY, a new causal explanation was built from the old causal explanations using rules together with a domain model. Transformational reuse requires a robust domain model plus a control mechanism to guide the application of this model.

Derivational reuse examines the problem solution in the retrieved case. This contains information about the solution method used for the retrieved problem. The retrieved method is then applied to the new case in the new context. During the process, previously successful alternatives will be considered first; new subgoals are pursued based on the old ones and old subplans can be retrieved as needed.

An exhaustive review of current adaptation practices can be found in [Hanney, 1996]. This review goes beyond the general summary given here, explicitly identifying existing adaptation taxonomies and presenting a novel role-based taxonomy for adaptation knowledge.

Evaluate solution. The evaluation task looks at how well the modified solution works as determined by an expert, use in the real world, or use in a model. This generally takes place outside the CBR system as it involves the application of a suggested solution to a real problem. Depending on the type of application, the results from this may take some time to become apparent. The case can still be retained and made available in the case library in the interim, but its questionable validity must be indicated.

Repair fault. Case repair is essentially an additional adaptation process which retrieves or generates explanations for detected errors in the solution. Causal knowledge can be used to generate an explanation of why certain goals of the solution were not achieved. This explanation is stored in a failure memory and can be used to predict possible shortcomings of initially retrieved solutions. Solution repair uses the failure explanations, modifying the solution to prevent failures. The revised solution can then be retained directly (assuming it is correct) or it can be evaluated and repaired again.

3.3.4 Saving the New Solution for Subsequent Use

The newly solved or interpreted case must now be stored in the case library. This process incorporates useful information from the new problem solving episode into the existing knowledge. The outcome of the evaluation (and possible repair) task dictates whether successful or failed case information is stored. Retaining case knowledge involves selecting the appropriate information from the case and determining the form in which it is to be retained. Actual storage of this information in the case memory is done by appropriately adjusting indices and other organizational mechanisms.

As for the other main processes of CBR, many different approaches for handling the storage of cases exist. Supervised learning systems store specific instances of cases and update the feature weighting in the case memory. Planning & design systems save partially ordered and annotated plans, failure information, and update utility estimates for existing cases. Diagnosis systems update the domain model and store concluded explanations (diagnoses) with cases. Argument construction systems keep the best arguments for and against a position, an explanation justifying the decision, and relate how comparable the input case is to other cases in the library.

Extract. The source of any learned information must be determined first. Case descriptors and solution methods are obvious candidates. Explanation(s) justifying a solution's applicability to the problem may also be included. In CASEY, explanations were included in retained cases and subsequently used for modifying an initially retrieved solution. Failures indicated in the Revise task may also be extracted and retained, either as separate failure cases or within individual cases. When the system encounters a failure it can be reminded of a similar situation and use the previous failure to facilitate a solution for the present situation.

Index. Indexing amounts to deciding what types of indices to use for future retrieval and how to structure the search space of indexes; as such, it is really more a knowledge acquisition problem and should be considered as part of the initial domain knowledge analysis and modeling. Syntax-based methods use a trivial solution to indexing by incorporating all input features as indices.

CASEY used a two-step indexing method. General causal states in the heart failure model were used as primary indices. Features were propagated in the heart failure model, and the states that explained the features were used as additional indices in the case memory. The observed features themselves are used as secondary indices only.

Integrate. This is the final step of incorporating new case knowledge in the case memory. If extraction and indexing have failed to contribute any guidelines for including this knowledge, integration becomes the main step of the Retain task. Modifying the indexing of existing cases is one way to improve the CBR system's similarity-assessment capabilities. Indices for a particular case or solution are adjusted as dictated by the applied case's success or failure. Successful case retrieval features have their association with the case strengthened; for features of unsuccessful cases, this association is weakened.

Integration of new information can also occur within the general knowledge model by using machine learning methods or through direct interaction with the user. Therefore, a system may incrementally extend and refine its general knowledge model, as well as its memory of past cases. All general knowledge in PROTOS was acquired in such a bottom-up interaction with a competent user.

3.4 CBR & Engineering Design Problem Solving

> *"Now one of the [best] ways of designing something is not to design it at all. Use [a part] that is designed already by someone else."* [Glegg, 1969]

A common way of doing engineering design is to reuse an existing design. The design practice of modifying past designs is used within nearly every engineering field. Domain- and task-specific techniques determine how the modification of previous designs is done. It is in performing such modifications, not to mention finding previous designs, where CBR has relevance to engineering design problem solving.

CBR uses the specific knowledge of previous concrete design problems. A new problem is solved by finding a similar past case and reusing it as determined by the new situation. As new problems are solved, the solutions are stored as cases. The archival nature of such a CBR system allows users of the system to find previous cases, speeding up the entire design process.

> *"Each problem I solved became a pattern, that I used later to solve other problems."*
> Rene Descartes.

Engineering design problems are difficult because many acceptable solutions exist, even for those designs that are well defined. The challenge is to effectively limit the design alternatives, as there are often many possible solutions. Designers also have to wade through an overwhelming amount of information. This information typically consists of experimental data and collections of unorganized heuristics, neither of which is immediately conducive for use in design. By using CBR, this information can be organized into a design archive that is immediately useful for solving new problems.

Just how well CBR works for solving engineering design problems is of course dependent on many things, few of which are the approach itself. An adequate source of design cases must first exist before CBR can be used. These cases must also be organized and indexed so that the appropriate cases can be retrieved as needed. A means of modifying retrieved cases must also exist. Several approaches to engineering design problems that use CBR, among other techniques, are shown in the following chapter. Some systems discussed use CBR successfully; others do not. It will be seen that the successful uses of CBR carefully consider the case representation, organization of the case library, and modification techniques. One or more of these considerations will be seen to be lacking (or poorly done) in those systems that do not work well.

Part III

Various Composite Designers

CHAPTER 4

Germane Approaches to Composites Design

The interest in producing composites designs without the extensive use of reference materials and tedious trial-and-error tests has led to the development of several computer systems for aiding the design process. Many different approaches exist, varying from expert systems that attempt to do everything from the initial selection of materials to the final finishing of the completed part, to systems that deal with only specific portions of the design and production processes.

The myriad systems that have been developed for composites design also cover a wide range of domains, be it polymers or ceramics, and methodologies, from purely database retrieval to sophisticated artificial intelligence techniques.

Often, the scope of a design support system is determined by the domain or the personnel for which it is intended; less complicated domains lend themselves to a comprehensive approach, as do systems intended for novice users. At least that is generally the case with successful systems (e.g., the Composites Design Assistant - see page 93ff).

There are instances of systems which have failed because of an overzealous attempt at handling everything from initial specification to final manufacturing in a complex domain. However, whether these systems have failed, succeeded or are still under active development, there are lessons to learn from both the computational techniques utilized and the philosophies that drive composites design.

The following discussion examines specific systems covering only a small sampling of fielded approaches, focussing mainly on those that emphasize computational techniques with an AI emphasis, as those are the most pertinent to this dissertation. Other computational techniques that have been used for designing composites include pure database retrieval, detailed finite element analysis, and others. Most of the systems considered here also emphasize the design of the composite *material*.

A total of five computational approaches to the design of composites is discussed here. These systems are presented in roughly chronological order, spanning a period from the early-to-mid 80's up to today. In order of presentation, the systems considered are: the Composites Design Assistant (CDA) from Lockheed Missiles & Space, Composites Part Design (CPD) from Germany, the Decision Support System for Preforming (DSSPreform) from the University of Delaware, the Composites Designer (COMDES) from England, and Expert Assisted Design of Composite Structures (EADOCS) from the Netherlands.

To facilitate the comparison of these systems, a checklist of items will be examined for each. The discussion in each section will begin with a look at the historical context within which the system was developed, then look at the specific design problem addressed and the stated goals for the system. An overview of the underlying design philosophy behind the system follows.[1] After that, the discussion will turn toward the details of the implementation, specifically looking at the problem-solving techniques utilized, the ease with which the system could be modified (if at all), and its capability for explanation. To get a feel for how the system operated, a quick summary of its run-time behavior will be given. Finally, the discussion for each system will conclude with a brief synopsis of the pros and cons of what the system did and how it measured up to its intended function.

4.1 Composites Design Assistant

This system was one of the preliminary attempts at automating the design process for composite materials. The Composites Design Assistant (CDA) was developed in the early 80's by Zumsteg & Pecora at the Lockheed Missiles and Space Center as a part of the Expert Systems in Materials Selection and Design project. Examination of its structure and approach to composites design serves as a baseline for comparison with some of the later, more sophisticated, approaches to the design of composites.

The particular design problem that the CDA addressed was the design and analysis of facings on sandwich panels for aerospace structural components, some of which could be polymer composite laminates. The overall goal of the CDA was to demonstrate the use of an expert system for coordinating and facilitating access to external software [Zumsteg *et al.*, 1985]. This external software provided material property data and calculated mechanical properties of the laminates.

1. Several of the following systems utilize an approach to design which is referred to as "conceptual." There is no consistent use of that term. Each system's particular definition of conceptual design will be discussed within the context of the specific design problem it addresses.

By looking closely at the structure and behavior of the CDA, it can be seen that the instances in which the system actually was used for polymer composite design were limited. Recall, the specific application of the CDA was for sandwich panels, the cores of which were assumed to be metallic. Upon further inspection, it can be seen that the CDA did not perform any material design per se; instead, the focus was on the validation and analysis of the user's descriptions of these panels, the generation of which the CDA facilitated. The CDA was never intended to perform material design.

The design process for these panels was well defined, enabling the use of a backward-chaining rule-based expert system. This expert system used a PROLOG-based expert system shell. At the time this system was developed, such expert systems were quite popular for engineering design, hence the existence of the parent project for this work. The rule-based portion of the CDA served as an intelligent front-end to a commercial materials database and a laminate analysis program [Zumsteg *et al.*, 1985]. See Figure 14 for a schematic of the intended interactions between these components.

As the CDA was intended to facilitate interaction with external systems and not instruct users how to generate sandwich panels, its explanation capability was limited. In fact, the only way to explain decisions was to view the rules utilized as they were fired. User modification of the design equations within the CDA was not possible.

An example of a rule from the CDA shown in Figure 15 reads:

> "if there is a current design in progress, and the face sheet spacing (`h_calc`) and the facing thickness (`facing_thickness`) are known, then the total panel thickness is the sum of the face sheet spacing and the facing thickness." [Zumsteg *et al.*, 1985].

Comparing this translated version of the rule with the encoded version shows one the most typical difficulties with rule-based expert systems: nonintuitive verbosity. This

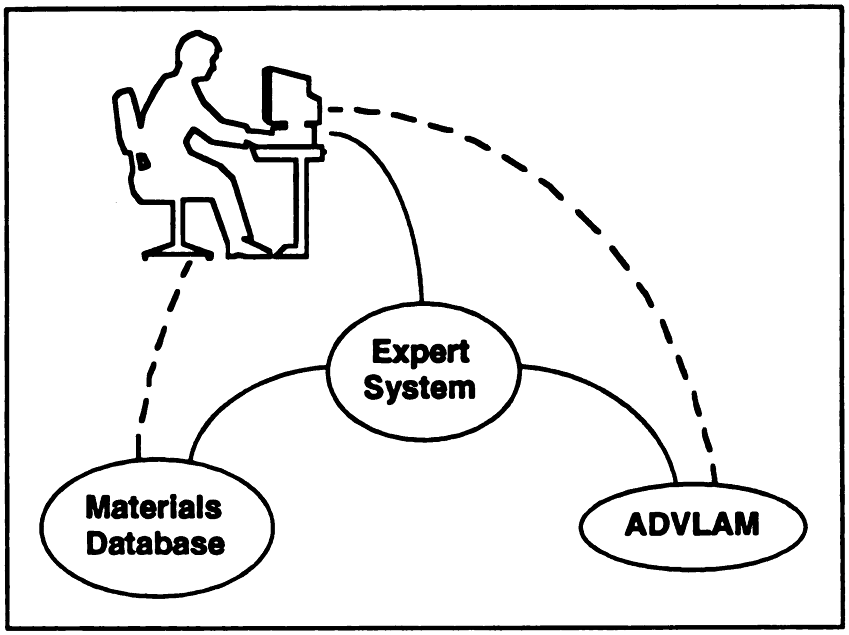

Figure 14. The architecture of the Composites Design Assistant.

```
rule 8:

if name_of_the current_design has_value
v(current_design) and h_calc of_the
v(current_design) has_value v(h) and
facing_thickness of_the v(current_design)
has_value v(thick) and
v(ans) is v(h) + v(thick)
then
total_panel_thickness of_the v(current_design)
has_value v(ans)
```

Figure 15. An example rule from the Composites Design Assistant.

simplistic rule can also be compare to rules from other rule-based systems, as mentioned in Chapter Two. The rule from the CDA shown here is essentially an arithmetic equation, whereas the rules for MYCIN, R1/XCON, and PROSPECTOR given in Chapter Two were more sophisticated rules for diagnosis, conditional connection assignation, and probability assessment, respectively.

A typical example of the operation of this system was as follows. The system presented the user with a sequence of menus from which to select appropriate choices (e.g. external loads, skin material, and so on) and also queried the user for a guess at the skin thickness of the panel. The rule-based portion of the system then calculated the panel thickness by accessing a materials database to obtain the relevant material properties. The user was then prompted for the choice of core material, after which the system checked the design for various safety and compatibility issues through the use of additional rules and by interfacing with a commercial laminate analysis program (ADVLAM). This program estimated the mechanical properties of the panel based on the specified laminate properties. If the generated sandwich panel did not meet the design requirements, the user could iterate the design for the panel until it met the given specifications.[1]

Figure 16 shows the results from a typical run of the CDA. It can be seen that the features one would typically associate with any kind of material design (e.g. `facing_material_name`, `core_material_name`) were simply assigned by the user. The exact meanings of the numbers generated by the various utilized rules are unclear. The one result of note, however, is that from rule15: `design_check = not_acceptable`.

1. Any item indicated as 'User input' in Figure 16 could be modified by the user in an attempt to find a panel that met the given specifications.

span_length	96.0	User input
beam_load	0.8333	User input
beam_type	simple_support_uniform_load	User input
deflection_limit	0.35556	User input
safety_factor	2.0	User input
facing_material_name	Woven_roving_PE	User input
facing_material_descr	*	User input
core_material_name	Kraft_paper	User input
core_material_descr	KP-3/8-60(25)	User input
total_panel_thickness	5.61349	rule8
facing_thickness	0.09	User input
core_thickness	5.43349	rule11
facing_safety_factor	19.6782	rule19
core_safety_factor	3.47993	rule24
deflection	0.385525	rule29
face_wrinkling_safety_factor	15.7685	rule32
design_check	not_acceptable	rule15

Figure 16. Sample output from the Composites Design Assistant.

Therefore, the user must iterate this design through the CDA again, perhaps choosing a different core material for the next run.

The benefit of the CDA was that it was able to generate a design for a sandwich panel without the user ever needing to access the materials database, run the laminate analysis package, know the relevant equations, or check for secondary effects. This was exactly what the system was intended to do. However, as alluded to above, the explanation capability was limited to the indication of which rules were fired, and as any further examination of the rules themselves (as in Figure 15) was not particularly illuminating, iteration of the failed designs was more hit and miss than thoughtful and guided.

The CDA was never very sophisticated in either its approach to the design of composites or its enabling technology, but it did what it was intended to do: assist the designer. It also demonstrated a proof-of-concept and served as a springboard for the development of additional systems.

4.2 Composite Part Design

This preliminary prototype for designing polymer composite parts was developed in 1994 by Michaeli & Biswas at the Institut für Kunststoffverarbeitung in Germany [Michaeli & Biswas, 1994]. Although the researchers never explicitly stated the specific design problem for which this system was intended, it appears to be the design of generic (or possibly automotive) polymer composite parts. The intent of the Composite Part Design (CPD) was to support an engineer in the first steps of the design of composite parts.

The CPD was to design the composite part at a conceptual phase of the design using a combination of rules and previous experience. Within the CPD the concept phase was defined as that portion of the design process in which "[t]he task is defined and all function structures are established. Also, the solution principles are fixed. Last, a technical and economical evaluation is carried out" [Michaeli & Biswas, 1994]. The knowledge available at this concept phase is very coarse in nature, as opposed to the more detailed knowledge which has been set through previous experience (i.e. from a previously designed part).

A traditional linear design process (i.e. one used for neat resins or metals) was not attempted within this system. Instead, the researchers conceded the inherent complexity of composites design and focused the CPD where it was most useful. See Figure 17 for the interaction of composites' design issues underlying the CPD. The material (i.e. fiber and resin) was determined in conjunction with a production process, after which the laminate structure was determined and the part geometry optimized.

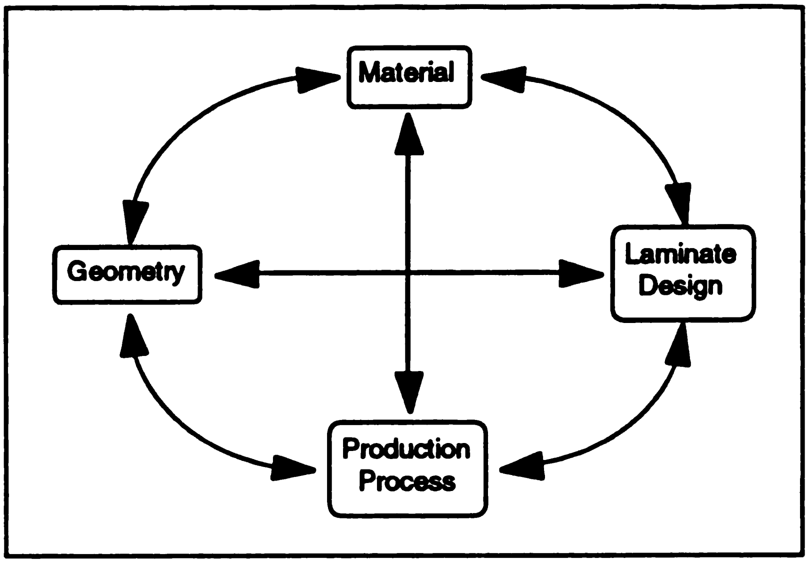

Figure 17. Interaction of design issues for composites.

The CPD was implemented by combining a classical rule-based expert system with a case-based reasoning system. The user was to input a list of general requirements to the rule-based portion which would then generate combinations of matrices, fibers and accompanying processing procedures. Optionally, an experienced user could bypass the rule-based portion and proceed directly to the design retrieval (CBR) portion.

The knowledge which comprised the rule-based portion of the CPD was embedded in simple rules and based upon generalized data about materials and processing methods. Modification of this rule base was possible through an interface provided by the commercial expert system shell within which the CPD was generated.

The output from the rule-based portion of the CPD included the material & processing combinations together with the rules and associated data utilized in their generation. In this way, the system could teach novice designers the design process. These results could then used as a seed (along with the input general requirements) for the retrieval of matching composite parts from the library of stored designs. See Figure 18 for both the architecture of the CPD and the knowledge flow through the system.

The case based portion of the CPD was implemented using a customized data base management system. The retrieval of matching cases was based on a similarity approach. This approach matched first against the material and production process combinations. If design cases which matched these exactly were found, additional matching was performed. This entailed three levels of increasing similarity. The lowest level included those data which were considered to be the most important (e.g. geometry, loads, dimension, etc.). The other two levels included less important requirements. Therefore, partial matches emphasized the more important design variables first.

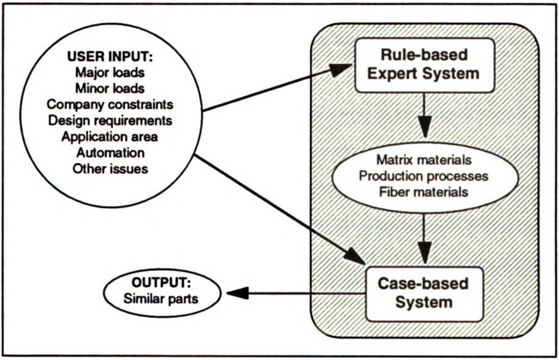

Figure 18. The architecture & input/output behavior of the CPD.

For any variable to be considered matching, regardless of the similarity level, the value had to match exactly. Explanation of the retrieved designs (other than documentation embedded within the designs) was limited to a listing of the variables used to determine the match. Modification of the existing designs and the addition of new designs was possible through a standard interface.

One of the shortcomings with the CPD system is the lack of a detailed implementation. It appears that this approach is relatively mature in the minds of the researchers, but there are very few detailed examples of anything to lend it credence. This lack of detail can be directly attributed to the limited development of the CPD.

On a theoretical level, there is no reason why this approach should not be able to generate useful results; however, there is nothing that leads one to believe that this implementation is as yet useful. The lack of details about the size of the rule-based system and the number of cases in the library precludes drawing any conclusions about the effectiveness of this approach.

The procedure by which the matching cases were to be retrieved appears to be indicative of a carefully generated case base. Such case bases tend to oversimplify the knowledge representation of the design problem, which can in turn lead to problems upon scaleup to a robust case base. In practice, much more flexibility to deal with close matches should be included. The researchers do admit, however, that "the competency of this system is limited at the moment" [Michaeli & Biswas, 1994].

4.3 DSSPreform

The DSSPreform (Decision Support System for Preforming) system was developed by Pitchumani and Karbhari at the University of Delaware in conjunction with Lanxide Corporation roughly during 1993 to 1995 [Pitchumani *et al.*, 1993; Pitchumani & Karbhari, 1994; Pitchumani *et al.*, 1994]. This system was developed to aid in the design and manufacture of porous preforms which are subsequently infiltrated with a matrix material to form ceramic- and metal-matrix composites. The goal was to provide a comprehensive tool for realizing rapid, efficient manufacturing that would be able to start with a material specification and finish with control of the manufacturing process, covering all points in between.

The design philosophy upon which DSSPreform was based distinguished between material selection, process selection, and the parameterization of the manufacturing process. A linearized design process beginning with the determination of the material selection, proceeding to the process selection and ending with process parameterization was developed and implemented utilizing what the researchers labelled a "Knowledge-based Decision Support System (KBDSS)." This KBDSS can be considered to be a mixture of a knowledge-based system and model-based reasoning. See Figure 19 for the entire design process from initial problem specification to the final finished part, with the KBDSS providing the bridge in between.

DSSPreform was comprised of three main portions: a Materials Expert, a Process Selection Expert, and a Process Design Expert. Each of these experts was to determine a portion of the design and then hand off the results to the next expert. See Figure 20 for the

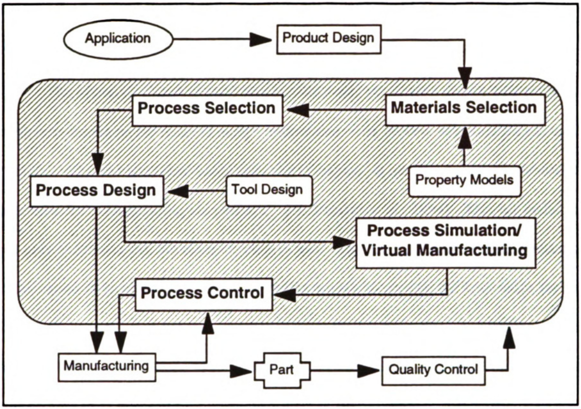

Figure 19. Comprehensive KBDSS design vision for manufacturing applications.

architecture of DSSPreform. The three experts comprising DSSPreform addressed only 60% of the total comprehensive design vision shown in Figure 19.

The overall process for generating the design and manufacturing details implemented within DSSPreform can be thought of as linear. This is interesting, as most composites designers admit to the inherent difficulty of strict linear design due to the many interactions between various portions of the design. However, through a judicious specification of design variables and some limitations imposed by the preform manufacture domain, DSSPreform accounted for these interactions. A caveat to this is that DSSPreform was never fully integrated; each of the three experts could be accessed separately, with the user providing the inputs which may have been provided by another expert and potentially overriding recommendations from other portions of the system if so desired [Pitchumani *et al.*, 1994].

The Materials Expert utilized simple if-then-else rules based on structure-property models to determine the material design information. The Process Selection Expert used "[s]imple rule- and procedure-based selection strategies...to obtain a list of recommended processes for manufacturing the preform" [Pitchumani *et al.*, 1994]. This list included not only exact matches, but close matches as well. The Process Design Expert utilized a combination of heuristics and simple empirical models to provide the knowledge for its rules. The model-based portion served to validate the heuristic recommendations and to handle situations not covered by the established heuristic knowledge. Each of the experts was comprised of teams of hierarchically organized specialist modules.

Figure 20 shows some of the details within each of the experts, with the caveat that the grayed-out portions were never fully implemented. Of particular interest is the group of specialist modules comprising the Material Selection Expert: Reinforcement Geometry, Matrix, Reinforcement & Reinforcement Volume Fraction. This effectively defines the composite material in terms of the matrix, the reinforcement, and the reinforcement's geometry and volume fraction.

Given the object-oriented environment (Gensym's G2) in which DSSPreform was implemented, the addition of more options (e.g. additional processes, materials, etc.) for any of the experts was possible by simply creating a new object of the proper class. Similarly, additional rules could be added to the experts. There was no apparent explanation capability within DSSPreform.

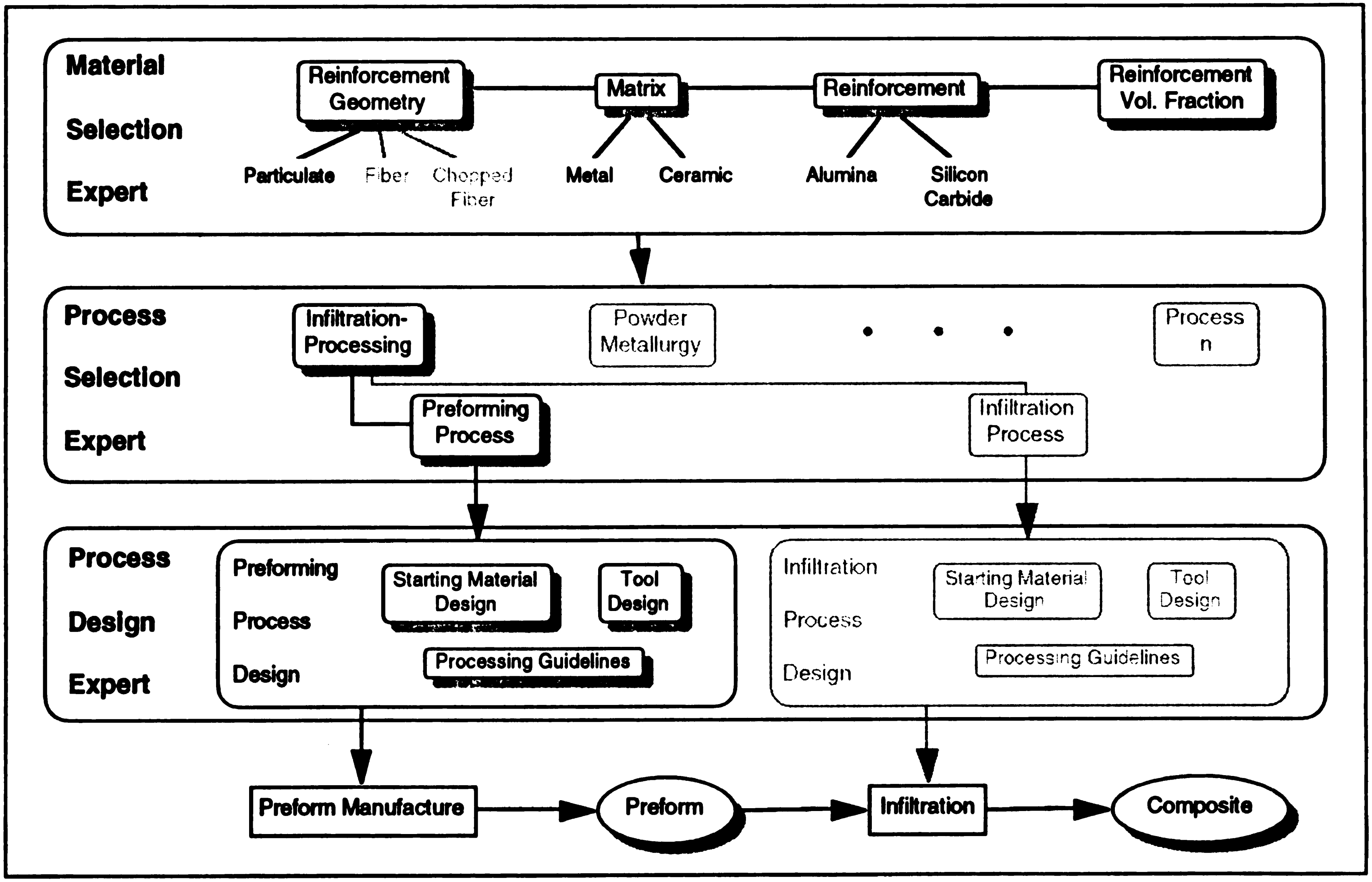

Figure 20. The architecture of DSSPreform.

DSSPreform was to initially determine the reinforcement and matrix materials, reinforcement geometry and reinforcement volume faction in the composite through the use of the Material Expert. The first three of these have choices listed in Figure 20, with the Material Selection Expert selecting one choice from each based on a combination of relevant material-specific variables (e.g. thermal conductivity, coefficient of thermal expansion, ductility, etc.). The volume fraction of the reinforcement was determined by combining the other material choices and an empirical model of composite physical properties. The output from this portion was to be routed to the Process Selection Expert, which was to use this information together with additional user inputs and specific process knowledge to suggest an appropriate processing method. After this, the Process Design Expert was to take the suggested processing method and determined detailed parameters for the starting material design, the mold design and the processing plan.

There are some important lessons to learn from this system and its development, both positive and negative. On the positive side, the approach to the design of the entire process from material selection to production was quite detailed and the interactions between the varying portions of the design were enumerated explicitly, something that has proven to be difficult for other researchers to handle. While the framework for DSSPreform was quite detailed, the implementation was never much more than a prototype. Despite having promised the detailed capabilities of the comprehensive KBDSS approach, DSSPreform never fully achieved maturity.

The Materials Expert was never able to handle anything more than particulate reinforcements, and while the Process Selection Expert could recommend 22 different processing methods, the Process Design Expert was never able to handle more than two of these methods (the second of which was added just prior to the termination of this project).

A direct cause of this sparse implementation was the combination of a corporate sponsor (Lanxide) who was providing proprietary information for inclusion into the system[1], and an attempt to make the system general enough to deal with processing techniques not utilized by Lanxide. This conflict of interests contributed substantially to the demise of DSSPreform.

The stated goal of providing a comprehensive tool for realizing rapid and efficient manufacturing was never fully met, as the system was never completely implemented. The portions of DSSPreform which were implemented, most notably the Materials Expert, were quite detailed and did indeed provide a portion of the intended comprehensive tool.

4.4 COMDES

The Composites Designer (COMDES) system was developed during 1991-1994 by Edwards and Ige at the Cambridge Engineering Design Centre with support from Rolls-Royce [Edwards *et al.*, 1994]. COMDES was intended to assist in designing polymer matrix composite materials, specifically addressing the optimization of laminate stacking sequences. The primary objectives of this planned assistance were to improve the quality of decision making in, and to reduce the time required for, the design process.

The vision of engineering design behind the development of COMDES can be presented as an idealized design process comprised of three main portions: conceptual, embodiment, and detail. See Figure 21 for this three-phase design process. As defined

1. This industrial sponsorship with Lanxide led to the most detailed portion of the system: process design for low temperature molding.

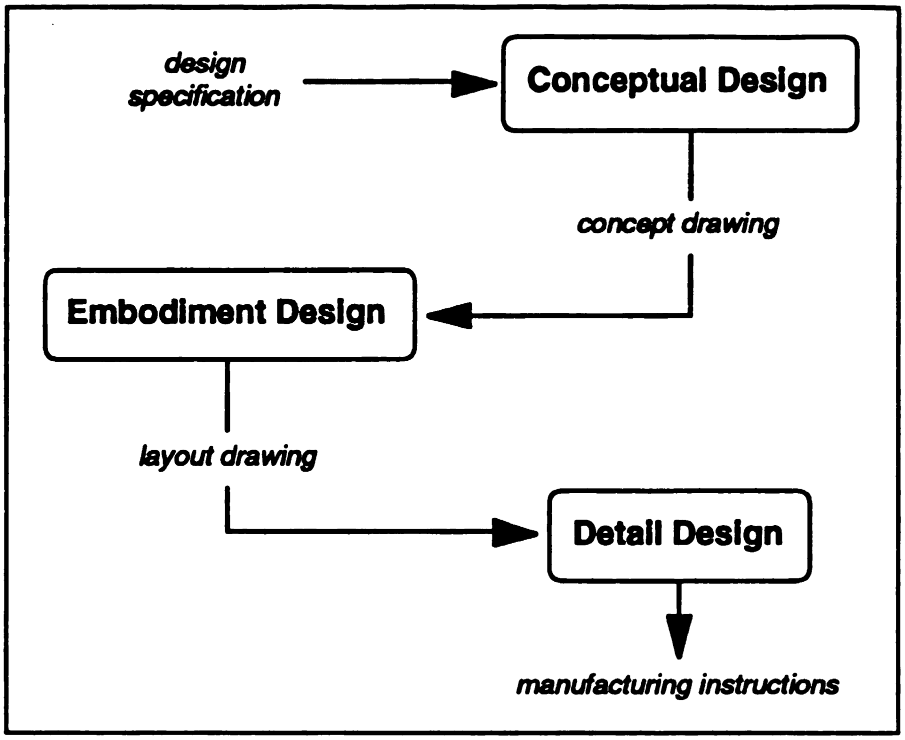

Figure 21. The idealized three-phase design process.

within COMDES, conceptual design deals with the generation of concepts likely to fulfill the design specifications, resulting in a concept drawing. In embodiment design, foundations are laid for a detailed design through a structural development of the concept. The output from this phase is a layout drawing. Finally, detail design defines the precise geometry and materials and generates manufacturing instructions. COMDES addresses design at both the conceptual and embodiment levels.

The approach to the design of polymer composites taken by this system is intriguing, as it uses a task-level model to describe the philosophy behind the computational implementation. Here, tasks are used as a description of a solution to a technical problem (e.g. generate candidate materials, generate candidate geometry, generate candidate processing route). As used in COMDES, tasks should not be equated with Generic Tasks.

The approach of breaking the overall design process into discrete portions or tasks provides guidance on how to proceed with an implementation of the design process. As a result of this task analysis, a task-level model of the engineering design process has been generated. This model can be thought of as an ordered listing of the engineering problems which must be solved in the design process and the requisite interactions between them, the ordering of which has been optimized to minimize feedback between the individual problems.

The task-level model of the design process can be seen in Figure 22. Shaded nodes in this figure are dependent upon feedback, necessitating an iterative approach for solution. For example, the node "identify suitable fiber orientation & proportion" has feedback contributions from the nodes "compute elastic properties of laminate," "evaluate laminate vs. structural reqs," and "evaluate laminate vs. strength criteria."

COMDES used a pseudo-blackboard system for an overall control mechanism, within which operated systems for performing specific tasks from the engineering design process model mentioned above. These systems included: MatSel, which selected the most suitable composite materials; LamCon, which selected the laminate layup, optimized the stacking sequence of the laminate and evaluated the elastic properties of the laminate; StructAn, which analyzed the laminate configurations for stress and strain as well as appropriate failure criteria; and Critic, which strategically managed the iterations between these systems. The user interacted directly with COMDES as it ran, controlled the information flow between each of these systems, and accepted or rejected advice given by each system. Due to this user interaction, COMDES can not be considered to be a true blackboard system (hence, the pseudo- prefix). A true blackboard system must control

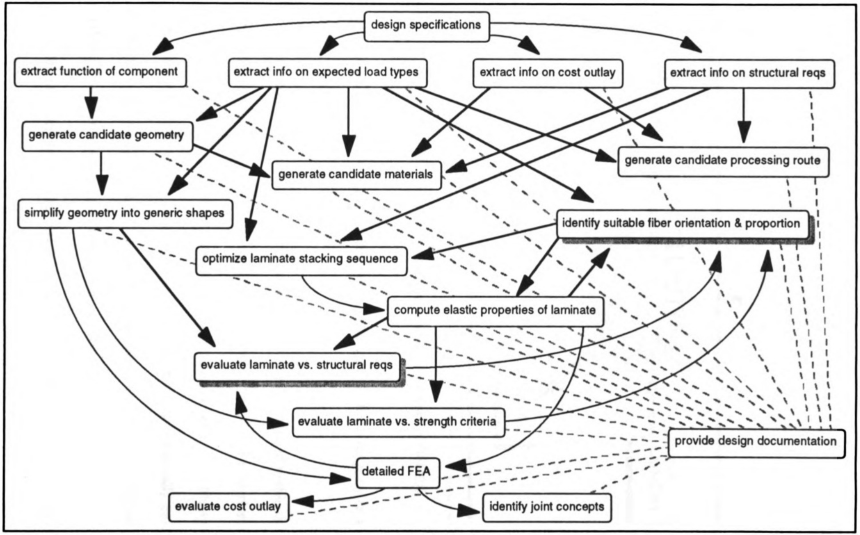

Figure 22. Design tasks & interactions for designing composite materials in COMDES.

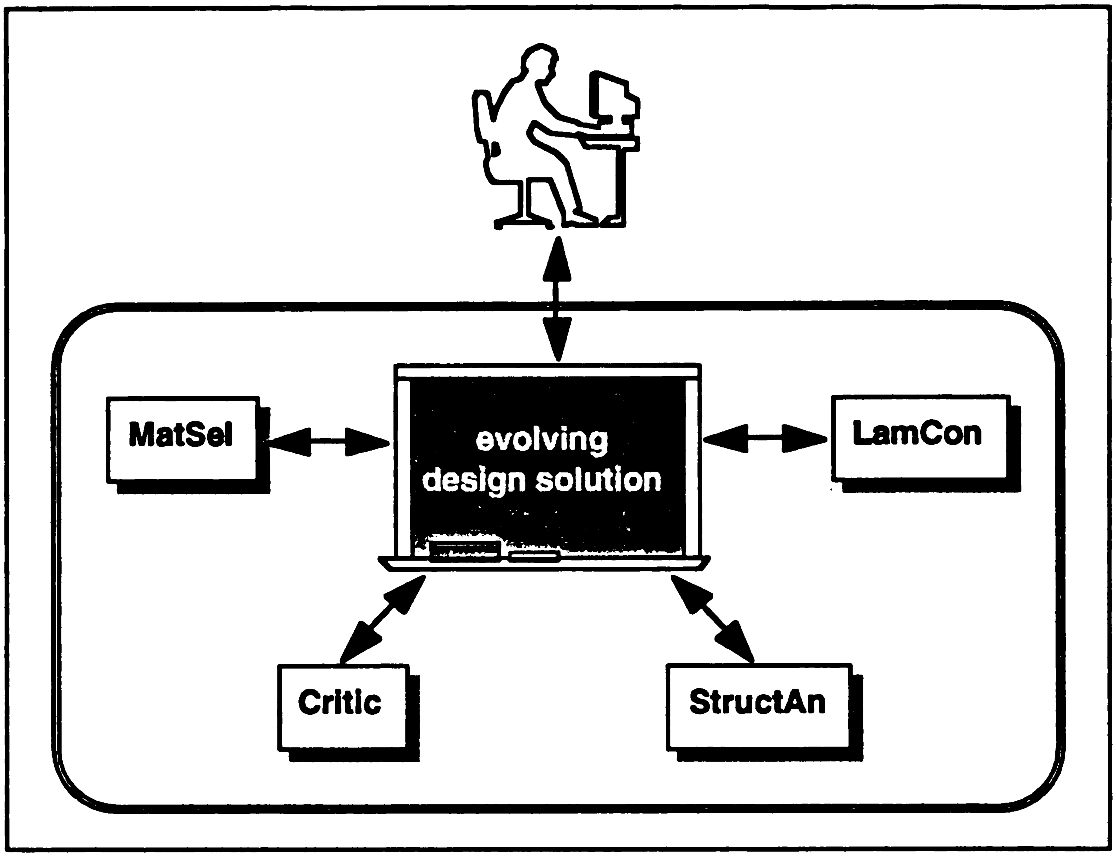

Figure 23. Blackboard system architecture for COMDES.

itself, and COMDES was not autonomous. See Figure 23 for a diagram of the overall system.

No specific mention of the problem solving techniques utilized within the subordinate systems was made, but based upon the design tasks they were to perform one can deduce that MatSel & Critic were classical rule-based expert system. As the system generated new composite designs in consultation with the user, it automatically modified and added to its knowledge base. There were no apparent explanation capabilities within COMDES aside from following a trace of its execution.

By stepping through the task-level model of the design process for composites as shown in Figure 22, a feel for the operation of COMDES can be gained. However, not all of these tasks were included within the system. A design specification was the starting point. From this, information was extracted about the function of the component, expected load types, cost outlay and structural requirements. Candidate geometries for the composite were then generated from the function and load types. MatSel then determined the most suitable composite materials from the candidate geometries, structural requirements, cost outlay and expected load types. After which, the candidate processing plan was determined from the candidate geometries, structural requirements and cost outlay. The geometry was then simplified into generic shapes through the use of the expected load types. LamCon then selected the fiber orientation, optimized the laminate layup and computed the elastic properties of the laminate. StructAn evaluated the laminate layup against the structural requirements and strength criteria. COMDES then iterated (under the control of the user) between LamCon and StructAn until an appropriate laminate layup was determined. A detailed FEA was performed with the generic shapes and laminate elastic properties. Joining concepts were identified, the cost outlay evaluated and the final design documentation was generated.

The underlying premise of COMDES is that a rigid phase level design process for composites is a ridiculous endeavor. This assertion arises from recognizing the impossibility of solving the design problem at the correct level of detail in an strictly imposed linear design process. Rigid phase-level design, which includes distinct conceptual, embodiment, and detail design phases, does not allow for the complex interaction of design issues so prevelant in composites design (e.g., material choice impacts the processing method, which impacts structural geometry, which in turn impacts the material choice).

COMDES, while cast in the flavor of the phase level design process, utilizes a task level design process. As implemented in COMDES, various design tasks cross the boundary between conceptual and embodiment design. So, while the phase level design process is a convenient way to think about the design process, COMDES shows that the details of the design can occur across the boundaries set by phase level design.

COMDES has been successfully tested on several simple case studies, and there are plans to extend its capabilities [Edwards *et al.*, 1994].

4.5 EADOCS

This system is still under development and in its prototype phase; consequently it is as yet incomplete. However, there are some interesting features which warrant consideration here, even though not all of the details are yet determined.

EADOCS, which stands for Expert Assisted Design Of Composite Structures, is an expert system for the conceptual design of thin laminated composite panels in aircraft structures. Netten & Vingerhoeds at the Delft University of Technology in the Netherlands have been developing EADOCS since 1993 [Netten *et al.*, 1995; Netten & Vingerhoeds, 1996a; Netten & Vingerhoeds, 1996b]. It is an offshoot of Netten's M.S. work which dealt with the discrete numerical optimization of composite aircraft structures [Netten, 1996]. The main thrust of EADOCS is to generate solutions at a conceptual level which are useful as a starting point for more detailed numerical analysis and optimization.

As stated above, EADOCS designs composite panels at a conceptual level. As defined within EADOCS, this conceptual level of the design process involves the fluid navigation of a dynamic mix of potentially contributing factors, ranging from qualitative objectives to explicit quantitative limitations (e.g., minimize weight, minimize cost, panel must be at least one inch thick, and so on).

Design specifications and objectives are used to determine a "good" concept for further preliminary design and optimization. The system handles the description of the composite panels by specifying materials, fiber orientation, numbers of plies, panel type and components for stiffening from a limited catalog of choices. See Figure 24 for some possible design elements for representing panel concepts [Netten *et al.*, 1995].

The panel designs within EADOCS are declared at three levels: prototype design, initial conceptual design, and conceptual design. Prototype design involves simply specifying what types of components and design elements are to be incorporated in a final design. Initial conceptual design quantifies the prototype by assigning initial values to the attributes. Conceptual design modifies this initial concept to the point where it is a good starting point for further design optimization.

Within the context of EADOCS, therefore, conceptual design involves three steps: prototype selection, concept selection, and concept adaptation. Figure 25 presents a logical progression of the conceptual design process as embodied within EADOCS.

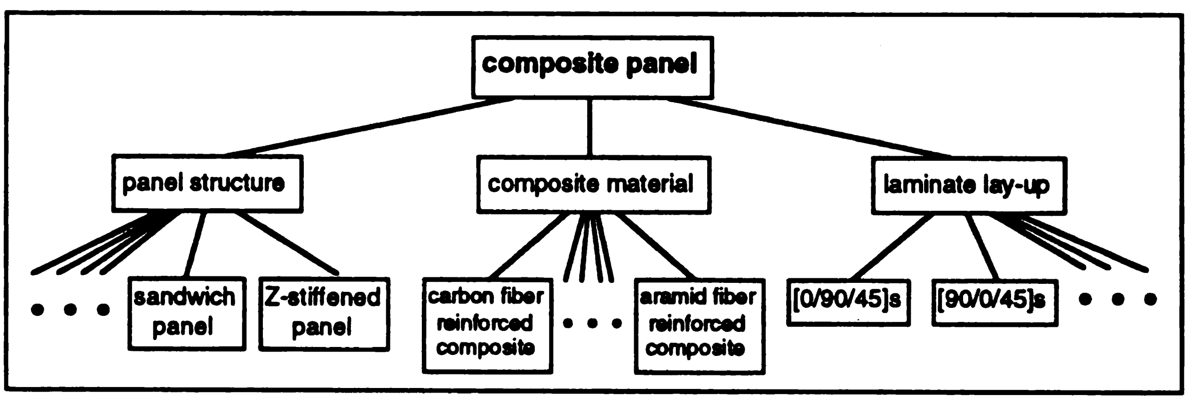

Figure 24. Design elements for composite panel prototypes in EADOCS.

The intent of EADOCS is to utilize the computational technique most suited to a particular portion of the conceptual design process. How well a particular technique is suited to a design task is determined explicitly by the reasoning process the design task is to perform and the representation of the available knowledge. Prototype selection utilizes constraint-based reasoning to limit the design space. Case-based reasoning is then used to determine initial concepts. The adaptation of the resulting concepts is performed through the use of expert-governed numerical optimization (EGNOP). These three computational approaches are embedded within a meta-level rule-based system that controls the overall design process, data flow and user interaction [Netten *et al.*, 1995]. EADOCS is not autonomous, and requires substantial user interaction.

Prototype selection qualitatively evaluates the design specifications and subsolutions and then combines the subsolutions to generate a prototypical panel design. As seen in Figure 25, this can be interpreted as a propagation of the design specifications to limit the allowable design space to an area where design optima can be expected. This qualitative evaluation utilizes the achieved functionalities of the subsolutions and the interactions between these subsolutions. The selection is performed by mapping the achievable behavior of the subsolutions onto the specified functionality the panel must meet and is driven by an optimality criteria (i.e. get as close to the intended functionality as possible).

The prototype is then used (in conjunction with the design specifications) to retrieve matching cases. These cases are comprised of the attributes of both complete and incomplete designs and their respective functionalities. Again, by consulting Figure 25, this can be logically represented as the determination of the initial concept(s) by finding the region of optimal practical designs. Those cases which match the prototype most

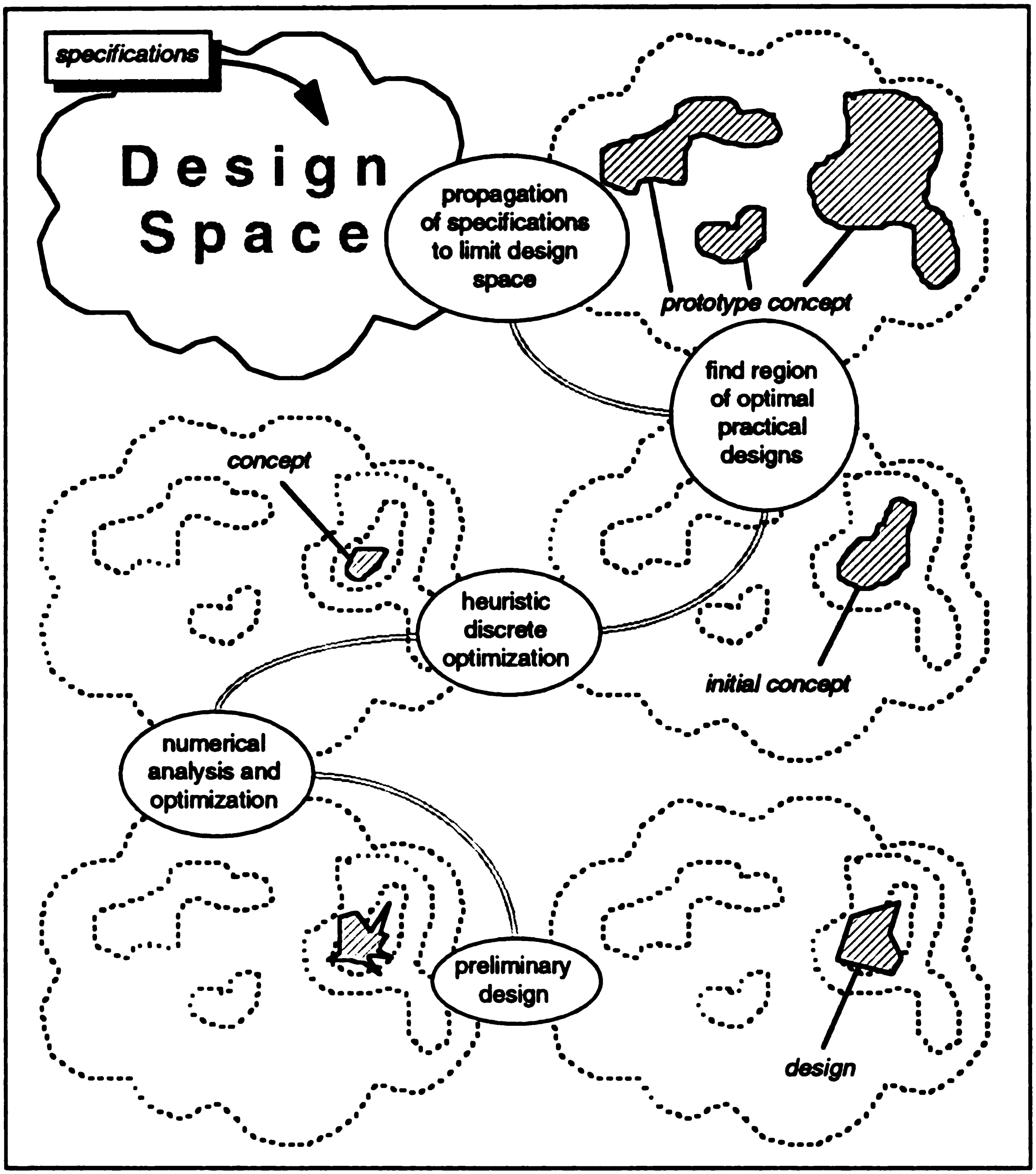

Figure 25. Logical progression of the conceptual design strategy in EADOCS.

closely, even if the cases are incomplete, are then adapted and combined to form the final conceptual panel design. The adaptation involves first replacing the subsolutions in the prototype with the corresponding design objects (non-optimal portions of the overall panel design).

The evolving design concept is further transformed through the use of heuristic discrete optimization and standard numerical optimization techniques. The target for this optimization is the specified behavior which the panel must meet. Finally, preliminary design stage generates a design from the optimized panel concept.

An unanswered question about this system and the utilized approach include is whether or not it will actually work. Due to its immaturity, this system has yet to be tested for validity (not to mention efficiency). As of this writing, the case library has yet to be completely filled [Netten, 1996]. Any evaluation of the capabilities of the system must wait until enough detail exists to permit validation studies. This work is ongoing, with plans to significantly increase the overall depth and rigor of the implementation by the end of Summer 1997 [Netten, 1996].

It is not clear whether or not the AI techniques used in this approach are appropriate for the design problems considered. This uncertainty is due in part to the lack of any explicit system knowledge (e.g system run traces, utilized cases) in the literature, which is in turn due to the incompleteness of the current development. Even if this system never reaches maturity, general lessons will have been learned by the fact that it was attempted. The manner in which the adaptation of the conceptual designs is approached is intriguing. There are two reasons for this: the detailed listing of design possibilities, and the mixed levels of complexity that exist among these possibilities. It does appear, however, that what is already a difficult design problem has been made even more difficult by this design methodology.

4.6 Capabilities & Implications of These Systems

Throughout the preceding discussion, various capabilities of the systems considered have become apparent. When taken with the computational technique(s) used, the presence (or lack) of these capabilities has served as a road map for successful composites design implementations.

To uniformly address these capabilities and define the road map for successful approaches to composites design, the discussion that follows refers to the comparison framework presented at the beginning of this chapter. The capabilities necessary for successful approaches to designing composites are identified by comparing specific features from that framework. A discussion of these capabilities in light of other general comments concludes this chapter.

4.6.1 Comparison Framework Analysis

Revisiting the comparison framework presented earlier helps to bring trends among the considered systems to light. See Table 1 for a side-by-side comparison of all five systems within this framework. The discussion here considers three aspects summarized in Table 1: the specific design problem addressed (the domain), the design philosophy, and problem solving techniques used. Often, a distinction between the design domain and the design philosophy cannot necessarily be made.

Table 1: A detailed comparison of previous computational approaches to composites design.

System	Domain	Design Philosophy	PS Technique	Explanation	Modification
Composites Design Assistant	polymer composite laminate facings on aerospace sandwich panels	assist in gathering info and presenting in the proper format to materials database & laminate analysis systems	PROLOG backward-chaining rule-based expert system as front end to commercial systems.	examination of specific rules as fired	design equations can't be changed at all
Composite Part Design	polymer composite parts (specific application is unclear)	determine list of possible materials & production processes at a concept phase, then use to find similar parts	rule-based expert system & a CBR tool	examine partial answers & rules used to determine	add new rules or add new cases to library
DSSPreform	ceramic and metal matrix composites	select materials, processes & design the processing details	Knowledge Based Decision Support System; heuristics & model-based reasoning; implemented using G2	none apparent	modify KBDSS knowledge base or models
COMDES	laminated polymer composite structures	conceptual & embodiment design for material selection, laminate design, structural analysis & design criticism	blackboard system (Edinburgh Prolog Blackboard Shell) w/ dedicated expert systems, optimization	none apparent other than following a system trace	no means for modification stated explicitly
EADOCS	thin laminated composite panels for aircraft	conceptual design; laminate optimization; amalgamation of design portions; adaptation	constraint-based reasoning, CBR with adaptation, 'expert governed' numerical optimization	none apparent	add new cases to library

Design Domain. The specific design problem addressed in four of the five systems was that of polymer composite materials. In three of the systems (CDA, COMDES, and EADOCS) the main emphasis of the entire design process was on generating appropriate laminate layup schemes. This emphasis is disconcerting, especially considering that only one of these systems dealt with both detailed geometrical information and processing technology determination (COMDES). Specifying a laminate layup without taking into account changes in the structure caused by processing has catastrophic consequences if the layup has been optimized for particular loads.

Design Philosophies. A total of five separate emphases within the design philosophy can be seen in Table 1; these have been extracted and are presented in a more visible format in Table 2. The checked cells indicate systems which are capable of determining the corresponding facet of the design. Note that in this table a system is only credited with covering a specific portion if it is determined without direct user intervention.

Table 2: Delineated design philosophies for the systems considered.

	CDA	CPD	DSSPreform	COMDES	EADOCS
architecture determination	✗		✗	✗	✗
material selection		✗	✗	✗	✗
process selection		✗	✗	✗	✗
similar part comparison		✗			✗
analysis				✗	✗
Relative Design Scope	0.2	0.6	0.6	0.8	1

For example, the CDA only entails architecture determination within its design vision. This is because in the CDA the material is chosen by the user; by the guideline above material selection cannot be included within its design scope.

Notice that the systems on the right side of the table have more expansive design scopes, i.e., a great deal more is included within the overall design task. EADOCS and COMDES both have wide-ranging design scopes, as the design philosophies included within each cover nearly the entire realm of possibilities.

There is potential for some confusion here, especially among design tasks, scopes, and philosophies. Admittedly, the distinctions between these three terms are not clear cut. However, there is utility for the use of each. Design philosophies can be thought of as abstract, large-grain design tasks. The design scope is simply a reference to how many different design philosophies are included in the overall approach taken by a system: the CDA has a small design scope as it only considers architecture determination, whereas EADOCS has a relatively large design scope as it considers all five design philosophies.

A trend present within the design philosophies of three of these systems is design at a conceptual level of detail (CPD, COMDES, and EADOCS). The main driver for performing design at this level is that this is the point where the maximum leverage for the work exists: investing the effort pays off tremendously. This is a very logical trend, and one which should continue if these systems are to continue to have an impact. As there is a dearth of generic computational tools for designing at this level, systems like these help to expand the design community's capabilities.

The discrepancies which exist between what each of the systems considers to be conceptual design can be traced back to knowledge representation and the specific design tasks addressed. What a system attempted to design "conceptually," be it a laminate layup or a material specification, dictated the level of abstraction at which it could be designed. Therefore the definition of conceptual design must be context-dependent. Indeed, EADOCS admits to this when the researchers discuss how the conceptual design process changes depending upon the target design specifications [Netten, *et al.*, 1995]:

> "... in designing a panel [for] high strength and stiffness [at] a minimum weight, [the] focus will be on the panel structure. [When] designing a panel [for] aerodynamic shape, smooth surface and low production cost, [the] focus will be on the production process, materials and laminate lay-ups, leading to a... different design procedure."

Computational Techniques. Another portion of the comparison framework for consideration is the utilization of combined problem-solving techniques. Again, the relevant information presented in Table 1 has been extracted and is presented in Table 3. This table shows the different computational techniques and combinations of these techniques utilized within the systems considered.

Table 3: Computational techniques for the systems considered.

	CDA	CPD	DSSPreform	COMDES	EADOCS
numerical optimization				✗	✗
rule-based reasoning	✗	✗		✗	✗
model-based reasoning			✗		
constraint-based reasoning					✗
case-based reasoning		✗			✗
knowledge-based reasoning			✗	✗	
Relative computational complexity	0.17	0.33	0.33	0.50	0.67

The CDA used a rule-based system in conjunction with a materials database and a laminate analysis system. The CPD system used a rule-based system together with a case-based reasoning system. COMDES used a pseudo-blackboard system to monitor various expert systems. Only one system (the CDA) used a single computational technique to accomplish the composites design. The other five systems used combinations of various techniques.

The point born out by this common use of multiple computational techniques was driven home in the discussion of the COMDES system: use the appropriate technique for the appropriate problem. The task analysis of composites design done for COMDES delineated explicitly how it is indeed possible, and in fact desired, to have multiple problem solving techniques interacting to solve the overall design problem. This is an important trend which should be carefully considered in any further implementations for composites design.

4.6.2 Commentary

While alluded to in the previous comparison framework discussion, the issues of scope, focus and degrees of abstraction bear further consideration. Although these issues were not present within the comparison framework, they have a definite impact on any composites design implementation.

Finding a balance between spanning a sufficient design scope and focussing in on particular design aspects is not simple. Some items to keep in mind for achieving this balance are presented in the following discussion. Related to the issues of scope & focus is the level of abstraction at which the design is performed. How designing abstractly influences the resulting capabilities of an associated approach is also discussed.

Design Scope vs. Focus. The issue of scope is perhaps the most important to highlight. Only by focussing on a specific portion of the complete composites design process can complex computational techniques be used effectively. Scope creep contributes all too readily to an increasing sloppiness in the interaction of computational techniques. This must be avoided.

Graphing the relative design scopes and the relative computational complexities for all of the systems considered (Figure 26), shows that a corresponding increase in computational complexity accompanies an increased design scope. EADOCS attempted to design every portion of the composite, albeit at a conceptual level, and as a result needed to use the largest number of computational techniques. This is not entirely unexpected, as more difficult problems intuitively should require more complicated solution approaches.

A warning against scope creep arises from the trend toward increased computational complexity with an increased design scope. An increase in the scope of the design problem without a corresponding increase in the computational complexity will most likely result in a system which functions poorly, if at all. Definite limits on the design problem(s) to be solved must be set before any implementation. This will help to limit the resulting computational complexity of the resulting approach. This, in turn, will reduce the number of potential interacting problem solving approaches included within the system.

Degrees of Abstraction and Capabilities for Design. The approach taken by the three systems (CPD, COMDES, and EADOCS) of designing at a conceptual level of detail is an evocative twist on handling the design process. However, by employing such an abstraction in the design process, the possible avenues for design are restricted. At whatever level

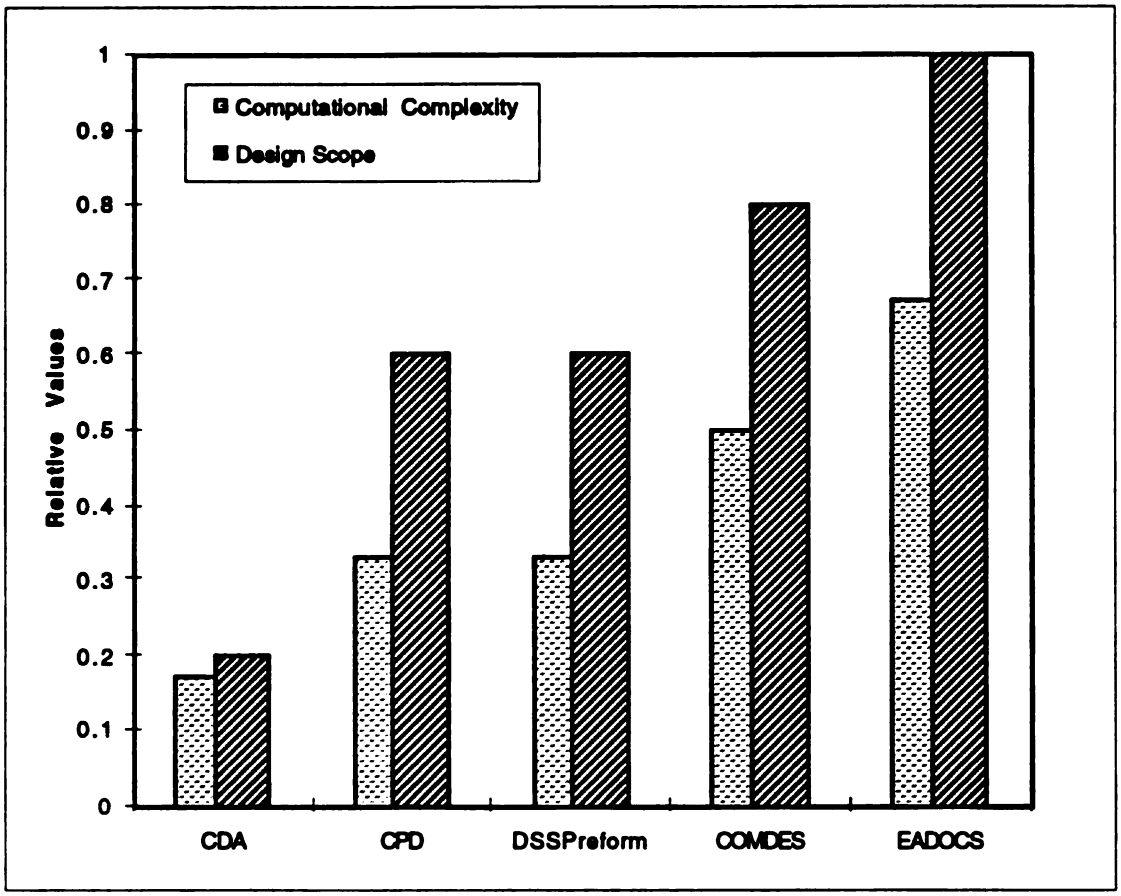

Figure 26. A comparison of relative computational complexities & design scopes.

of detail the design process is to be performed, the driving force must be the design task at hand.

It may very well be that certain design tasks (e.g. ordering laminate layouts) can not possibly be represented in a conceptual manner. In such instances, those design tasks should be performed at a more detailed level. The degree of abstraction must be driven by the design task.

Final Thoughts. The observations made above show that there is still work left to be done in this field. While these observations are drawn from the examination of judiciously selected systems, the systems are characteristic of the various facets of the composites design research community.

While the consideration of the systems here has uncovered several issues to keep in mind, these issues are hardly a complete prescription for designing composites. Other issues are important as well. An important additional item is knowledge representation. Only when an approach to designing composites has appropriate representation and use of design knowledge (beyond the issues mentioned previously) will it be successful. The following chapter addresses uniform means of knowledge acquisition, representation and use.

CHAPTER 5

The Evolving Generic Task Approach to Composites Design

Over the last several years, research in the ISL has focused on the application of knowledge-based systems and other artificial intelligence techniques to designing, processing, and controlling the manufacturing of polymer composite materials. Of specific interest is the research in designing polymer composite materials. Three material design problem solvers have been implemented during this time: Epox-I, Thermos-I & CMatDesign.

The problem solving theories of task specific architectures and generic tasks have guided this effort in intelligent decision support. This chapter presents the development and evolution of both the ideology and methodology behind the work on this project, and discusses how the use of these problem solving theories has facilitated this research.

The three material designers are compared by examining both the dynamic global vision of polymer composites design and the focused approach for material design. Notable features of each system are elucidated through the use of a benchmark design example. Finally, the salient lessons which were learned during the development process are discussed. Portions of the discussion which follows have appeared in [Lenz *et al.*, 1995a] and [Lenz *et al.*, 1996b].

5.1 Epox-I

The initial goal of the polymer composite design research was to produce composite material fabrication plans on the basis of prior experience. Epox-I was developed with this goal in mind. This system focused on only a portion of the possible composite materials domain, specifically on materials with a thin-film epoxy resin matrix and continuous, uni-axial fiber reinforcement. It was assumed that the fabrication process for the composite used hand-layup and autoclave curing [Kamel & Sticklen, 1990; Sticklen *et al.*, 1992b].

5.1.1 Global View of Composites Design

Fabricating a new composite material follows a certain chronology [Sticklen *et al.*, 1992a]. First, the designer chooses the desired macroscopic properties of the completed composite by considering the finished part's geometrical properties (such as thickness), its intended use, cost constraints, and so on. This initial step sets the desired properties of the final composite, such as tensile modulus and corrosion resistance. Using these desired properties as a benchmark, the designer proposes an initial production plan, including a list of initial ingredients and a preliminary plan stating how to process these materials. Samples of the material are produced and tested, and the procedure iterates until the design meets the desired properties. This architecture is shown in Figure 27.

Prototype systems were to capture the experience-based static design of fabrication plans (implemented in Epox-I) and process-control knowledge for the manufacture of composites [Sticklen *et al.*, 1992b].

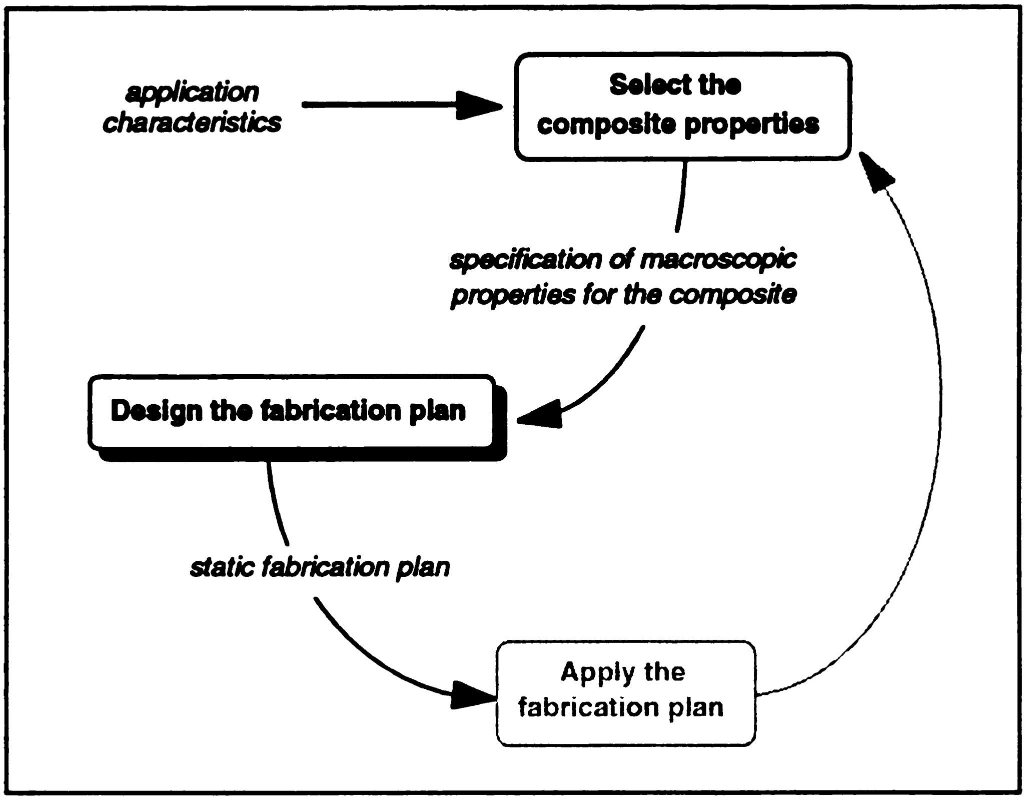

Figure 27. Overall design vision guiding the development of Epox-I.

A Routine Design approach was used for the compiled-level part of the problem-solving architecture to produce a static design, which in turn was a part of the complete architecture mentioned above. Routine Design, a generic task, is a typical experience-based, compiled-level knowledge-based problem-solving technique. Like other compiled problem-solving techniques, Routine Design by itself will fail in the face of a novel situation. In order to handle this potential failure, the complete architecture for static fabrication plan design (see Figure 28) included a model-based reasoning component called the functional reasoner, which was to have a single task: checking the applicability of compiled-level decisions [Sticklen *et al.*, 1992a]. As the routine designer systematically made design decisions, the functional reasoner was to use model-level understanding about the

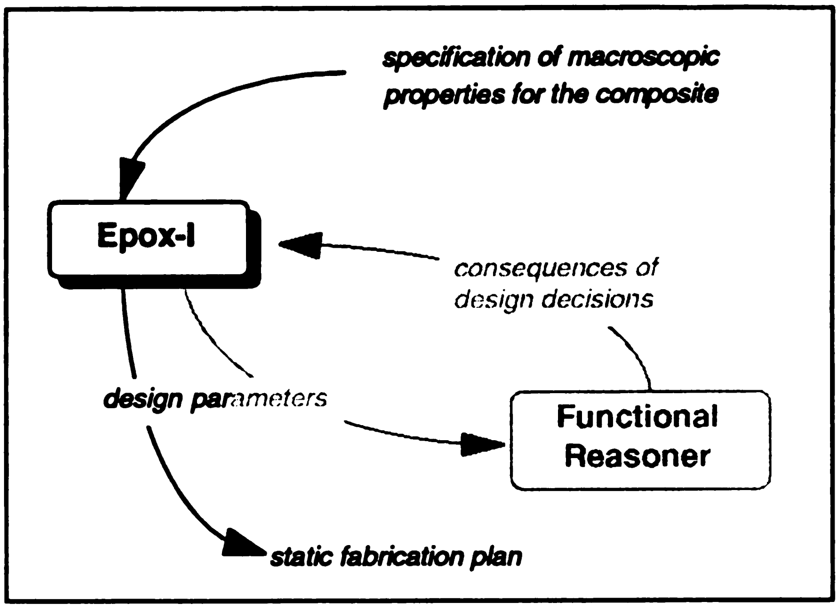

Figure 28. Architecture for determining static fabrication plans using Epox-I.

composite material domain to check the impact of each decision for the particular design problem.

The proposed interaction between these two modules included an information exchange. The routine designer would set design parameters and pass them to the functional reasoner, which would then determine the consequences of that design decision. The system could use these consequences to check the evolving design or to help the routine designer make the next parameter-determining decision [Sticklen *et al.*, 1992b].

5.1.2 Material Design

The initial plan for the production of the composite included both an ingredient list for all initially present materials and a preliminary protocol stipulating how to process the initial mixture. The DSPL problem solver developed for this small testbed problem depended on compiled knowledge [Kamel & Sticklen, 1990; Sticklen *et al.*, 1992b].

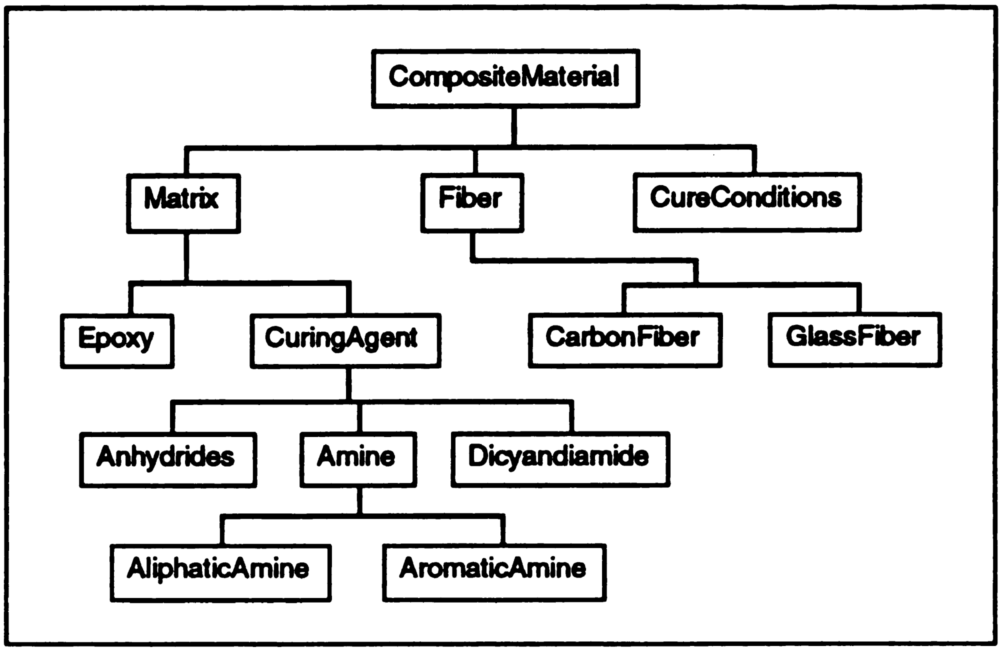

Figure 29. Specialist hierarchy for Epox-I.

Figure 29 shows the routine design decomposition for designing a thin-film, epoxy-resin composite with fiber reinforcement. This decomposition of the fabrication plan includes the determination of both materials and cure conditions [Sticklen *et al.*, 1992a].

The representation of the generated fabrication plans can be seen in Figure 30, together with the variables utilized in the decision process. Only three input variables were required for Epox-I to make its decisions: ApplicationType, Glass Transition Temp, and Tensile Modulus. The output from Epox-I gave the best possible fabrication plan in terms of seven material & processing specifications.

A sample problem was solved using Epox-I [Kamel & Sticklen, 1990]. The problem was to design a composite material for an outdoor structure. The specified inputs to the problem solver required that the material was to have a glass transition temperature of

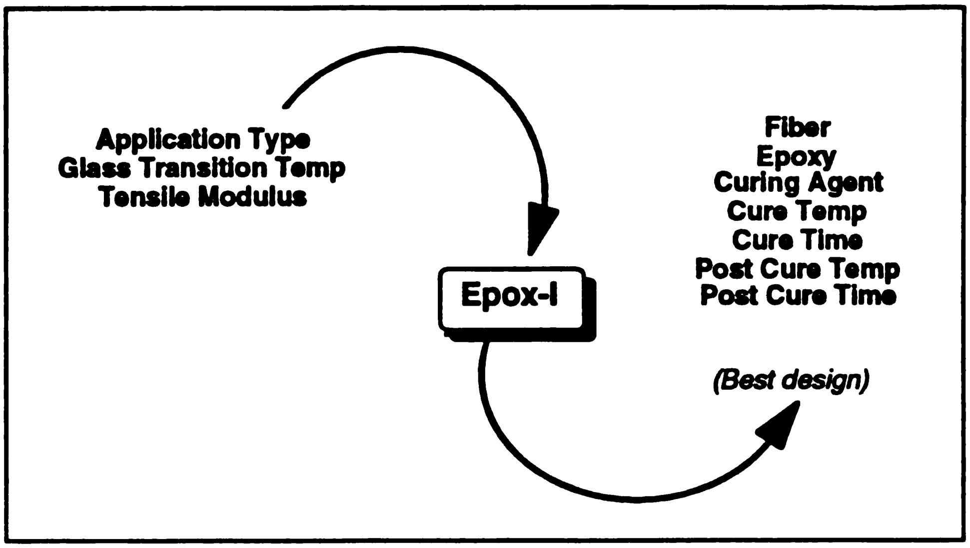

Figure 30. Input-output behavior of Epox-I.

at least 200°C and a tensile modulus of at least 50 GPa. Epox-I selected the default epoxy, diglycidyl ether of bisphenol-A (DGEBA), as the Epoxy, diaminodiphenylsulfone (DDS) as the CuringAgent, E-glass fiber, and a cure cycle of one hour at 150°C followed by three hours at 220°C at atmospheric pressure. See Figure 31 for this example.

This example is straightforward from a composite materials perspective. The Routine Design framework, which had not been used previously for selecting raw materials or planning a fabrication process, facilitated the generation of this solution [Kamel & Sticklen, 1990].

5.1.3 Planned Continuations

The stated continuing goal of the project was to apply the broad framework of the generic task approach to realistic composite material design problems and to thereby uncover any weaknesses in the approach. In addressing these weaknesses, principled

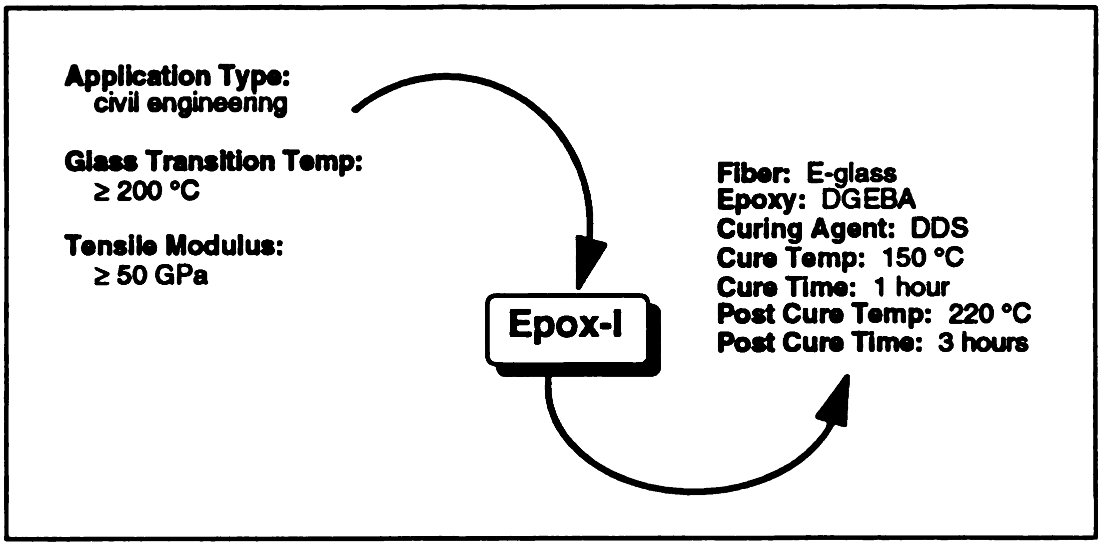

Figure 31. Example problem from Epox-I.

extensions could be made to the generic task tools and approaches, with the aim of enhancing overall problem solving capabilities [Sticklen *et al.*, 1992a].

Being preliminary, the initial structure of the architecture in Epox-I was confined in its coverage of the domain. An expanded coverage of the composite materials domain was to be the first extension to Epox-I. Additional domain knowledge would include a greater range of epoxies and fiber types and would eventually branch out to include interphase relations (particularly fiber-treatment conditions) and different matrix materials (especially ceramic and metal matrix composites) [Kamel & Sticklen, 1990; Sticklen *et al.*, 1992a]. A longer term extension would incorporate the functional reasoning component into the design subsystem of the overall architecture as a qualitative device model for the composite material [Sticklen *et al.*, 1992a; Sticklen *et al.*, 1992b].

5.2 Thermos-I

The next system, Thermos-I, was essentially an extension of the previous system, Epox-I, in that it had a slightly expanded domain coverage. However, the surrounding architecture had evolved to present a broader picture of the design of polymer composite materials. The organizational structure of Routine Design significantly facilitated this expansion.

5.2.1 Global View of Composites Design

Thermos-I proposed a more comprehensive architecture in which separate modules existed for material design and fabrication technology selection. This marked a new direction for the overall approach to composites design. Previously, the fabrication technology had been defaulted to only one choice: hand layup with autoclave cure.

It was recognized that the default was not necessarily the best approach in every situation, so the overall design picture was made more flexible. A module to selectively determine material/process compatibility and generate a combination of the results from the material design and fabrication technology selection modules was planned, but was not implemented [McDowell *et al.*, 1993]. See Figure 32 for a diagram of this comprehensive architecture.

5.2.2 Material Design

Thermos-I essentially maintained the specialist hierarchy established in Epox-I. Instead of the Epoxy specialist, Thermos-I had a more general Polymer specialist-an indication of its expanded domain coverage. An additional sub-specialist to the Curing Agent specialist was also added to deal with Cure Agents for Non-Epoxies. These pieces were

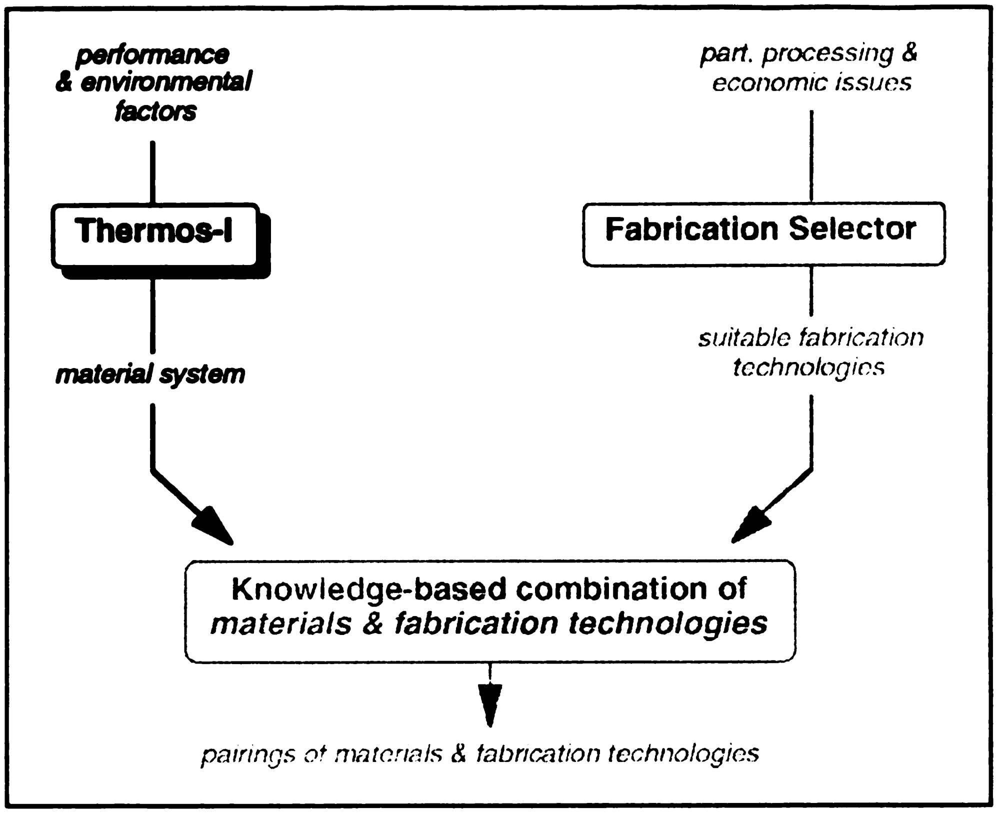

Figure 32. Planned integration of Thermos-I and a fabrication method selector.

incorporated into the existing Epox-I structure. This addition was facilitated in part through the Routine Design foundation upon which this problem solver was built. These additional specialists are indicated in Figure 33.

Other improvements included the capability of selecting from nine different resins, as opposed to only one in the previous system (Epox-I), and graphical interfaces for both the input and output. The designer could now change input values directly using an input interface with a display for features and their values. The output browser also provided the designer with an explanation in the form of the specific match table used to determine a particular value. For example, if the user chose Polymer Material and requested an

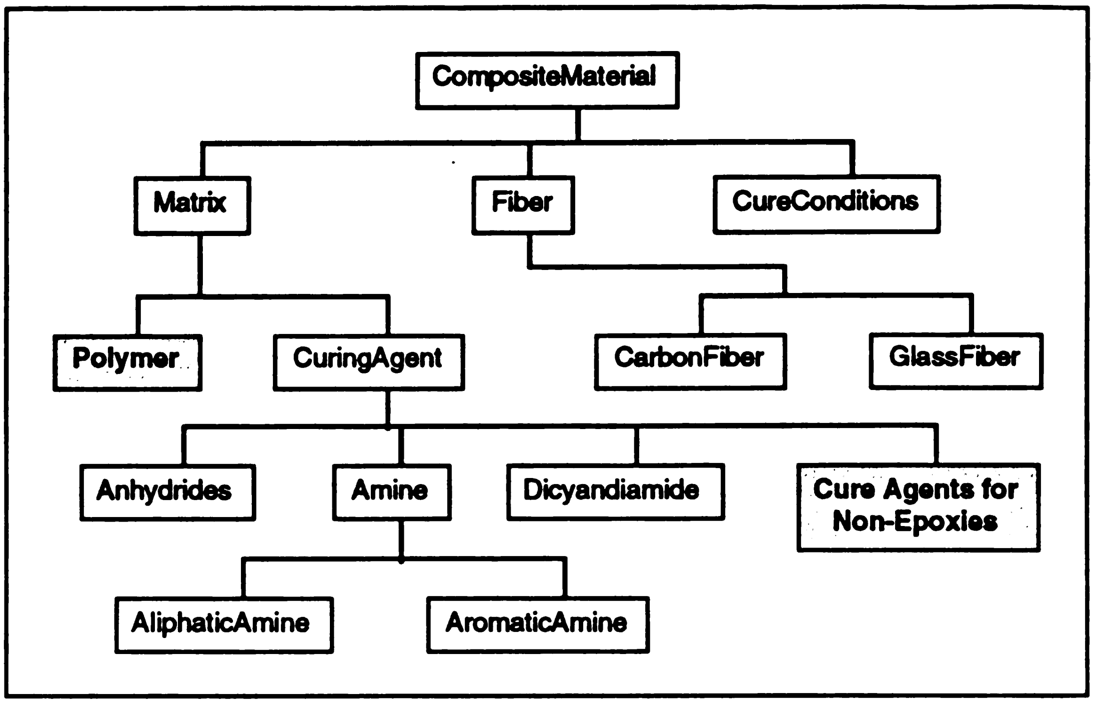

Figure 33. Specialist hierarchy for Thermos-I.

explanation, the browser would automatically display the corresponding matching table as a first level explanation for this design decision.

The list of variables used to make decisions was expanded to include five additional inputs. The three inputs used previously in Epox-I remained, and are shown in gray type in Figure 34. Note that the variables in the generated fabrication plan are identical to those in Epox-I.

By way of comparison, the sample problem presented for Epox-I was solved using Thermos-I [McDowell *et al.*, 1993]. The inputs to the problem solver were as before; additional inputs set the upper use temperature at 100°C, did not require resistance to either corrosion or humidity, and did not specify the material's rigidity or flame retardance. The output was identical to that provided by the previous system, as was expected

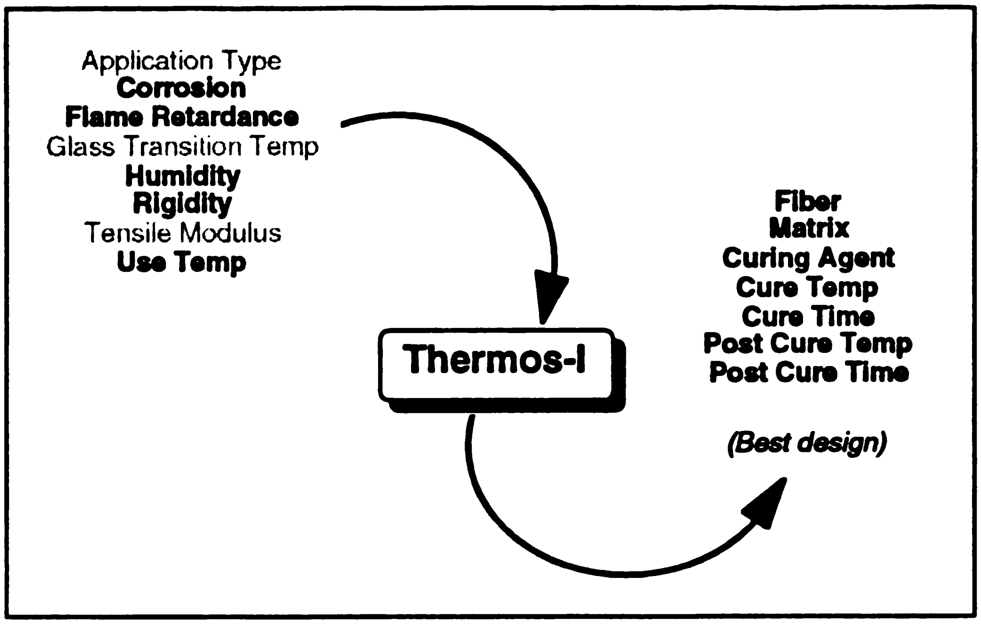

Figure 34. Input/output behavior for Thermos-I.

given the similarity of knowledge in the two problem solvers. See Figure 35 for this example.

5.2.3 Planned Continuations

The expansion of the polymer composites domain coverage seen in Thermos-I was a planned extension of Epox-I. A key point in this transition was the fact that the knowledge in Epox-I was totally reused without starting over. However, it was realized that the full complexity of the material design could not be completely depicted using the previous approach. Engineering problems rarely have a single solution, and in most cases, several composite systems might satisfy the performance requirements. On the basis of discussions with industry representatives, it was determined that the output of a family of designs would present a more effective aide to composite designers than would the output of a single design.

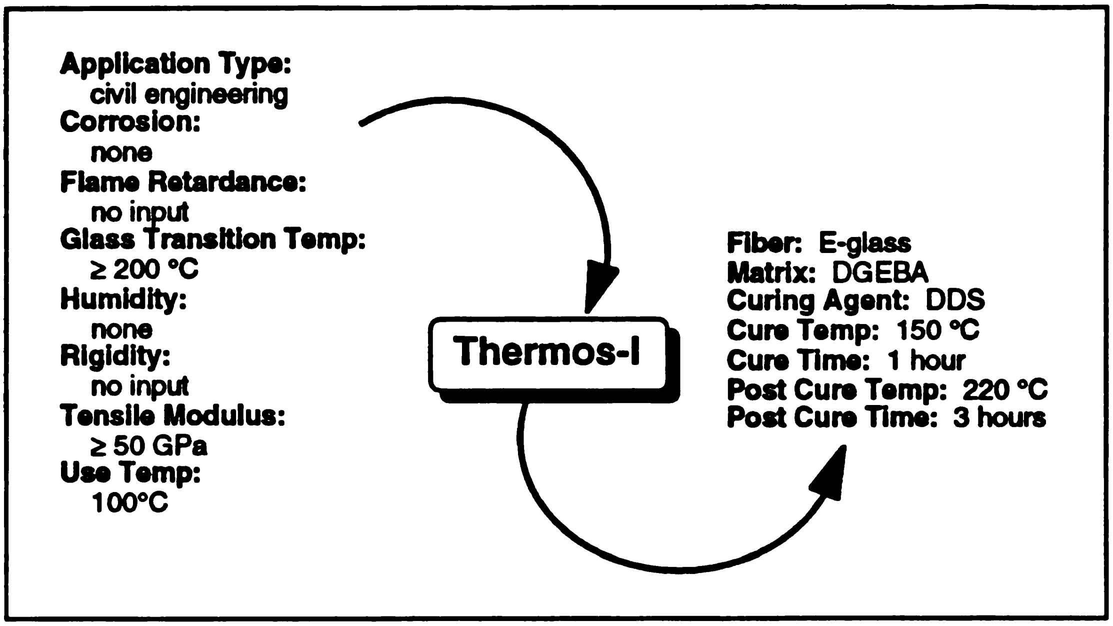

Figure 35. Comparison problem input and results for Thermos-I.

A modification of the Routine Design paradigm would be required to produce such a family of designs. Modifications to the inference strategies and knowledge representations were planned to enable this extension. The module for handling the selection of manufacturing methods was also to be completed, but the method for doing this selection had not yet been determined.

5.3 CMatDesign

This system represented a marked departure from the previous methodology for the design of polymer composite materials. Not only were multiple outputs to the same problem given, but the overall approach to composite material design had significantly broadened in scope. Two new prototypes to aid in composites design had been implemented: CMatDesign for material design [McDowell *et al.*, 1996], and CTechSel for fabrication technology selection [McDowell *et al.*, 1993]. These resulted directly from the planned expansion of the previous system, Thermos-I.

5.3.1 Global View of Composites Design

Composites design was now envisioned along three dimensions: material design, assembly design, and process design. Material design involves mapping performance requirements to choices for the fiber, the matrix, the necessary curing agents, and the appropriate processing conditions. Assembly design determines the overall artifact geometry as well as the fiber architecture, fiber fraction, orientation, stacking, integration of other materials (honeycomb or foam), and issues of bonding and fastening. Process design involves an array of complex problem-solving activities, from selecting the processing technology to determining the parameters at which the chosen technology will operate [McDowell *et al.*, 1996].

CTechSel, the current implementation for process design, uses hierarchical classification to perform process technology selection. The combination of the material design and processing technology selection envisioned for CMatDesign and CTechSel can be seen in Figure 36.

5.3.2 Material Design

The domain coverage and specialist hierarchy in CMatDesign were the same as in Thermos-I (Figure 33), as was the input/output behavior (Figure 34). However, CMatDesign could handle families of designs instead of the best single design. To accomplish this, modifications were made to the Routine Design paradigm. For a detailed description of these changes to Routine Design, see [Kamel *et al.*, 1994].

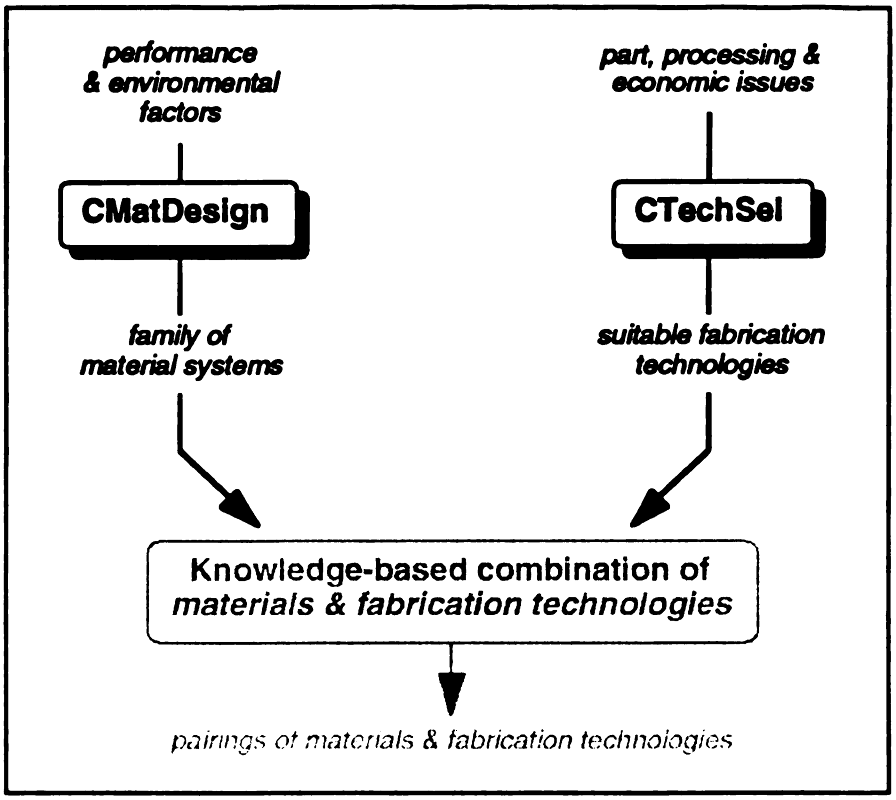

Figure 36. Planned combination of CMatDesign and CTechSel.

The comparison sample problem was solved using CMatDesign. The specified inputs to the problem solver were the same as those for Thermos-I. CMatDesign selected three epoxies: hydroxyphenyl, DGEBA (the previous default choice from Epox-I), and epoxidized phenolic novolac, as the Matrix, diaminodiphenylsulfone (DDS) as the Curing-Agent for all three Matrices, four types of fiber: E-glass, S-glass, AS4-carbon, and graphite, and a cure cycle of one hour at 150°C followed by three hours at 220°C and atmospheric pressure [Kamel *et al.*, 1994]. In Figure 37, a single trace through the displayed output nodes from Matrix to Fiber represents a design.

This output included the output provided by the previous system, but found 11 additional designs that met the specified conditions. Recall that this was without any

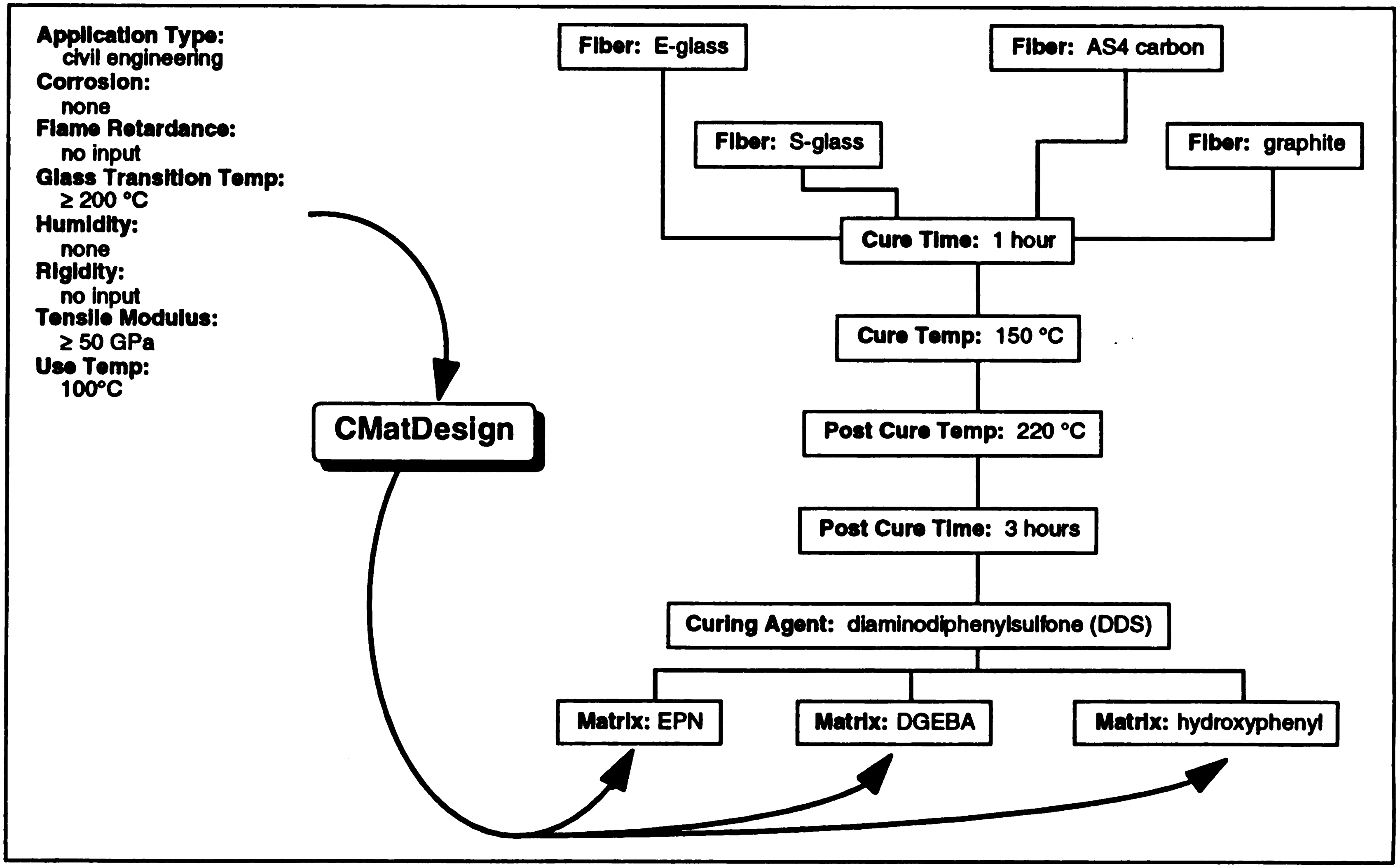

Figure 37. Comparison problem input and results for CMatDesign.

modification of the domain content. Once again, the use of Routine Design allowed modification to only the evaluation portion of the program and not the knowledge-base, which was left intact from the previous system.

5.3.3 Planned Continuations

With the impetus of presenting families of designs, CMatDesign was implemented using a new approach called multiple design. With essentially the same depth as the Thermos-I system, CMatDesign generated a family of designs, instead of the first best design as the previous two systems had done. Again, an important note to mention is that the knowledge base was essentially unchanged from Thermos-I to CMatDesign.

For the material design portion of the architecture there were immediate plans to implement additional portions, including a case-based reasoner to archive successful designs and a model-based reasoner to evaluate proposed designs [McDowell *et al.*, 1996]. This model-based reasoner was first proposed as an extension to Epox-I. Additionally, CMatDesign's organizational structure was to be scrutinized and modified as necessary. This was to be carried out in conjunction with an expansion of the domain coverage to include not only thermosets but also thermoplastics.

5.4 Lessons from the Evolution of these Systems

As the development of these systems evolved, several issues became apparent. The simple formulation of the design problem utilized for the initial system was not comprehensive enough for use in the later systems. This was in part attributable to the intentionally restricted domain space for which Epox-I was intended. Other lessons learned in this evolution can be seen by considering the changing global vision of polymer composites design and the focused approach for material design.

This tour through the development of an architecture for polymer composite material design has shown the value in learning from intentionally restricted domains. The solution to the restricted domain problem can be reevaluated as the domain size and complexity is expanded, and modified as necessary. The relatively simplistic approach to composite materials design in Epox-I was incapable of fully encompassing the complexities of an expanded material domain. A more detailed approach utilizing additional inputs and an expanded output set was necessitated in Thermos-I and CMatDesign.

The views of global composites design and the focused composite material design envisioned for Epox-I were streamlined as well. Initial assumptions which enabled this streamlining included a default fabrication technology. Once the overall design vision was expanded to include the selection of appropriate fabrication technologies, an additional problem solver was required. This is not unexpected, as an increase in the complexity of the design problem nearly always leads to a more complex implementation scenario. This idea was bourn out in the previous chapter.

While planned and discussed at a high level from the very beginning, other computational aspects of the composites design process were not implemented. These included the functional reasoner for design consequence checking and the case-based reasoner for designing from previous design solutions. The reasons for this postponed implementation were a lack of fundamental composite material functional knowledge and a lack of design space complexity (for the functional reasoner and the case-based reasoner, respectively).

In the course of the development of these three systems, it was realized that the full complexity of the polymer composite material design could not be completely depicted using the first, best design. Engineering problems rarely have a single solution, and in many cases, the performance requirements may be met by several composite systems. Representatives from the composites industry indicated that a family of designs would present a more effective aide to composite designers than would the output of a single design. This capability was added to the design approach, with CMatDesign presenting families of designs as output.

Just as the expanding domain coverage forced modifications to the accepted design approaches for the previous development, so too should changes be expected for any additional domain expansion. The planned extensions to CMatDesign stated that thermoplastics were to be added into the materials domain coverage. As design issues for thermoplastics are quite different from those of thermosets, an additional evolution of the material design approach should be anticipated for a system incorporating thermoplastics.

Part IV

Augmenting the Legacy

CHAPTER 6

Refocusing Composites Design

An observation of the developments detailed in the previous chapter shows several components of the design approach intimated but never realized. A lack of cohesiveness in portions of the design vision is also apparent. With the ISL's composites design history firmly in mind, the remainder of this dissertation extends and expands the capabilities for and the understanding of the composite material design process.

Of particular interest is the implementation of an extended composite material designer that includes thermoplastics, and the establishment of protocols for and the development of a case-based reasoning approach to composite material design. These new systems and approaches will facilitate the use and reuse of engineering design knowledge, thus enabling the transfer of expertise. By streamlining the mundane design processes, these systems will also free design engineers for more creative design activities.

The remainder of this chapter examines the previously used composites design visions (global and material-specific) and reevaluates the content of these visions. Modifications to these visions that serve as the guidelines for the research in this dissertation are presented as well. This chapter concludes with a presentation of several explicitly stated research goals.

6.1 Rethinking the Global Composites Design Vision

Throughout the previous chapter, the overall approach to designing composites evolved with the implementation of each new system. The driving forces behind this evolution of the design vision alternated between domain-specific issues and the methods by which the systems were implemented.

While computational developments have pushed the understanding of the design process, representation of the polymer composites domain has not advanced as rigorously. A recommitment to the polymer composites domain, driven by a chemical engineering perspective, has resulted in a refocused overall vision for composites design.

This discussion presents the initial view of the design to manufacturing cycle for composites and compares it with the new vision that now guides the composites design research.

6.1.1 The Previous Global Design Vision

The overall design vision within which all of the previous composites design development in the ISL occurred was essentially linear, proceeding from initial design specifications through to the finished composite part. Figure 38 shows this design vision.

The first step in the design vision designed of the composite material by considering factors related to company interests, economic considerations, and to the specific application. The details of this design were dependent upon the specific computational technique used. Once the best composite material design was determined, the appropriate

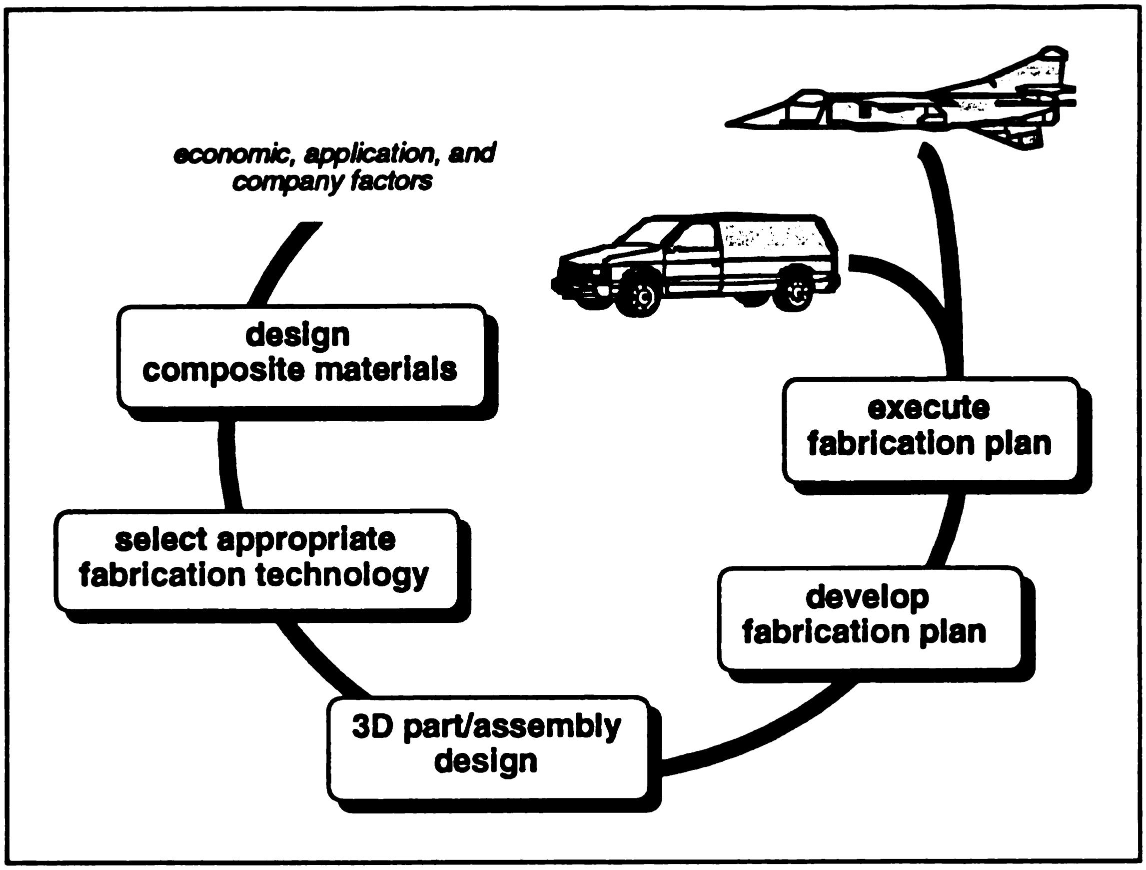

Figure 38. Originally envisioned design-to-manufacture cycle for composites.

fabrication technology for producing that material was selected. After this process selection, the detailed geometry of the part was designed. Combining information from the material design, the fabrication technology, and detailed part geometry, a detailed fabrication plan was established. This plan listed a specific recipe for producing the composite part. Finally, the fabrication plan was carried out and controlled, resulting in the finished composite part.

For an initial guiding vision, this design-to-manufacture cycle served its purpose well: it served as a foundation upon which preliminary composites design efforts were based. It was at the time a seemingly logical progression of design tasks to accomplish in the design and production of a composite material. Only by revisiting the design vision over the course of the previous developments were inconsistencies with existing design practices identified.

Unfortunately, this design vision was entirely too dependent upon typical design processes in other, more traditional, application domains (e.g., metals, chemical processes). In such domains, a linear design process is possible due to the detailed understanding of the interactions between various portions of the design. For composites, that level of understanding for the interactions for materials, processes, and geometry simply does not exist.

If this design vision were to be used for additional decision support systems, it would have to be modified, with several iterative loops inserted between material design, fabrication technology selection, and geometry design. Perhaps there is another way to envision the composites design to manufacturing process other than linearly.

6.1.2 The New Overall Design Vision

Over the course of the previous composites design development, it became clear that the best vision of the composites design process should not be rigidly specified. In fact, the design process is likely connected on many levels. The details of how these levels interact are unclear. It does seem clear, however, that separating out the individual design domains will facilitate the design process by simplifying the domain coverage for each system developed to solve them. This, in turn (as we have seen before), should reduce the required computational complexity.

The new multiply-interacting design vision of intelligent decision support for polymer composites is composed of several modules based on various intelligent technologies contributing design advice, design criticism and design generation for composite materials, composite parts and composite fabrication processes. See Figure 39 for this design vision.

Although not explicitly depicted in Figure 39, the understanding of the composites design process has been advanced as well. The design of composite materials involves the mapping from performance requirements to choices for the fiber, fiber length, matrix, and any necessary chemical agents. In the design of composite parts and assemblies, requirements include partial geometric information and tolerances and also various environmental loadings. Part design determines the overall part geometry and fiber architecture, fiber fraction, orientation, stacking, integration of other materials (honeycomb and/or foam) and issues of bonding and fastening. Fabrication process

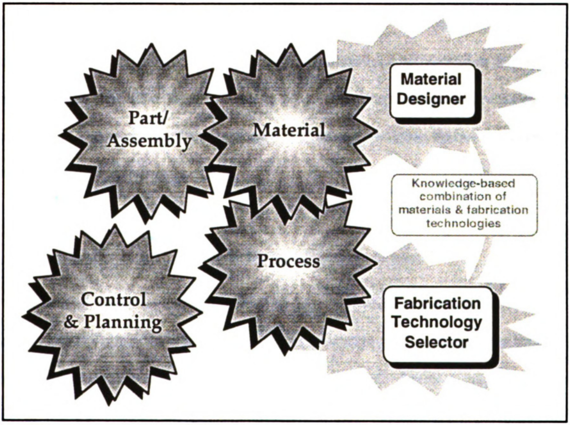

Figure 39. Revised global design/manufacturing vision for composites.

design involves an array of complex problem solving activities, ranging from selecting the fabrication technology to deciding the parameters at which the chosen technology will operate.

Three of the four starbursts in Figure 39 overlap with each other, but not in any "easy" gear-like fit. This depicts the expected multiple interactions between the design areas, but neither the method of implementation nor the order of execution is necessarily mandated. The best way to find the detailed interactions between these design areas to continue the research into and development of decision support systems for composites design.

Part of the design vision that has been defined is shown on the right of Figure 39, where material design and process design are partially accomplished by a material designer and a fabrication technology selector, respectively. The results from both systems are then combined to decide appropriate combinations of fabrication technologies and materials. The fabrication technology selector finds appropriate technologies using hierarchical classification. Details of how the material designer figures out composite material designs are in a state of flux.

6.2 Rethinking the Composite Material Design Vision

As the overall vision for the composites design process has evolved, the more detailed aspects of this vision have also changed. Part of this change can be attributed to fitting the detailed portions into the context of the new design vision. However, since the overall design vision's evolution was guided in part by developments at the material design level, additional consideration of the details of the design process is warranted.

This discussion compares the new vision for the design of composite material systems with the previously used vision. A key difference between the new vision and the previous one is the representation of the composite material design problem itself.

6.2.1 The Previous Material Design Vision

Material design as conceived previously involved mapping performance requirements to choices for the fiber, the matrix, the necessary curing agents, and the appropriate processing conditions. This problem formulation was used most recently in CMatDesign. See the vision for material design within which CMatDesign was developed in Figure 40.

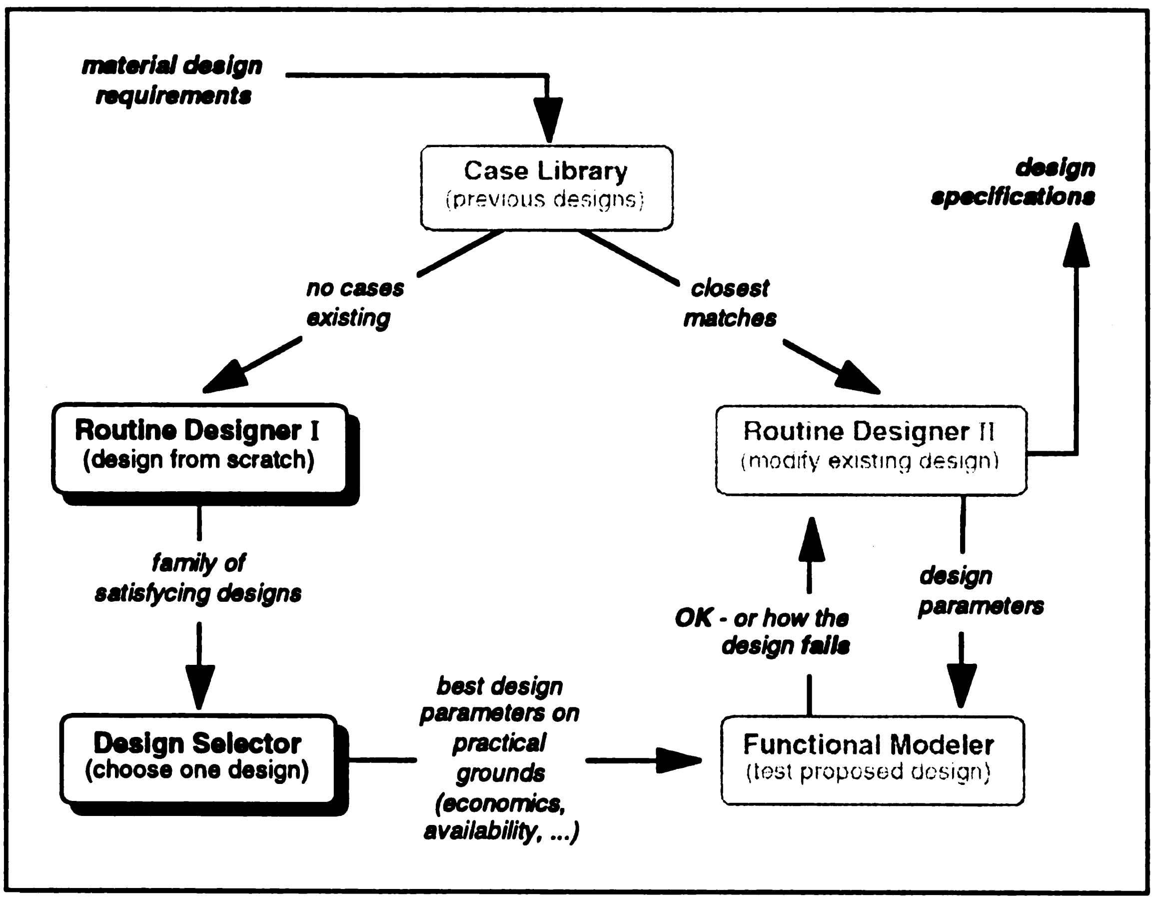

Figure 40. CMatDesign-era vision for composite material design.

While not implemented at the time of CMatDesign, the plan was to use a case-based reasoning approach to retrieve previous design solutions. If a design solution was found, it would be modified by another (again, unimplemented) routine design system. The design-from-scratch routine designer was to be used only when no previous design solutions were found by the CBR system. CMatDesign embodied the design-from-scratch portion of the material design vision. While an intermediate result of CMatDesign was a family of designs, these designs were pared and only the "best" design was presented as the final answer.

6.2.2 The New Material System Design Vision

The material design vision which guides the developments in this dissertation is a slight variant of the previous vision. Figure 41 shows this new vision. The main distinction between this and the previous design vision is the lack of a Design Selector. The output from the material designer is to be a family of design solutions and there is no need to explicitly select only one solution. This family of material designs will then be matched to the output from the fabrication technology selector to generate a matched list of acceptable combinations of materials and fabrication technologies (see Figure 39). At that point, there may be a need for a design selector.

Material design for polymer composites involves the mapping from environmental and performance requirements (mechanical, thermal and chemical) to choices for fibers and matrices. Depending on the type of matrix, a detailed recipe of reactive diluents, inhibitors, fillers, initiators, and other additives may be specified as well. Notice the lack of any processing information in this design solution, as had previously been included.

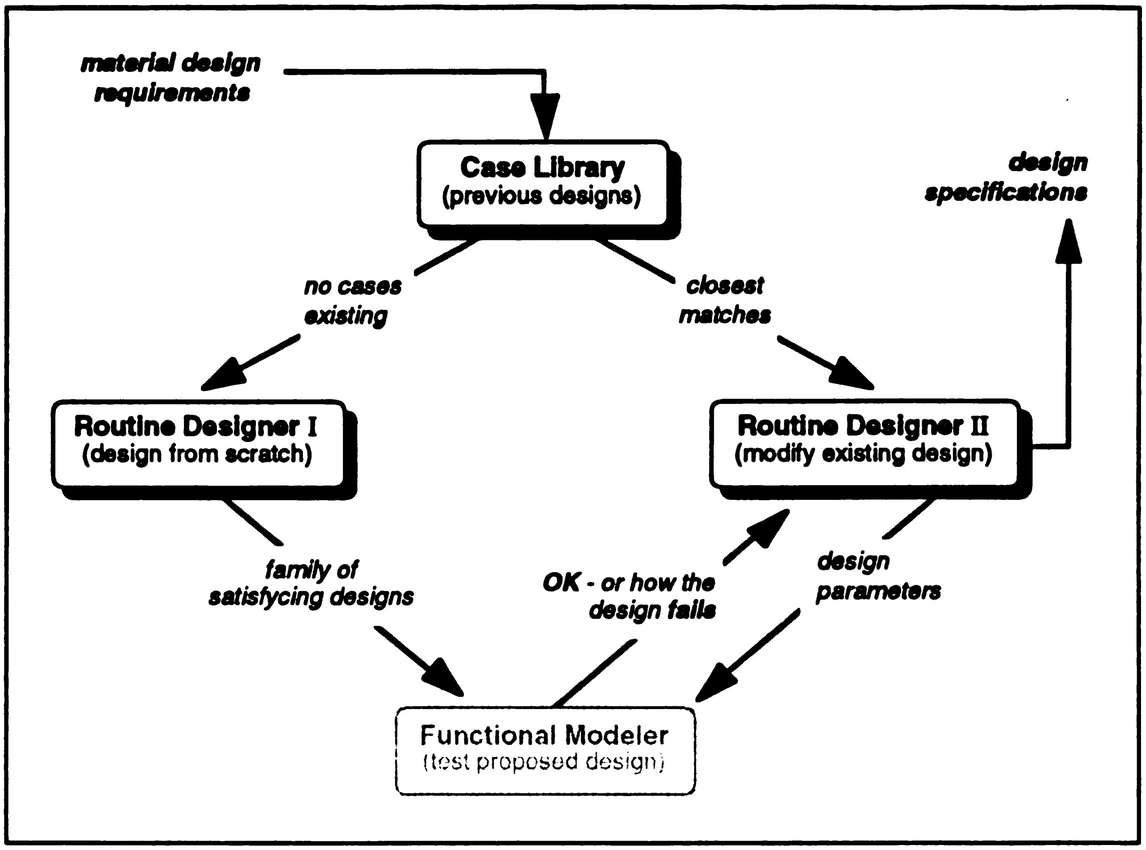

Figure 41. Post-CMatDesign envisioned material system design approach.

A distinction must be made between polymer composite materials and polymer composite material *systems*. This distinction is especially important when considering the implications of the design of either. Design of polymer composite materials implies a creative aspect to the design, with new classes of materials potentially being invented. As that is not the intent in this dissertation, another term must be used to explicitly avoid this misinterpretation. Consequently, the term "polymer composite material systems" is used.

As used within this dissertation, polymer composite material systems specify combinations of polymer matrix materials, chemical agents (curing, co-curing, reactive diluents), fiber materials and fiber lengths. The design of polymer composite material systems specifies nothing other than finding valid combinations of material system constituents.

6.3 Research Plan

The research presented in this dissertation focuses on the application and development of engineering design techniques for designing polymer composite materials. Principles of engineering design in general, and chemical engineering design in particular, are prevalent throughout this domain and will guide this development. Of specific interest is the design of composite material systems.

The planned work for composite material system design focuses on two main issues: the accumulation and organization of design knowledge, and an examination of the philosophical nature of the design process. This accumulation and organization will result in a new paradigm for composite material system design.

Not only will this paradigm serve to formulate the design process, but it will also serve as the basis for the construction of tools that enable its use. The tool of interest for embodying the paradigm is a knowledge-based approach to composite material system design. Instructive case histories of the design process and a way to reason with them forms the other contribution. This involves the development of a case-based reasoning approach to composite material system design. The following discussion briefly examines each of these issues.

6.3.1 Implement an Expanded Material Design System

Following the development of CMatDesign, there were immediate plans to incorporate additional capabilities in the material design portion of the composites design vision. CMatDesign's organizational structure was to be scrutinized and modified as necessary. This was to be carried out together with an expansion of the domain coverage to include not only thermosets but also thermoplastics.

The specific charge at the inception of the research for this dissertation was to expand the composite material system domain coverage to include: a total of ten fibers, two to five more thermosets, five to ten thermoplastics (this is to be a totally new addition), and to reevaluate the existing knowledge structuring and inference patterns, and modify them as needed.

Not only is design knowledge about composite material systems to be accumulated, but it is to be organized in a way that is useful for design. The open-ended nature of the design process mandates the determination of abstracted heuristic knowledge. Such knowledge is useful for limiting the scope of the design space. Any problems due to ill-structure are significantly limited by the establishment of a detailed structure of the polymer composite material system design problem (seen in the previous section).

Ideally, the composite material system designer will be used with other computational tools that address the design of the processing and structural aspects of the composite material. An additional computational tools with which the composite material system designer will be used designs by reusing past design solutions.

6.3.2 Develop a CBR Capability for Material Design

The other computational capability to be developed for this dissertation is designing new composite material systems using previous design solutions. A case-based reasoning approach can accomplish just that.

A CBR approach can use the specific knowledge of previous concrete design problems. This is an experience-based method that attempts to deal with current problems through a comparison to past solutions. A new problem is solved by finding a similar past case and reusing it as indicated by the new situation. As new problems are solved, the solutions are stored as cases. This allows incremental learning, which in turn allows the reasoner to become more efficient. The incremental learning occurs as a CBR system accumulates both specific and general knowledge. The mixture of knowledge level abstractions facilitates the solution of problems requiring the simultaneous handling of multiple interacting considerations.

Use of a CBR system together with the previously mentioned material system designer will help to shed light on the design process. The elucidation of that design process is the final goal of this dissertation.

6.3.3 Scrutinize & Illuminate the Design Process

The focus of this dissertation is to examine the ideas behind the design of polymer composite material systems at varying levels of complexity and to develop a cohesive approach for efficaciously performing design.

While a design vision currently exists as a guide to development, the details of this vision are unclear. These details will be made more clear throughout this dissertation. This is to be done with an emphasis on knowledge-based systems, carefully engineering the possible interactions of information and structuring the problem of material design in a way that provides not only substantive research results, but also genuinely usable answers.

The applicability of the rigid phase level design process for composites as a way of thinking about the design and as a guideline to implementation of the design process are also to be considered. An examination of the design tasks comprising the problem to be solved must guide this deliberation. It will also be decided if the conceptual phase of the design should be exclusively compartmentalized when implemented.

The first step toward achieving an understanding of the material design process is to examine in detail the foundations upon which developments are to be built. As CMat-Design was the immediate precursor to the research in this dissertation, the details of its approach to material design, including design problem formulation and knowledge representation, must be examined closely.

CHAPTER 7

Dissecting CMatDesign

A detailed examination of the foundations for the new composite material system design developments will give a fuller understanding of the material design process. As CMatDesign was the immediate precursor to the research in this dissertation, it is the system to examine in detail. CMatDesign's approach to material design, including design problem formulation and knowledge representation, must be examined closely.

Although CMatDesign was discussed in Chapter Five, there are still untapped details within it that merit further consideration. The previous discussion was at a high level, looking at the global design environment into which CMatDesign fit and its behavior within that environment. An in-depth analysis of CMatDesign was completed to more fully understand the knowledge structuring and decision processes it used. The results of this analysis will serve to guide the reengineering and expansion to the next generation of the material designer.

The analysis here examines three aspects of CMatDesign: domain coverage, knowledge structuring, and inferencing procedures. With each of these three areas, a baseline for the continued development of the composites design approach is established.

An enumeration and critical evaluation of CMatDesign's domain coverage sets the stage for the planned expansion of the materials coverage. Both the knowledge structures and the specific inferencing procedures used within CMatDesign are dependent upon this domain coverage. By examining the knowledge structures in light of CMatDesign's domain, potential avenues for improvement can be identified. Similarly, an examination of the inferencing procedures in CMatDesign helps to point the way for the next development.

7.1 Domain Coverage

CMatDesign was intended for the design of thermoset polymer composite materials [Kamel, 1994]. The thermosets considered within CMatDesign for defining composite materials were polyesters, polyimides, and epoxies. The domain coverage within CMatDesign was influenced by the representation of the composite material design problem, and therefore only constituents of the generated design needed to be included. As defined within CMatDesign (and its predecessors, Thermos-I and Epox-I), a polymer material system included a polymer material and a curing/co-curing agent.

The entire list of materials included within CMatDesign was copied directly from Thermos-I, which in turn copied all of the epoxies in its database from Epox-I [Kamel & Sticklen, 1990]. When Thermos-I was created, several polyesters and polyimides were added to its database [McDowell *et al.*, 1993]. Therefore, the polymer materials in

Table 4: The 105 (*130*) material systems included within CMatDesign.

Polymer Type	Polymer Material	Curing/Co-cure Agent	Fiber Type
Polyester (4)	Orthophthalic acid Maleic acid	Chlorostyrene Styrene	None AS-4 Graphite S-Glass E-Glass
Epoxy (15/*20*)	Hydroxyphenyl Epoxidized Phenolic Novolac DGEBA *TGMDA*	MPDA DDS DEAPA Anhydride Dicyandiamide	
Polyimide (2)	BMI	Olefin	
	PMR-15	Alcohol solvent	

CMatDesign's database were inherited from Epox-I (epoxies) and Thermos-I (polyesters & polyimides). Table 4 shows all of the composite material systems in CMatDesign.

By combining the 21 possible polymer materials (four polyesters, 15 epoxies, and two polyimides - see Table 4) with the five fiber types (including 'None'), CMatDesign covered a design space of 105 possible composite material systems. If the final epoxy in the database (TGMDA) had been added into the knowledge structure, CMatDesign would have contained a total of 130 composite material systems.

CMatDesign's representation of its knowledge and the organization of its approach to determining composite material designs were largely predicated on the material domain coverage shown in Table 4. However, CMatDesign did not address the representation and structuring of knowledge anew with its domain coverage; instead, it used the representations and structures of Thermos-I and Epox-I. This led to inconsistencies in CMatDesign's knowledge structures.

Any additional expansion of the material domain coverage should be accompanied by a fresh examination of the required knowledge representation and structuring. This examination must be done with the goal of capturing the material domain as aptly as possible.

7.2 Knowledge Representation & Organization

As CMatDesign was a routine design problem solver, certain ways of representing and reasoning about domain knowledge were fixed (see related discussion in Chapter Two):

- the design problem must be represented in terms of specific types of variables (e.g., Yes/No, OneOf, and so on);
- the design problem must be decomposable into smaller subproblems; and
- the domain knowledge must be represented in terms of structured pattern matching tables.

The discussion in this section looks at how CMatDesign represented each of these three aspects. An examination of how the design problem was represented within CMatDesign immediately follows. Variables used to make decisions about the constituents of the design problem are also presented. Finally, the decomposition of the design problem and the resulting knowledge organization is discussed.

7.2.1 Design Problem Representation

CMatDesign's design problem representation included a total of nine constituents. Table 5 shows the complete design problem representation used in CMatDesign. Not only did CMatDesign include material systems (**Polymer Material, Curing/Co-Cure Agent and Fiber Type**) in its output, but also material specific processing information (**Cure Temperature, Cure Time, Post-Cure Temperature, and Post-Cure Time**), conditional thermal (**Glass Transition Temperature**), and mechanical (**Tensile Modulus**) properties.

Despite the existence of a system for the selection of fabrication technology (CTechSel) and an overall vision for combining the results of CMatDesign and CTechSel (Figure 39 on page 150), CMatDesign still included fabrication knowledge in its design output. This processing information (as listed above) was specific to the autoclave processing of thin film laminate composites. This processing information was also incomplete, as an autoclave curing cycle must include additional specifications. For the processing information to be complete, it should have also specified information about ramping cycles for both pressure and temperature as well as initial and final temperatures and pressures.

Table 5: Output variables used in CMatDesign.

Variable Name	Variable Type	Valid Values
Polymer Material	OneOf	(see Table 4, column two)
Curing/Co-cure Agent	OneOf	(see Table 4, column three)
Glass Transition Temperature	Numerical	160 or 220 °C[a]
Fiber Type	OneOf	(see Table 4, column four)
Tensile Modulus	Numerical	800 GPa max
Cure Temperature	Numerical	80 - 200 °C
Cure Time	Numerical	3 - 240 or 0 - 3[b]
Post Cure Temperature	Numerical	0 - 250 °C
Post Cure Time	Numerical	0 - 240 or 0 - 3[b]

a. Values were assigned only for epoxies and when the Curing/Co-cure Agent was an aromatic amine: MPDA (160 °C) or DDS (220 °C).

b. No units were specified within CMatDesign for this variable. An examination of the specific variable value assignations within the system indicates several inconsistencies. Most likely, certain values assigned should have units of *minutes*, while others should have units of *hours*.

By reusing the existing knowledge in the databases and knowledge structures of Epox-I and Thermos-I without carefully considering the underlying assumptions for that knowledge, CMatDesign acquired unintended baggage and was not as focussed in its design task as it should have been. A result of this is that CMatDesign's output cannot be considered a composite material system design, but should instead be called a composite material fabrication plan.[1]

An examination of the initial problem description for Epox-I [Kamel & Sticklen, 1990] shows the same design problem as used in CMatDesign. However, as materials included within Epox-I were only a subset of those in CMatDesign, the design problem should have been more simple. Instead, there was no difference between the design problems. That is incongruous, especially when the significantly larger input set for CMatDesign is considered (see next section). Every indication points to a more complex design problem representation for CMatDesign.

7.2.2 Input Variables

CMatDesign used nine input variables to generate composite material fabrication plans. Types of variables in the input set included thermal properties (Use Temperature and Required Glass Transition Temperature), mechanical properties (Required Tensile Modulus, Rigidity, and Maximum Tensile Modulus), chemical & environmental properties (Corrosion, Humidity, and Flame Retardance), and the type of application for which the composite was to be used. Table 6 shows these variables.

1. The output from Epox-I was called a 'composite material fabrication plan.' Recall, however, that at the time of Epox-I there was no vision for combining the results of the material designer and fabrication technology selector. As such a vision existed at the time it was built, CMatDesign should have fit into the scope of that vision.

Table 6: Input variables used in CMatDesign.

Variable Name	Variable Type	Valid Values
Application Type	OneOf	Adhesives Casting Civil Engineering Domestic Application Electrical Encapsulation Lamination
Corrosion	Yes/No	Yes or No
Flame Retardance	Yes/No	Yes or No
Humidity	Yes/No	Yes or No
Maximum Tensile Modulus	Numerical	800 GPa max
Required Glass Transition Temperature	Numerical	220 °C max
Required Tensile Modulus	Numerical	800 GPa max
Rigidity	Yes/No	Yes or No
Use Temperature	Numerical	-270 to 350 °C

Nearly all of the input variables used by CMatDesign were shrouded in one sort of mystery or another. This enigmatic nature was partly due to the lack of any documentation within CMatDesign. As a result, any assumptions underlying the input variables were completely hidden. Because of this, questions arise for nearly every input variable. Each of CMatDesign's input variables therefore requires further consideration of the questions, apparent inconsistencies, and the potential negative interactions with other variables.

Application Type. This was the most widely used of the variables in CMatDesign and, unfortunately, also the most enigmatic. Although it had six different values from which to choose (see first row, third column in Table 6), exactly what each value meant was never made clear. No documentation to distinguish the different application types or to detail the underlying assumptions for each application was included within CMatDesign. The only assistance CMatDesign gave to the user for understanding this variable (besides the valid value names themselves) was the query prompt: *What type of application is this composite going to be used for?*

The idea of specifying a particular end-use application has much potential utility. However, the extreme dearth of documentation for this variable negated much of that potential. Figuring out how and why decisions in CMatDesign were made with this variable is a tedious process. Often, the decisions that were made seem nonintuitive. There may very well have been reasons for the choices made, but without documentation of the rationale the knowledge is suspect.

Examples of the unclear use of Application Type can be found for epoxies (=Adhesives, ~=Domestic Applications), polyesters (=Domestic Applications), and polyimides (~=Domestic Applications, ~=Civil Engineering). Other examples can also be found in the Curing Agent and Fiber specialists. There were implicit concerns behind each of these restrictions (e.g., structural integrity, safety and health factors, failure modes, and so on), but those concerns remain unclear.

The inclusion of Electrical Encapsulation and Adhesives as valid values for Application Type is disconcerting, especially considering that designs generated using these application types do not contain any fibers. That would be a non-composite material. These application types are holdovers from the Epox-I system, as both applications are possible uses of epoxies. However, in such instances, the material is not a composite. Furthermore, under no circumstances can such materials be produced using hand layup and an autoclave cure. Inconsistencies in design scope and intent definitely exist for this variable.

Corrosion. Because this variable was a Yes/No variable, its descriptive power was obviously limited. The documentation for this variable within CMatDesign was also limited. The extent of the documentation was the user prompt question: *Is this material going to be used in a corrosive environment?*

A typical response to this question could be: "Yes, but...what about x?" Simply stating whether the environment was corrosive was not a sufficiently detailed distinction. Nothing was said about the nature of the corrosive environment. An environment that may be very corrosive for a particular polymer can be totally harmless for another.

The lesson to be learned from CMatDesign's use of this variable is that if this question is to be asked in a new system, the nature of the chemical environment must be more detailed. Added details, such as a distinction between an acid and an organic solvent, will greatly increase the utility of the routine design approach for designing composite material systems.

Flame Retardance. Again, because this variable was a Yes/No variable, its descriptive power was limited. Documentation of this variable within CMatDesign was also limited. As for Corrosion, the extent of the documentation was the user prompt question: *Does the resulting material need to be flame retardant?*

On the surface, this variable seems straightforward and sufficient. However, there is a lot more to the flammability of polymers than whether they are flame retardant. The most important distinction to be made is the difference between flame-retardance and flame-resistance. A flame retardant material generally does not burn in the presence of a flame. A flame resistant material will burn, but it will extinguish itself when the source of heat is removed. There is a significant difference between those two extremes.

Perhaps the best way to delineate a material's desired flammability characteristics is to use the Underwriters' Laboratory (UL94) rating tests for the flammability of plastic materials.[1] The UL94 ratings are quantifiable and therefore have more utility in decision-making than does the specification of whether a material is to be flame retardant.

Humidity. This was another Yes/No variable with nonexistent documentation. As for the other Yes/No variables, the extent of the documentation was the user prompt question: *Is the resulting material going to be used in a humid environment?*

There is not much to be done with Humidity other than a simple yes/no distinction. The utility of this variable depends upon how it is used. That is detailed in a later section.

Maximum Tensile Modulus. This variable seems superfluous within CMatDesign given the existence of the variable Required Tensile Modulus. Its utility was limited, at best. Again, without any documentation there is no way to discover what role it was intended to fulfill. The user query is the only clue: *What is the maximum acceptable tensile modulus (in GPa)?*

The best explanation for this variable is that it was a leftover from an attempt at eliminating over-design. An examination of the use of this variable within CMatDesign (see next section) shows only one occurrence; that use of Maximum Tensile Modulus was not particularly significant. This variable can be eliminated from CMatDesign with no discernible consequence.[2]

1. There are five criteria characteristics for the UL94 flammability ratings. Generally, a 94 V-0 rating is satisfactory, while 94 HB, 94 V-1, and 94 V-2 are less satisfactory. A 94 V-5 is the highest rating. See p. 358 of [ASM International, 1987] for more detail.
2. Upon further examination and after closely reading [Kamel, 1994], it appears that this variable was only included in CMatDesign as part of an academic exercise to explicitly show the utility of the (then newly implemented) design limiters in MDSPL.

Required Glass Transition Temperature. A problem with this variable was the lack of units. That deficiency was due to an incapability within the ISL's implementation of routine design.[1] The only way CMatDesign could convey that the required T_g was in Celsius was with the user query: *What is the minimum required glass transition temperature (in celsius)?*

The specific request for a required T_g, especially considering that Use Temperature was also requested, seems redundant. There does not appear to have been any great utility gained by the distinction of the T_g from the Use Temperature. Given that the T_g marks the temperature at which a polymer's properties (e.g., refractive index, specific heat, shear modulus, and thermal conductivity) begin to degrade, there does appear to be some utility for its inclusion in the list of variables with which CMatDesign reasoned. However, this variable could very easily be eliminated from CMatDesign by judiciously employing the Use Temperature.

Required Tensile Modulus. There was an inconsistency between the name (and use) of this variable and the user query: *What is the minimum required tensile modulus (in GPa)?* This was most likely due to the confusion with the Maximum Tensile Modulus variable mentioned above. It appears that this was a result of a changing picture of how to specify and limit the mechanical properties of the composite material.

An inconsistency in the use of this variable arises when examining the details of CMatDesign. There was never any mention of an assumed fiber volume fraction or orientation.[2] These factors both significantly affect the resulting mechanical properties (e.g.,

1. That deficiency has since been rectified. Numerical variables within the ISL's GT toolset now possess default units and can convert between different (e.g., SI and AE) units for a variable.

tensile modulus) of the composite material. Unless these considerations were hidden in the numbers used for the specific fiber tensile moduli, it appears that CMatDesign neglected to consider them.

Rigidity. This was yet another Yes/No variable with nonexistent documentation. As for the other Yes/No variables, the extent of the documentation was the user prompt question: *Does the resulting material need to be rigid?*

Again, as for Flame Retardance, more needs to be specified than just one of two possible extremes. How does Rigidity = Yes translate into a value for the modulus of rigidity?[1] How are materials that are neither rigid nor non-rigid assessed? Another way of requesting (and using) information about the required rigidity of the composite material must be found to effectively use this variable.

Use Temperature. As for Required Glass Transition Temperature, the lack of units was a problem for this variable. Again, the way that CMatDesign conveyed that this temperature was in degrees Celsius was with the user query: *What is the maximum temperature the resulting material will be used in (in celsius)?*

The limits set for Use Temperature (see in Table 5 on page 163) of -270°C and 350°C for the lower and upper bounds were too extreme. The only way that the lower bound could ever be approached would be if the composite were to be used in space. As there was not a corresponding Application Type value for such a situation, this limit was

2. Although, if the assumption made in Epox-I of unidirectional thin-film laminates holds here, then these considerations are much less important. In such composite materials, properties such as the tensile modulus are strongly driven by the fiber properties. However, as has been seen elsewhere, this assumption was inconsistently applied within CMatDesign.
1. The modulus of rigidity is the ratio of stress to strain within the elastic region for shear or torsional stress. It is also known as shear modulus or torsional modulus [ASM International, 1987].

outside the bounds of possibility. Any terrestrial application will have a low temperature bound of -65°F (-54°C), and such a bound should more than adequate for CMatDesign.[1] Conversely, given the materials included in CMatDesign's database, 350°C as an upper bound is too high. None of the materials considered can perform at that temperature for any extended period. Polyimides can be used at temperatures up to 290°C for short periods without significant consequences (i.e., property degradation). Therefore, a more realistic upper temperature bound would be 290°C.

As alluded to in the discussion of **Required Glass Transition Temperature**, a certain degree of overlap existed between the two variables. **Use Temperature** could have easily encompassed all of the knowledge expressed in terms of the required T_g, and done so without compromising the nature of its own knowledge.

7.2.3 Knowledge Structures

The design problem representation and the set of input variables presented above served as the building blocks for CMatDesign. The organizational structure within which these blocks were used can be considered on two levels: large-grain, and finely detailed. As CMatDesign was a routine design problem solver, these two levels of detail were expressed by the specialist and agent hierarchies, respectively.

Specialist Hierarchy. Design problem solving in routine design is based on the hierarchical structure of design specialists, each of which offers a collection of expertise or knowledge on how to complete part of the design problem. This hierarchical organization implies that the design problem is decomposable or nearly decomposable into appropriate

1. This is the specification as set by the U.S. Government for low temperature performance of terrestrial aerospace applications.

subproblems. The specific arrangement of the specialists depends on the regularities that exist in a particular domain. For CMatDesign, as for its predecessors, the decomposition of the composites design problem was relatively simple. Figure 42 shows the specialist decomposition used in CMatDesign.

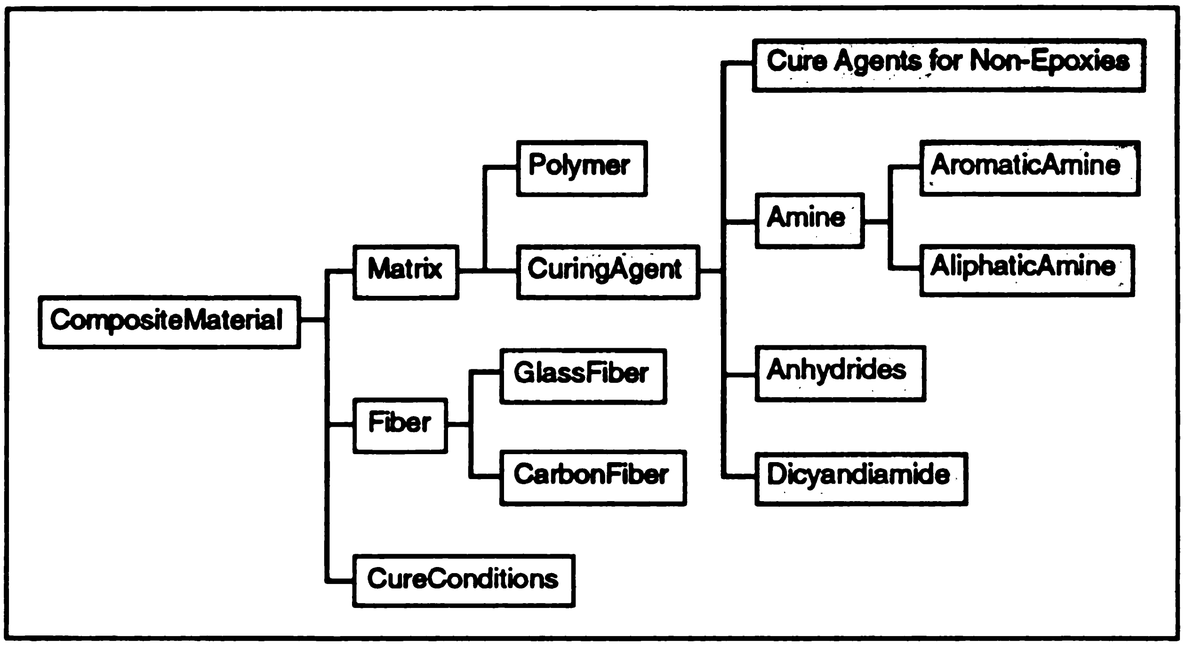

Figure 42. Specialist hierarchy for CMatDesign.

The specialist hierarchy shown in Figure 42 was identical to Thermos-I's, which was shown in Figure 33 on page 136. Thermos-I's hierarchy essentially maintained the specialist hierarchy established in Epox-I. Instead of the Epoxy specialist, Thermos-I and CMatDesign had a more general Polymer specialist. An additional sub-specialist to the Curing Agent specialist was also added to deal with Cure Agents for Non-Epoxies. These pieces were incorporated into the existing Epox-I structure. This addition was facilitated by the routine design framework within which these problem solvers were built.

Agent Hierarchy. Besides specialists, routine design includes several other knowledge constructs (selectors, sponsors, plans, constraints, tasks and steps). Together with specialists, these knowledge constructs can be collectively called agents. Therefore, an agent hierarchy shows every knowledge construct in a system. As the complete agent hierarchy for CMatDesign is quite complex and unwieldy to present, Figure 43 shows only the Curing Agent portion of the complete hierarchy. The shaded nodes in Figure 43 correspond to those in Figure 42.

A great benefit of using a routine design approach is that it allows modification and adaptation of previous systems; that capability is also a potential pitfall, and one into which CMatDesign fell. The problem with reusing a previous system's knowledge structure is that if the design problem in the new system is significantly different the previous structure will not be appropriate. This is exactly what happened with the transition from Epox-I to Thermos-I to CMatDesign. The epoxy curing agent section of the agent hierarchy was explicitly defined from its creation in Epox-I. Curing agents for the non-epoxy polymers were handled in very simple knowledge structures. The detail that existed for the epoxy curing agents is clearly superfluous, especially given the limited choices of curing agent for epoxies.

Many nodes shown in Figure 43 can be eliminated by strategically incorporating the knowledge in them into more complicated nodes. In fact, the entire decomposition of the curing/co-cure agent determination used in CMatDesign may not have been the best way to address the design problem. As will be seen in the following section, redundant knowledge exists in the nodes for the curing/co-cure agent and the polymer. It would

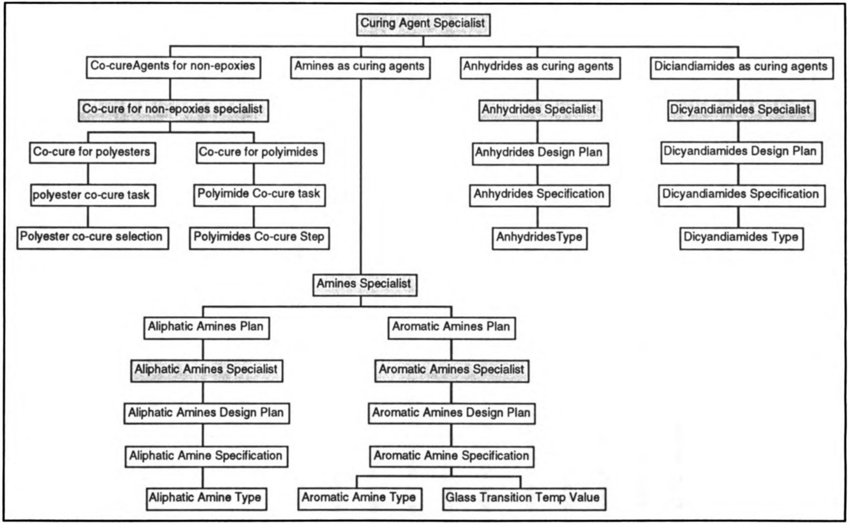

Figure 43. CMatDesign's curing/co-cure agent hierarchy.

significantly simplify the overall agent structure if this knowledge could be used just once, instead of the multiple times it was used in CMatDesign.

7.3 Inferencing Procedures

The final portion of the analysis in this chapter looks at the inferencing procedures in CMatDesign. These inferencing procedures were dependent upon the knowledge structures and the design representation, both of which have already been examined. An examination of the details behind CMatDesign's inferences completes the evaluation required to learn the guidelines for an expansion to the next generation material designer.

The analysis of the inferences in CMatDesign includes two main aspects: first, a dependency analysis that looks at the impact of input variables on the generated output; second, a mapping of specific decision procedures through the agent hierarchy. The former of these two analyses is on a general level, while the latter is at a more detailed level. Variables are only considered by name, with no regard for their values, in the dependency analysis. The mapping & chaining analysis deals with both the variables and their values.

7.3.1 Overall Variable Dependencies

The results of the dependency analysis for CMatDesign can be seen in Table 7. The dependencies of the three different types of variables used within CMatDesign (output, internal, and input) are presented, with the dependent variables in the columns. Within each of these variable types, those variables with the greatest influence are listed first.

Table 7: Overall CMatDesign variable dependencies.

		Dependent Variables									
		CCCA	PM	Glass Transition Temperature	Tensile Modulus	PT	FT	Post-cure Time	Cure Temperature	Post-cure Temperature	Cure Time
Output	Curing/Co-cure Agent (CCCA)			✗				✗	✗	✗	✗
	Polymer Material (PM)	✗				✗					
	Fiber Type (FT)				✗						
Internal	Polymer Type (PT)	✗		✗							
Input	Application Type	✗	✗	✗	✗	✗	✗				
	Corrosion	✗	✗	✗							
	Use Temperature		✗			✗					
	Required Tensile Modulus				✗		✗				
	Maximum Tensile Modulus						✗				
	Humidity		✗								
	Rigidity		✗								
	Required Glass Transition Temperature	✗									
	Flame Retardance	✗									

Dependencies are inherited, i.e., a variable dependent upon another variable includes that variable's dependencies. For example, Glass Transition Temperature is shown to be dependent upon both Polymer Type and Curing/Co-cure Agent. The T_g was directly dependent only upon the Curing/Co-cure Agent, but since Curing/Co-cure Agent was dependent upon the Polymer Type, Glass Transition Temperature inherited that dependency.

Several observations can be made from the information collected in Table 7. Two of the most significant of these observations concern the dependents of Curing/Co-cure Agent and Application Type. Another minor observation concerns the limited use of the variable Maximum Tensile Modulus. As previously mentioned (see footnote on page 168) this variable existed in only one limiting table. Although Table 7 shows that Fiber Type was dependent upon it, Maximum Tensile Modulus limited the choice of only carbon fibers.

Much was made earlier about the disjointedness of the design problem as represented in CMatDesign, and especially the inclusion of processing information. The four processing elements of the output design (Post-cure Time, Cure Temperature, Post-cure Temperature, and Cure Time) were dependent only upon the Curing/Co-cure Agent variable. An examination of the matching tables used to determine these variables shows simple look-up tables. For every Curing/Co-cure Agent, there was a corresponding time or temperature. There was no dependence on any other portion of the design, or any other input variable. This implies that the processing information as determined within CMatDesign was disconnected from the design problem. Most likely, information should not have been included within the confines of a material design system. This illustrates the negative possibilities mentioned in Chapter Six about the need to separate design tasks.

The variable Application Type was very influential in CMatDesign, as every material related aspect of the generated designs (Polymer Material, Curing/Co-cure Agent, Fiber Type) was dependent upon it. However, the ambiguity present in this variable made the specific impact on each variable nebulous. It is disconcerting that such a large portion of the generated design was contingent upon such an enigmatic variable. Specific details this variable's use are shown in the following section.

7.3.2 Variable Chaining & Value Assignation

Again, as for the discussion of the agent hierarchy, the extensive nature of the knowledge in CMatDesign precludes a comprehensive presentation of this analysis. Therefore, only a representative portion of the chaining analysis is presented here.

Table 8 shows the specific variable flow and step level reasoning for the determination of the polymer material in CMatDesign. As presented in this table, the inference chaining is delineated by the type of polymer; polyesters, epoxies, and polyimides each have characteristic inference chains. The second and third columns in Table 8 show specific values for and combinations of variables at the plan and step level, respectively.[1] Combinations of variable values at the step level (column three) indicate a particular polymer material in column four.

Polyesters, for example, must have Application Type = Domestic Application at the plan level for that type of polymer to be evaluated further. At the step level, the distinction between the two possible polyesters is made based on Rigidity and Application Type. Notice here, the redundancy that exists between the plan and step level knowledge. The same

1. To increase the visibility of the information in the table, the variable names are not completely specified. They can, however, be easily inferred from the valid values listed.

Table 8: Specific polymer material input variable inferencing.

Polymer Type	Plan Level	Step Level	Polymer Material
Polyester	= Domestic Application	? Rigidity = Domestic Application	Orthophthalic Acid
		= No Rigidity = Domestic Application	Maleic Acid
Epoxy	~=Domestic Application <=175 Use Temperature	<=100 Use Temperature = Adhesive Application = No Humidity = No Corrosion	Hydroxyphenyl
		<=100 Use Temperature ~=Domestic Application ? Humidity = No Corrosion	
		? Corrosion = No Humidity ~=Domestic Application <=100 Use Temperature	Epoxidized Phenolic Novolac
		= No Corrosion = No Humidity ~=Domestic Application <=175 Use Temperature	DGEBA
Polyimide	<=350 Use Temperature ~=Domestic Application ~=Civil Application	~=Domestic Application <=245 Use Temperature	BMI
			PMR-15

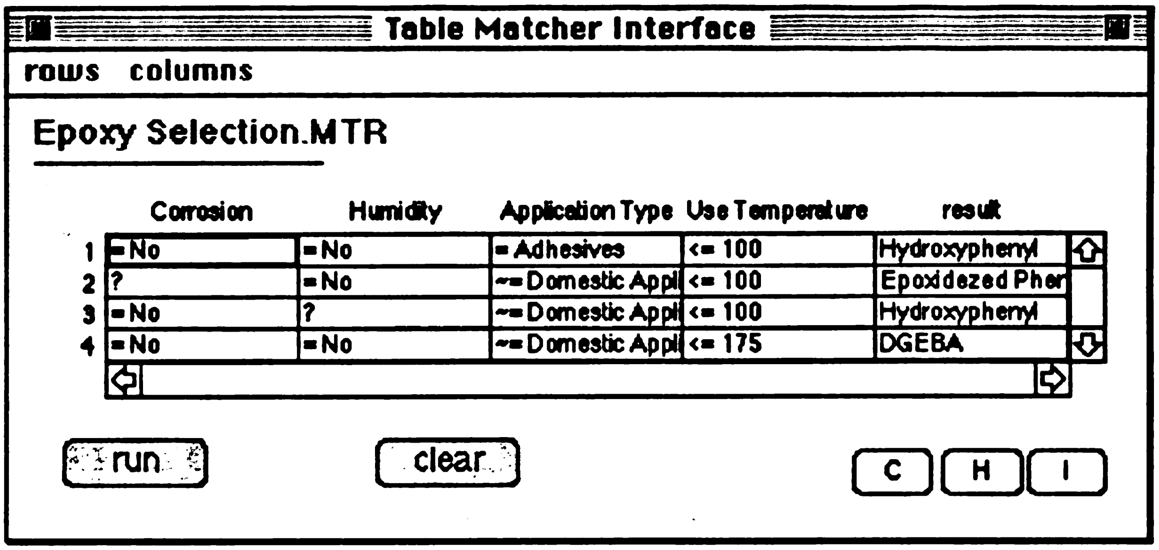

Figure 44. Epoxy structured pattern matching table in CMatDesign.

values of Application Type were used twice. Nothing was gained (other than an increase in computational effort) by matching against the Application Type at the step level. This variable should be eliminated from the step level matchers for polyesters.

The step level inferencing shown in Table 8 for epoxies can be seen as encoded within CMatDesign in Figure 44. This shows the structured pattern matcher for choosing a particular epoxy. Notice that Hydroxyphenyl appeared twice within this matcher. Most likely, this was an error; the second instance (row three) should probably have been TGMDA.[1]

These examples of redundancy and incorrect knowledge within CMatDesign are indicative of a pervasive inchoateness that has crept into the routine design approach to composite material design. This has been fostered by the ingenuous reuse of existing knowledge representations and structures. To properly address the design of composite

1. Recall the discussion in "Domain Coverage" on page 160, and specifically Table 4 on page 161, where TGMDA was shown to be included in the database but not used elsewhere.

materials, any reused knowledge must be carefully examined to decide its applicability within any new design system. Given the abundance of minute inconsistencies found in CMatDesign, the best approach for a new design system would be to start from scratch. This does not mean, however, that the lessons learned from the analysis of CMatDesign and its predecessors should be ignored.

DESIGNING COMPOSITE MATERIAL SYSTEMS USING GENERIC TASKS AND CASE-BASED REASONING

VOLUME TWO

By

Timothy Joel Lenz

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Chemical Engineering

1997

CHAPTER 8

COMADE: A Composite Material System Designer

One of the stated goals of this dissertation is to develop 'persuasive paradigms' for the design of composite material systems. These paradigms are intended to augment the capabilities of composite designers by providing abstract design guidelines and implementations of these guidelines with which to design composite material systems. COMADE fulfills both intents. Not only does COMADE contain design knowledge about composite material systems, but it organizes that knowledge to be useful for design. This accumulation and organization of knowledge addresses both the ill-structured and open-ended nature of the composite material system design process.

It has been well documented that the early conceptual design phase for polymer composites fixes many critical cost drivers for the detailed design of the final composite. Providing support for the conceptual design of polymer composites therefore has great potential for providing substantial downstream cost control. However, the efficacy of rigid phase level design process as a way of thinking about composite material system design is not certain. Within COMADE, the emphasis is on handling material design at the appropriate level of detail, regardless of phase level design.

Although the composites designer has an overwhelming amount of information available, this information is typically not easily used. Additionally, given the vast database of potential polymers and reinforcing fibers from which a composite material may be fabricated, the probability of any one person having complete or even representative knowledge of the entire domain is quite low. It is more likely that the knowledge of the engineer is restricted to selected portions of the domain. Some may be aware of a substantial amount of thermosets, while others may be primarily informed about thermoplastics.

When given a set of material requirements, the engineer will concentrate on those materials that are most familiar, and in the process may not even consider materials that may be superior to those originally considered. COMADE provides a focus for composite material system design, while also presenting possibilities for families of composite material systems that may not be immediately obvious.

An enumeration of COMADE's domain coverage sets the stage for the reconsidered polymer composite material system design problem. Both the knowledge structures and the specific inferencing procedures used within COMADE are based upon this domain coverage. The details of how COMADE solves the material system design problem are presented by a thorough examination of the knowledge representation and organization. How the heuristics comprising the knowledge used to make the decisions within COMADE are generated is also discussed. The chapter continues with a comparison of the designs generated by COMADE with those of the previous systems, and concludes with an eye toward extending the capabilities of COMADE.

8.1 Domain Coverage

The initial mandate for the development of COMADE was to cover a materials domain of 10 fibers, two to five more thermosets than CMatDesign, and a total of five to ten thermoplastics. The domain coverage in COMADE greatly exceeds these expectations. The following discussion details the polymer composite material system domain coverage within COMADE. Table 9 shows all of the material systems included within COMADE.

CMatDesign was structured so that distinctions were made between polyesters, epoxies and polyimides. These three categories of thermosets remain within COMADE. The polymer material options within each of these have been modified to give a more comprehensive representation of thermosets currently used in industry. Two additional material distinctions are also included (phenolics and bismaleimides), for a total of five thermoset material classes. The thermoset coverage in COMADE represents an approximate two-fold increase in thermoset resin choices over CMatDesign.

The addition of thermoplastics to the material designer presents the opportunity to explore a much larger domain of composite materials. The distinction is made in COMADE between amorphous and semicrystalline thermoplastics, and also blends of the two. Materials included within these categories include both common engineering and specialty thermoplastics. COMADE includes a total of 15 thermoplastics.

Depending on the specific polymer, additional chemical agents (reactive diluents, modifiers, hardeners, etc.) may also be required. Thermoplastics are assumed to have no additional chemical agents, while thermosets may or may not have associated chemical agents. Whether a thermoset matrix material has a chemical agent associated with it depends upon both the use specifications and the matrix material itself. All told,

Table 9: The 1188 material systems included within COMADE.

Matrix Material		ChemicalAgent	FiberType	FiberLength
Semicrystalline Thermoplastic	PBT, PA66, PET, PPO, PPS, PEEK	None (TP matrix)	No fiber	N/A
Amorphous Thermoplastic	ABS, PC, PSU, PAS, PEI, PAI		E-glass S-glass C-glass Quartz AS-4 carbon P-55 graphite P-100 graphite Boron SiC	Chopped
Blend Thermoplastic	ABS/PBT PC/PBT PC/ABS			
Epoxies	DGEBA Epox Phenolic Novolac TGMDA TGETPE	DEAPA Dicyandiamide MPDA Clorendic anhydride DDS/BF_3·MEA		
Phenolics	Phenolic novolac	HMTA		
	Phenolic resole	None (heat cure)		
Polyesters	Orthophthalic resin Isophthalic resin BPA fumarate resin Chlorendic resin	Vinyl toluene Styrene	E-glass S-glass C-glass Quartz AS-4 carbon P-55 graphite P-100 graphite Kevlar-29 Kevlar-49 Kevlar-149 Boron SiC	Continuous
	Vinyl ester resin	DAP		
Bismaleimides	4,4'-MDA-BMI	Dicyanate BPA 0,0'-Diallyl BPA MABA 1,4-Tolyl BMI		
Polyimides	PMR15 oligomers	Methanol		
	Thermid600 oligomers	Ethanol		

COMADE's database of 15 thermoplastics and 39 thermosets gives a total of 54 possible matrix material/chemical agent combinations.

The other component of the polymer composite material system is the fiber reinforcement. Distinctions for the type of fiber material are made between glasses, carbons/graphites, aramids and specialty fibers. The length of the fiber is also specified, with continuous and chopped as the two options. However, not every fiber material is available in both forms. Aramid fibers are never used in chopped form, mainly because the fibers are much too difficult to chop. With a choice of twelve fiber materials and a neat resin (no fiber) option, and the possibility of chopped fibers, this gives 22 possible fiber material/length combinations for COMADE to use. This represents a 140% increase in fiber material selection from CMatDesign, with a choice of eleven fiber materials and a neat resin (no fiber) option.

The expanded polymer composite material system domain coverage allows for COMADE to choose from up to 1188 possible material designs. Taken completely, the expanded domain coverage within COMADE represents an order of magnitude increase in the domain coverage over CMatDesign.

The data used within COMADE for the determination of applicable material designs has been drawn from various public domain sources [Carlsson, 1991; Phillips, 1989; ASM International, 1987; Strong, 1989], and where possible avoids including specific brand names or product formulations. Instead, generic types of materials have been used to enable COMADE to generate material systems that give viable solutions to the use requirements. The prime motivation for many selected materials was to obtain as wide a range of potential performance properties as possible within COMADE.

This expansion of the material domain coverage is accompanied by a fresh examination of the required knowledge representation and structuring. This examination is done with the goal of capturing the design of polymer composite material systems as aptly as possible.

8.2 Knowledge Representation & Organization

The use of domain knowledge within a routine design system is multi-faceted. Knowledge about the domain coverage is used to decide what variables are to be used to figure out the design solution. Knowledge is used to decompose the design problem into appropriately solvable subproblems. Abstract and general knowledge is used to populate the high level nodes within the design decomposition hierarchy and provide navigational guidance. Specific knowledge is used at the step level to assign values to variables. How these varying types of knowledge are determined and defined directly affects the capabilities of a routine design problem solver such as COMADE. Only if the design problem is accurately represented and the heuristics are meticulously generated from experiences and experiments will the knowledge included within the system be sound. This is the guiding vision behind the construction of COMADE.

The discussion in this section looks at how COMADE represented each of these three aspects. An examination of how the design problem is represented within COMADE immediately follows. Variables used to make decisions about the constituents of the design problem are also presented. Finally, the decomposition of the design problem and the resulting knowledge organization is discussed.

8.2.1 Design Problem Representation

The analysis of CMatDesign in the previous chapter pointed out the importance of carefully considering the design problem to be solved. By not reconsidering the design problem and instead simply copying the design problem from its predecessor, CMatDesign lost its focus. That oversight has been eliminated for COMADE.

The design problem has been carefully constructed to accurately describe a polymer composite material system. There is no processing information included in this description, as that information has been relegated to the process design portion of the overall composites design vision. Figure 45 shows the design problem as represented within COMADE. The details of the output variables MatrixMaterial, ChemicalAgent, FiberType, and FiberLength are shown in Table 9.

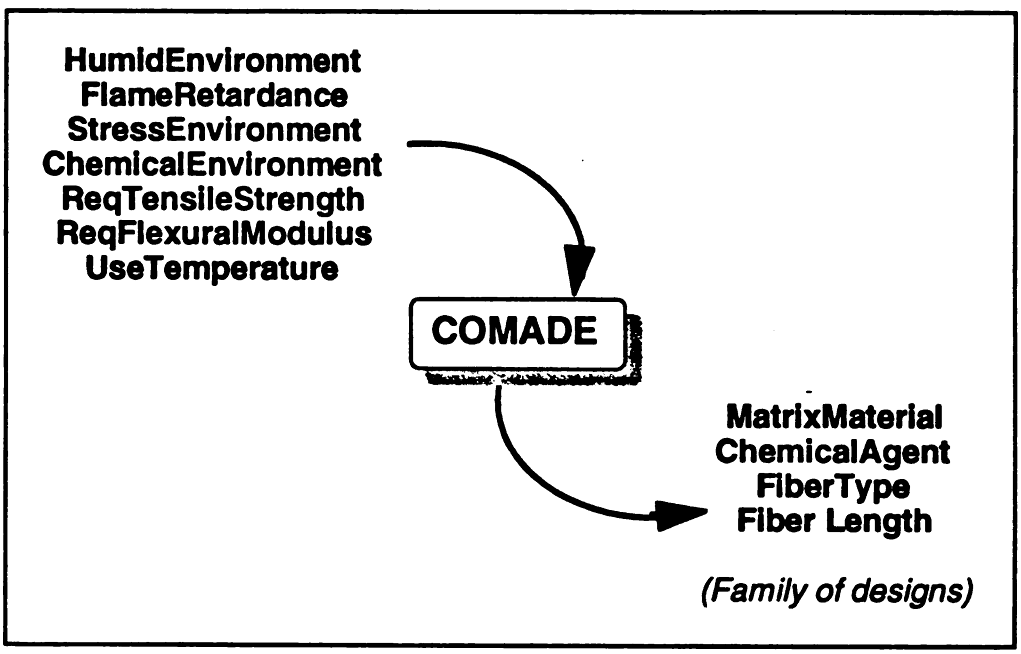

Figure 45. Input-output behavior of COMADE.

8.2.2 Input Variables

With the new design problem representation and expanded domain coverage in COMADE, the variables used to guide the design of composite material systems had to be considered carefully. Many properties can be considered in the design of composites, including mechanical, thermal, electrical, optical, and chemical properties. The variables used to design the composite material systems in COMADE were specified in part by the availability of sufficient data in the literature for the materials included in the domain coverage mentioned previously.

Variables included in the input set for COMADE addressed thermal properties (Use Temperature and Flame Retardance), mechanical properties (Stress Environment, Required Flexural Modulus, and Required Tensile Strength), and environmental properties (Chemical Environment and Humid Environment). Table 10 shows these variables.

Several of the variables shown in Table 10, while similar in name to variables used in CMatDesign, are different enough to merit further discussion. These variables include ChemicalEnvironment, UseTemperature, and StressEnvironment.

ChemicalEnvironment. A lesson learned from CMatDesign's use of the variable Corrosion was that the nature of the chemical environment must be more detailed. Additionally, thermoplastics are now possible matrix material choices in COMADE. More information is required when choosing thermoplastics than is needed for choosing thermosets. Specifically, an explicitly defined chemical environment becomes important in making distinctions between amorphous thermoplastics, which by their very nature are more susceptible to chemical attack than other materials. Details regarding the resistance of a polymer to a particular type of chemical is then a necessity. Therefore, use must be

Table 10: Input variables used in COMADE.

Variable Name	Variable Type	Possible Values
Chemical Environment	OneOf	none hydrocarbon weak acid strong acid weak base strong base organic solvent
Flame Retardance	Yes/No	yes no
Humid Environment	Yes/No	yes no
Use Temperature	Numerical	-40 °C to 300°C
Stress Environment	OneOf	none constant cyclic increasing
Required Flexural Modulus	Numerical	0 GPa to 420 GPa
Required Tensile Strength	Numerical	0 MPa to 2800 MPa

made of a comprehensive variable that can give a choice of potential chemical environments. ChemicalEnvironment, as used in COMADE, is such a variable.

UseTemperature. Recall the discussion in the previous chapter about the overlap that existed in CMatDesign between the Use Temperature and Required Glass Transition Temperature. All of the knowledge expressed in terms of one of these could have been expressed in terms of the other. In COMADE, the definition of UseTemperature has been expanded to include both such considerations, and also other thermal properties. See the following section ("Use Temperature: A Concisely Comprehensive Thermal Property" on page 197) for a detailed discussion of how this was done.

StressEnvironment. The mechanical loads to which the polymer composite material is exposed are also influential in the determination of material choice. Take a brittle phenolic material, for example. It would be inappropriate to use a composite material fabricated from such a brittle material in a dynamic loading environment, as unwanted part fatigue and/or failure of the composite could occur. Therefore, an additional input to specify the mechanical loading on the polymer composite is required. In COMADE, this is addressed with the variable StressEnvironment. This variable allows for a distinction between constant, cyclic, ramped and no mechanical loading.

On a related note, a numerical flexural modulus (ReqFlexuralModulus) and a numerical tensile strength (ReqTensileStrength) are also employed within COMADE. These present the mechanical properties of the composite material system in more quantifiable format than did CMatDesign. A numerical variable such as ReqFlexuralModulus presents a richer expression of the flexibility of the composite than the Yes/No variable (Rigidity) employed within CMatDesign.[1]

8.2.3 Knowledge Structures

The design problem representation and the set of input variables presented above provided the foundation for COMADE. Unlike the development for the previous systems, the implementation of COMADE was specifically driven by the new design problem representation. The knowledge structures comprising COMADE were developed to accurately capture the decomposition of the design problem.

1. Along with these numerical variables come certain difficulties. The most notable is that when COMADE was implemented, the ISL toolset version of routine design could not perform numerical calculations. It would then seem foolish to include numerical variables within the input set. However, the heuristics used within COMADE have circumvented this problem. See "Approximate Mechanical Properties of Composites" on page 201.

As was done for the discussion of CMatDesign's knowledge structures, COMADE's structures can be considered at two levels of detail. Again, as COMADE was a routine design problem solver, these two levels of detail can be expressed by the specialist and agent hierarchies, respectively.

Specialist Hierarchy. Design problem solving in routine design is based on the hierarchical structure of design specialists, each of which offers a collection of expertise or knowledge on how to complete part of the design problem. This hierarchical organization implies that the design problem is decomposable or nearly decomposable into appropriate subproblems. The specific arrangement of the specialists depends on the regularities that exist in a particular domain. The decomposition of the design problem for COMADE is significantly more streamlined than its predecessors'. Figure 46 shows the specialist hierarchy used in COMADE.

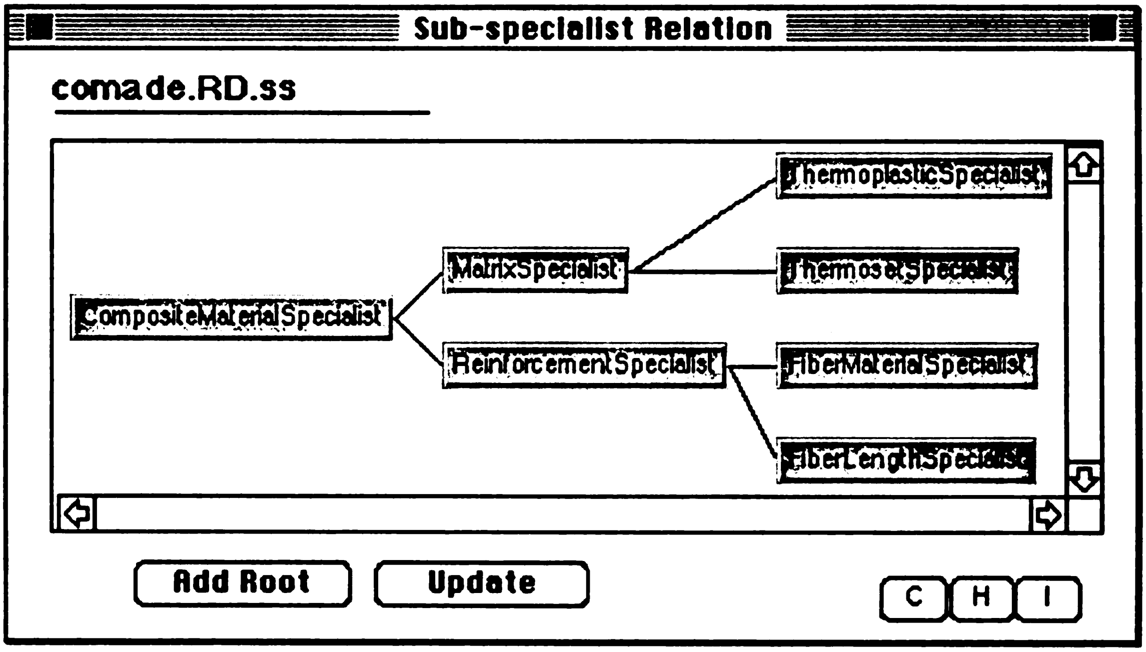

Figure 46. Specialist hierarchy for COMADE.

A lesson learned from the critique of CMatDesign was the importance of matching the design decomposition to the design problem. This must be done to properly address the design of composite material systems. Given the abundance of minute inconsistencies found in CMatDesign, the specialist hierarchy in COMADE was generated from scratch. There are, however, similarities between the hierarchy shown in Figure 46 and that used in CMatDesign (Figure 42 on page 172). These are to be expected, as the design problem still addresses composite materials.

The specialist hierarchy in COMADE is much simpler than the hierarchy in CMatDesign. This is direct result of using COMADE's design problem as the foundation for the hierarchy construction. COMADE used only seven specialists to capture the decomposition of the composite material system design problem, while CMatDesign used fourteen. With this decrease in the number of specialists comes a decrease in computational complexity.

Notice that the hierarchy for COMADE does not include any mention of curing agents or cure conditions. The cure conditions have been totally eliminated from COMADE's design problem, and therefore need not be considered within the specialist hierarchy. The curing agents have not been eliminated from the design problem, but have instead been incorporated into the agents hierarchy, a more appropriate use of the routine design methodology. This was done to eliminate redundant design knowledge within the agent hierarchy.

Agent Hierarchy. Besides specialists, routine design includes several other knowledge constructs. These knowledge constructs (selectors, sponsors, plans, constraints, tasks and steps), with specialists, can be collectively called agents. Therefore, an agent hierarchy shows every knowledge construct in a system. Figure 47 shows the entire decomposition of the design problem in COMADE at the agent level. The shaded nodes in Figure 47 indicate the specialists shown in Figure 46.

Within the agent hierarchy in Figure 47, the overall problem of designing the composite material is decomposed into successively smaller portions, finally reaching a level of detail that can easily be decided. Variable values are specified in routine design at the step level. The agents hierarchy within COMADE is significantly different from that in CMatDesign. This difference can be attributed to three things: a refocusing on the composite material system design problem, an elimination of redundant and inappropriate agents, and the expansion of the materials domain coverage.

This hierarchy describes all of the possible paths that COMADE used to figure out composite material systems. The matrix is handled separately from the reinforcement. Within the matrix portion of the hierarchy, thermoplastics and thermosets are considered separately. This decomposition continues down to the level of different thermosets (e.g., epoxies) and all the way down to the distinction between the specific epoxy and complementary chemical agents. Recall, COMADE has more domain coverage than did CMatDesign. A result of this increased domain coverage is the increased decomposition of the matrix material design problem to distinguish between thermoplastics and thermosets. This addresses the specification of the matrix material in much greater detail than did CMatDesign.

Note that each step for deciding the value of particular type of matrix material is accompanied by a step for deciding the chemical agent. As the chemical agent is dependent on the matrix material, it makes sense to specify it immediately after specifying the

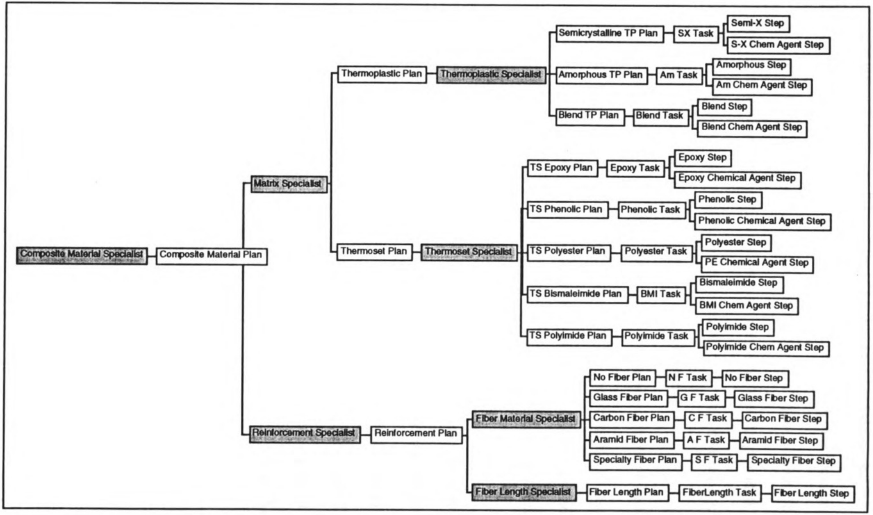

Figure 47. Agents hierarchy for COMADE.

matrix material. This is a marked change from how CMatDesign decided the chemical agent. Recall, CMatDesign had a complex specialist hierarchy for deciding the chemical agent. COMADE's approach to determining the chemical agent is more efficient for two reasons: it eliminates a significant amount of redundant knowledge within the matching tables for the agents, and it eliminates seven specialists (and all of their associated agents - see Figure 43 on page 174) from the overall agents hierarchy.

The reinforcement portion of the hierarchy breaks the design of the fiber into two portions: the specification of a fiber material, and the specification of the fiber length. Within the Fiber Material Specialist, distinctions are made between the various classes of fiber materials included within COMADE's database. This also includes the possibility that the final composite will not contain any reinforcement.

Notice that there is only one specialist for the fiber material (Fiber Material Specialist) in COMADE, unlike CMatDesign which had a specialist for every class of fiber material (Glass Fiber Specialist, and Carbon Fiber Specialist). In COMADE, the applicability of each of the different classes of fiber materials is addressed at the plan level. This is a more appropriate use of routine design agents for expressing the decomposition of a design problem. This has resulted in a more streamlined agent hierarchy within COMADE than in CMatDesign.

8.3 Generating Heuristics from Raw Data

Nearly every node within the agent hierarchy shown in Figure 47 contains knowledge for either navigating the hierarchy or deciding the value of a variable. This knowledge is almost entirely heuristic in nature. Guidelines for deciding, say, when semicrystalline thermoplastics are appropriate for use, have been abstracted from data available in the literature. Such heuristics have been generated for every input variable used in COMADE.

This section discusses two examples of how heuristics were generated from raw data. The first example looks at the variable UseTemperature and how the values used for it have been abstracted from the mass of available thermal properties. The second example presents assumptions for the mechanical properties of the final composite, discusses how they were generated, and why they were used.

8.3.1 Use Temperature: A Concisely Comprehensive Thermal Property

Figure 48 shows every value of the variable UseTemperature used within COMADE. This combines all of the heuristics (at least the portions relevant to UseTemperature) used to decide the MatrixMaterial and ChemicalAgent throughout COMADE. The heuristics exist at the plan sponsor level all the way down to the step level.

The various shaded lobes in Figure 48 represent the values of UseTemperature embedded within COMADE for the different material types. Within each lobe, the inflection points indicate the value used for a specific instance of that material type. For instance, the lobe at between six and seven o'clock corresponds to the semicrystalline thermoplastics included within COMADE.

The upper limit of the use temperature increases in a clockwise progression. Polyesters have the lowest temperatures, and polyimides have the highest. The material types

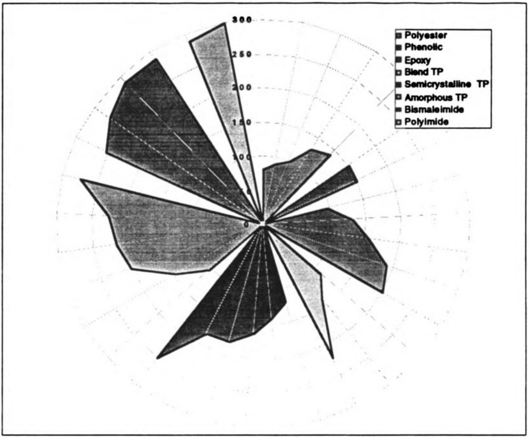

Figure 48. Use temperatures for every matrix material in COMADE.

as listed in the legend follow a clockwise progression: polyesters are between twelve and one o'clock, followed by phenolics at two o'clock, and continuing until the polyimides at eleven o'clock.

When picking a polymer, the temperature at which the material will be used must be considered. If the continuous use temperature is above the glass transition temperature (T_g) of the material, problems such as creep are exacerbated and the part may fail completely. The T_g requirement is implicitly incorporated into the variable UseTemperature by specifying upper use temperature limits for materials at or near the T_g for amorphous

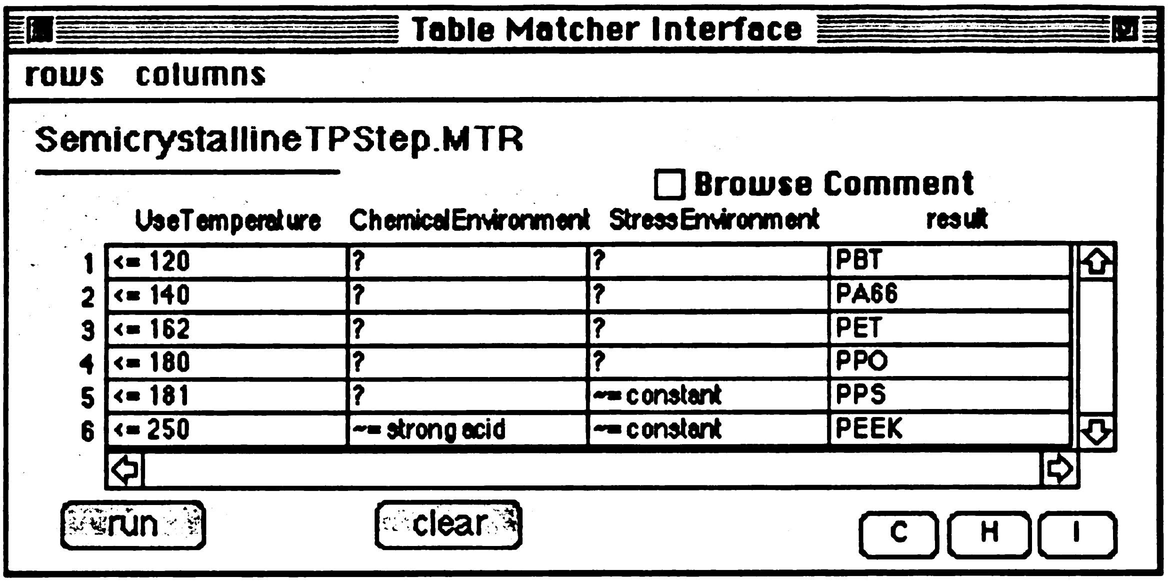

Figure 49. UseTemperature as used in COMADE for semicrystalline thermoplastics.

materials (thermosets and amorphous thermoplastics) and above the T_g (but below the T_m) for semicrystalline thermoplastics.

The upper limits are explicitly encoded within the matching tables for each material. Figure 49 shows the matcher used to choose semicrystalline thermoplastics. The first column shows the values of UseTemperature that decide a semicrystalline thermoplastic.

The values used in column one of Figure 49, as mentioned above, represent a combination of thermal property data. Included within these values are data for the melting temperature, the glass transition temperature, the heat deflection temperature, and where available, Underwriters Lab's recommended use temperature. Table 11 shows the raw data for the thermal properties of the six semicrystalline thermoplastics in COMADE used to determine the values of UseTemperature in Figure 49.[1]

1. The raw data in Table 11 was gathered from various sources in the literature, including: [ASM International, 1987], [Carlsson, 1991], [Phillips, 1989], and [Strong, 1989]. More detailed references of the data are given in Appendix D.

Table 11: Thermal properties of semicrystalline thermoplastics.

Matrix	T_{melt} (°C)	T_{glass} (°C)	HDT (°C)	UL (°C)	Use Temp (°C)
PBT	**234**	**29**	**68**	**120**	**120**
	240	17	50		
	228	40	85		
Nylon 6,6	**266**	**48**	**83**	**130**	**140**
	267	45	75		
	265	50	90		
PET	**268**	**73**	**41**	**140**	**162**
	270	69	41		
	265	70	--		
	--	80	--		
PPO	**286**	**85**	**100**	**N/A**	**180**
	286	85	100		
PPS	**282**	**89**	**135**	**N/A**	**181**
	288	85	135		
	273	88	--		
	285	93	--		
	--	90	--		
PEEK	**339**	**144**	**165**	**250**	**250**
	340	144	165		
	335	143	--		
	343	--	--		

For each semicrystalline thermoplastic listed in Table 11, there are several numbers given for each thermal property. The numbers in bold immediately across from the abbreviation for the material are the averages of the raw data from the literature directly below. By using the average values of multiple material formulations, COMADE avoids using any particular brand of material. This gives a better feel for the general capabilities of the materials instead of concentrating on the exotic formulations commonly used in industry.

Figure 50 graphically shows the averaged thermal properties from Table 11. This gives a visual comparison of the thermal properties for semicrystalline thermoplastics, and illustrates how the UseTemperature values fall between the T_m and T_g & HDT of the materials. Notice that the values for UseTemperature are similar to or slightly higher than the Underwriters' Lab recommended use temperatures.

The combination of four thermal properties into just one variable is an example of how heuristics can be used to effectively consider many different concerns. The variable UseTemperature as used in COMADE encompasses many such concerns. Not only does it include abstractions of thermal properties, but it also includes interpolations from those abstractions. UseTemperature exemplifies the richness of the design heuristics incorporated within COMADE. It also shows the power of using heuristics to guide decision making.

8.3.2 Approximate Mechanical Properties of Composites

The use of the variables ReqFlexuralModulus and ReqTensileStrength within COMADE was strongly driven by heuristics, the details of which are not immediately apparent in the matching tables of COMADE. The main reason for using heuristics with

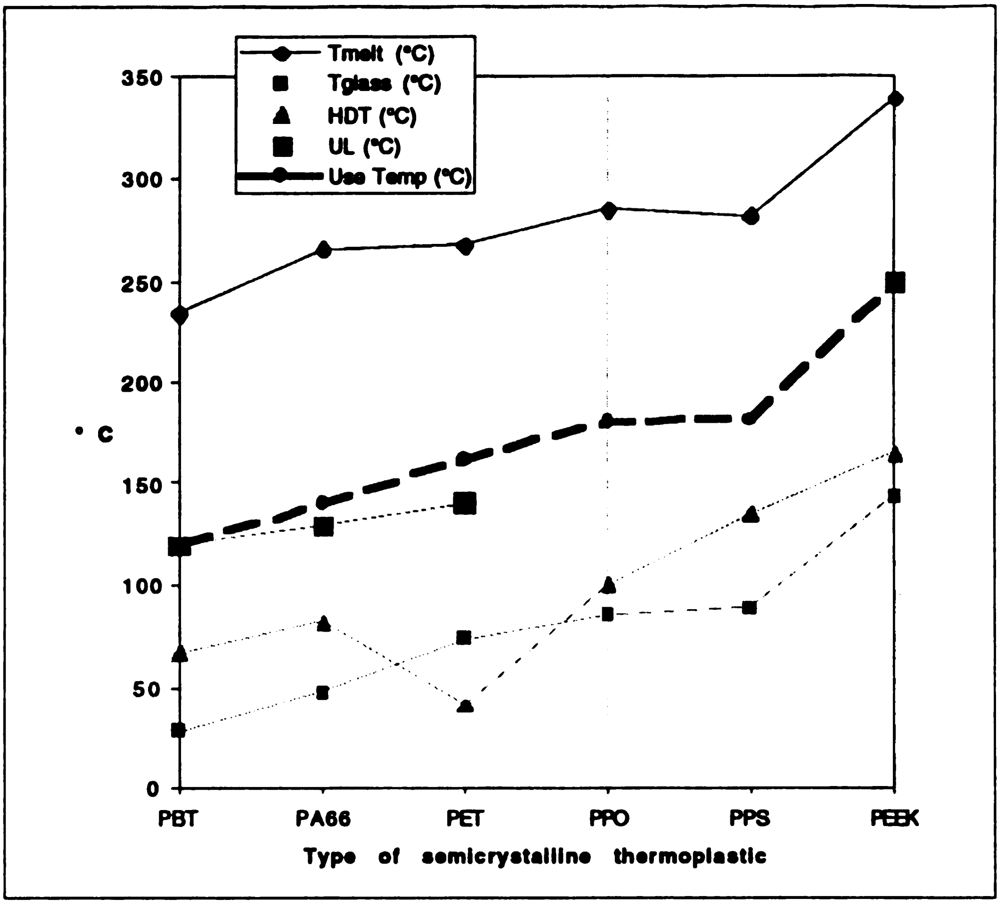

Figure 50. Thermal properties of the semicrystalline thermoplastics in COMADE.

these variables was the incapability of the ISL generic task toolset to do any numerical calculations. That capability has since been added to the toolset. However, at the time COMADE was built, another way to effectively consider the mechanical properties was needed. The following discussion details how COMADE decides the mechanical properties of the final composite without numerical calculations.[1]

As encoded within COMADE, the MatrixMaterial does not affect the mechanical properties of the final composite. Since COMADE could not do numerical calculations, such a varying contribution was not permissible. Another way to account for the

1. A detailed analysis of how well this data abstraction works is given in Chapter 12.

contribution of the MatrixMaterial to the mechanical properties of the final composite had to found. An upper bounding contribution from the MatrixMaterial to the mechanical properties was used. The estimated upper limits for neat resin properties for both the tensile strength and the flexural modulus are shown in Table 12.

Table 12: Estimated final composite material properties versus fiber type.

Fiber Type	Tensile Strength (MPa)		Flexural Modulus (GPa)	
	Chopped	Continuous	Chopped	Continuous
No Fiber	50[a]		5[a]	
E-glass	169	2110	5	46
S-glass	224	2800	5	50
C-glass	163	2040	4	42
Quartz	170	2120	4	39
AS-4 carbon	196	2450	14	135
P-55 graphite	87	1090	22	220
P-100 graphite	111	1390	42	420
Kevlar-29	--	2210	--	52
Kevlar-49	--	2210	--	77
Kevlar-149	--	2090	--	113
Boron	177	2210	23	230
SiC	191	2390	24	240

a. These are the estimated upper limits for neat resin properties in COMADE.

The values for the composite material properties shown in Table 12 have several assumptions buried within them. Besides the estimated contribution from the matrix, assumptions were made about the architecture of the fibers in the composite. The assumption of continuous and unidirectional fiber reinforcement forms the basis of the additional assumptions. A volume fraction of 60% was assumed for the continuous fibers. This is a typical proportion for fiber reinforced polymer composites. Furthermore, a rule-of-mixtures assumption was used to predict the properties of the composite.

The mechanical properties for the composite were calculated using the rule-of-mixtures with the estimated matrix contribution and a 60% unidirectional continuous fiber reinforcement. The simple rule-of-mixtures formula used was [Phillips, 1989]:

$$CompositeProperty = (Property_{matrix})(0.4) + (Property_{fiber})(0.6)B$$

where: B = 1 for unidirectional continuous reinforcements.[1]

The mechanical properties for the composite with chopped reinforcements were not determined using this formula. Instead, a correlation between the relative values of moduli and strengths for different lengths of reinforcements was used to predict at the values for the chopped fibers.[2] Table 13 shows the correlation between the mechanical properties and reinforcement lengths.

Table 13: Relative property values versus reinforcement length.

	Moduli	Tensile Strength	Flexural Strength
Neat	1	1	1
Short	3	2	2
Continuous	30	25	10

1. Although not used for the calculation of mechanical properties in COMADE, B = 1/2 for bidirectional continuous fibers and = 3/8 for random in-plane chopped fibers.
2. This is taken from [Carlsson, 1991]. Even though this correlation was for thermoplastics, it was also used for determining thermoset values.

This correlation was used instead of the rule-of-mixtures due to a lack of reliable data in the literature for the mechanical properties of short fibers. Despite this, COMADE could predict the properties of composites with chopped fibers by using the relative property correlations in Table 13.

Therefore, the tensile strength values found for continuous fibers were divided by 12.5 to find the tensile strengths for composites with chopped fibers. Similarly, the values calculated for the flexural modulus with continuous reinforcements were divided by 10 to arrive at the values for the chopped. This correlation also helped to determine the estimated contribution of the matrix mentioned earlier. Flexural strength values were not calculated, as that variable was not considered within COMADE.

Table 14 shows a comparison between the tensile strengths of the fibers and the final composite as calculated with the heuristics in COMADE.[1] This table shows the progression from raw data[2], to the values determined from the rule-of-mixtures calculation for the continuous fiber composites, to the relative properties of the chopped fiber composites.

The heuristics used to determine the mechanical properties of the final composite give a bounding upper limit upon which to base the choice of fiber length. If the required tensile strength and flexural modulus are both less than the determined composite material value estimation, then the fiber material and fiber length combinations meeting those requirements are given as possible design constituents. The assumptions & heuristics used

1. Table 14 shows a portion of the data used to determine the mechanical properties in COMADE. This data was culled from several sources, including: [ASM International, 1987], [Carlsson, 1991], [Phillips, 1989], and [Strong, 1989]. See Appendix D for more detailed references.
2. Actually, the tensile strengths given here are averages of several values found in the literature.

Table 14: Tensile strengths of fibers and fiber-reinforced composites.

Fiber Type	Tensile Strength (MPa): Fiber	Composite: Continuous	Composite: Chopped
No Fiber	–	50	
E-glass	3440	2110	169
S-glass	4580	2800	224
C-glass	3310	2040	163
Quartz	3450	2120	170
AS-4 carbon	4000	2450	196
P-55 graphite	1730	1090	87
P-100 graphite	2240	1390	111
Kevlar-29	3600	2210	–
Kevlar-49	3600	2210	–
Kevlar-149	3400	2090	–
Boron	3180	2210	177
SiC	3900	2390	191

here give an estimate of the composite properties accurate within about 20%. This error is about as close as is possible for a preliminary design.

8.4 An Example from COMADE

The comparison sample problem used in Chapter Five to compare the results of Epox-I, Thermos-I, and CMatDesign was solved with COMADE. As mentioned above, the inputs to COMADE are significantly different from those used previously. Therefore, comparison of COMADE results with those from the previous systems must be done carefully.

The caveat must be that although a common comparison baseline was attempted, the comparison is still not exactly congruous. Realistically, it is still an apples to oranges comparison.

In Epox-I, the original problem specification was to design a composite material for an outdoor structure. The inputs required that the material was to have a glass transition temperature of at least 200°C and a tensile modulus of at least 50 GPa. Thermos-I, while it used the inputs from Epox-I, specified additional inputs that set the upper use temperature at 100°C, did not require resistance to either corrosion or humidity, and did not specify the material's rigidity or flame retardance. CMatDesign's inputs were identical to those of Thermos-I. COMADE's inputs, while similar to these in some respects, were markedly different in others.

Figure 51 shows the comparison problem inputs used with COMADE. The values for COMADE's inputs were as representative of the other systems' inputs as possible. The values for ChemicalEnvironment, FlameRetardance, HumidEnvironment, ReqFlexuralModulus, and UseTemperature were all suggested by the analogous previous variable values. The variables ReqTensileStrength and StressEnvironment had no analogous variables in the previous systems. Therefore, the input values for these variables were chosen in an attempt to capture the spirit of the original design problem.

COMADE generated 592 material system designs, nearly 50 times more than found by CMatDesign (Figure 37 on page 141), and 591 more designs than either Thermos-I (Figure 35 on page 138) or Epox-I (Figure 31 on page 133) had found. Figure 52 shows a sampling of the generated composite material systems. Notice that the sampling

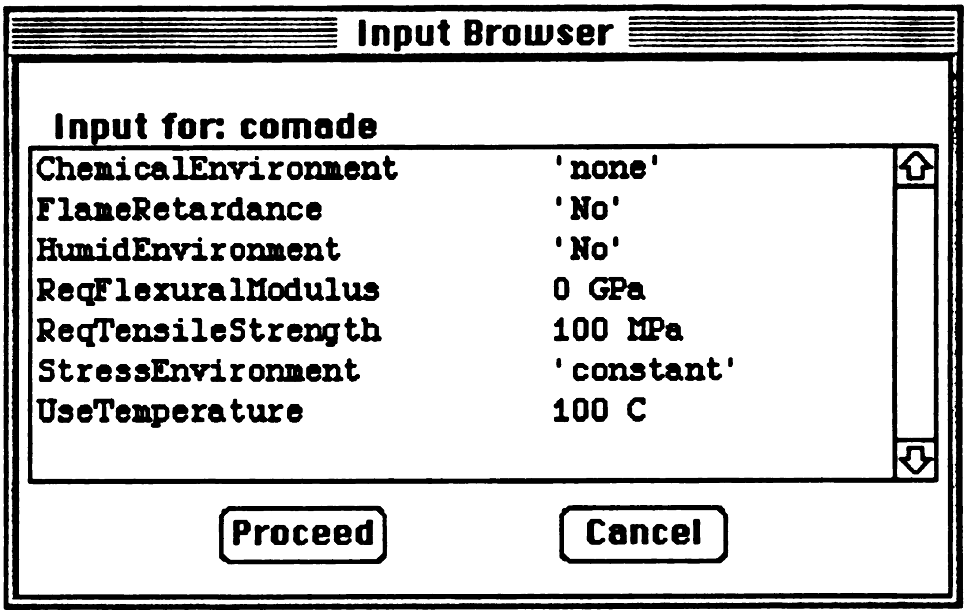

Figure 51. Comparison problem input for COMADE.

of results covers only a small portion of the output. This is depicted by the small box around only the center of the entire results tree on the right.

The materials found suitable for the comparison problem included not only epoxies (as found by all of the previous systems), but also polyesters, polyimides, and various thermoplastics. Notice the highlighted design in Figure 52. This was a design found by the previous systems, modified within COMADE's design representation.

This increase in the number of generated design, while impressive at first glance, is a hindrance. Such a large set of designs is an overload for a user of COMADE. Generating large families of material systems is not the intent of COMADE. Realistically, this output family must be significantly condensed before it can be useful.

The dramatic increase in the number of material system designs found for this example was partially due to the design problem reorganization and the knowledge base expansion in COMADE. Another factor behind this large output family lies in the input

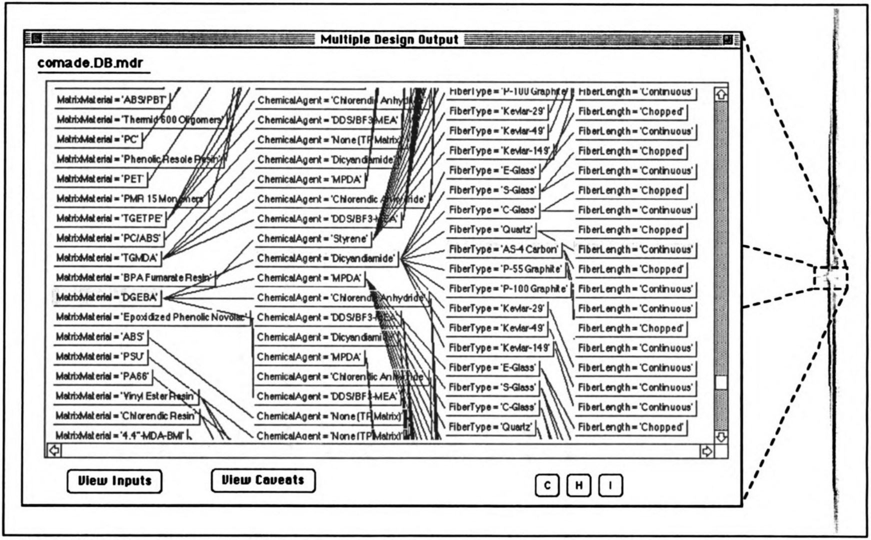

Figure 52. Comparison problem results for COMADE.

set used for COMADE. By setting variables to values used by previous systems, any advantage gained by the reorganization in COMADE was lost. Because of this attempt to force a comparable design problem specification upon COMADE, an unrealistic input set was specified. In this forced input set lies the root of the apples to oranges caveat mentioned earlier.

8.5 Extensions

The capabilities of COMADE's original version have been expanded by several related efforts. An extensive approach that includes the design of the polymer matrix and fiber interphase within COMADE has been explored [Wilenski *et al.*, 1995]. COMADE has been made available over the web for large scale distribution. Also, a case-based reasoning approach has been used to extend the capabilities of COMADE.

The emergence of the web was underway at the time COMADE was originally completed. At the urging of the main funding agency behind COMADE (DARPA), COMADE and CTechSel were made available on the web in June of 1994.[1]

More recently, a case-based reasoning approach to designing composite material systems has been developed. Instead of using a commercial CBR tool, a CBR tool was generated within the context of the current ISL programming tools. A main reason for doing so was to preserve compatibility between the CBR and stand-alone versions of COMADE. This system used the knowledge within COMADE as a source of design

1. This version of COMADE can be reached via the ISL homepage at **http://isl.cps.msu.edu** by following the links for composites research. Also available is the most recent version of the ISL's fabrication technology selector, COFATE.

cases, for the organization & navigation of its case memory, and for the adaptation of retrieved cases.

The details of this new approach to CBR are detailed in the following chapters. Chapter Nine outlines the creation of the programming tool, and Chapter Ten describes how that tool was used with COMADE to implement a CBR system for the design of polymer composite material systems.

CHAPTER 9

CBR within a Generic Task Environment

> *"Now one of the [best] ways of designing something is not to design it at all. Use [a part] that is designed already by someone else."* [Glegg, 1969]

The immediately preceding chapters have shown the development of a knowledge-based approach to the design of composite material systems. This approach has reused design knowledge as heuristics and other abstracted information. A principle of engineering design, that of using assumptions or back-of-the-envelope calculations, has been embodied in the generated approach. Nothing has been said yet, however, about the other guiding principle: use of previous designs.

The reiteration of Gordon Glegg's quote serves to place the focus on the direct reuse of previous composite material system designs. Finding a new solution by modifying a previous design can be accomplished using case-based reasoning. The entire premise of case-based reasoning is the reuse of previous design solutions. Representation and reuse of knowledge serve as the foundation for case-based reasoning.

The previously detailed development of the knowledge-based approach for composite material system design and the subsequent implementation of COMADE were facilitated by using the ISL's generic task toolset. The specific tool most utilized was that for routine design. Unfortunately, the ISL toolset does not contain a tool for case-based reasoning.[1]

There are many commercially available tools for case-based reasoning (e.g., Remind, CBR Express, Kate-CBR, The Easy Reasoner). However, none of the available tools are directly compatible with the tools in the ISL generic task toolset.

Given the design knowledge that exists in COMADE[2], and a mandate to prove the use of generic tasks as a framework for structured knowledge reuse[3], a case-based reasoning methodology compatible with generic tasks was developed. This implementation of a GT-influenced CBR approach used many existing generic task themes, extending the capabilities and definitions of these themes as needed.

This chapter presents the methodologies and computational tools developed for implementing a CBR system within the ISL's generic task toolset. These CBR development tools are intended for use in a specific task, namely, routine design problem solving. Because of this focus, the tools have been implemented specifically to be used in a rigidly structured problem solving environment. The CBR tasks included within the methodological approach developed are the retrieval and subsequent modification of design cases. For

1. Previous work in the ISL [Wallingford, 1992] addressed the development of a CBR system (CRISTA). However, the results of that work were never incorporated into the ISL toolset. This was due in part to the highly specialized nature of CRISTA's implementation.
2. Since the process of design alteration requires explicit knowledge about the effects of changes in the design specification, the existence of the design knowledge in COMADE can serve as a resource for the adaptation portion of the CBR process.
3. An initially proposed component of the overall composites design vision was a case-based reasoner. The inclusion of this CBR component was intended to show that generic task knowledge structures can be used for multiple purposes - that structured knowledge reuse is possible provided an approach with strong organizational principles is used.

an application of these methods to reusing composite material system designs, see the following chapter.

This chapter is divided into three sections, the first of which looks at some general issues involved in combining CBR with generic tasks. It also presents the CBR tasks implemented in the new tools. After that, a new tool for retrieving cases from a case library, CaseFinder, is presented. Finally, this chapter concludes with a discussion of the new design modification tool, DesignMod.

9.1 Combining CBR and Generic Tasks

The idea of combining case-based reasoning with a generic task approach is not new. A generic task description of case-based reasoning (including the subtasks retrieve, reuse, and store) has previously been constructed [Chandrasekaran, 1990]. This description focused on using CBR for the proposal of design solutions only; no explicit task description of case modification was included.[1] In actuality, this task description has proved more beneficial for understanding the mechanisms of CBR itself than inspiring combinations of CBR and generic tasks.

Three issues can be considered for the combination of CBR and generic tasks. The first is the common emphasis each approach places on knowledge representation. Typical modifications to a generic task, say, hierarchical classification, to facilitate its use in CBR

1. This lack of detail about a generic task description for design modification is not at all surprising. Such a description may very well be an impossibility. Design modification is not well understood, and as a result there are many different approaches to performing it. In spite of the confusion, however, a common theme among all the different approaches does exist: the method used for design modification is dictated by the domain to which CBR is applied.

are also considered. Finally, the task decomposition of CBR is revisited and examined considering the CBR/GT approach developed for this dissertation.

9.1.1 Knowledge Representation in CBR and Generic Tasks

A main facet of a generic task approach to problem solving is the existence of a structured knowledge representation. This representation forms the basis for the organization and execution of every type of generic task. CBR also requires a structured representation of knowledge. As mentioned previously, the representation and reuse of knowledge serve as the foundations for CBR.

The knowledge in CBR is represented as cases. These cases are generalized according to design solution states and retrieved once the specific requirements of a new case are known. Generic tasks, on the other hand, leverage commonalities between several applications and embody them to enable the engineering of future systems. Individual solutions are the strength of CBR, while generic tasks emphasize the problem solving techniques common to groups of solutions.

Although these two approaches exploit different aspects of knowledge-based systems, there is no immediately apparent way to integrate their advantages into a single system [Allemang, 1994]. CBR requires that knowledge be represented in uninterpreted cases, while generic tasks represent knowledge in task structures. Perhaps the best way to explain the integration of the two approaches is to examine the application of a specific generic task to CBR.

9.1.2 Augmenting Hierarchical Classification for Use in CBR

The use of hierarchical classification in CBR is the most natural of any generic task, especially considering the hierarchical case memory organizational structures introduced in Chapter Three (shared feature networks, category-exemplar networks, and redundant discrimination networks). A specific implementation of CBR using hierarchical classification to diagnose faults in manufactured textiles is considered here. This consideration includes an examination of the extensions made to hierarchical classification and the details of the implementation itself [Allemang, 1994; Allemang, 1995].

Hierarchical classification uses an establish-refine inference procedure to navigate through a hierarchy. Knowledge in a high-level node is used to *establish* its applicability; if it is established, the classification is *refined* by repeating the algorithm on the subnodes. The control of this algorithm (depth-, breadth-, or best-first) does not affect the knowledge structure. This lack of impact has some significance when combining hierarchical classification with CBR. The case memory navigation for a new case will most likely be different from that for previous cases; the only similarity is the knowledge requirement for a solution.

The use of a hierarchical case memory structure relies on specific observations about hierarchical classification [Allemang, 1994]:

- Hierarchical classification models the application domain as a tree; the structure of the case memory follows this tree.

- Knowledge exists in the tree to explicitly discriminate between sibling nodes, but not between nodes farther apart in the hierarchy.

- Less precise matching knowledge occurs high in the tree, and more precise knowledge, lower in the tree.

These features of hierarchical classification do not influence the algorithm used to navigate the tree, but rely instead on the knowledge-level description of what the nodes in the tree mean.

For the specific implementation under consideration here, the three-step processing cycle of CBR is performed at each node in the hierarchical classification tree [Allemang, 1994]. The possible textile failures are classified, and cases are retrieved based on specific failures. An algorithm combining case-based and rule-based reasoning reuses the retrieved case. If the reuse fails (i.e., an improper case is retrieved), a blame assignment algorithm reevaluates the correct location of the case in the hierarchy.

The main emphasis of this system is that of blame assignment, and making sure that cases are located properly within the case memory. Allemang has developed a method for checking knowledge errors in the case hierarchy (the blame assignment above). In this method, a case is checked against each partially relevant node in the hierarchy. A parable has been developed to describe this method [Allemang, 1995]:

> *A new case... asked if he could pass the door to a part where he thought he might be interested. The guard at the door said 'There is nothing relevant for you here - you may not pass' (if you have read Kafka* [The Castle], *then you might want to think about the phalanx of guards who stand behind this guard).*

The parable goes on to discuss how the new case calls out to all of the other cases and asks if any of them has a good reason for it to pass the door. A series of criteria for those cases that respond must be met in order for the case to pass the door: similarity (how similar are the cases?), accuracy (what percentage of the old cases responded?), and significance (what is the chance that this percentage would have responded randomly?). The parable then continues:

> *From these three measurements, we decide whether or not we walk past the guard* [the case has been anthropomorphized now] *(you will remember that the guard in the Kafka story said that you could pass if you wish, but you should watch out for the other guards, who are behind him - this is so relevant, that I sometimes wonder if maybe Kafka developed this method long before me), and continue with the establish/ refine method* [until the case is stored in the 'correct' location].

In this way, a failed case is placed properly within the case memory.

This implementation shows that hierarchical classification does indeed have utility as applied to CBR. The use of hierarchical classification (together with a governing rule-based system) in this diagnostic approach, enabled the implementation of the complete three-part CBR process.

9.1.3 A Task Decomposition for CaseFinder & DesignMod

The specific implementation of hierarchical classification and CBR in the previous section was driven by Chandrasekaran's three-part generic task description of the CBR process: retrieve, reuse, and store. The GT/CBR approach developed in this dissertation was inspired by the four-part CBR process description presented earlier (retrieve, reuse, revise, and retain), was specifically focused on design rather than diagnosis, and was not necessarily intended to completely capture each of these CBR tasks.

By reexamining the task decomposition for CBR, the location of the tasks addressed by the developed GT-influenced CBR tools in the four-part description of the CBR process can be seen. Figure 53 shows the CBR tasks implemented in CaseFinder and DesignMod. Taken together, these two new tools cover roughly half of the tasks comprising the CBR process. CaseFinder performs the tasks of **search, initially match**, and **select**, while DesignMod performs the tasks of **copy** and **adapt**. Notice that neither CaseFinder nor DesignMod is confined to a single main CBR tasks; CaseFinder spans

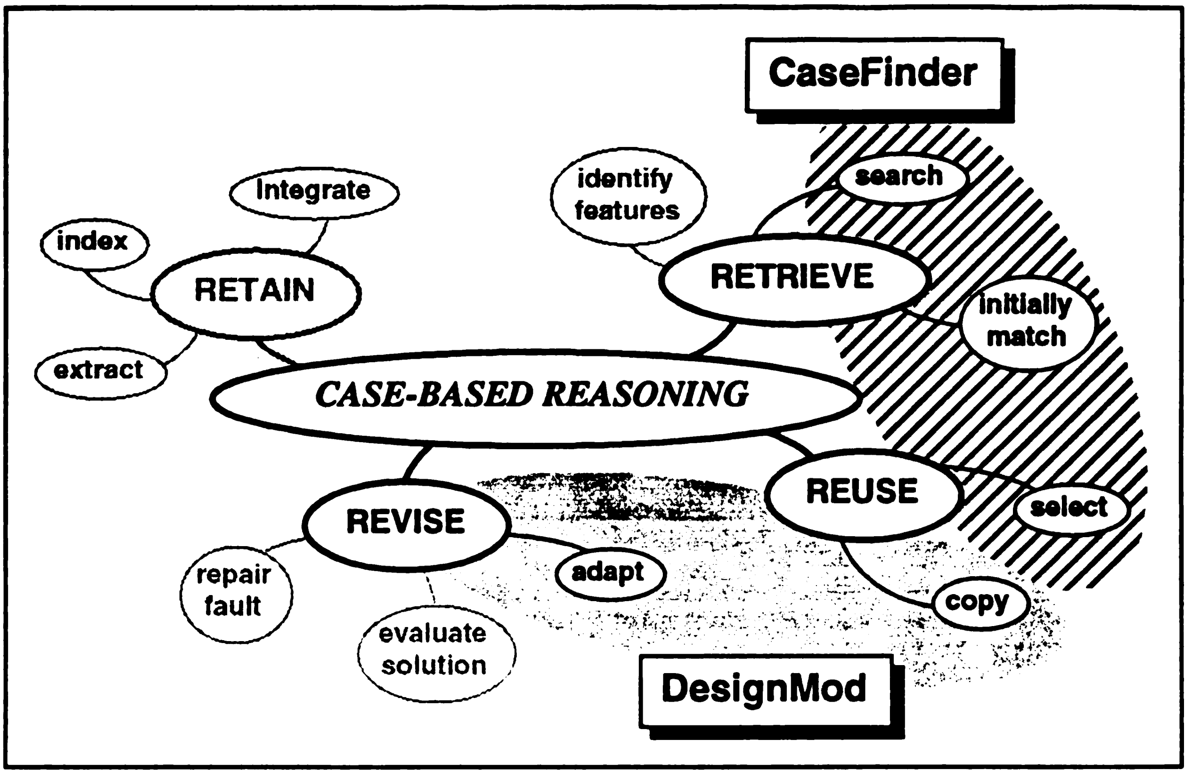

Figure 53. CBR tasks implemented in CaseFinder & DesignMod.

both **Retrieve** and **Reuse**, and DesignMod spans **Reuse** and **Revise**. The methods used to accomplish these CBR (sub)tasks in CaseFinder and DesignMod can be seen in the following sections.

9.2 CaseFinder

The GT influenced approach to CBR developed for this dissertation embodies two elements, as mentioned previously. The first element, CaseFinder, retrieves relevant cases from the case memory. It then chooses and sends the 'best' case to the second element, DesignMod, for subsequent modification. The integration of these two portions of the GT/CBR approach uses the ISL's GT integration methodology (see Chapter Two).

The case memory organizational structure used within CaseFinder is hierarchical and utilizes the principles of hierarchical classification for its knowledge representation. The details of a particular case memory tree cannot be predetermined, as the knowledge representation (and resulting case memory structure) is dependent upon the specific application domain. The following chapter presents this structure for composite material systems.

This lack of concrete predetermination of case memory structure may be perceived as an inordinate amount of flexibility. Such flexibility is a desired characteristic of a GT influenced CBR tool. CRISTA used case memory structures and indices that were very specific to its domain.[1] Unlike CRISTA, the flexibility in CaseFinder allows it to be used in disparate applications. A possible reason CRISTA was never ported into the ISL GT toolset was that its representations were not universally applicable. CRISTA's rigid structure and representation precluded its reuse in dissimilar domains. CaseFinder's flexibility is therefore a positive feature.

This section continues by examining the details of the CaseFinder implementation. CaseFinder's specific components are briefly discussed, with similarities to the analogous generic task components indicated. The overall knowledge flow through the CaseFinder is outlined. This section concludes with the details of the case retrieval and case sorting methods as implemented within CaseFinder.

9.2.1 Components

The components of a CaseFinder implementation include: a database, a case hierarchical classifier, and a case library. The top-level CaseFinder user interface, through

1. CRISTA addressed the domain of justificatory reasoning applied to tax law.

Figure 54. The top-level user interface for CaseFinder.

which all of these components are accessed, is shown in Figure 54. Buttons for each of the three components are included in the user interface, as is a button for initiating a stand-alone run of the CaseFinder.

Database. This component of CaseFinder is exactly the same as implemented within any standard generic task in the ISL toolset. This similarity permits a mapping between problem solvers of varying kinds, using the GT integration methodology to enable communication between different GT databases. The resulting ease with which database variables can be exchanged between CaseFinder, DesignMod, and a Routine Designer is a direct result of the generic task approach.

Hierarchical Classifier. As implemented within CaseFinder, the hierarchical classifier is quite similar to the standard implementation in the ISL toolset. There is, however, an important distinction. Within the CaseFinder HC, a node can access the match level for its parent (or parents) and use that value to normalize the possible results for it. This capability facilitates the retrieval of partially matching cases via branches in the case tree which seem fruitless at first, but whose relevance may increase with further exploration. The specific details of this match level normalization are left to a particular domain. See Chapter Ten for the application of this to composite material system design.

The match level normalization provides the novelty of CaseFinder's retrieval process. This is unlike CRISTA, which was more rigid in its case retrieval algorithm. If a node did not match, CRISTA eliminated all of the cases under that node from consideration. No partial matching of any kind was permitted within CRISTA's case retrieval algorithm.

Organization of the case memory is dictated by the content of the cases included in the library; different case libraries may necessitate different hierarchies. The hierarchies use syntactical indices for delineating cases. These indices are based on abstractions of both specific design attributes (e.g., within a certain range of a quantitative variable, similar types of answers for a qualitative variable) and additionally abstracted values (e.g., semi-crystalline thermoplastics, epoxy thermosets, and so on). Redundant paths to cases in the library may exist. The tip level nodes in the hierarchy point to a case within the case library.[1]

Due to the current nature of the tools in the ISL toolset, the case hierarchies must be generated manually - there is no automated hierarchy generation method, a definite requirement for any case archiving procedure. The indices must also be manually predetermined. This limitation precludes the use of CaseFinder (at least in its current

1. Non-tip level nodes may also point to cases within the case library. Such intermediate cases are similar to the cases as represented in [Allemang, 1994].

implementation) for automated case storage, i.e. the Retain task in the four-part CBR process. See Chapter Ten for an example of such index determination.

Case Library. The case library within CaseFinder is essentially a collection of cases, each stored as an object within the system's memory. The case library user interface shown in Figure 55 allows the user to examine the details of a case, modify an existing case, or add a new case to the library. The user interface includes four portions: the case names, a list of case attributes for a particular case, the values corresponding to those attributes, and the comments for a particular case. The modification of the attributes and their values is accomplished through the use of a standard GT database interface.

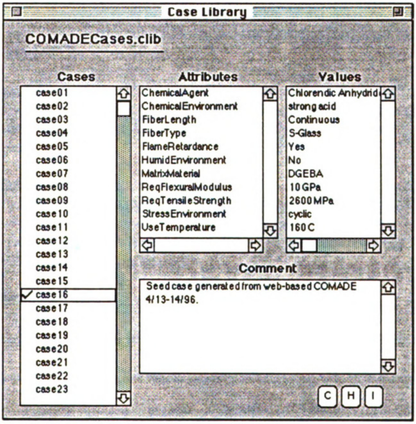

Figure 55. CaseFinder's case library user interface.

9.2.2 Knowledge Flow

The case retrieval process as implemented within CaseFinder can be broken down into four steps: initial query, case retrieval, initial matching, and case evaluation & sorting. Figure 56 depicts the case retrieval process. CaseFinder is intended to fit into an integrated CBR/GT system, the general details of which can be seen in Figure 57 on page 230; there are occasional references to other portions of this integrated system in the discussion which follows.

1. **Query to case library.** An initial query is made to the case library. This consists of variables of interest and their associated values. It is also possible to just specify a variable without giving a value (e.g. "I'm interested in cases which contain **FlameRetardance**, but don't care what the particular values are.").

2. **Case retrieval.** The query is tested against nodes within the library tree; partial matches are allowed to enable exploration down paths not necessarily matching exactly. This continues until one of two things occurs: a) a case is indicated explicitly; or, b) enough failure occurs to block further probing of the query tree. If no cases are matched, pass off to design from scratch with the Routine Designer. See 9.2.3 "Case Retrieval Method" for more details.

3. **Initially matching cases.** The probe of the query tree continues until all paths have either ended in failure or have indicated a case. If there are indicated cases, all of them have at least a moderate level of match with the query. No explicit level of match for a case exists at this point.

4. **Case evaluation & sorting.** The initially matching cases are ranked according to the quality of match, and the highest quality case is passed along for modification. Quality is a measure of the relative importance of the matching variables taken together. Those portions of the case which do not match are flagged for modification. See 9.2.4 "Case Sorting Method" for more details.

9.2.3 Case Retrieval Method

As has been mentioned several times previously, the details of any case retrieval method are inextricably tied to the organizational structure of the case memory. The case memory's organizational structure is in turn determined by the specific domain to which the CBR approach is applied. Therefore, discussing any specific details of the case

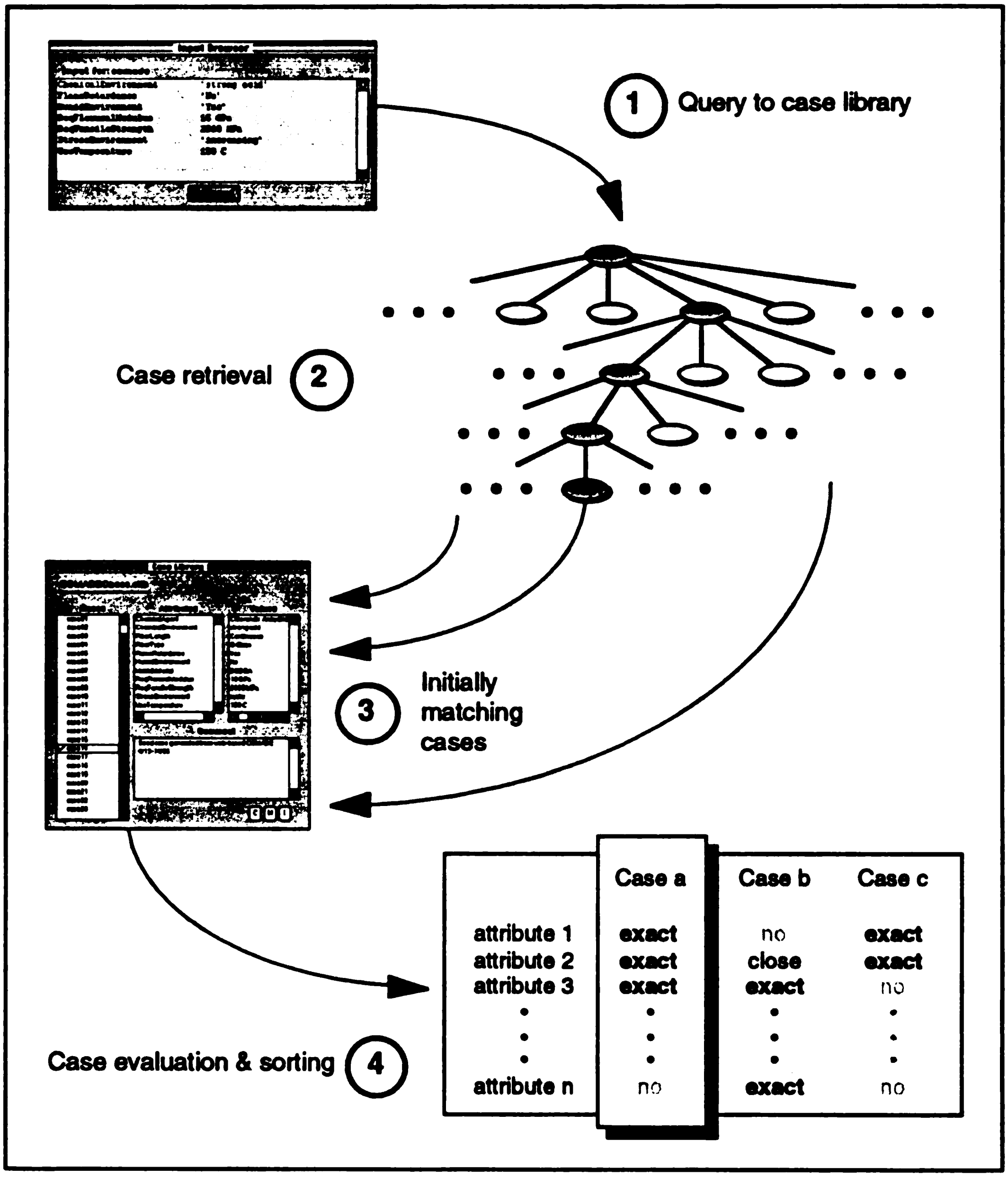

Figure 56. Process diagram for CaseFinder.

retrieval method in CaseFinder is pointless without a particular domain implementation. That discussion is reserved for the chapter immediately following. An overview of the capabilities of CaseFinder's case retrieval method is given here instead.

The general mechanism for retrieving a case is the same as hierarchical classification. The knowledge representation of the domain drives the details of the knowledge in the nodes, with general knowledge at the top of the tree and detailed knowledge at the tip level. The case retrieval uses the establish-refine method to probe the case memory tree structure. Depending on the knowledge within a node, the path through that node may have any one of seven qualitative confidence values: strongly match, match, weakly match, neutral, weakly against, against, strongly against. This is directly taken from the standard ISL GT hierarchical classification implementation. The nodes may also check the confidence value of their parent nodes as mentioned earlier. A benefit from this is with the controlled exploration of partially matching portions of the case library.

As the query is tested against nodes within the library tree, partial matches can be specified. These enable exploration down paths which do not match the current input exactly. The specifics of this partial match propagation are, again, determined by the application domain of interest. Chapter Ten presents an application focus for the detailed presentation of partial match propagation. The case retrieval continues until a case is either indicated explicitly or enough failure occurs to block further probing of the query tree (i.e., a non-tip node's confidence value is evaluated to be neutral or worse).

Only when the case memory hierarchy is structured appropriately will this retrieval process work properly. This case retrieval process is intended to obtain a small set of closely matching cases; an absolute degree of match is not explicitly determined. This set of initially matching cases is then evaluated and sorted to find the best matching case. As this retrieval method was developed for use in a routine design situation (i.e. explicitly structured domain attributes and values), it may not scale directly to other types of design environments.

9.2.4 Case Sorting Method

The list of initially matched cases must be evaluated and sorted in order to identify the case with the easiest modification process. Once identified, this case is sent to DesignMod to be modified. Although specific details of the case sorting method can not be given outside of a particular application domain, a summary of the approach utilized for sorting the cases is given here.

The weighting of the attributes, as well as the attachment of significance to their values, is predetermined by the structure of the routine design and the DesignMod implementations with which the CaseFinder is to be used. The results from the DesignMod Dependency Relationship Generator are used to attach importance to the variables in the ordering process.

The variable with the most dependent variables is identified as the most important variable to match. Other variables are ranked after that by their impact on the redesign process. An ordered list of variables is determined by this ranking. It is more desirable to match a variable higher on this list than lower, and a corresponding weight is given to each variable. This weight, or significance, is multiplied by the accuracy of the match to determine a ranking of a case's importance.

The accuracy weighting is determined by how closely the value for a retrieved case's attribute matches the input value. For numerical variables, there is a five-level scale for the accuracy weighting: match within 10% = 4, match within 20% = 3, match within 30% = 2, match within 40% =1, match within > 40% = 0. For other variable types, the accuracy weighting is more simple: exact match = 4, similar match (as determined by the user) = 2, no match = 0. The ranking of a case is determined by summing up the significance and accuracy products for each attribute/value pair. The case with the highest weight is the best match.

Currently, this case sorting routine is not implemented in a generic manner and must be hard-coded by the user. This hard-coding can take place only after the attribute dependencies and their respective effects on the modification of design cases have been evaluated within DesignMod. Of course, DesignMod must first examine the Routine Designer containing the design knowledge to determine the attribute dependencies. Once the sorting routine has been implemented by the user, the CaseFinder can evaluate and sort the initially matched cases.

The case which is determined to be the best match is passed to DesignMod for modification, along with a list of non-matching attributes. The ranked list of design cases is maintained, however, as the case sent to DesignMod is not necessarily guaranteed to be modifiable. If DesignMod indicates it was unsuccessful in modifying the case, the next best case is sent by CaseFinder, and the process repeats as needed.

9.3 DesignMod

The tool for modifying retrieved cases, DesignMod, was developed in conjunction with the development of CaseFinder. Because DesignMod was not developed explicitly for this dissertation, it will not be discussed in as much detail as was CaseFinder. Nevertheless, it is a portion in the combined GT/CBR approach to designing composite material systems from past design solutions. This integrated system makes use of the GT integration architecture discussed in Chapter Two.

DesignMod depends upon the existence of a routine design system for much of its knowledge. In fact, the behavior of DesignMod can be summarized by saying that it determines how to selectively run portions of a routine designer in order to modify a design case to fit the input specifications.

The location of DesignMod and its constituents in an overall GT/CBR system can be seen in Figure 57. There are three portions which comprise a DesignMod implementation [Kamel *et al.*, 1996]: the Dependency Relationship Generator, the Case Analyzer, and the Design Modifier. Here, each of the three portions are discussed briefly, with the roman numerals from Figure 57 indicating each portion. See Chapter Ten for a detailed example of these three steps.

I. The Dependency Relationship Generator examines the data structures of an existing Routine Designer and generates a list which specifies all design attributes dependent upon another attribute. The output is a list of each design attribute and its dependents (if any), e.g. UseTemperature: MatrixMaterial, ChemicalAgent, FiberType.

II. CaseFinder sends a retrieved case to the Case Analyzer and the case's flagged, non-matching attributes are examined. These attributes are matched against the design-attribute dependencies generated in I to determine if any design variables are dependent on the non-matching attributes. If such dependencies are found, those dependent variables are added to the list of flagged variables. This list indicates those variables that must be modified.

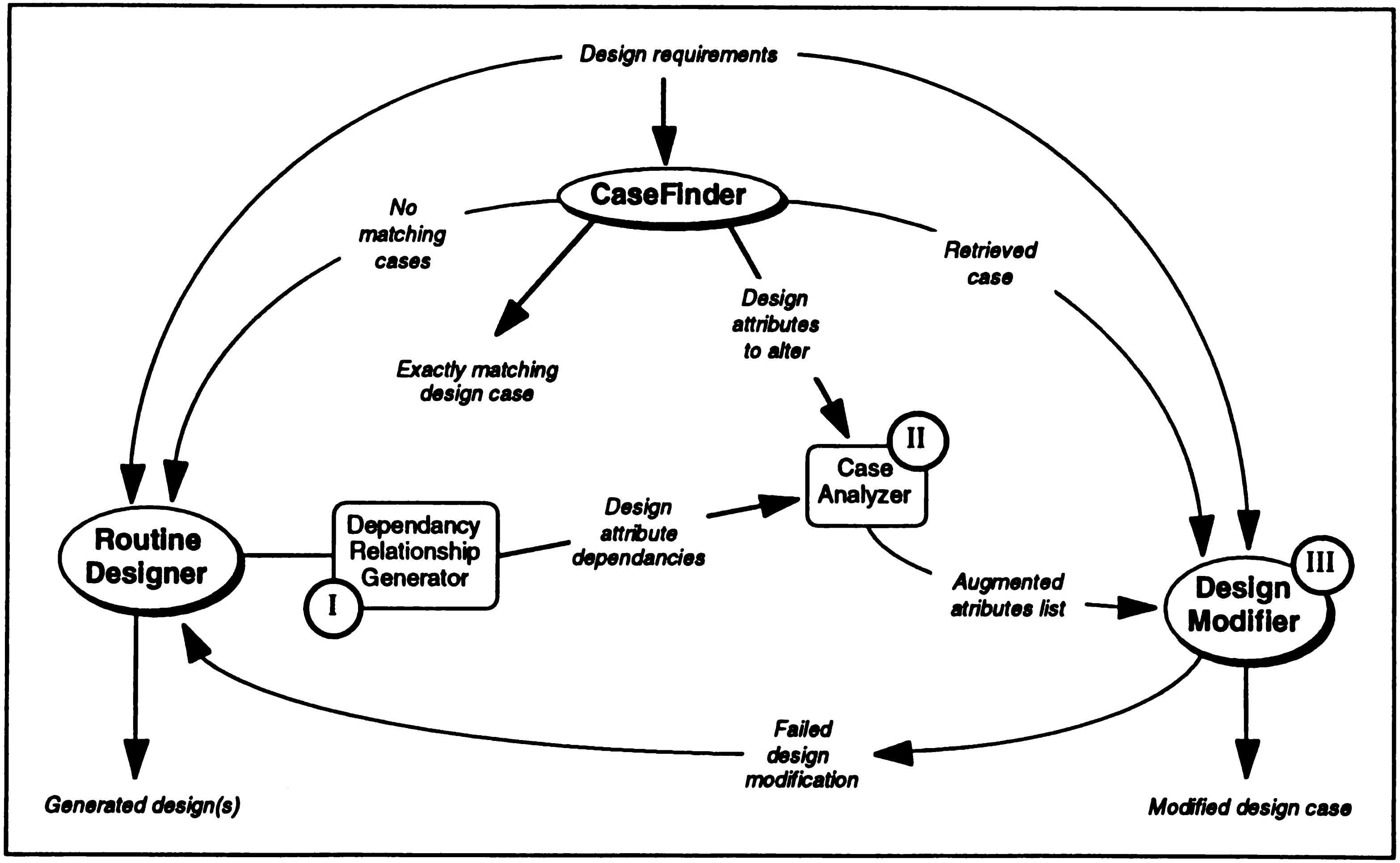

Figure 57. Information flow for DesignMod within an overall CBR/design process.

III. The Design Modifier takes the augmented attributes list, the retrieved case, and the originally specified design requirements, and utilizes the existing routine designer to selectively modify those non-matching portions of the retrieved case. Only portions of the routine designer are invoked to effect these changes. The result is a modified design case.

Figure 57 shows the two other overall behaviors for the CBR/design system which are possible:

1. An exactly matching case may be found in the case library. In this instance, no modification need be done, and the retrieved case is output as the design solution.

2. No cases are found within the case library that match the design requirements. Here, the routine designer is run completely and the design is generated from scratch. This output design should then be incorporated into the case library.[1]

The chapter immediately following presents the specific application of these techniques to the design of composite material systems through the reuse of existing design solutions. Within the confines of that specific application domain, more detailed discussions of the capabilities of both DesignMod and CaseFinder can take place.

1. As mentioned previously, this must be done manually, as no automated methods for hierarchy creation or index abstraction exist in the current ISL GT toolset.

CHAPTER 10

Designing Composite Material Systems with CBR

The presentation of the underlying methodology and framework for the GT influenced case-based reasoning tools in the previous chapter did not include many specific examples. The domain-dependence of several portions of the methodology was a constant caveat. This chapter provides such a domain focus.

The discussion that follows revisits the major aspects of CaseFinder and provides specific examples for composite material systems. Design representation is at the heart of this entire dissertation, and at no point is it more important than for the application of CBR to composite material system design. The design representation of composite material system cases is presented briefly to begin this discussion. After that, the interconnected nature of the case hierarchy and retrieval algorithm is presented. The details of the retrieved case sorting mechanism for composite material systems are also given. Finally, this chapter concludes with an example of the how the integrated CaseFinder/DesignMod system finds and modifies composite material system designs.

10.1 Representing Material System Designs as Cases

Depending upon the domain and task for which the CBR approach is used, the exact definition of a case may change, varying in shape and size, temporal coverage, and associational structure (e.g., solutions with problems, outcomes with situations). Alternatively, cases can contain a lesson and a specific context in which the lesson was applicable. The context is used to decide when the lesson may apply again.

CADRE incorporated only shallow cases describing building designs in its case library. Shallow cases are those cases that contain only a design solution, and no associated design history. Otherwise stated, shallow cases consist of simple flat feature-attribute pairs. For the design of composite material systems, and specifically output designs from COMADE, a case consists of variable and value pairs for composite material systems. Table 15 shows an example case generated from COMADE.

Table 15: A sample case generated by COMADE (case11).

Variable	Value	Units
UseTemperature	220	°C
ChemicalEnvironment	strong base	--
ReqFlexuralModulus	170	GPa
ReqTensileStrength	1600	MPa
FlameRetardance	yes	--
HumidEnvironment	no	--
StressEnvironment	increasing	--
MatrixMaterial	PAI	--
ChemicalAgent	None (TP Matrix)	--
FiberType	SiC	--
FiberLength	Continuous	--

10.2 Organization & Navigation of a Material System Case Memory

After the case representation has been specified, it is then possible to consider the case library organization. This includes both indices and organizational structures. Both are fixed by the specific domain to which the CBR approach is applied, in this instance, composite material systems. The details of the case retrieval method are domain dependent as well. The case retrieval method is inextricably tied to the organizational structure of the case memory.

In the construction and navigation of a case hierarchy, the advantages of structured knowledge reuse become apparent. By using the inferencing knowledge and design representation in COMADE, it is possible to construct a readily navigable case hierarchy.

The discussion that follows considers the influence of the composite material system domain upon the indexing vocabulary, organizational structures, and retrieval methods for a case memory. The reuse of structured knowledge from COMADE enables the definition of each of these case memory characteristics.

10.2.1 Determining Indices for Composite Material Systems

Recall the basic principle of CBR presented in Chapter Three: "Generalize the indices, not the cases!" [Hammond, 1989]. This statement is an excellent starting point when considering the definition of case indices for composite material systems. The flat nature of the case representation presented above suggests that the indices for these cases must be quite syntactic. There simply is not enough richness in the case representation to enable the abstraction of sufficiently semantic indices.

There are two types of indices that can be used for composite material systems: strictly syntactic, and partially semantic abstractions. The design representation from

COMADE used for the case representation limits strictly semantic indices to input variables (e.g., StressEnvironment and values of it). By then generalizing these indices, it may be possible to generate case hierarchies that can represent the entire case memory. Other indices that can be used are design abstractions from the agents hierarchy in COMADE (e.g., semicrystalline thermoplastic, epoxy, carbon fiber, and so on).

10.2.2 Case Hierarchies for Composite Material Systems

Before a case hierarchy can be specified, two items must exist. First, indices for the representation of archived cases must be defined. The previous section discussed these indices. Secondly, a library of cases must exist. Two case libraries have been generated and were used to determine case hierarchies for composite material systems. The discussion here focuses on a case library that resulted from a design-of-experiment to explicitly represent the domain coverage of COMADE. Consequently, this case library is called the structured library. It consisted of 27 composite material system designs. The other case library was randomly generated. The details of both libraries are shown in Appendix E.

Several possible case memory organizations were discussed in some detail in Chapter Three. These included category-exemplar networks, redundant discrimination networks, and shared feature networks. Many attempts were made to model the case memory for the structured library. Several variations that used the more syntactic indices are shown in Appendix E. The case memory hierarchy found to be the most appropriate is the most similar to a shared feature network. Figure 58 shows the entire case hierarchy for the structured cases. A more detailed (and larger) version of this hierarchy is shown in Appendix E.

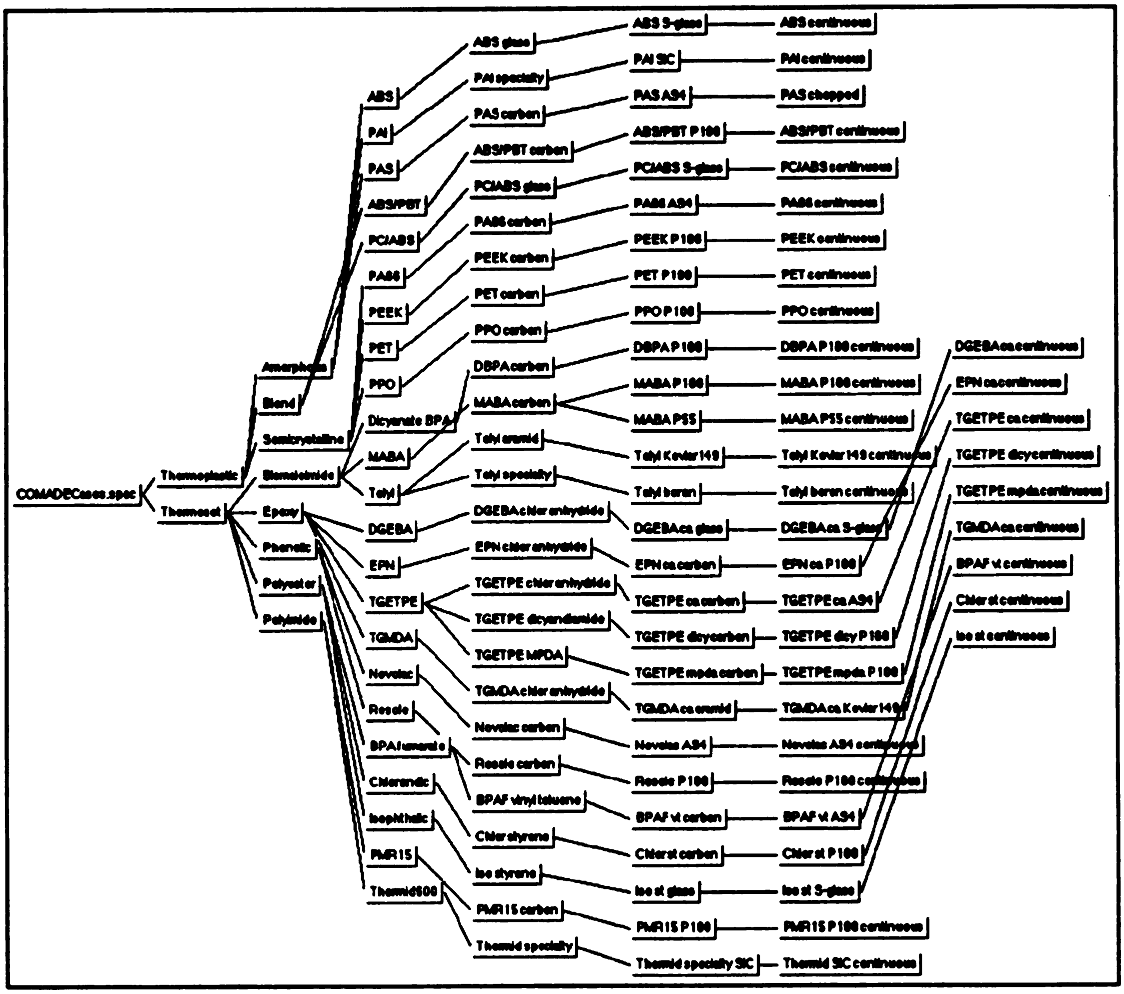

Figure 58. A case memory hierarchy for COMADE cases.

This case hierarchy used the class abstractions from COMADE with abstractions of design solutions. Depending upon the cases in the library, this structure will most likely change. It currently contains 124 nodes for the 27 structured cases. Specific knowledge from COMADE about the composite material system domain exists within each of these nodes. However, that knowledge is secondary to the control knowledge used to guide the case retrieval process that is also embedded within each of these nodes.

10.2.3 Case Retrieval for Composite Material Systems

The control knowledge within each of these nodes is called match level normalization. This adjusts the maximum match level for a node based upon the match level its parent had. Figure 59 shows a simple example of a node with match level normalization from the structured case hierarchy.

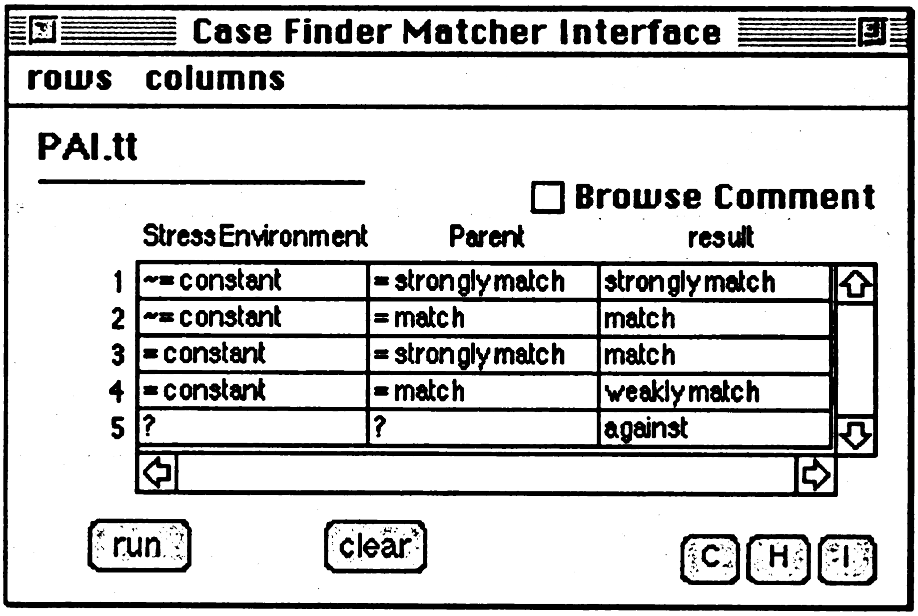

Figure 59. An example of match level normalization.

In this pattern matcher, the domain knowledge of interest is StressEnvironment. If the StressEnvironment is not equal to constant, then the node is established. However, as shown in Figure 59, the match level is dependent upon the parent's match level (in this example, the match level of AmorphousThermoplastic). If the StressEnvironment condition is met, the match level of this node can be either match or strongly match, depending upon its parent's level. A partial match also can decrease the match level of a node. Abject failure to meet a node's requirements results in a match level of against.

Table 16 shows a progression of the case hierarchy navigation for part of the structured case hierarchy. As the probe extends further down the branch, more resistance is built up. Eventually, however, a case is retrieved (case11). This uses the input case shown in Table 18 on page 241.

Table 16: A partial case retrieval process showing match level normalization.

Node	Match Level	
Thermoplastic	strongly match	
Amorphous	strongly match	
PAI	match	
PAI specialty	weakly match	
PAI SiC	weakly match	
PAI continuous	weakly match	
		case11

10.3 Evaluating & Ranking Retrieved Material System Cases

Once the CaseFinder has searched its case hierarchy and retrieved a small set of matching cases, it then decides which case is the "best." This is the case that will be sent to DesignMod for modification.

As outlined in the previous chapter, the sorting routine is driven by the eventual modification process. An ordered list of variables is found by ranking their impact on the redesign process. For COMADE, the small list of variables is highly interdependent. The significance of the variables within COMADE is assigned as follows: 4 - UseTemperature, ChemicalEnvironment; 3 - ReqFlexuralMod; 2 - ReqTensileStr; 1 - StressEnvironment, FlameRetardance, and HumidEnvironment. The combination of the significance and the accuracy weighting outlined in the previous chapter, suggests how easy a case will be to modify. The higher the sum of the accuracy and significance products is for the case, the less modification will theoretically be needed. Table 17 shows an example of this evaluation and sorting process for two cases.

The details of case3 and case11 are given at the top of this table[1], with the input case against which they are to be compared given in the lower left corner. The values given below each of the retrieved cases are the accuracy weights. For instance, compared to the input case's UseTemperature, case3's UseTemperature of 160 matches within 20%. According to the numerical value accuracy scale, this receives a weight of three. When multiplied by the significance of the UseTemperature variable (4), this contributes 12 (out of a total possible 60) points toward the case ranking. The significance values are not shown in this table. Clearly, case11 is the more closely matching case.

1. These variable names and values are abbreviated. However, the abbreviations should be apparent. The order of the variables in cases three & eleven is the same as for the input case.

Table 17: Evaluation and sorting of retrieved cases.

	CASE	3	11
	UT	160	220
	CE	wb	sb
	RFM	250	170
	RTS	1100	1600
	FR	yes	yes
	HE	yes	no
	SE	incr	incr
UseTemperature	184	3	3
ChemicalEnvironment	strong base	2	4
ReqFlexuralModulus	215	3	2
ReqTensileStrength	1251	3	2
FlameRetardance	yes	4	4
HumidEnvironment	no	0	4
StressEnvironment	constant	0	0
	Score:	39	46
	% Match	0.61	0.72

10.4 An Example of Reusing Designs from COMADE

The following example shows the capabilities of the CBR/GT tools applied to the reuse of composite material system designs. The 27 cases presented in the beginning of this chapter were used as the case library for this example. The case organizational structure shown in Figure 58 and the navigational methodology presented earlier were also used. The following example traces through the entire case retrieval process shown in Figure 56 on page 225, and the case modification using DesignMod.

Table 18 show the input case to the CaseFinder. This was a random case generated to produce the alternate case memory hierarchy mentioned earlier. Notice the difference

Table 18: Input case for the composite material system example.

Variable	Value	Units
UseTemperature	184	°C
ChemicalEnvironment	strong base	--
ReqFlexuralModulus	215	GPa
ReqTensileStrength	1251	MPa
FlameRetardance	yes	--
HumidEnvironment	no	--
StressEnvironment	constant	--

Table 19: The nine retrieved cases for the composite material system example.

Match Level		
strongly match	match	weakly match
case5 (0.16)	case8 (0.30)	case24 (0.34)
case25 (0.28)	case6 (0.22)	case21 (0.34)
	case3 (0.61)	case11 (0.72)
	case19 (0.13)	

between this input case and the archived case shown in Table 15. The input case contains no material system description.

The composite material system CaseFinder queried its case hierarchy with this input and retrieved the eleven cases shown in Table 19. Notice that there were cases with different matching levels retrieved: two strongly match, four match, and three weakly match.

The ingenuous conclusion is that one of the two cases that were found to strongly match the input would be the best case. However, that is not what happens in this example. The number in parentheses after each case name in Table 19 shows the relative match for each case after sorting. These values were determined according to the procedure detailed above. Notice that the two strongly matching cases have low values. Based on the relative match level, case11 (a weakly matching case) represents the best match (see Table 17 on page 240 for the details of this evaluation and sorting). Table 15 on page 233 shows the details of this retrieved case.[1]

Only one constituent of case11, the StressEnvironment, does not match the input at all. Other variables match the input partially (UseTemperature, ReqFlexuralModulus, and ReqTensileStrength). These four variables are flagged, and the case is sent to DesignMod for modification. DesignMod checks the dependencies of these four variables, and finds that every variable in the COMADE input set is dependent upon these variables. DesignMod runs the entire COMADE system.

1. That a so-called weakly matching case was found the "best" case points out the danger in reading any significance into the labels assigned to retrieved cases. These labels do not show how well a retrieved case matches the input. The match level attached to these retrieved cases is only a measure of the case retrieval algorithm's built-up resistance. The lower a retrieved case's match level (i.e., closer to neutral or against), the more resistance to the case hierarchy probe for that branch.

Table 20: The modified composite material system designs.

MatrixMaterial	ChemicalAgent	FiberType	FiberLength
4,4'-MDA-BMI	Dicyanate BPA	P100 graphite	Continuous
4,4'-MDA-BMI	1,4-Tolyl BMI	P100 graphite	Continuous
PEEK	None (TP Matrix)	P100 graphite	Continuous
PC/ABS	None (TP Matrix)	P100 graphite	Continuous
PSU	None (TP Matrix)	P100 graphite	Continuous

Each row in Table 20 shows a composite material system design given as the output from this modification process. Nothing remains of the retrieved case in these designs, as DesignMod decided a complete modification was necessary.

It seems that the results of this modification process invalidate the use of the retrieved case for this example. However, that is far from the truth. Through several variable dependencies, the example case shown here happened to influence the entire composite material system design. The methodology is sound, regardless.

Part V

Denouement

CHAPTER 11

Validating COMADE & the Composites Design Vision

After the expanded approach to composite material system design has been examined in detail, unanswered questions still exist. The main question that must be answered is that of validity. How valid is the polymer composite material system designer COMADE? How valid is the design vision for composites? These are not simple questions to answer, especially given the natures of COMADE and the design vision.

COMADE is comprehensive in its material domain coverage, but is definitely restricted in its use of decision variables. The design vision, while providing a sense of how the composites design-to-manufacture process must occur, is still abstract and does not explicitly detail the interactions between material design, process design, and part design. Somehow, a metric for determining the validity of these two items must be found.

The difficulty in asking users of COMADE about its validity is that everyone has a different perspective and understanding of what they think COMADE should be able to do. Trying to draw conclusions about the validity of COMADE with such an approach is like asking people to say if a painting is good or not, and why. Everyone will have a different opinion and different reasons for it. The best that can be done is to get as many opinions as possible and then attempt to make sense of them.

Three methods were used for evaluating both COMADE and the composites design vision that encompasses it. The first method involved the use of COMADE in a specific design experiment. The design experiment, called MADEFAST, gave insight on the usability of COMADE and the interactions between the portions of the design vision. The second means of evaluating COMADE and the design vision was through the specific application of COMADE's knowledge and the ideas of the design vision to an application for which they were not explicitly designed. A new system for designing composite material systems and fabrication technologies has resulted from this project. Finally, a two-pronged plan of attack was used to expose COMADE to as many individual users as possible. The two methods of exposure entailed conducting demonstrations of COMADE in person and making a version of COMADE available that could be run over the web.

11.1 An Experiment in Distributed Design

This project was to develop a collaborative infrastructure and community within DARPA's MADE (Manufacturing Automation and Design Engineering) initiative. As such, it was not a design experiment, but rather a design collaboration experiment. Within the context of developing this infrastructure the collaborators were to rapidly deliver a working infrared seeker prototype.

The ISL's portion of the project was the design and fabrication of a composite casing for an infrared seeker mechanism in a tactical missile. This served as a high profile showcase for the agile manufacturing and rapid prototyping enabled by intelligent decision support systems for polymer composites. This was also an excellent testing ground for COMADE and the refocused design methodology presented in Chapter 6.

Where applicable, the systems for material design (COMADE) and technology selection (COFATE) were used in the design of the seeker casing. However, since these systems covered only part of the design vision, outside interaction with composites design experts was used to complete the design.

Other than the fact that the casing was to be cylindrical and made of composites, little information was initially provided. Through interactions with members of the MADEFAST community, final geometric tolerances and performance requirements for the casing were established. As the casing was only to be a prototype, rigorously designing for the actual use environment was ruled out. This also meant that the casing was to be fabricated from existing stock if possible.

Based on a combination of qualitative and quantitative use requirements for the casing, COMADE suggested several materials. Matching these suggestions with the available supply narrowed this list to one material. This was a Hercules graphite/epoxy prepreg (AS4/3501-6). A Quantum sheet molding compound (QC-8800) was suggested by a collaborator, as this material had proven to be useful in similar applications and a supply of it was available. As this type of material was in the list originally suggested by COMADE, it met the requirements for use. Other aspects of the material design process are shown in Figure 60.

Based on a combination of availability, recommendation by COFATE and processing requirements of the materials, prepreg autoclave was chosen as the fabrication technology. After discussions with two collaborators, a simple cylindrical aluminum tool for the casing was designed. Consolidation in the material was achieved by using a heat-curable shrink tape as suggested by a collaborator. Use of this technique, co-curing the SMC with the epoxy prepreg, and finishing the part by smoothing the outer surface enabled the production of a superior finished part.

The seeker casing was completed and shipped out for inclusion into the final seeker assembly. The casing was a hybrid composite, with an interior of quasi-isotropic epoxy co-cured to an exterior of vinyl ester/glass SMC. This hybrid nature combined the structural benefits of the epoxy with the impact resistance of the SMC. The part had a wall thickness of approximately 4 mm, had a 5" O.D. and was 8" in length.

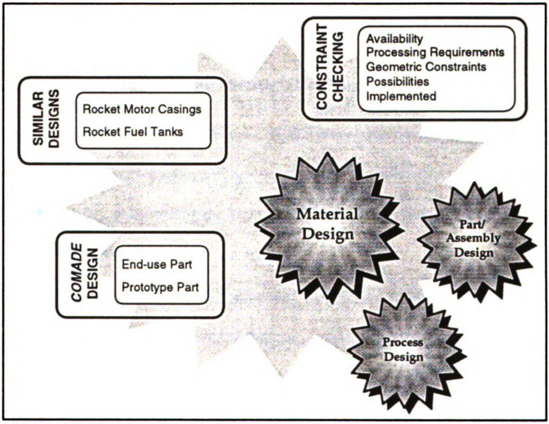

Figure 60. Material design considerations in MADEFAST.

Other than the actual production of a composite material, this design experiment had two major results. One was the use and acceptance of COMADE as a viable source of composite material designs. Collaborators ran COMADE and discussed the results with ISL researchers, accepting the generated material designs as valid. The final working composite was then fashioned from material designs suggested by COMADE. The other major outcome of this experiment was an increased understanding of the composites design process. The three starburst design vision espoused by the ISL (Figure 39 on page 150), while descriptive of an abstraction of the design process used, does not fully capture the interactions existing between material design, process design, and part design. MADEFAST helped to clarify some ideas about how these interactions should and do take place.

11.2 Integrated Design in an Industrial Setting

This work was to modify COMADE and COFATE into systems specific to high temperature polymer composites used by GE Aircraft Engines. The effort had two goals: developing the systems, and assessing the impact on the composites design vision that such a development may have.

The use of high temperature composites in jet engines holds promise for reducing both weight and manufacturing costs of these large, complex systems. Jet engines have a significant role in power generation and airline propulsion. Focusing on this market segment is an important example of how COMADE can be modified to serve the specific issues of a market and facilitate the use of composites in that market.

Both COMADE and COFATE show the capability to organize and reason about composites design knowledge. However, much of the current comprehensive domain coverage becomes superfluous or ambiguous when applied to a specific industrial application or product line. This effort continued the development of these systems by focusing their current knowledge and reasoning on an application market segment, specifically aircraft engine components.

Many inputs used in COMADE mapped directly into salient inputs for the design of aircraft engines. However, much of the material in COFATE was superfluous. Knowledge beyond that already present in the two systems was necessary to make design decisions for composites used in aircraft engines. This knowledge included items such as specific material systems and processing technologies, and information required to make decisions with these additions.

An overall architecture that combined both material & processing issues was established. This architecture involved a total of six problem solving units (Material, FiberForm, MaterialSystem, Layup, Prep, Cure) combined in an integration environment.[1] This integration work for multiple agents involved an approach not previously applied in the composite materials domain. Figure 61 shows this architecture and the accompanying knowledge flow through the GEAE Composites Designer.

GEAE-CODE was subjected to preliminary testing and received mixed approval from GEAE. The decomposition of the design problem was accepted enthusiastically; however, the processing designs generated by the system were not as focussed as GEAE wanted.

1. This used the ISL's generic task Integration Framework that was discussed in Chapter Two.

This lack of focus was addressed in two different ways. A reevaluation of the variables in the GEAE-CODE system was performed. Specifically, the influence of the variables on the final design was evaluated. Because of this evaluation, the list of variables used to make decisions was streamlined significantly. Additionally, further processing knowledge was obtained from GEAE and added to the system.

The structure of the GEAE-CODE system calls into question some ideas that have been held about the overall design vision for composites. The combination of both material and process design within the same system is something that has always been implicitly forbidden by the composites design vision. However, a closer examination of the structure of GEAE-CODE shows that the process design is effectively isolated from the

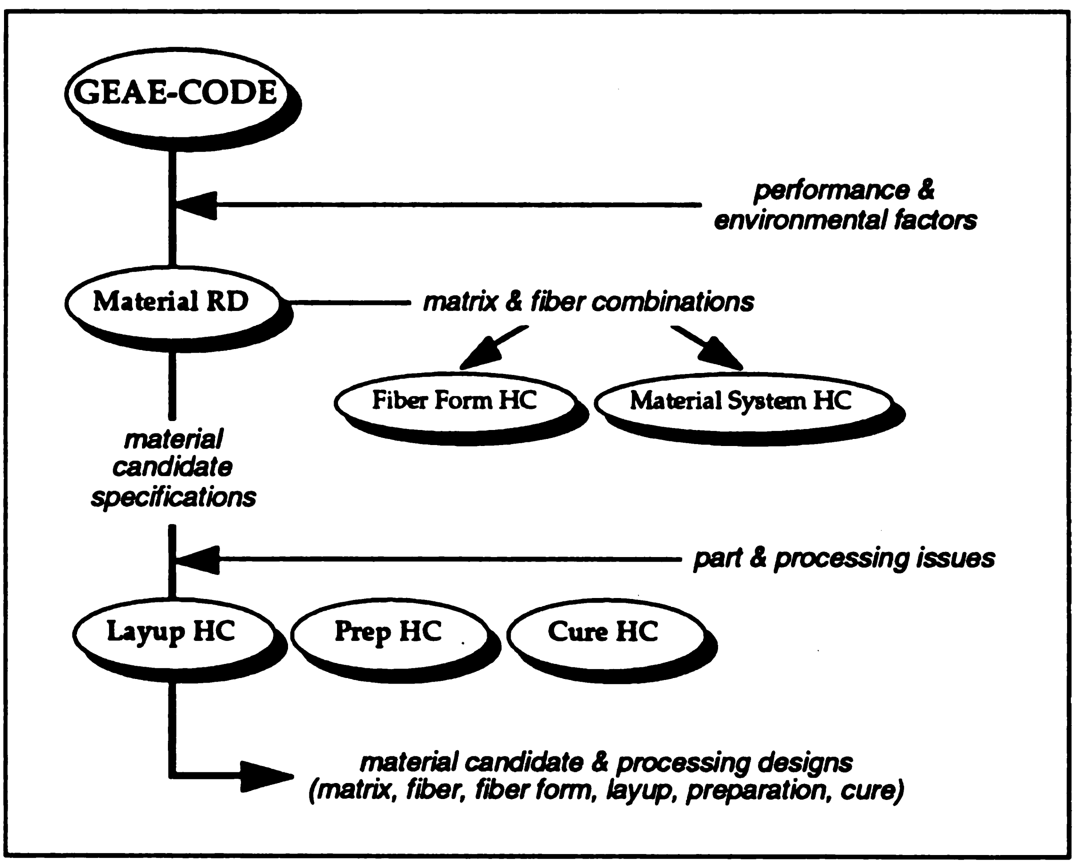

Figure 61. Design problem decomposition & knowledge flow for the GEAE system.

material design. The design process within GEAE-CODE is linear, with the material design serving as a precursor to the process design.

The validity of COMADE and its approach to the design of composite materials was borne out by the development of the GEAE-CODE system. The decomposition of the material design problem and a substantial portion of the domain knowledge within COMADE were directly applicable within GEAE-CODE. This was in direct contrast to COFATE, which, besides a few variables, was entirely unusable in the new application domain.

11.3 In-service Evaluation of COMADE

Unlike the two previous portions of this chapter, this section deals with the evaluation of COMADE on its own. The MADEFAST and GEAE validations of COMADE took place with highly focussed environments that set specific ground rules for its use. Because of those limitations, the metrics with which COMADE and the design vision were judged were straightforward. There are no such metrics within the following discussion.

The following discussion looks at the two ways used to expose COMADE to as many individual users as possible. These methods involved conducting demonstrations of COMADE in person and making a version of COMADE available that could be run over the web. The responses from these users serve as additional material for the assessment of COMADE's validity.

11.3.1 Demonstrations of COMADE

A concerted effort to show COMADE to industrial users has been underway since the completion of COMADE. As of April 1997, nearly 250 people have had COMADE

Table 21: Selected demonstrations of the COMADE system.

COMADE shown to (or at)	Location	Date
NSF Center Industrial Advisory Board members	East Lansing, MI	May 1994
Advanced Design Group at GE Aircraft Engines	Cincinnati, OH	December 1994
Himont International personnel	East Lansing, MI	January 1995
Chrysler research personnel	Auburn Hills, MI	January 1995
Aeroquip personnel	Ann Arbor, MI	February 13, 1995
Cadillac research personnel	Flint, MI	March 2, 1995
Ford research personnel	Dearborn, MI	March 7, 1995
Booth at the 40^{th} International SAMPE Exhibition	Anaheim, CA	May 8-10, 1995
Booth at the Advanced Research Projects Agency (ARPA) Software Technology and Intelligent Systems Symposium	Chantilly, VA	August 28-31, 1995
Prospective MSU Chemical Engineering graduate students	East Lansing, MI	March 13, 1996
Staff members for Michigan Congressman Dick Chrysler	Washington, D.C.	March 18, 1996
Booth at the Coalition for National Science Funding Exhibition and Reception	Washington, D.C.	March 19, 1996
Montell International personnel	East Lansing, MI	May 28, 1996
TRP/Michigan State University/University of Delaware Joint Composites Workshop	East Lansing, MI	Oct. 24-25, 1996
Prospective MSU Chemical Engineering graduate students	East Lansing, MI	March 15, 1997
Prospective MSU Chemical Engineering graduate students	East Lansing, MI	March 22, 1997

demonstrated to them. COMADE has also been installed at 12 companies, ranging from the Big Three to General Electric Aircraft Engines to AutoAir Composites. It has been demonstrated on Capitol Hill, at SAMPE's Annual Symposium & Exhibition, and at DARPA's Software Technology and Intelligent Systems Symposium. Table 21 shows a selected listing of the demonstrations of COMADE given over the past three years.

These demonstrations have provided significant exposure for COMADE. They have also provided an impetus for several industrially focussed systems, including the GEAE-CODE system mentioned previously. That COMADE has sparked additional research in composites design speaks volumes about the favorable light in which it is viewed.

While a necessary distribution, publicity, and validation mechanism early in the life of COMADE, recent efforts at demonstrating the system in person have declined. That mode of distribution consumed significant time and resources. The preferred mode of demonstration now lets users come to COMADE, rather than taking COMADE to the users. This has been accomplished by making COMADE available over the web.

11.3.2 Web Version of COMADE

The emergence of the web as a popular medium for the exchange and delivery of information was underway at the time COMADE was originally completed. COMADE was first demonstrated in May of 1994, and in June of 1994 a version of COMADE was made available on the web[1]. This version could be used by anyone, anywhere in the

1. This version of COMADE can be reached via the ISL homepage at **http://isl.cps.msu.edu** by following the links for composites research. Also available is a version of the ISL's fabrication technology selector, COFATE.

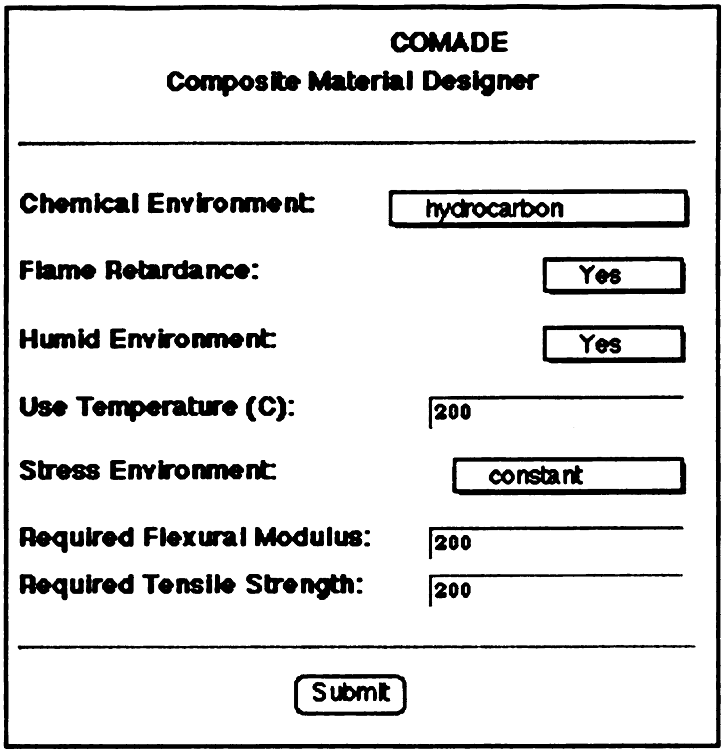

Figure 62. Input prompt for web-based version of COMADE.

world, and any time. Nearly simultaneously, extensive documentation of the rationale behind and the specific knowledge within COMADE were also made available on the web.

By today's standards for the web, the original interface for using COMADE was quite primitive. However, as the technological capabilities became available the web version of COMADE was improved. Figure 62 shows the currently presented user interface for COMADE on the web[1].

Using the web as a presentation medium limits the method of presentation. The results of COMADE presented to the user are simple lists of material systems and no documentation of these results is possible. This is in contrast to the stand-alone version of COMADE, in which an explanation may be requested for any generated portion of a

1. This is current as of April 1997.

design. The user interface capabilities and bandwidth of the web have not advanced far enough to make this sort of documentation easy.

Over almost three years over 1,800 different users have run COMADE over the web. This may seem like a lot, but in standard web terms this is a negligible trickle of activity. Averaging this out over the three years gives about 20 uses of COMADE a week. Use of this system continues even now, three years after its inception[1].

While small by web standards, this kind of exposure is far greater than is possible with individual demonstrations as it is not limited by availability of personnel, scheduling, or finances. Granted, not every user of COMADE is seriously interested in composite material system design, nevertheless, enough legitimate use of the web version of COMADE has taken place over its lifetime to generate useful response.

There have been three main types of legitimate users of COMADE on the web. The first type was exemplified by the MADEFAST experiment, detailed earlier. The second use is by academic institutions, either researchers or students. An example of such a use is presented below. The final use is by industrial researchers who are looking for ways to apply COMADE or a similar approach to their specific research problems. An example of this industrial use concludes this section.

Australian Material Science Class. The web versions of COMADE and COFATE were used by students at the University of Technology, Sydney in Australia, for a capstone course in material science. The instructor for the course sent a request for additional explanation and paid a compliment [Heness, 1996a]:

1. The week of March 23-29, 1997 there were 12 separate users who ran the web-based version of COMADE.

```
I just tried using the fabrication selector. Would you mind explaining
what match means as opposed to against? Nil I take it means it does
not match at all. Thanks for your help. Both this and the materials
selector are good work.
```

After a response to these questions was sent, the instructor responded with comments about the systems from the students (and another compliment) [Heness, 1996b]:

```
Many thanks, that clears it up. I have some students in a final year
subject use it. You might be interested in some comments.

Generally they think that you often end up with too many choices, par-
ticularly with the materials selector. I suggest they refine their
search parameters. The second is that for the materials, especially
the matrix materials, there are many abbreviations and trade names
which don't allow them to work out what it is.  I must admit that I
have not looked too closely at this.  The last is that they often get
no matches for the fabrication.  Again, I suggest that they refine
their parameters.  All in all, this is a very useful site and congrat-
ulations to all involved.
```

A limitation of the web-based version of COMADE, as mentioned earlier, was the lack of detailed documentation. Providing the kind of documentation available with the stand-along version of COMADE would have eliminated the confusion over abbreviations.

That the students were getting too many material systems as output speaks to a couple of issues. The first is that the user interface (as shown in Figure 62) does not give enough of an explanation of how the variables are used to make decisions. As a result, users can have unrealistic expectations about the types of answers COMADE should generate. Secondly, and more importantly, is that the knowledge within COMADE most likely needs to be augmented to reduce the number of designs. A similar development to that discussed for the GEAE-CODE system earlier may be required.

Aeroquip Request. Industrial researchers are always looking for new ways of solving their specific research problems. By having COMADE available on the web, such researchers can test the capabilities of COMADE to see if it is useful to them. These researchers are looking to apply COMADE or a similar approach to their specific research problems. One such example of an industrial researcher who found COMADE and its methodology to be possibly useful for their research sent the following email [Hofmann, 1996]:

> I am with Aeroquip Corp, a TRINOVA company. We are a world-leader in fluid conveying products, for example, rubber hose. I am part of Aeroquip's Advanced Computer Technology and Information Systems group. One of our projects takes a look at the our compounding process. During my investigation, I discovered your COMADE tool/project. It looks like your tool could be used to support the compounding process of rubber hose? If so, I would like to learn more about your project?

An interesting sidenote to this is that COMADE was shown at Aeroquip, but not to the Advanced Computer Technology and Information Systems group. The contact above was independent of that demonstration, and was initiated due to the presence of COMADE on the web. By having COMADE available on-line, those who are interested in either the domain or the approach used in COMADE can find and use it for themselves and request more information as needed.

Although nothing resulted from this contact, it shows two things. One, people can get their hands on COMADE on-line without being hand-held through the process. Two, it showed that the approach used in COMADE is perceived as potentially useful even for widely disparate application domains such as hose compounding. While COMADE as it is cannot design for such applications, the underlying rationale and tools with which it was built are primed for use in different domains. That the tools can be used in such a way is no surprise, as that is the intent behind their creation. The perception of the rationale as useful in other domains is a very favorable reflection on the capabilities of COMADE.

11.4 Summary

A typical evaluation process for knowledge based systems involves comparing the behavior of the system against that of several experts. Each expert and the system are posed the same design problems and the results are compared. However, a difficulty with such a validation approach is that design is a very subjective activity. The odds are that every generated design will be different. If the designs are different, then there can be no real basis for comparison.

Such an approach is also inherently flawed, at least as potentially applied to validating COMADE, because the design undertaken in COMADE is fundamentally different from design as performed by most composites designers. The industrial ambiguity of COMADE precludes any deep consideration of many factors that drive the typical material system design process. Additionally, the level of abstraction at which the design in COMADE is performed is dissimilar from that used by composites designers. A final distinction is the design intent: COMADE looks for a family of satisfycing material systems, while the composites designer wants the best design.

Another way of assessing the validity of COMADE and the design vision must be used. An idea for doing so can be seen in the definition of the word valid[1]:

> having the power to impress others as right and well-founded; founded on truth or fact, capable of being justified or defended.

1. Definition taken from Merriam-Webster's web-based dictionary, WWWebster. This reference can be accessed at http://www.m-w.com/netdict.htm.

How valid is the polymer composite material system designer COMADE? How valid is the design vision for composites? The metric for answering these two questions that can be extrapolated from the above definition is that they must be accepted by others as "right and well-founded."

What has arisen throughout the discussion in this chapter is that COMADE is perceived to be good at what it does. The MADEFAST experiment showed the use and acceptance of COMADE as a viable source of composite material designs. The validity of COMADE and its approach to the design of composite materials was borne out by the successful development of the GEAE-CODE system. The systems developed from COMADE show the favorable light in which it is viewed. Perception of the rationale as useful in other domains is a very favorable reflection on the capabilities of COMADE. In addition, COMADE has received accolades from its users [Heness, 1996a; Heness, 1996b] and other researchers (1993-1994 Michigan Materials & Processing Institute award for best research).

The design vision for composites has never been proclaimed as definitive. The interactions existing between material design, process design, and part design remain unclear. These interactions have become more defined because of feedback from users and examination of the composites design practice in industry. The separation of the design vision into portions for material, processing, and assembly has been accepted as potentially useful. However, due to the abstract nature of the design vision and the lack of explicitly detailed interactions, this acceptance cannot be translated into validation without further elicitation of the details of the design vision.

Just because COMADE and the design vision can be considered to have been validated does not mean that they are complete. It just means that they have been considered good at what they do. There is always room for improvement. Users of COMADE want it to do more and different things. The details of the design vision need to be discovered. To expand the capabilities of COMADE and the understanding of the design process, a detailed examination of the focus and content of COMADE must first be done. This examination should be done with an eye toward augmentation and customization of COMADE. The following chapter presents just such an examination.

CHAPTER 12

A Critical Examination of COMADE's Legacy

> *"The significant problems we face cannot be solved at the same level of thinking we were at when we created them."* Albert Einstein

In the ongoing search for knowledge and understanding, there are advances upon which one must reflect to fully appreciate their significance. The development of the material in this dissertation is one such advance. The understanding of composite material system design developed in this dissertation exhibits a marked maturity over its intellectual roots.

This chapter provides a recap and analysis of the advances for composite material design. The increased comprehension of the composites design process is presented in the following section with the material system design process. These two visions give a succinct global picture of composite design.

All engineers learn early in their education that learning is not all about glowing successes. With failure comes increased understanding. Only by critically examining what has been done can true advances be made. Thus, after the global picture, the discussion in this chapter turns to an evaluation of the limitations of the current understanding.

However, even with the increased capabilities and understanding that have been presented, there is always something else to consider. Therefore, this chapter concludes by discussing two approaches for expanding the capabilities of COMADE.

12.1 Composites Design Milieu

Throughout this dissertation, a gradual understanding of a new way to think about composites design has been building. From the initial consideration of the attempts of other researchers in Chapter Four, to the overview of previous research in the ISL in Chapter Five, this understanding had reached a plateau in Chapter Six. An initial conception that the old ways were not what the new way would be had formed.

Now, after several new contributions have intimated something different, reconsideration of the origins brings to light what has been discovered. This discussion examines the two aspects of composites design with which this conception started in Chapter Six: composite material system design, and global composite design.

12.1.1 Composite Material System Design Vision

Figure 63 shows the modified vision for composite material system design. The most significant difference in this vision is the absence of a projected functional reasoning component. During the development of COMADE, a reconsideration of the requirements for such a component was made. For this component to be implemented, substantial phenomenological models correlating fiber/matrix interactions with the mechanical properties of the final composite are needed. No such models exist, nor are they likely to ever exist.

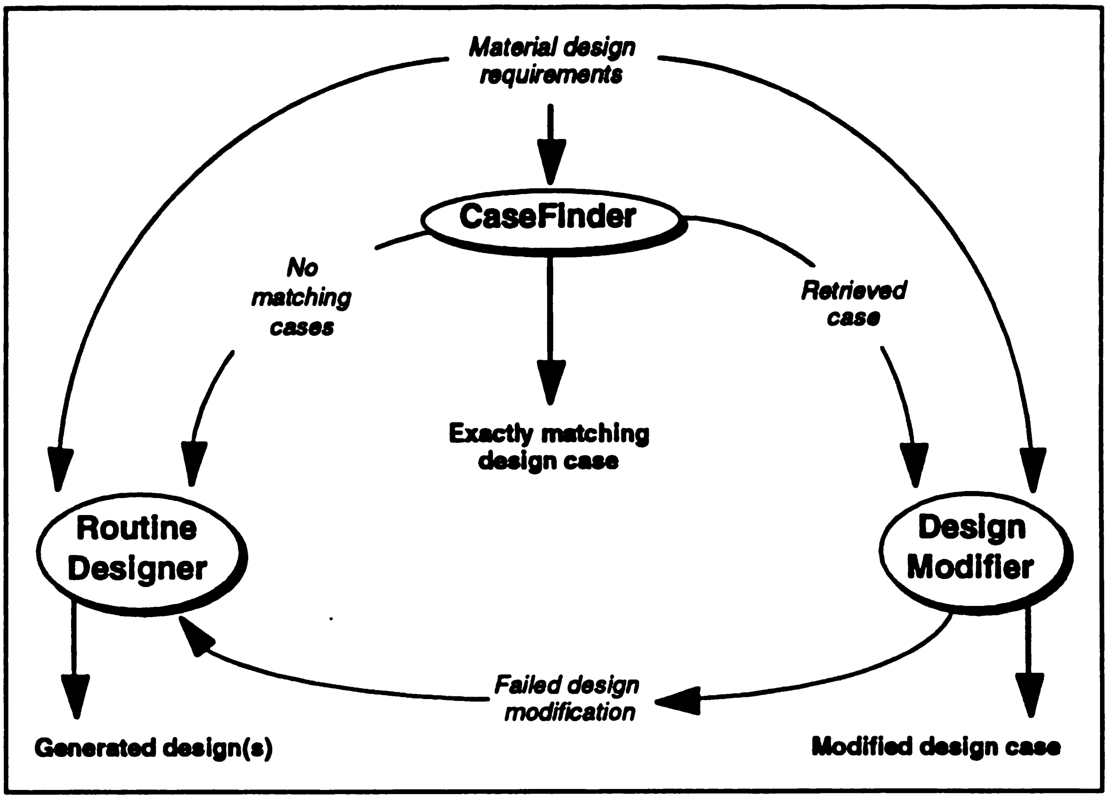

Figure 63. The current composite material system design vision.

12.1.2 Global Composites Design Vision

At the inception of this research, it was apparent that the vision of the composites design process should not be linear. The conception was that the design process was connected on many levels. However, the details of how these levels interact were unclear. Separating out the individual design domains was predicted to ease the design process by simplifying the domain coverage for each system developed to solve them.

The separation of the design process into three portions that interacted in some unknown way was depicted using three starbursts that did not mesh in any "easy" gear-like fit. This depicted the expected multiple interactions between the design areas, but neither the method of implementation nor the order of execution was mandated.

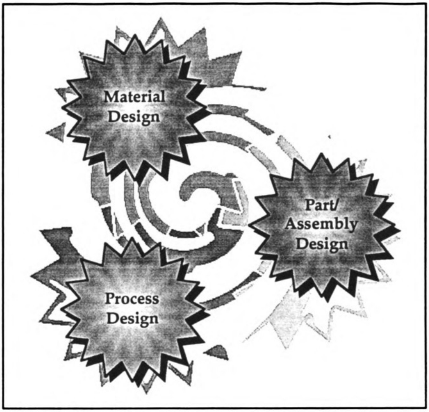

Figure 64. Design spiral for composites.

Figure 64 presents a modified version of the three starburst vision for the composites design process. The starburst overlay a design spiral that mingles aspects from each area with others. This intermingling occurs on several concentric rings. The design spiral tells a story of the entire design process, but does so without mandating particular problem solving techniques.

The design spiral story starts on the periphery and works toward the center. Taking material design as the starting point, notice the underlying shadow of the material design starburst. Beginning at approximately two o'clock in this shadow, notice that this area is completely isolated from the other portions of the design process: it is pure material design. By going through material design, an interaction with process design occurs, the first of three possibilities. At this point, the results of the pure material design are combined with process design to give a combined material/process depiction of the developing design. The spiral continues to wind it way toward the center, and in doing so encounters part/assembly design. The developing design now considers material/process/assembly, but at a less than detailed level. This process continues, with each portion of the design vision being revisited, but each time it is, the developing design has become more detailed and concrete. Finally, the spiral reaches the center, and the detailed design is complete. Similarly, the design spiral can start at either process design or assembly/part design.

This design spiral presents the composite design process as a combination of rigid phase level design (made much less rigid) with the segregated design vision of material-process-part. These two seemingly antithetical conceptions of the design process have been made compatible with the design spiral vision. While nothing is a panacea, this new vision represents a significant advance in thinking about designing composites.

12.2 Dissecting COMADE

Much like the analysis of CMatDesign in Chapter Seven pointed the way toward this research, this section presents an evaluation of the limitations of the current research. The intent is to learn what shortcomings, if any, exist in the current research, and how to address them in future research.

Specifically, three issues are addressed here. The general nature of the variables and data within COMADE has influenced the resulting understanding of the composite material system design process. The extent to which it has done so, and the changes to the understanding a more focussed approach could give, are presented. Several assumptions and heuristic used within COMADE to decide the final composite material properties are examined in detail. By analyzing the heuristics, an understanding of ways to improve such assumptions can be gained. Finally, this discussion presents the issues concerning case-based reasoning for the design of composite material systems.

12.2.1 The General Nature of COMADE's Domain Coverage

From the first formulation of the expanded material domain coverage in COMADE, an attempt was made to make the considered materials as generic as possible. This was done with the intent of keeping the entire composite material system design process at a conceptual level of representation. As discussed above, such unnatural forcing of a design problem to a level of the rigid phase level design process causes problems.

The general nature of COMADE's domain contributed to two specific problems. The first was the actual accumulation of data for materials. Very little of the data available in the literature is of a generic nature. Most of this data is highly specialized, and completely non-generic. The only way to get generic material data was to gather as much specialized data as possible about classes of materials, and then attempt to abstract out general descriptive data. The second problem caused by the general nature of the domain was generating inferencing patterns for this abstracted general descriptive data.

Much estimation and assumption in the property values and inference patterns could be avoided by using specific material formulations within the domain coverage. With such materials, a great wealth of detailed material properties data exists. By building COMADE on such a solid foundation, less abstraction would be required.

However, such a focus almost by necessity implies the targeting of specific material suppliers' product lines. Because COMADE was not intended to be company specific, and probably should remain so to have the greatest appeal, this presents a stalemate. The detailed materials knowledge exists, but it would be more appropriate to not use it. This is an issue that needs to be resolved. Perhaps the methods for extending the capabilities of COMADE presented at the end of this chapter provide an avenue for such a resolution.

12.2.2 An Analysis of Mechanical Properties As Used in COMADE

In Chapter Eight ("Input Variables" on page 189), the methodology used within COMADE to determine the mechanical properties of the final composite was presented. COMADE specified fiber materials and fiber lengths based on the mechanical properties of the material system constituents. It did so without using any numerical calculations. The absence of a numerical calculation capability within the ISL GT toolset forced the use of heuristics to enable this specification.[1] The following discussion analyzes how well the methodology used within COMADE worked.

The assumptions used within COMADE included:

- a 60% volume fraction of continuous, unidirectional fiber reinforcement,
- an upper bounding contribution from the MatrixMaterial to the mechanical properties,
- a simple rule-of-mixtures formula was used to determine the continuous fiber composite mechanical properties (See page 204), and
- a correlation between the relative moduli and strengths for different lengths of reinforcements was used to predict at the values for the chopped fiber composite.

The first assumption was merely a convenience to simplify the design problem by eliminating numerical calculations. A survey of commonly used industrial formulations of composites with continuous reinforcement shows a range of 50% - 70% reinforcement by

1. This capability has since been added to the toolset.

volume [Phillips, 1989]. Therefore, the use of 60% volume provides a representative depiction of possible material properties. The other three assumptions merit further consideration.

It can be seen by examining the mechanical properties of the neat resins (not shown here) that while the assumed upper bounding flexural modulus value (5 GPa) representa an upper bounding limit, the tensile strength value (50 MPa) underestimates the values of certain matrices by up to 200%. Whether this underestimation has any impact on the calculated mechanical properties of the composite remains to be seen.

Table 22: Calculated properties for composite materials versus fiber type.

	Tensile Strength (MPa)		Flexural Modulus (GPa)	
Fiber Type	**Chopped**	**Continuous**	**Chopped**	**Continuous**
No Fiber	50		5	
E-glass	546	2084	15	49
S-glass	717	2768	16	53
C-glass	527	2006	15	50
Quartz	548	2090	13	43
AS-4 carbon	630	2420	38	141
P-55 graphite	290	1058	60	229
P-100 graphite	366	1364	112	436
Kevlar-29	--	2180	--	52
Kevlar-49	--	2180	--	76
Kevlar-149	--	2060	--	114
Boron	507	1928	61	236
SiC	615	2360	63	242

Table 22 shows what the values for the mechanical properties would have been without using the relative value correlation (Table 13 on page 204). The values shown in Table 22 were determined using the rule-of-mixtures assumption with volume fractions of 60% and 40% continuous and chopped fibers, respectively. The values also still assume an upper bound for the matrix contribution to the property (as indicated).

The continuous fiber values for tensile strength are fairly close to those used in COMADE (Table 12 on page 203), as expected, except those for boron fibers. There was obviously an oversight in the original calculation for boron. As expected, the continuous values for flexural modulus are also similar. The chopped values determined without the use of the relative property correlation are significantly higher than those encoded within COMADE.

To further clarify the influences of the relative property correlation and the constant matrix contribution assumptions, the mechanical properties for both continuous and chopped fiber composites were calculated for two additional situations. Both calculations used the rule-of-mixtures heuristic. The first did not assume either a constant matrix contribution or a relative property correlation. The second used the assumed constant matrix contribution, but not the relative property correlation. The results of these two calculations were compared with the values encoded within COMADE.

Figure 65 compares all three sets of values for the tensile strength of E-glass fiber composites. The values generated assuming a constant matrix contribution -- without a relative property value correlation -- closely approximate a best linear fit of the data generated using the individual matrix values. This held for both chopped and continuous

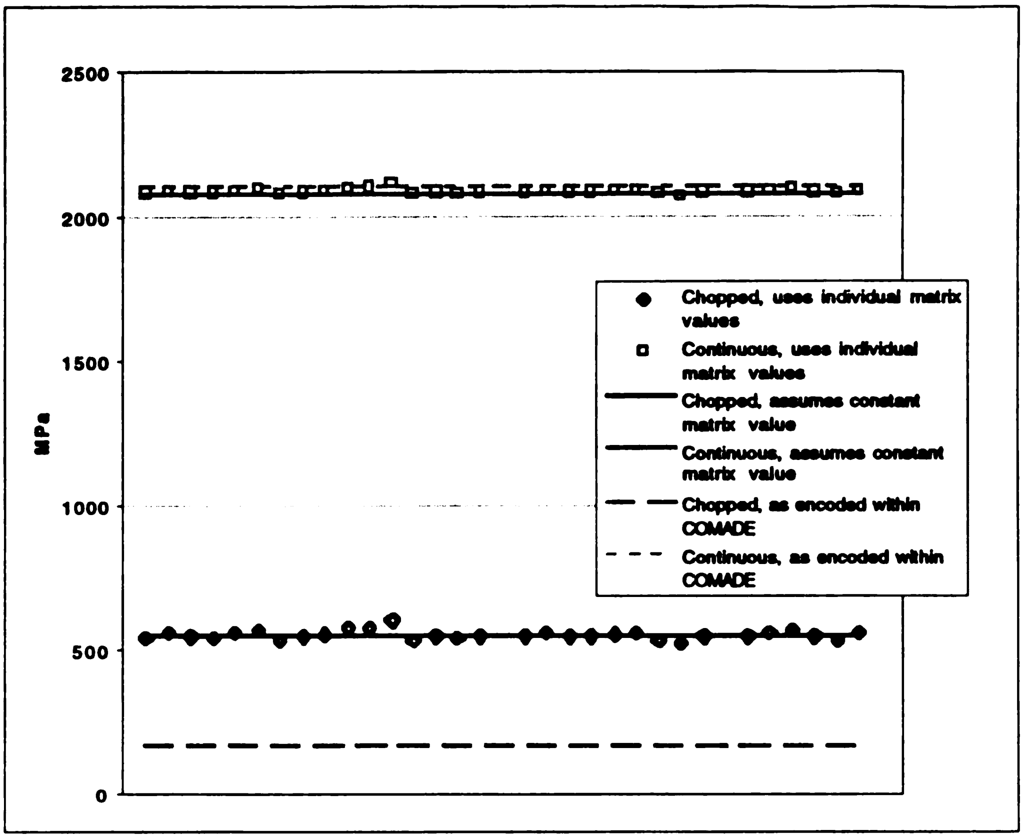

Figure 65. Comparison of tensile strength values for E-glass.

fiber composites. Therefore, at least for determining the tensile strength, the use of a constant matrix contribution is valid.

The tensile strength value encoded within COMADE for continuous E-glass is very similar, as expected. However, the value for the chopped E-glass tensile strength encoded within COMADE is significantly lower than the values calculated for the chopped E-glass composite. This suggests that the relative value correlation is not the best heuristic for determining the properties of chopped fiber composites. Therefore, the values within COMADE for the chopped fibers underpredict the tensile strengths of the final composite.

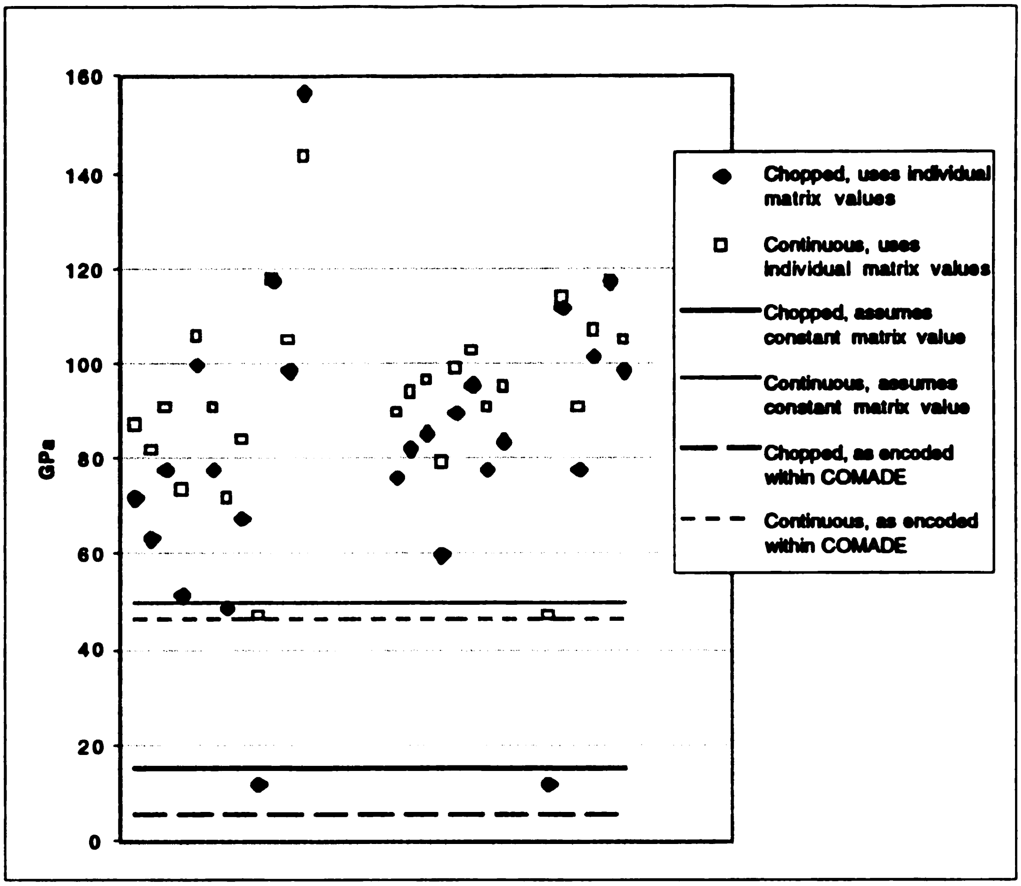

Figure 66. Comparison of flexural modulus values for E-glass

A similar comparison for the flexural modulus of E-glass composites yields quite different results. Figure 66 summarizes these results. The constant matrix contribution (without the relative property correlation) shows that the values in COMADE are too low, underpredicting the continuous fiber value by 3% and the chopped fiber value by 67%. However, the flexural modulus values determined without the continuous matrix contribution or the relative property correlation are significantly different from either of these. Even if there were a linear correlation for the flexural moduli, neither of the constant matrix contributions would be within 50%. Clearly, this lack of agreement suggests a significant discrepancy in the assumptions used for determining the flexural modulus.

The discrepancy in the flexural moduli points to the rule-of-mixtures assumption used to calculate all of the values in this analysis. An examination of the assumptions behind the rule-of-mixtures theory verifies this [Strong, 1989]. Most important among the assumptions is that the mechanical properties of the constituents whether they are in a composite. Therefore, a rule-of-mixtures assumption turns out to be a poor choice for determining the flexural modulus in COMADE.

A summary of the preceding analysis shows the difficulty facing composites designers when attempting to predict mechanical properties of composites. Certain assumptions work well for a few mechanical properties, and fail miserably for other properties. The validity of the assumptions used in COMADE was mixed.

The use of the relative value correlation was not very accurate for chopped fibers. The use of the constant matrix contribution and rule-of-mixtures for determining tensile strengths worked well. None of the assumptions used for determining the flexural modulus were consistent. However, lacking any better estimation, the use of the constant matrix contribution yields consistent values. Nevertheless, the mechanical property values decided using the heuristics are only estimates. Since they give a close (say, within 20-30%) estimate of the composite material properties, they are effective.

12.2.3 Case-Based Reasoning for Composite Material Systems

Given the typically cautious mindset toward the use of composite materials, the capability of designing a new composite artifact from an explicitly documented and approved previous design is very evocative. A review of how CBR was used in the CPD and EADOCS systems, and also with COMADE, shows three different emphases.

The CBR portion of the CPD system was never fully implemented. A case base with a few composite designs served as the inspiration for the retrieval methods used by the system. Due to the artificial environment created by such a small collection of sparse cases, these retrieval methods were skewed. Matches were made according to a four-tiered similarity method, but only if a case attribute was the same as that of the problem under consideration. An offshoot of this matching technique was that there was no allowance for close matches. Due to the sparseness of the case library, no explicit modification of the retrieved cases was ever attempted.

The EADOCS system, which also has a sparse case library, emphasizes the modification of retrieved cases much more than the retrieval process itself. The complexity of the design representation clouds the picture a bit, but the focal point of this system is the attempt to modify partial matches by using optimization.

The major issue that can be taken with CPD and EADOCS is the sparseness of the cases and case libraries considered. This sparse nature has contributed to misguided assumptions about how to do case retrieval and matching. Any case retrieval process must allow for close or partial matches to an attribute, or to a case. Those cases that match more closely obviously require less modification for use in the design process.

In addition, design modification is not well understood, and as a result there are many different approaches to performing it. In spite of the confusion, a common theme among all the different approaches does exist: the method used for design modification is dictated by the domain to which CBR is applied.

To properly address a design process with case-based reasoning, fully-populated case libraries, detailed case structure, principled modification methods, and retrieval with partial-match capability must be incorporated. Up to this point, one or more of these necessary ingredients has been missing from previous CBR implementations, which has in turn has limited the effectiveness of these systems.

Within the CBR extension to COMADE, an attempt was made to build upon the existing problem solving structure already present within COMADE and to use that for the modification and redesign problem. Case retrieval was performed using a probe-like similarity matching scheme, with an emphasis on those variables that affected the design problem most significantly. Allowance was made for partial matches within this matching scheme.

The modification method as outlined with the GT-influenced CBR tools in Chapters Nine and Ten seems principled. DesignMod for COMADE, while an effective proof of concept for design modification, often completely redesign whatever cases are sent to it. The example given in Chapter Ten resulted in such a complete redesign. This is because of the small and highly interconnected nature of COMADE's set of variables. Table 23 shows just how tangled the variable dependencies within COMADE are.

This analysis served as the basis of the sorting mechanism used for evaluating retrieved cases in CaseFinder. However, what it implies for DesignMod's approach to redesign of composite material system designs is not encouraging. When a case is sent to DesignMod to be modified, it's non-matching variables are flagged. DesignMod fashions its modification process around those variables. As Table 23 clearly shows, whenever either UseTemperature or ChemicalEnvironment is flagged, a total redesign will occur. This completely invalidates the use of CBR. Not only does COMADE have to run on its own, but there is also the overhead of the case retrieval process. Essentially the only time that

Table 23: COMADE variable dependencies and the impact on modification.

Variable	$\text{Direct}^{\text{Indirect}}$ Dependencies	% Redesign[a]	Direct Redesign	Implied Redesign
UseTemperature	$\text{MatrixMaterial}^{\text{ChemicalAgent}}$ $\text{FiberType}^{\text{FiberLength}}$ ChemicalAgent	100	75	50
		100	80	30
ReqTensileStrength	$\text{FiberType}^{\text{FiberLength}}$ FiberLength	50	50	25
		50	50	30
ReqFlexuralModulus	ChemicalAgent FiberLength $\text{FiberType}^{\text{FiberLength}}$	75	75	25
		60	60	20
ChemicalEnvironment	$\text{MatrixMaterial}^{\text{ChemicalAgent}}$ $\text{FiberType}^{\text{FiberLength}}$ ChemicalAgent	100	75	50
		100	80	30
StressEnvironment	$\text{MatrixMaterial}^{\text{ChemicalAgent}}$ ChemicalAgent	50	50	25
		50	50	10
FlameRetardance	$\text{MatrixMaterial}^{\text{ChemicalAgent}}$ ChemicalAgent	50	50	25
		50	50	10
HumidEnvironment	$\text{MatrixMaterial}^{\text{ChemicalAgent}}$ ChemicalAgent	50	50	25
		50	50	10

a. The two values given are for the raw redesign (first row) and weighted redesign (second row). The weighted redesign value considers the computational effort to determine each output variable. The ordered ranking is: MatrixMaterial (4), FiberType (3), FiberLength (2), and ChemicalAgent (1).

DesignMod does not do a complete redesign (i.e., run COMADE from scratch) is when a single variable other than UseTemperature or ChemicalEnvironment is flagged for modification.

Over the course of this development, shortcomings of the approach used in COMADE have become apparent. With the current set of inputs and outputs, there is no real benefit to justify the use of CBR, as typical run times for generating results from previous cases are longer than generating the designs from scratch with the stand-alone version of COMADE. This is not to say that there is no advantage in using the CBR approach. The sparse nature of the input/output set, coupled with the fact that all cases within COMADE have the same structure, hide the benefits of a CBR approach. The advantage of a CBR approach should become apparent when it is applied to a more rich design space.

12.3 Expanding the Capabilities of COMADE

COMADE, while industrially relevant in its domain coverage, was developed to be generic in its considerations (See "The General Nature of COMADE's Domain Coverage" on page 267.). This restriction was made with the intent of making COMADE as broadly applicable as possible. However, as presented in the previous chapter, this restriction has provoked specific requests for expansions of COMADE's capabilities.

Although widely varied, these requests for expanded capabilities in COMADE can be loosely grouped as:

- a detailed consideration of a particular processing technology,
- a focussed consideration of a particular industry's concerns, or

- a replacement of COMADE's materials with those of a particular materials supplier.

More detail exists beneath the surface of these requests. By specifically considering how COMADE would have to be modified to address these requested scenarios, two different types of extension become apparent. The first type involves the incorporation of additional capabilities, most likely by the addition of new knowledge. This is called interstitial extension. The second type requires the explicit refocusing of COMADE's design problem emphasis, and is called mutational extension. The distinction between these types of extension can be encapsulated by examining the design intent. Interstitial extensions augment the material design capabilities of COMADE, while mutational extensions expand the scope of the design problem beyond material design.

The following discussion further delineates interstitial and mutational extension through the presentation of concrete modification examples. The details of interstitial expansion are brought out in the discussion of an implemented extension to COMADE for interphase design. The concluding section presents mutational extension using a planned application of COMADE to resin transfer molding.

12.3.1 Interstitial Extension

The composites interphase system was developed using COMADE as the foundations upon which it was built [Wilenski *et al.*, 1995]. Without changing the existing design knowledge structures in COMADE, this system generated composite material systems by considering additional theoretical calculations and heuristics. The interphase designer used COMADE as a skeletal structure upon which it built, adding further levels of detail to the polymer composite material system design problem representation.

The interphase designer used the concepts of surface wetting, miscibility, and functional group chemistry to define an interphase providing ideal compatibility between the constituents of a potential material system (COMADE's output). The interphase was represented within the system as a combination of three things:

- a fiber surface treatment that identifies the surface chemistry and energy of the fiber,
- a binder that both wets the fiber and is miscible in the polymer, and
- a chemical coupling agent that chemically bonds the fiber to the bulk matrix.

The interphase system retained the entire specialist hierarchy of COMADE, adding two specialists after those in COMADE. The COMADE agent hierarchy was modified by expanding the number of steps for each COMADE design task. Figure 67 shows how the interphase system fit within the existing structure of COMADE. The shaded nodes are new to the concerns of the interphase system, while the other nodes are from COMADE.

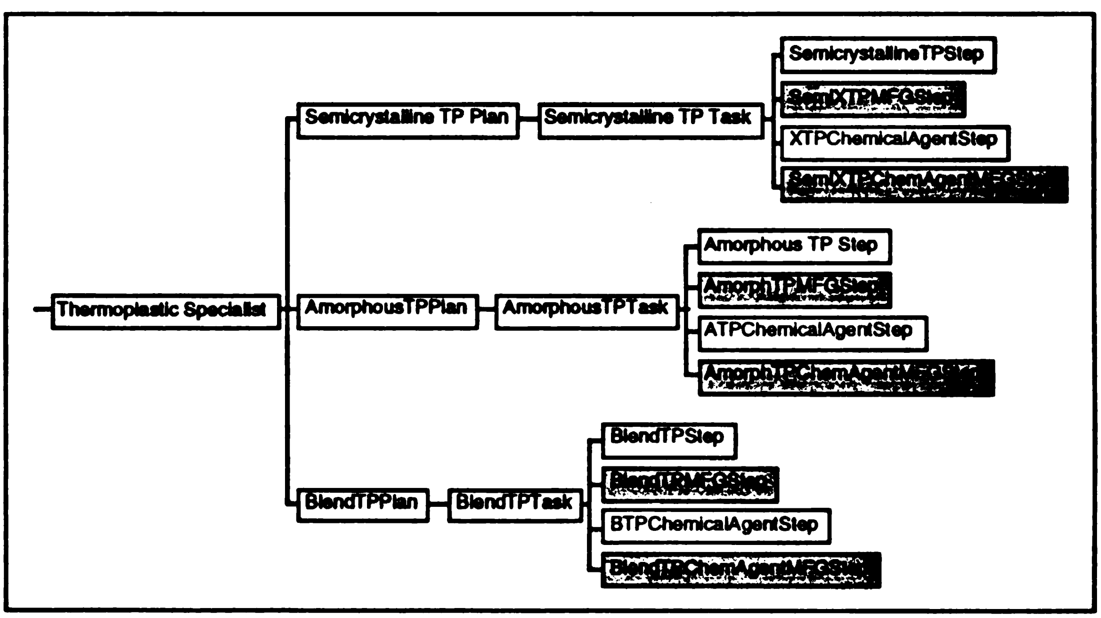

Figure 67. Interstitial extension of COMADE's thermoplastics agent hierarchy.

The interphase system added 26 steps to the structure of COMADE, and an additional 24 agents within its unique agent hierarchy addition. This increase in the number of agents was directly connected to the modified formulation of the design problem. The total design space corresponding to this design problem within the interphase system was three orders of magnitude larger than the design space in COMADE. Unfortunately, this design space was larger than is currently computationally feasible to consider using the ISL's implementation of routine design.

At the root of this extension to COMADE lay a carefully constructed design problem. It was easily incorporated into the existing structure of COMADE. However, the design space this system attempted to consider was overambitious. A scaled down design space would enable the interphase system to serve not only as a material system designer, but also as a predictor of viable and novel material systems.

12.3.2 Mutational Extension

A substantial alteration of the makeup of the design problem in COMADE typifies a mutational extension. Portions of COMADE's knowledge structure remain, but the specific details embedded within that structure are significantly different. The GEAE-CODE system development ("Integrated Design in an Industrial Setting" on page 249) can be considered to have been a combination of a mutational extension of COMADE and an inspiration by COFATE.[1] Instead of revisiting that development, this discussion centers on the mutational extension of COMADE to consider a fabrication technology, specifically resin transfer molding (RTM).

1. Think of the accompanying 'Inspired by actual events' tag that is often seen on broadcast television's movies of the week. In reality, very little of the actual story ever is presented, and often in nearly unrecognizable variations. The same can be said for how COFATE contributed to the GEAE--CODE system.

To properly apply COMADE to a particular fabrication technology, principled adjustments must be made. For the modification to specifically address concerns for RTM several portions of COMADE must be changed. The material domain coverage must be altered to address not only specific brand names of materials, but also to address only those materials intended to be resin transfer molded. Table 24 shows possible material systems for use in such a version of COMADE.

Table 24: Possible material systems for an RTM version of COMADE.

<table>
<tr><th>Matrix Material</th><th>Chemical Agent</th><th>Fiber</th><th>Fiber Form</th></tr>
<tr><td>polyesters</td><td rowspan="8">as dictated by
specific choice
of resin</td><td rowspan="8">graphite
carbon
glass
ceramic
aramid
boron</td><td rowspan="8">continuous strand mat
woven roving/fabric
unidirectional roving/fabric
chopped-strand mat
preform
veil/surface mat</td></tr>
<tr><td>vinyl esters</td></tr>
<tr><td>polybutadienes</td></tr>
<tr><td>epoxies</td></tr>
<tr><td>polyimides</td></tr>
<tr><td>bismaleimides</td></tr>
<tr><td>low-performance thermoplastics</td></tr>
<tr><td>engineering-grade thermoplastics</td></tr>
</table>

The design problem representation could potentially be very similar to that used in COMADE: MatrixMaterial, ChemicalAgent, Fiber, and FiberForm. However, such a representation is obviously a simplification of the material description, as typical resin systems used for RTM are complicated (and most often proprietary) mixtures of a resin, curing agent, catalysts, fillers, pigments, promoters and inhibitors.

There are many variables to be considered when choosing a resin for RTM. Typical rough selections of resins are based on the generic categories of application/performance areas and resin types/characteristics.[1] Generally, RTM resin systems have a long pot life (at least two hours) and low viscosity at the transfer temperature, short gel time at the curing temperature, and low levels of outgases, volatiles, and cure by-products.

Reinforcement selection is mainly guided by the performance and cost requirements of the end-use application. However, other issues such as the physical form of the reinforcement, the sizing (if any), and the type of stitching, can also be considered.

Depending upon the specific materials included in the domain coverage for such a system, many or all of the variables for resin and reinforcement must be considered. The combination of the input variables and the design problem representation guides how the knowledge structure of COMADE must be changed.

The overall knowledge structure will be inspired (at least) by that of COMADE. However, modifications must be made to reflect the modified domain of interest. The decomposition of the matrix design problem within COMADE will not be sufficient for an RTM-targeted system. The distinction between thermoplastics and thermosets as general classes of materials must be eliminated, and the individual materials considered separately. The structure of the reinforcement specialist will change markedly, depending upon the specific knowledge used to figure out appropriate combinations of fiber materials and fiber forms. The details of these changes to the knowledge structures of COMADE cannot be explicitly delineated until such a system is built.

1. See pp. 168-171 of [ASM International, 1987] for a detailed discussion of these categories.

Besides the above modifications, all of the knowledge within the RTM system's pattern matchers must be generated from scratch. Due to the data abstractions and generalizations within COMADE's matching tables, none of that knowledge will be applicable for the RTM-specific considerations. There is minimal overlap between the existing variables used in COMADE and those projected for use in the RTM system.

Despite the detailed modifications to COMADE that such a processing technology-focussed system would have, the basic design intent would be different. The RTM system must consider (and possibly suggest) processing information, which removes this system from the strict material design scope of COMADE. While vestiges of COMADE may exist in the final RTM system, the surroundings will have mutated beyond the approach and intent of COMADE.

CHAPTER 13

Summary and Conclusions

Throughout the preceding chapters, a comprehensive narrative about the design of polymer composite material systems has been developed. Common threads and repeating patterns of thought have been woven into the fabric of this narrative. All of the detail in the tapestry may overshadow the underlying ideas, but they exist nonetheless.

This chapter presents a condensed version of these ideas. An itemized summary of the specific contributions of this research is presented first. Following the summary, several conclusions about the underlying ideas in this dissertation are presented. Finally, several possible paths for future exploration are identified.

13.1 Summary of Contributions

This research has cultivated a significant and substantial understanding of the engineering design process for polymer composite materials. Specific contributions to this increased understanding include:

- a new way of thinking about the overall composite design vision - the design spiral. ("Global Composites Design Vision" on page 264) and (Figure 64 on page 265.)

- the accumulation and organization of composite material system design knowledge into a usable framework - COMADE. ("COMADE: A Composite Material System Designer" on page 182)

- the development of a generic task influenced case-based reasoning methodology for storing and retrieving past designs - CaseFinder. ("CaseFinder" on page 219)

- the specific application of CaseFinder to the reuse of composite material system designs. ("Designing Composite Material Systems with CBR" on page 232)

- a detailed analysis of the previous composite material design systems in the ISL, and an identification of many unclear assumptions. ("Dissecting CMatDesign" on page 159)

- an increased understanding of the composites design process in general, and the composite material system design process in particular.

- the successful use of COMADE within DARPA's MADEFAST experiment. ("An Experiment in Distributed Design" on page 246)

- the ongoing use of COMADE as a framework for the development of an industrially-focussed decision support system - GEAE-CODE. ("Integrated Design in an Industrial Setting" on page 249)

- structured representation of potential modifications to COMADE - interstitial and mutational extension. ("Expanding the Capabilities of COMADE" on page 277)

- an analysis of previous related approaches to composites design. ("Germane Approaches to Composites Design" on page 91)

- identification of the problem of scope creep and the accompanying increase in computational complexity in these approaches to composites design. ("Design Scope vs. Focus" on page 124) and (Figure 26 on page 125)

- detailed use of the preliminary integration architecture within the ISL's GT toolset, and testing of its capabilities.

- expansion of various capabilities of the ISL's GT toolset implementations: commenting individual row within matching tables, caveats for input variables. Also, provided an impetus for the addition of units to numerical variables and the capability to do numerical calculations.

13.2 Conclusions

Of all the specific contributions listed above, three ideas merit further discussion. Each of these refers to an original goal of this research. The first idea addresses how the contributions listed add up to a persuasive paradigm for engineering design, and what that means for the design of composite material systems. The second idea builds on the generation of the paradigm by discussing instantiations of it that other engineers can use. Finally, it is shown how the lessons learned in the discovery of the paradigm influence the comprehension of phase level design intertwined with a segmented design vision.

13.2.1 A Paradigm for Composite Material System Design

The initial conception of the design of composite materials included many things. Throughout the development from Epox-I up to CMatDesign, more items were added to this vision. These items included redundant and unclear material properties. Also included in this vision of composite materials was information related to the processing of the material, but only for a specific fabrication technology: autoclave processing with hand-layup. This assumption of a fabrication technology affected the design representation by forcing the inclusion of a partial list of processing parameters. Also assumed was that the composite material was a thin film laminate.

Before any progress could be made toward the initially-mandated expansion of the composite materials domain coverage, the design representation had to be crystallized. And even before that could occur, the assumptions within the previous conception of a composite material had to be identified and eliminated. Therefore, before there was a system called COMADE, there was a significant clarification of the definition of composite materials. All of the assumptions made about processing and architecture were removed from the definition of a composite material. Composite materials were defined to be a combination of a polymer matrix (plus any necessary chemical additives) and a fiber, either chopped or continuous. The processing information was relegated to process design. The architecture assumptions were discarded.

Later, the term *composite material design* was even modified to composite material system design to avoid confusion. By cleaning up the design representation, the stage was set for implementing a new system for designing composite material systems: COMADE.

13.2.2 The Instantiated Paradigmatic Design Process

With the design representation for composite material systems firmly specified, the accumulation and organization of design knowledge into a usable tool were then completed. The decision support system for polymer composite material systems design, called COMADE, embodied this material-specific design representation. It also presented a usable structure for the additional accumulation and reuse of design knowledge. COMADE was an instantiation of Petroski's persuasive paradigm for engineering design. Simply put, COMADE is the state of the art for designing composite material systems. No other system exists that specifically addresses the design of composites as succinctly or as effectively as does COMADE.

COMADE was a new way of designing composite materials in that it only considered the design of the material and did so without any processing information. COMADE also included an order of magnitude expansion over the capabilities of the previous system (CMatDesign) in the materials it could design. COMADE has been tested in various forums and has proven its capabilities for material design. It served as an integral part in the design of the MADEFAST seeker casing ("An Experiment in Distributed Design" on page 246) and has also served as the framework for an extensive commercial development for GE Aircraft Engines ("Integrated Design in an Industrial Setting" on page 249). COMADE has also served as the foundation of a principled extension for designing a composite material by considering the matrix/fiber interphase [Wilenski *et al.*, 1995].

The knowledge and design representation upon which COMADE was based has also proved useful for the specific application of a generic task influenced implementation of a case-based reasoning tool (CaseFinder). With DesignMod, CaseFinder presents a framework for specifically reusing past design experiences. To a large extent, the successful application of these tools to composite material systems was due to the knowledge used from COMADE.

Combining all three of these tools provides a complete implementation of the composite material system design vision shown in Figure 41 on page 154. This is the first time in the ISL's development of decision support systems for composites design that an entire design vision has been implemented. This is a significant contribution.

13.2.3 Abstractions of the Design Process

One recurring issue in the work of other researchers and an important notion in this work (at least initially) is that of conceptual design. There is almost total agreement that it is the most efficacious portion of the design process for the application of AI systems. However, there is no tangible or consistent interpretation of what defines part of the design process to be conceptual. Most often, a particular interpretation is influenced directly by the specific design problem to be solved. This is not entirely unexpected. Any direct comparison of the various interpretations degrades quickly into a futile semantic exercise.

However, one point becomes abundantly clear: a specific label of being philosophically conceptual always results in an implementation that is at best only partly conceptual. The most common pitfall is the tendency to get distracted by details at a level where these details should be unimportant.

One would think that if every approach to composites design (and this includes the approach used in COMADE) were intended to be at a conceptual level and were not, they should be flawed and should not work. That is, of course, not so, as many other approaches have resulted in successful implementations. This suggests that something else is going on.

Designing by strictly following a rigid phase level design process does not address the design problem at the correct level of granularity, and therefore does not allow for the complex interaction of design issues. COMDES ("COMDES" on page 107) used a task level design process in which the boundary between conceptual and embodiment design was crossed repeatedly. From this system it was seen that while the phases of design are convenient for thinking about the design process, they should not dictate the approach taken in performing the design process.

While researchers may profess the benefits of the conceptual design phase, they cannot easily carry out systems at a solely conceptual level because it is not a natural approach for humans. There is always some mixing of the levels of abstraction. When an apple is mentioned as a concept, additional details automatically spring to mind (e.g., red, juicy, Grannysmith, etc.). This is already crossing the line into another phase of detail. Of course, one can argue that this is a false analogy, as an apple is really the embodiment of a more abstract concept of a fruit. This shows that it is difficult to be precise with the semantics of what are and are not concepts or embodiments.

The representation of the composite material system within COMADE was not strictly conceptual either, in spite of an attempt to make it so. A typical design output from the system is: ABS Matrix Material, no Chemical Agent, P55 carbon Fiber Type, and continuous Fiber Length. More truly conceptual design would be something like: Semicrystalline Thermoplastic Matrix, High Modulus Carbon Fiber, and Continuous Fiber Length. But even this could be expressed in a more abstract form of: Thermoplastic Matrix, Carbon Fiber, Continuous Fiber Length. One could very easily lose focus on the design process by getting bogged down in the semantic games of expressing things as entirely conceptual.

Continuing this demonstration of the difficulty of completely conceptualizing the design process (or representation), it pays to look at EADOCS ("EADOCS" on page 113). This system has the most conceptual approach to problem solving and to the representation of the composite of all the systems considered. It is no great coincidence then that it also is the most unwieldy and forced of them all.

The lesson to take away from all of this is, again, that the rigid phase level design process for composites is useful only as a way of thinking about the design and not as a guideline to accomplishing the design process. This implementation should be driven specifically by the tasks comprising the problem to be solved. There should be no concern given to compartmentalizing the conceptual phase design apart from the embodiment or detail phase design.

The design spiral vision (Figure 64 on page 265) developed during this dissertation provides a convenient remedy for all of this confusion. It presents the composite design process as a combination of rigid phase level design (made much less rigid) with the segregated design vision of material-process-part. The spiral allows for so-called conceptual design to occur between material & process, or part & material, and so on. It also acknowledges similar recirculation ideas present in EADOCS. These two seemingly antithetical design process visions have been made compatible with the design spiral vision. While nothing is a panacea, this new vision represents a significant advance in thinking about designing composites.

13.3 Future Directions

In the last portion of Chapter Twelve, two classes of extension to COMADE were outlined. These definitions help to provoke additional developments based on this research, including applications of COMADE specific to industry and fabrication technologies. These developments and others remain to be done. Some possible future directions for research that have appeared throughout the course of this research include:

- developments that capitalize on computational capabilities within the ISL GT Toolset that have recently been added as offshoots of this and other related research. Specific capabilities to add include numerical calculations, caveats, and structured pattern matcher row commenting.

- update the current web-based version of COMADE to take advantage of the significant advances that have occurred since the demo was placed on the web in June of 1994. This could potentially include the reimplementation of COMADE in Java, among other things.

- add in a capability to store cases using the CaseFinder methodology. This would be a natural extension of the retrieve algorithm already in place. A new case could query the existing case hierarchy to find out where it should be located.

- readdress the determination of the mechanical properties within COMADE using the numerical variable functions.

- add in consideration of more choices for abstract fiber forms within COMADE: unidirectional continuous, random inplane chopped, bidirectional continuous. Get away from the idea of needing to eliminate all part/assembly and processing information within the material system designer.

- develop systems that are usable at multiple levels of design detail, alá the design spiral vision. It would be very evocative to have a system that could be used once to do design at a more conceptual level, and then again (depending upon the input to it) to do more detailed designing - whatever its emphasis may be. This might be possible using rough design in the current ISL MDSPL tool.

- implement a resin transfer molding targeted version of COMADE, taking advantage of the mutational extension description in Chapter Twelve. Possibly include portions of the design approach used in COFATE.

- fully utilize the numerical calculation capability within the ISL toolset to reassess the way COMADE addresses the consideration of numerical variables.

- figure out a better way to explicitly monitor the users (and use) of COMADE on the web; collect demographics, restructure the knowledge within COMADE to more adequately address the most common users.

- apply the methodology used in (and learned in the development of) COMADE to other chemical engineering domains (e.g., chemical processes).

- get the interfacing between CaseFinder and DesignMod stabilized.

- continue to expand a case library of composite material system designs. Right now, these must be generated from COMADE. However, as the understanding of the composite material system design process grows, cases from outside COMADE's experience could probably be included in this library.

- explore ways of placing the library of designs on the web, with enhanced documentation features. Also, consider ways to use the web for delivery and presentation of the CBR approach developed here.

- modify the current implementation of CaseFinder to include multiple case assignations to nodes. This is simply a matter of changing the case library interface to a multiple selection-in-list.

- automate the case hierarchy building process within CaseFinder. This would enable the approach to be used with large case libraries much more effectively.

All these possible avenues for further exploration speak to the wide-ranging nature of this research. They also point to the potential use of the developed methodologies and tools in widely disparate engineering domains, not just for composites. Think back to the Introduction to this dissertation and the hydrodealkylation process presented there. It is interesting to speculate how the methodologies developed for COMADE and composites design can be applied to more mainstream chemical engineering design problems. The potential utility is obvious. The execution is yet to come.

APPENDICES

Appendix A

Nomenclature within COMADE

There are a lot of abbreviations and acronyms present within COMADE, mostly for ease of representation within the routine designer. All of the abbreviations used for MatrixMaterial and ChemicalAgent within COMADE are expanded and explained below.

The abbreviations (and explanations) used for MatrixMaterial are:

- 4,4'-MDA-BMI: 4,4'-Bismaleimido- Diphenylmethane
- ABS: Acylonitrile-butadiene-styrene
- ABS/PBT: Acrylonitrile-butadiene-styrene/Polybutyleneterephthalate
- BPA Fumarate Resin: Bisphenol A...
- Chlorendic Resin: blended polyester resin
- DGEBA: Diglycidylether of bisphenol A
- Epoxidized Phenolic Novolac:
- Isophthalic Resin: blended polyester resin
- Orthophthalic Resin: blended polyester resin
- PA66: Polyamide-66 (Nylon-66)
- PAI: Polyarylimide
- PAS: Polyarylsulfide
- PBT: Polybutleneterephthalate

- PC: Polycarbonate
- PC/ABS: Polycarbonate/Acrylonitrile-butadiene-styrene
- PC/PBT: Polycarbonate/Polybutyleneterephthalate
- PEEK: Polyetheretherketone
- PEI: Polyetherimide
- PET: Polyethyleneterephthalate
- Phenolic Novolac Resin: two stage cure resin
- Phenolic Resole Resin: one stage (heat) cure resin
- PMR 15 Monomers: Polymerizable monomer reactants, blend 15; (NE/MDA/BTDE) - monoethyl ester of nadic anhydride, 4,4'-methylene dianiline, diethyl ester of 3,3',4,4'-benzophenonetetracarboxylic acid dianhydride
- PPO: Polyphenyleneoxide
- PPS: Polyphenylenesulfide
- PSU: Polysulfone
- TGETPE: Tetraglycidyl ether of tetraphenol ethane
- TGMDA: Tetraglycidylmethylene Dianiline
- Thermid 600 Oligomers: BTDE/APB/APA (4,4'carbonyl-bis-1,2-benzende dicarboxylic acid, -diethyl ester; 1,3-bis (3-aminophenoxy) benzene; 3-aminophenyl actylene)
- Vinyl Ester Resin: blended polyester resin

The abbreviations (and explanations) used for ChemicalAgent are:

- 0,0'-Diallyl BPA: 0,0'-diallyl bisphenol-A
- 1,4-Tolyl-BMI: 1,4-bismaleimido-2- methylbenzene
- DAP: diallyl phthalate

- DDS/BF3-MEA: 4,4'-diaminodiphenyl sulfone with boron trifluoride monoethylamine
- DEAPA: diethylaminopropylamine
- Dicyanate BPA: dicyanate bisphenol-A
- EtOH: ethanol
- HMTA: hexamethylenetetramine
- MABA: m-Aminobenzoic Acid
- MeOH: methanol
- MPDA: m-phenylenediamine. Also known as "C1."

Appendix B

Enumerating COMADE's Knowledge

This Appendix details the decision-making knowledge within COMADE apart from the confines of the routine design methodology. Therefore, there are no agent or specialist hierarchies, nor or there any structured pattern matching tables. Instead, the knowledge in COMADE is presented in the form of lookup tables and figures. It is possible to determine a design for a polymer composite material system using the following material. However, finding multiple solutions to a set of input specifications is not feasible. In such a situation, the use of COMADE is required.

The material that follows presents a very entangled group of figures and tables. All of the interactions between the different input variables used in COMADE are impossible to disconnect, and therefore overlap exists for nearly every item presented.

The first group of figures presents the UseTemperature as a means of determining polymer types. An overview figure helps to pinpoint the overall type, and the following figures for each polymer type help to determine a specific polymer. In a couple of the polymer type UseTemperature figures, there is no mention of the MatrixMaterial options for that polymer type. Instead, the ChemcialAgents are listed. This is due to the lack of any UseTemperature dependance for the MatrixMaterial. A series of tables follow these figures. These table encompass the remaining design knowledge within COMADE.

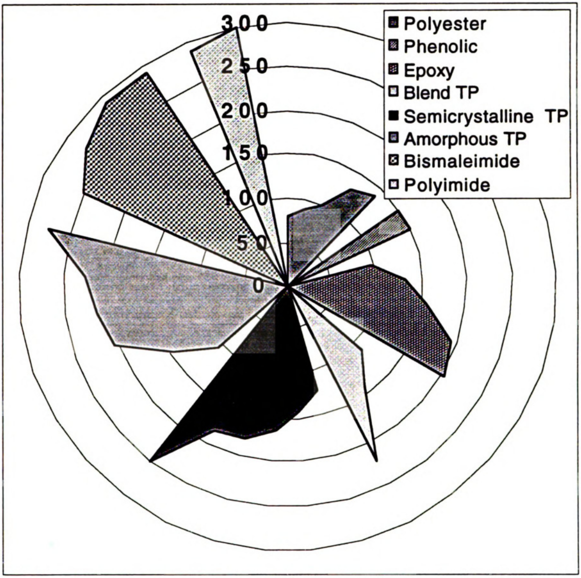

Figure 68. Use temperatures for all material types in COMADE.

A top-down progression through the vertical list of material types in the legend corresponds to a clockwise pattern of the lobes in the figure, starting just left of 12 o'clock.

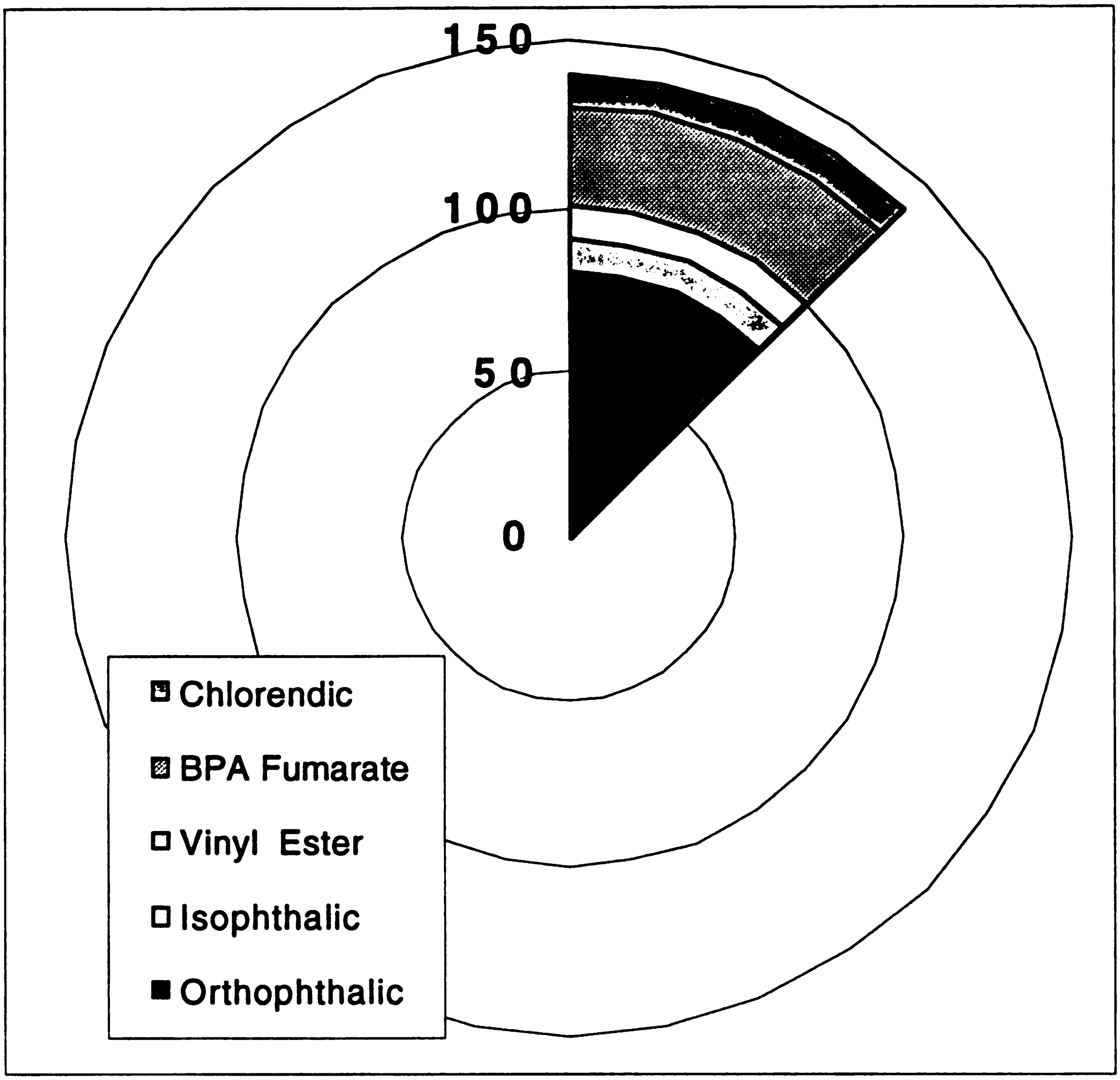

Figure 69. Polyester use temperatures in COMADE.

A top-down progression through the vertical list of materials in the legend corresponds to an outside-in progression through the multiply highlighted lobe.

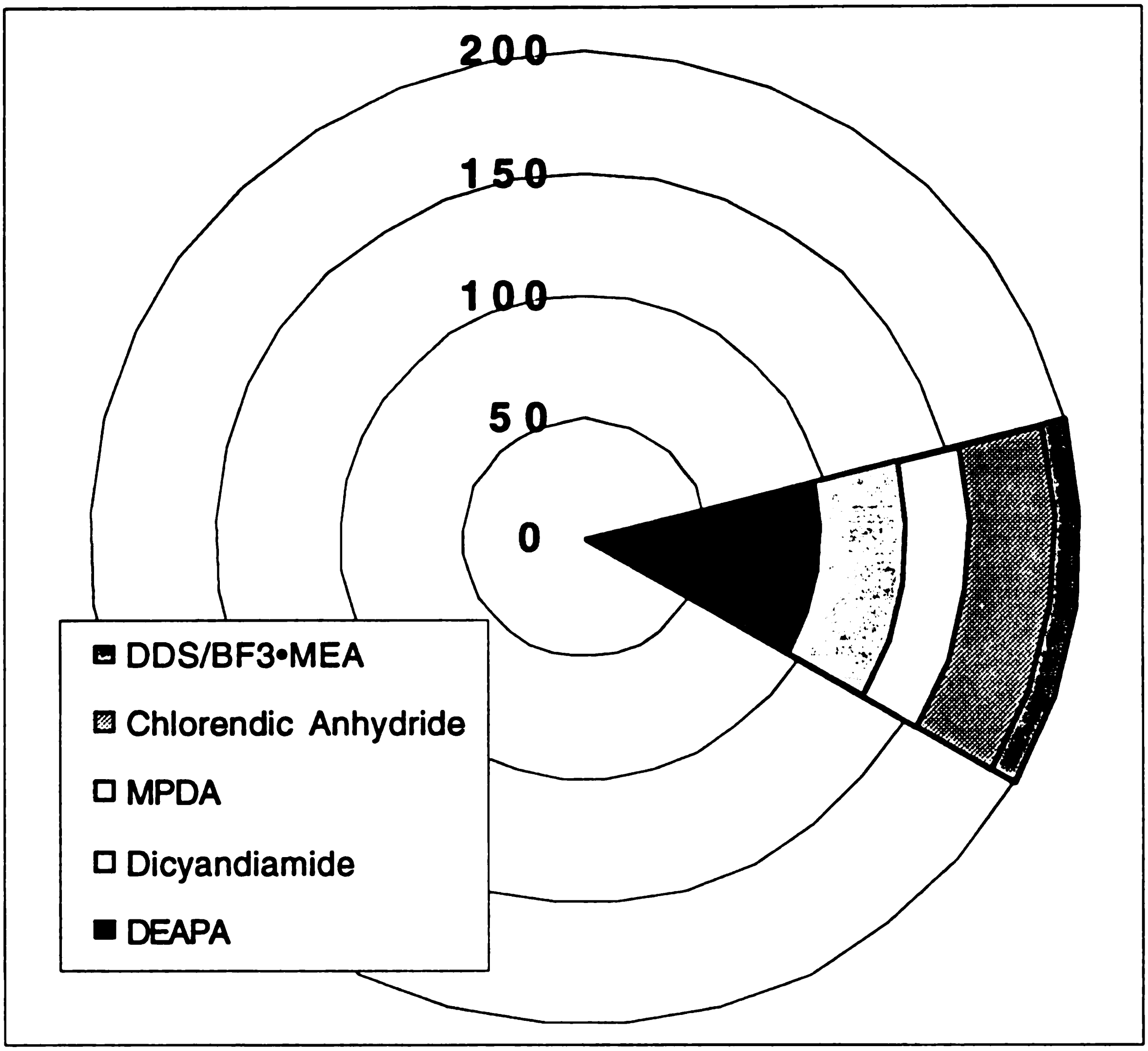

Figure 70. Epoxy use temperatures in COMADE.

A top-down progression through the vertical list of materials in the legend corresponds to an outside-in progression through the multiply highlighted lobe.

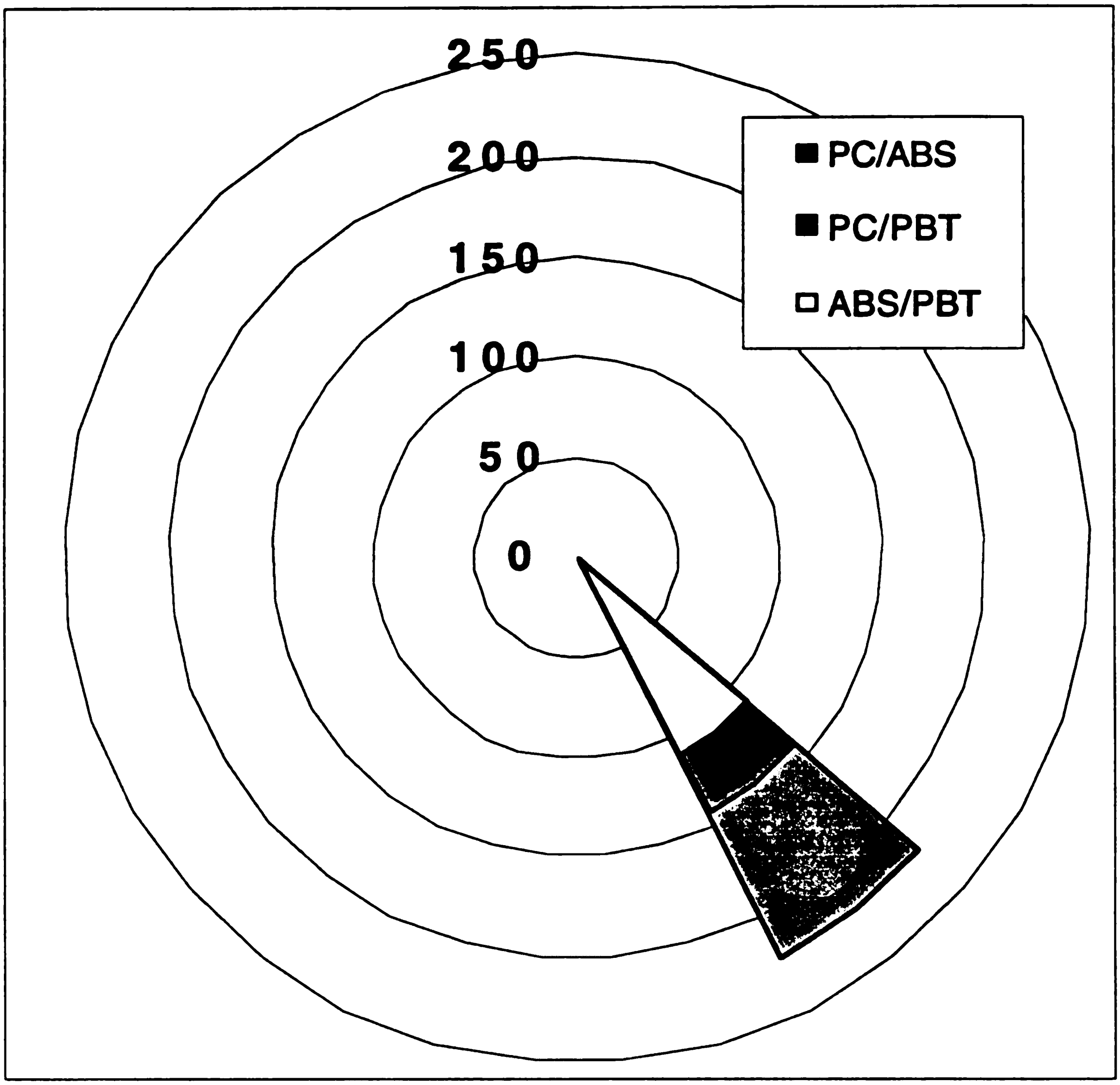

Figure 71. Blend thermoplastics use temperatures in COMADE.

A top-down progression through the vertical list of materials in the legend corresponds to an outside-in progression through the multiply highlighted lobe.

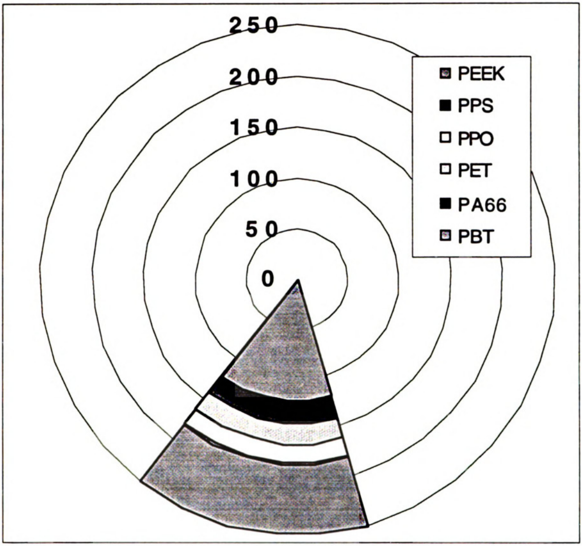

Figure 72. Semicrystalline thermoplastics use temperatures in COMADE.

A top-down progression through the vertical list of materials in the legend corresponds to an outside-in progression through the multiply highlighted lobe.

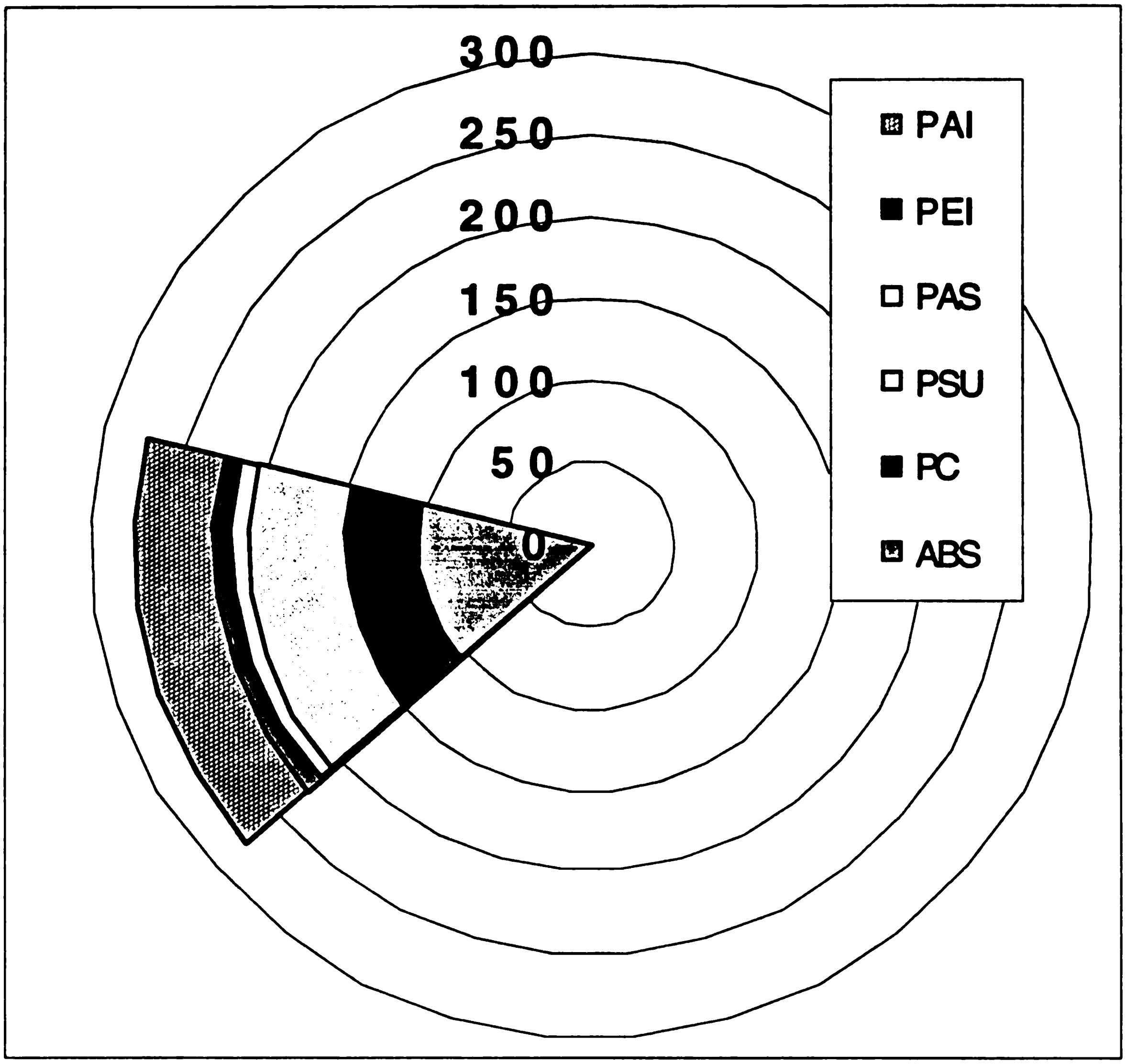

Figure 73. Amorphous thermoplastics use temperatures in COMADE.

A top-down progression through the vertical list of materials in the legend corresponds to an outside-in progression through the multiply highlighted lobe.

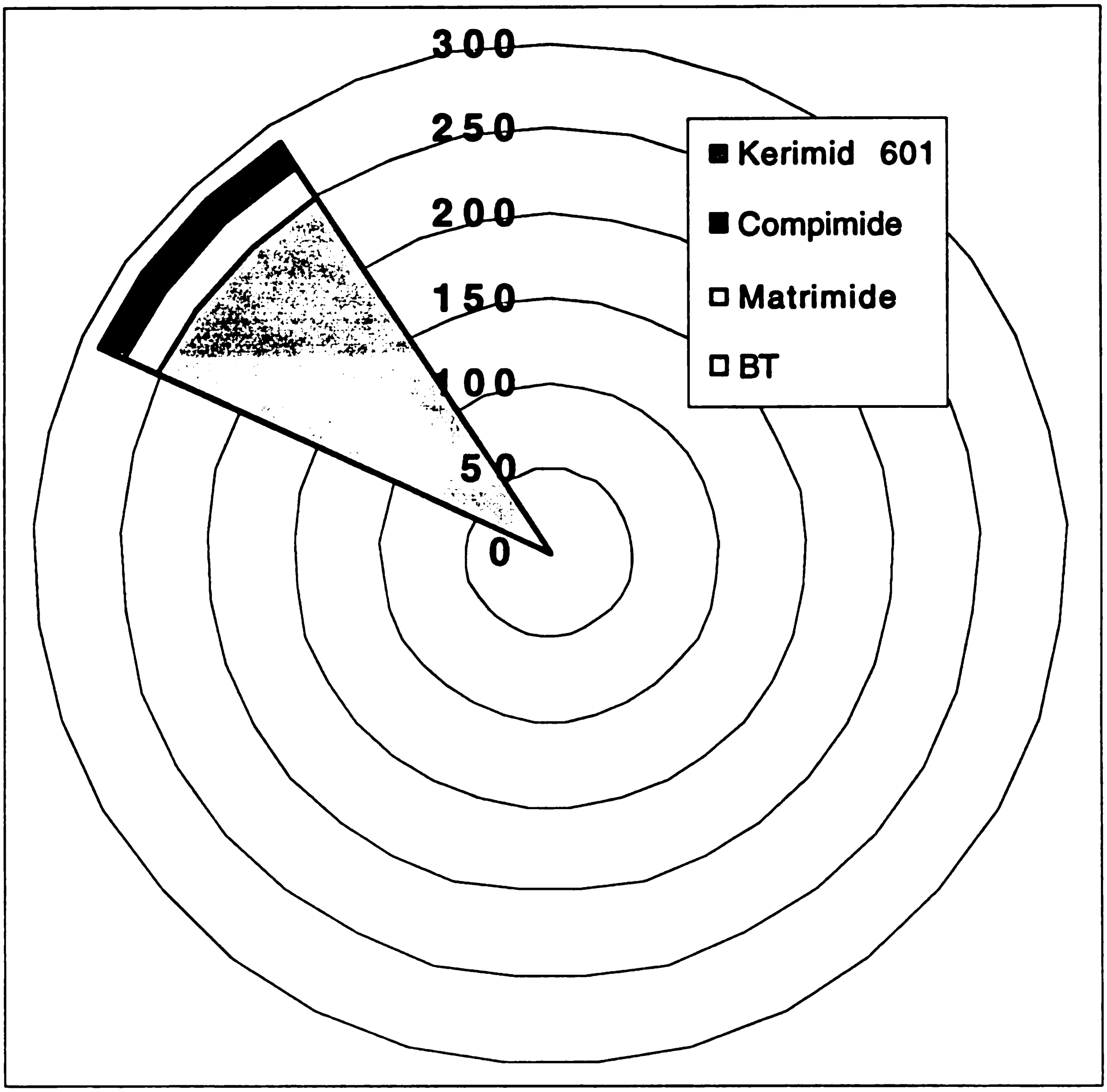

Figure 74. Bismaleimide use temperatures in COMADE.

A top-down progression through the vertical list of materials in the legend corresponds to an outside-in progression through the multiply highlighted lobe.

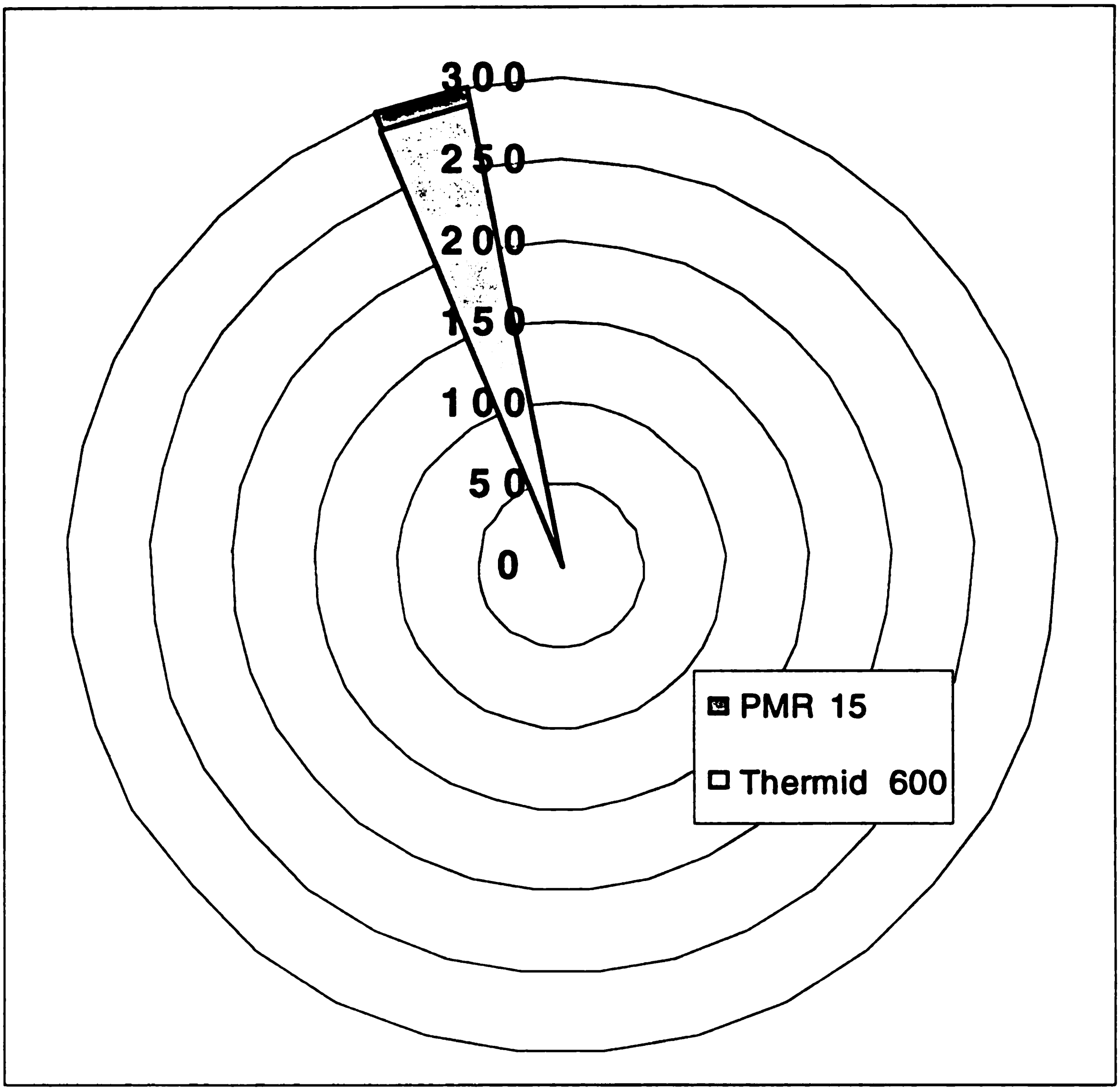

Figure 75. Polyimide use temperatures in COMADE.

A top-down progression through the vertical list of materials in the legend corresponds to an outside-in progression through the multiply highlighted lobe.

Table 25: Chemical environment contributions in COMADE.

		Chemical Environment[a]						
		none	weak acid	strong acid	weak base	strong base	hydrocarbon	organic solvent
Material Class	Amorphous Thermoplastics						not allowed	not allowed
	Semicrystalline Thermoplastics			not allowed[b]				
	Phenolics			not allowed	not allowed	not allowed		
	Polyimides			not allowed		not allowed		
Matrix Material	PEEK			not allowed				
	PEI				not allowed	not allowed		
	DGEBA							not allowed
Chemical Agent	Chlorendic Anhydride					not allowed		
	DEAPA	must be	must be	must be				
	MPDA							not allowed
Fiber Class	Aramid			not allowed		not allowed		
	Glass					not allowed		
Fiber Type	E-glass			not allowed		not allowed		

a. Blank cells indicate that there are no restrictions on that possibility.
b. 181 °C ≤ UseTemperature ≤ 250 °C

Table 26: Stress environment contributions in COMADE.

		Stress Environment[a]			
		none	constant	increasing	cyclic
Material Class	Bismaleimides			not allowed	
	Blend Thermoplastic		must be		
Chemical Agent	DEAPA	must be			
Matrix Material	PAI		not allowed		
	PAS		not allowed	not allowed[b]	not allowed[a]
	PEEK		not allowed	not allowed[a]	not allowed[a]
	PPS		not allowed	not allowed[a]	not allowed[a]

a. Blank cells indicate that there are no restrictions on that possibility.
b. UseTemperature ≤ 0 °C

Table 27: Flame retardance contributions in COMADE.

		Flame Retardance[a]	
		yes	no
Matrix Material	Phenolic Novolac	must be	
Chemical Agent	DEAPA		must be
	Dicyandiamide		must be
	MPDA		must be
	$DDS/BF_3 \cdot MEA$		must be

a. Blank cells indicate that there are no restrictions on that possibility.

Table 28: Humid environment contributions in COMADE.

		Humid Environment[a]	
		yes	no
Material Class	Amorphous Thermoplastic		must be[b]
	Epoxies		must be[c]
	Polyimides		must be[d]
Matrix Material	PEI		must be[e]
	DGEBA		must be
	Epoxidized Phenolic Novolac		must be
	TGETPE		must be
Chemical Agent	0,0'-Diallyl BPA		must be
	MABA		must be

a. Blank cells indicate that there are no restrictions on that possibility.
b. 210 °C ≤ UseTemperature ≤ 274 °C
c. 120 °C ≤ UseTemperature ≤ 200 °C
d. 260 °C ≤ UseTemperature ≤ 300 °C
e. 205 °C ≤ UseTemperature ≤ 225 °C

Table 29: Composite material mechanical properties versus fiber type.

Fiber Type	Tensile Strength (MPa)		Flexural Modulus (GPa)	
	Chopped	Continuous	Chopped	Continuous
No Fiber	50		5	
E-glass	169	2110	5	46
S-glass	224	2800	5	50
C-glass	163	2040	4	42
Quartz	170	2120	4	39
AS-4 carbon	196	2450	14	135
P-55 graphite	87	1090	22	220
P-100 graphite	111	1390	42	420
Kevlar-29	--	2210	--	52
Kevlar-49	--	2210	--	77
Kevlar-149	--	2090	--	113
Boron	177	2210	23	230
SiC	191	2390	24	240

Table 30: Miscellaneous material lookups for COMADE.

		Miscellaneaous COMADE Variables				Other Features	
		UseTemperature	Required Flexural Modulus	MatrixMaterial	FiberType	Superclass	Material Class
Matrix Material	TGMDA						Epoxy
	Phenolic Resole						Phenolic
	4,4'-MDA-BMI						BMI
Chemical Agent	None (TP matrix)						
	DAP	≥ 100 °C		Vinyl Ester		Thermoplastics	
	Vinyl Toluene	≥ 80 °C					Polyester
	Styrene		≥ 30 °C				Polyester
	HMTA			Phenolic novolac			Polyester
	None (thermal cure)			Phenolic resole			
	Methanol			PMR 15			
	Ethanol			Thermid 600			
Fiber Length	N/A (No fiber)				No Fiber		

Appendix C

COMADE in Detail

The following information complements the material given in Appendix B. Every bit of information encoded within COMADE is presented below. The material that follows was automatically generated from the COMADE source code using the GT Documenter.[1] Formatting modifications were made for the material in the following presentation.

```
************************************  DATA BASE ************************************

                                 ---------RDNumericalVar-------
*****ReqFlexuralModulus
INPUT VARIABLE
       upper limit: 420
       lower limit: 0
QUESTION:  What is the minimum required flexural modulus for the composite material?
DEFAULT UNIT:GPa
UNITS: MPsi

*****ReqTensileStrength
INPUT VARIABLE
       upper limit: 2800
       lower limit: 0
QUESTION:  What is the minimum required tensile strength for the composite material ?
DEFAULT UNIT:MPa
UNITS: Ksi

*****UseTemperature
INPUT VARIABLE
       upper limit: 300
       lower limit: -40
COMMENT:  This variable most strongly affects the choice of matrix material.
QUESTION:  What is the temperature at which the material will be used?
DEFAULT UNIT:C
UNITS: F

                                    ---------RDOneOfVar-------
*****ChemicalAgent
OUTPUT VARIABLE
```

1. **The GT Documenter was developed by Iliana Martinez.**

LEGAL VALUES:
    0,0'-Diallyl BPA
    1,4-Tolyl-BMI
    Chlorendic Anhydride
    DAP
    DDS/BF3-MEA
    DEAPA
    Dicyanate BPA
    Dicyandiamide
    EtOH
    HMTA
    MABA
    MeOH
    MPDA
    None (Thermal cure only)
    None (TP Matrix)
    Styrene
    unknown
    Vinyl Toluene
COMMENT: In the case of a thermoset material, this will be the curing/co-curing chemical (except for matrix choices like phenolic resoles, which are heat cured). Thermoplastics do not have curing agents per se, and therefore these materials will have no chemical agent: "None (TP Matrix)."
QUESTION: What is the chemical agent to be used with the matrix material?

*****ChemicalEnvironment
INPUT VARIABLE
LEGAL VALUES:
    hydrocarbon
    none
    organic solvent
    strong acid
    strong base
    unknown
    weak acid
    weak base
COMMENT: Examples for each of the variable choices are as follows:
    hydrocarbon: oils, hydraulic fluids, etc.
    organic solvent: cleaners, alcohols, ketones, etc.
    strong acid: mineral acids (>20% conc)
    strong base: conc. caustics (KOH, etc)
    weak acid: < 10% conc. acid
    weak base: < 10% conc. caustic
    Obviously, these are only qualitative groupings of chemicals, and there may be constituents within a particular group which are anomalously harmful. Check specific chemical interactions to be absolutely certain of the validity of a design.
QUESTION: To what type of chemicals (if any) will the material be exposed?

*****FiberLength
OUTPUT VARIABLE
LEGAL VALUES:
    Chopped
    Continuous
    N/A (No Fiber)
    unknown
COMMENT: Continuous fibers are assumed to have a fiber volume fraction of 60% and chopped fibers, 30%. This variable is affected by the required tensile strength and required flexural modulus.
QUESTION: What is the length of the fiber (if any) to be used in the material?

*****FiberType
OUTPUT VARIABLE
LEGAL VALUES:
    AS-4 Carbon

Boron
C-Glass
E-Glass
Kevlar-149
Kevlar-29
Kevlar-49
None
P-100 Graphite
P-55 Graphite
Quartz
S-Glass
SiC
unknown

QUESTION: What type of fiber is to be used?

*****MatrixMaterial
OUTPUT VARIABLE
LEGAL VALUES:

4,4'-MDA-BMI
ABS
ABS/PBT
BPA Fumarate Resin
Chlorendic Resin
DGEBA
Epoxidized Phenolic Novolac
Isophthalic Resin
Orthophthalic Resin
PA66
PAI
PAS
PBT
PC
PC/ABS
PC/PBT
PEEK
PEI
PET
Phenolic Novolac Resin
Phenolic Resole Resin
PMR 15 Monomers
PPO
PPS
PSU
TGETPE
TGMDA
Thermid 600 Oligomers
unknown
Vinyl Ester Resin

COMMENT: There are a lot of abbreviations and acronyms present within this variable, mostly for ease of representation within the routine designer. The material choices are all expanded and explained [in Appendix A].
QUESTION: What is the choice of material for the matrix?

*****StressEnvironment
INPUT VARIABLE
LEGAL VALUES:

constant
cyclic
increasing
none
unknown

COMMENT: The stress environment is a qualitative measure of what type of forces to which the material may be exposed. Generally, constant = structural, static loads; cyclic, increasing = componential, dynamic loads (therefore no brittle materials); none = not a

load bearing material.
QUESTION: To what type of stress (if any) will the material be exposed?

-------RDYesNoVar-------

*****FlameRetardance
INPUT VARIABLE
QUESTION: Does the material need to be flame retardant?

*****HumidEnvironment
INPUT VARIABLE
QUESTION: Is the material to be used in a humid environment?

```
******************************** AGENT HIERARCHY ********************************
---- <1>  CompositeMaterialSpecialist <RDSpecialist>  1
-------- <2>  CompositeMaterialPlan <RDPlan>  2
------------ <3>  MatrixSpecialist <RDSpecialist>  3
---------------- <4>  ThermoplasticPlan <RDPlan>  4
-------------------- <5>  ThermoplasticSpecialist <RDSpecialist>  5
------------------------ <6>  SemicrystallineTPPlan <RDPlan>  6
---------------------------- <7>  SemicrystallineTPTask <RDTask>  7
-------------------------------- <8>  SemicrystallineTPStep <RDStep>  8
-------------------------------- <8>  XTPChemicalAgentStep <RDStep>  9
------------------------ <6>  AmorphousTPPlan <RDPlan>  10
---------------------------- <7>  AmorphousTPTask <RDTask>  11
-------------------------------- <8>  AmorphousTPStep <RDStep>  12
-------------------------------- <8>  ATPChemicalAgentStep <RDStep>  13
------------------------ <6>  BlendTPPlan <RDPlan>  14
---------------------------- <7>  BlendTPTask <RDTask>  15
-------------------------------- <8>  BlendTPStep <RDStep>  16
-------------------------------- <8>  BTPChemicalAgentStep <RDStep>  17
---------------- <4>  ThermosetPlan <RDPlan>  18
-------------------- <5>  ThermosetSpecialist <RDSpecialist>  19
------------------------ <6>  TSEpoxyPlan <RDPlan>  20
---------------------------- <7>  TSEpoxyTask <RDTask>  21
-------------------------------- <8>  TSEpoxyStep <RDStep>  22
-------------------------------- <8>  TSEpoxyChemicalAgentStep <RDStep>  23
------------------------ <6>  TSPhenolicPlan <RDPlan>  24
---------------------------- <7>  TSPhenolicTask <RDTask>  25
-------------------------------- <8>  TSPhenolicStep <RDStep>  26
-------------------------------- <8>  TSPhenolicChemicalAgentStep <RDStep>  27
------------------------ <6>  TSPolyesterPlan <RDPlan>  28
---------------------------- <7>  TSPolyesterTask <RDTask>  29
-------------------------------- <8>  TSPolyesterStep <RDStep>  30
-------------------------------- <8>  TSPolyesterChemicalAgentStep <RDStep>  31
------------------------ <6>  TSBismaleimidePlan <RDPlan>  32
---------------------------- <7>  TSBismaleimideTask <RDTask>  33
-------------------------------- <8>  TSBismaleimideStep <RDStep>  34
-------------------------------- <8>  TSBMIChemicalAgentStep <RDStep>  35
------------------------ <6>  TSPolyimidePlan <RDPlan>  36
---------------------------- <7>  TSPolyimideTask <RDTask>  37
-------------------------------- <8>  TSPolyimideStep <RDStep>  38
-------------------------------- <8>  TSPolyimideChemicalAgentStep <RDStep>  39
------------ <3>  ReinforcementSpecialist <RDSpecialist>  40
---------------- <4>  ReinforcementPlan <RDPlan>  41
-------------------- <5>  FiberMaterialSpecialist <RDSpecialist>  42
------------------------ <6>  NoFiberPlan <RDPlan>  43
---------------------------- <7>  NoFiberTask <RDTask>  44
-------------------------------- <8>  NoFiberStep <RDStep>  45
------------------------ <6>  GlassFiberPlan <RDPlan>  46
---------------------------- <7>  GlassFiberTask <RDTask>  47
-------------------------------- <8>  GlassFiberStep <RDStep>  48
------------------------ <6>  CarbonFiberPlan <RDPlan>  49
---------------------------- <7>  CarbonFiberTask <RDTask>  50
-------------------------------- <8>  CarbonFiberStep <RDStep>  51
```

```
 ------------------------ <6>   AramidFiberPlan <RDPlan>  52
 ---------------------------- <7>   AramidFiberTask <RDTask>  53
 -------------------------------- <8>   AramidFiberStep <RDStep>  54
 ------------------------ <6>   SpecialtyFiberPlan <RDPlan>  55
 ---------------------------- <7>   SpecialtyFiberTask <RDTask>  56
 -------------------------------- <8>   SpecialtyFiberStep <RDStep>  57
 -------------------- <5>   FiberLengthSpecialist <RDSpecialist>  58
 ------------------------ <6>   FiberLengthPlan <RDPlan>  59
 ---------------------------- <7>   FiberLengthTask <RDTask>  60
 -------------------------------- <8>   FiberLengthStep <RDStep>  61

****************************** SPECIALIST HIERARCHI ******************************
 ---- <1>   CompositeMaterialSpecialist 1
 -------- <2>   MatrixSpecialist 3
 ------------ <3>   ThermoplasticSpecialist 5
 ------------ <3>   ThermosetSpecialist 19
 -------- <2>   ReinforcementSpecialist 40
 ------------ <3>   FiberMaterialSpecialist 42
 ------------ <3>   FiberLengthSpecialist 58

******************* KNOWLEDGE REPRESENTATION  **********************
------ 1------
CompositeMaterialSpecialist  <RDSpecialist>
DESIGN PLANS:  CompositeMaterialPlan

------ 2------
CompositeMaterialPlan  <RDPlan>
TYPE: design
PLAN ITEMS:  MatrixSpecialist  ReinforcementSpecialist

------ 3------
MatrixSpecialist  <RDSpecialist>
DESIGN PLANS:  ThermoplasticPlan  ThermosetPlan

------ 4------
ThermoplasticPlan  <RDPlan>
TYPE: design
PLAN ITEMS:  ThermoplasticSpecialist
SPONSOR:
  |UseTemperature      |StressEnvironment     |result
  |<= 274              |?                     |perfect
  |?                   |?                     |rule-out

------ 5------
ThermoplasticSpecialist  <RDSpecialist>
DESIGN PLANS:  SemicrystallineTPPlan  AmorphousTPPlan  BlendTPPlan

------ 6------
SemicrystallineTPPlan  <RDPlan>
TYPE: design
PLAN ITEMS:  SemicrystallineTPTask
SPONSOR:
  |UseTemperature      |ChemicalEnvironment          |result
  |<= 181              |?                            |perfect
  |<= 250              |~= strong acid               |perfect
  |?                   |?                            |rule-out

------ 7------
SemicrystallineTPTask  <RDTask>
STEPS:  SemicrystallineTPStep  XTPChemicalAgentStep

------ 8------
SemicrystallineTPStep  <RDStep>
ATTRIBUTE: MatrixMaterial
```

LIMITING AGENT:

MatrixMaterial	UseTemperature	StressEnvironment	result
= PEEK	< 0	= cyclic	exclude
= PEEK	< 0	= increasing	exclude
= PPS	< 0	= cyclic	exclude
= PPS	< 0	= increasing	exclude

EVALUATION AGENT:

UseTemperature	ChemicalEnvironment	StressEnvironment	result
<= 120	?	?	PBT
<= 140	?	?	PA66
<= 162	?	?	PET
<= 180	?	?	PPO
<= 181	?	~= constant	PPS
<= 250	~= strong acid	~= constant	PEEK

------ 9------

XTPChemicalAgentStep <RDStep>
ATTRIBUTE: ChemicalAgent
EVALUATION AGENT:

MatrixMaterial	result
?	None (TP Matrix)

------ 10------

AmorphousTPPlan <RDPlan>
TYPE: design
PLAN ITEMS: AmorphousTPTask
SPONSOR:

UseTemperature	ChemicalEnvironment	ChemicalEnvironment	HumidEnvironment	result
<= 210	~= organic solvent	~= hydrocarbon	?	perfect
<= 274	~= organic solvent	~= hydrocarbon	= No	perfect
?	?	?	?	rule-out

------ 11------

AmorphousTPTask <RDTask>
STEPS: AmorphousTPStep ATPChemicalAgentStep

------ 12------

AmorphousTPStep <RDStep>
ATTRIBUTE: MatrixMaterial
LIMITING AGENT:

MatrixMaterial	UseTemperature	StressEnvironment	result
= PAS	< 0	= cyclic	exclude
= PAS	< 0	= increasing	exclude

EVALUATION AGENT:

UseTemperatu	HumidEnviron	ChemicalEnvi	ChemicalEnvi	StressEnviro	MatrixMateri	result
<= 104	?	?	?	?	?	ABS
<= 147	?	?	?	?	?	PC
<= 204	?	?	?	?	?	PSU
<= 210	?	?	?	~= constant	?	PAS
<= 205	?	~= strong base	~= weak base	~= constant	?	PEI
<= 225	= No	~= strong base	~= weak base	~= constant	~= PEI	PEI
<= 274	?	?	?	~= constant	?	PAI

------ 13------

ATPChemicalAgentStep <RDStep>
ATTRIBUTE: ChemicalAgent
EVALUATION AGENT:

MatrixMaterial	result
?	None (TP Matrix)

------ 14------

BlendTPPlan <RDPlan>
TYPE: design
PLAN ITEMS: BlendTPTask

```
SPONSOR:
  |UseTemperature                    |StressEnvironment                |result
  |<= 220                            |= constant                       |perfect
  |?                                 |?                                |rule-out

------ 15------
BlendTPTask  <RDTask>
STEPS:  BlendTPStep  BTPChemicalAgentStep

------ 16------
BlendTPStep  <RDStep>
ATTRIBUTE: MatrixMaterial
EVALUATION  AGENT:
  |UseTemperature    |result
  |<= 110            |ABS/PBT
  |<= 140            |PC/PBT
  |<= 220            |PC/ABS

------ 17------
BTPChemicalAgentStep  <RDStep>
ATTRIBUTE: ChemicalAgent
EVALUATION  AGENT:
  |MatrixMaterial    |result
  |?                 |None (TP Matrix)

------ 18------
ThermosetPlan  <RDPlan>
TYPE: design
PLAN ITEMS:  ThermosetSpecialist

------ 19------
ThermosetSpecialist  <RDSpecialist>
DESIGN PLANS: TSEpoxyPlan  TSPhenolicPlan  TSPolyesterPlan  TSBismaleimidePlan  TSPoly-
imidePlan

------ 20------
TSEpoxyPlan  <RDPlan>
TYPE: design
PLAN ITEMS:  TSEpoxyTask
SPONSOR:
  |UseTemperature                    |HumidEnvironment                 |result
  |<= 200                            |= No                             |perfect
  |<= 120                            |?                                |perfect
  |?                                 |?                                |rule-out

------ 21------
TSEpoxyTask  <RDTask>
STEPS:  TSEpoxyStep  TSEpoxyChemicalAgentStep

------ 22------
TSEpoxyStep  <RDStep>
ATTRIBUTE: MatrixMaterial
EVALUATION  AGENT:
  |HumidEnvironment        |ChemicalEnvironment        |result
  |= No                    |~= organic solvent         |DGEBA
  |= No                    |?                          |Epoxidized Phenolic Novolac
  |= No                    |?                          |TGMDA
  |?                       |?                          |TGETPE

------ 23------
TSEpoxyChemicalAgentStep  <RDStep>
ATTRIBUTE: ChemicalAgent
LIMITING AGENT:
```

MatrixMaterial	MatrixMaterial	MatrixMaterial	MatrixMaterial	result
~= DGEBA	~= Epoxidized Phenolic Novol	~= TGETPE	~= TGMDA	exclude

EVALUATION AGENT:

UseTemperature	FlameRetardance	ChemicalEnvironment	StressEnvironment	result
< 95	= No	= weak acid	= none	DEAPA
< 95	= No	= weak base	= none	DEAPA
< 95	= No	= none	= none	DEAPA
<= 130	= No	?	?	Dicyandiamide
<= 155	= No	~= organic solvent	?	MPDA
<= 190	?	~= strong base	?	Chlorendic Anhydride
<= 200	= No	?	?	DDS/BF3-MEA

------ 24------

TSPhenolicPlan <RDPlan>
TYPE: design
PLAN ITEMS: TSPhenolicTask
SPONSOR:

ChemicalEnvironment	ChemicalEnvironment	ChemicalEnvironment	UseTemperature	result
~= strong base	~= weak base	~= strong acid	<= 150	perfect
?	?	?	?	rule-out

------ 25------

TSPhenolicTask <RDTask>
STEPS: TSPhenolicStep TSPhenolicChemicalAgentStep

------ 26------

TSPhenolicStep <RDStep>
ATTRIBUTE: MatrixMaterial
EVALUATION AGENT:

FlameRetardance	result
= Yes	Phenolic Novolac Resin
?	Phenolic Resole Resin

------ 27------

TSPhenolicChemicalAgentStep <RDStep>
ATTRIBUTE: ChemicalAgent
LIMITING AGENT:

MatrixMaterial	MatrixMaterial	result
~= Phenolic Novolac Resin	~= Phenolic Resole Resin	exclude

EVALUATION AGENT:

MatrixMaterial	result
= Phenolic Novolac Resin	HMTA
= Phenolic Resole Resin	None (Thermal cure only)

------ 28------

TSPolyesterPlan <RDPlan>
TYPE: design
PLAN ITEMS: TSPolyesterTask
SPONSOR:

UseTemperature	result
<= 140	perfect
?	rule-out

------ 29------

TSPolyesterTask <RDTask>
STEPS: TSPolyesterStep TSPolyesterChemicalAgentStep

------ 30------

TSPolyesterStep <RDStep>
ATTRIBUTE: MatrixMaterial
EVALUATION AGENT:

UseTemperature	FlameRetardance	ChemicalEnvironment	ChemicalEnvironment	result
<= 80	= No	~= strong base	~= strong acid	Orthophthalic
<= 90	= No	~= strong base	~= strong acid	Isophthalic

<= 100	= No	~= strong base	~= strong acid	Vinyl Ester
<= 130	= No	?	?	BPA Fumarate
<= 14	?	?	?	Chlorendic

------ 31------
TSPolyesterChemicalAgentStep <RDStep>
ATTRIBUTE: ChemicalAgent
LIMITING AGENT:

MatrixMateri	MatrixMateri	MatrixMateri	MatrixMateri	MatrixMateri	result
~=BPA Fumarate	~=Chlorendic	~=Isophthalic	~=Orthophthalic	~=Vinyl Ester	exclude

EVALUATION AGENT:

UseTemperature	MatrixMaterial	ReqFlexuralModulus	result
>= 100	= Vinyl Ester Resin	?	DAP
>= 80	?	>= 30	Vinyl Toluene
?	?	?	Styrene

------ 32------
TSBismaleimidePlan <RDPlan>
TYPE: design
PLAN ITEMS: TSBismaleimideTask
SPONSOR:

UseTemperature	StressEnvironment	result
<= 290	~= increasing	perfect
?	?	rule-out

------ 33------
TSBismaleimideTask <RDTask>
STEPS: TSBismaleimideStep TSBMIChemicalAgentStep

------ 34------
TSBismaleimideStep <RDStep>
ATTRIBUTE: MatrixMaterial
EVALUATION AGENT:

UseTemperature	result
<= 290	4,4'-MDA-BMI

------ 35------
TSBMIChemicalAgentStep <RDStep>
ATTRIBUTE: ChemicalAgent
LIMITING AGENT:

MatrixMaterial	result
~= 4,4'-MDA-BMI	exclude

EVALUATION AGENT:

UseTemperature	HumidEnvironment	result
<= 252	?	Dicyanate BPA
<= 273	= No	0,0'-Diallyl BPA
<= 290	= No	MABA
<= 290	?	1,4-Tolyl-BMI

------ 36------
TSPolyimidePlan <RDPlan>
TYPE: design
PLAN ITEMS: TSPolyimideTask
SPONSOR:

HumidEnvironment	UseTemperature	ChemicalEnvironment	ChemicalEnvironment	result
?	<= 260	~= strong base	~= strong acid	perfect
= No	<= 300	~= strong base	~= strong acid	perfect
?	?	?	?	rule-out

------ 37------
TSPolyimideTask <RDTask>
STEPS: TSPolyimideStep TSPolyimideChemicalAgentStep

------ 38------

TSPolyimideStep <RDStep>

ATTRIBUTE: MatrixMaterial

EVALUATION AGENT:

UseTemperature	result
<= 300	PMR 15 Monomers
<= 288	Thermid 600 Oligomers

------ 39------

TSPolyimideChemicalAgentStep <RDStep>

ATTRIBUTE: ChemicalAgent

LIMITING AGENT:

MatrixMaterial	MatrixMaterial	result
~= Thermid 600 Oligomers	~= PMR 15 Monomers	exclude

EVALUATION AGENT:

MatrixMaterial	result
= PMR 15 Monomers	MeOH
= Thermid 600 Oligomers	EtOH

------ 40------

ReinforcementSpecialist <RDSpecialist>

DESIGN PLANS: ReinforcementPlan

------ 41------

ReinforcementPlan <RDPlan>

TYPE: design

PLAN ITEMS: FiberMaterialSpecialist FiberLengthSpecialist

------ 42------

FiberMaterialSpecialist <RDSpecialist>

DESIGN PLANS: NoFiberPlan GlassFiberPlan CarbonFiberPlan AramidFiberPlan SpecialtyFiberPlan

------ 43------

NoFiberPlan <RDPlan>

TYPE: design

PLAN ITEMS: NoFiberTask

SPONSOR:

ReqFlexuralModulus	ReqTensileStrength	result
<= 5	<= 50	perfect
?	?	rule-out

------ 44------

NoFiberTask <RDTask>

STEPS: NoFiberStep

------ 45------

NoFiberStep <RDStep>

ATTRIBUTE: FiberType

EVALUATION AGENT:

ReqFlexuralModulus	result
?	None

------ 46------

GlassFiberPlan <RDPlan>

TYPE: design

PLAN ITEMS: GlassFiberTask

SPONSOR:

ChemicalEnvironment	ReqFlexuralModulus	ReqTensileStrength	result
~= strong base	<= 50	<= 2800	perfect
?	?	?	rule-out

------ 47------

GlassFiberTask <RDTask>

STEPS: GlassFiberStep

------ 48------
GlassFiberStep <RDStep>
ATTRIBUTE: FiberType
EVALUATION AGENT:

ReqFlexuralModulu	ReqTensileStrengt	ChemicalEnvironme	ChemicalEnvironme	result
<= 46	<= 2110	~= strong acid	~= weak acid	E-Glass
<= 50	<= 2800	?	?	S-Glass
<= 42	<= 2040	?	?	C-Glass
<= 39	<= 2120	?	?	Quartz

------ 49------
CarbonFiberPlan <RDPlan>
TYPE: design
PLAN ITEMS: CarbonFiberTask
SPONSOR:

ReqTensileStrength	ReqFlexuralModulus	result
<= 1090	<= 220	perfect
<= 1390	<= 420	perfect
<= 2450	<= 135	perfect
?	?	rule-out

------ 50------
CarbonFiberTask <RDTask>
STEPS: CarbonFiberStep

------ 51------
CarbonFiberStep <RDStep>
ATTRIBUTE: FiberType
EVALUATION AGENT:

ReqFlexuralModulus	ReqTensileStrength	result
<= 135	<= 2450	AS-4 Carbon
<= 220	<= 1090	P-55 Graphite
<= 420	<= 1390	P-100 Graphite

------ 52------
AramidFiberPlan <RDPlan>
TYPE: design
PLAN ITEMS: AramidFiberTask
SPONSOR:

ReqTensileStrengt	ReqFlexuralModulu	ChemicalEnvironme	ChemicalEnvironme	result
<= 2210	<= 113	~= strong acid	~= strong base	perfect
?	?	?	?	rule-out

------ 53------
AramidFiberTask <RDTask>
STEPS: AramidFiberStep

------ 54------
AramidFiberStep <RDStep>
ATTRIBUTE: FiberType
EVALUATION AGENT:

ReqTensileStrength	ReqFlexuralModulus	result
<= 2210	<= 52	Kevlar-29
<= 2210	<= 77	Kevlar-49
<= 2090	<= 113	Kevlar-149

------ 55------
SpecialtyFiberPlan <RDPlan>
TYPE: design
PLAN ITEMS: SpecialtyFiberTask
SPONSOR:

```
  |ReqFlexuralModulus       |ReqTensileStrength        |UseTemperature        |result
  |<= 240                   |<= 2390                   |>= 200                |perfect
  |?                        |?                         |?                     |rule-out

------ 56------
SpecialtyFiberTask  <RDTask>
STEPS: SpecialtyFiberStep

------ 57------
SpecialtyFiberStep  <RDStep>
ATTRIBUTE: FiberType
EVALUATION  AGENT:
  |ReqTensileStrength                |ReqFlexuralModulus                |result
  |<= 2210                           |<= 230                            |Boron
  |<= 2390                           |<= 240                            |SiC

------ 58------
FiberLengthSpecialist  <RDSpecialist>
DESIGN PLANS:  FiberLengthPlan

------ 59------
FiberLengthPlan  <RDPlan>
TYPE: design
PLAN ITEMS:  FiberLengthTask

------ 60------
FiberLengthTask  <RDTask>
STEPS: FiberLengthStep

------ 61------
FiberLengthStep  <RDStep>
ATTRIBUTE: FiberLength
LIMITING AGENT:
  |FiberType                         |FiberLength                       |result
  |= Kevlar-29                       |= Chopped                         |exclude
  |= Kevlar-49                       |= Chopped                         |exclude
  |= Kevlar-149                      |= Chopped                         |exclude
EVALUATION  AGENT:
  |ReqFlexuralModulus       |ReqTensileStrength        |FiberType          |result
  |?                        |?                         |= None             |N/A (No Fiber)
  |<= 5                     |<= 169                    |= E-Glass          |Chopped
  |<= 46                    |<= 2110                   |= E-Glass          |Continuous
  |<= 5                     |<= 224                    |= S-Glass          |Chopped
  |<= 50                    |<= 2800                   |= S-Glass          |Continuous
  |<= 4                     |<= 163                    |= C-Glass          |Chopped
  |<= 42                    |<= 2040                   |= C-Glass          |Continuous
  |<= 4                     |<= 170                    |= Quartz           |Chopped
  |<= 39                    |<= 2120                   |= Quartz           |Continuous
  |<= 14                    |<= 196                    |= AS-4 Carbon      |Chopped
  |<= 135                   |<= 2450                   |= AS-4 Carbon      |Continuous
  |<= 22                    |<= 87                     |= P-55 Graphit     |Chopped
  |<= 220                   |<= 1090                   |= P-55 Graphite    |Continuous
  |<= 42                    |<= 111                    |= P-100 Graphite   |Chopped
  |<= 420                   |<= 1390                   |= P-100 Graphite   |Continuous
  |<= 52                    |<= 2210                   |= Kevlar-29        |Continuous
  |<= 77                    |<= 2210                   |= Kevlar-49        |Continuous
  |<= 113                   |<= 2090                   |= Kevlar-149       |Continuous
  |<= 23                    |<= 177                    |= Boron            |Chopped
  |<= 230                   |<= 2210                   |= Boron            |Continuous
  |<= 24                    |<= 191                    |= SiC              |Chopped
  |<= 240                   |<= 2390                   |= SiC              |Continuous
```

Appendix D

Referenced COMADE Data

All of the material property data used in COMADE is presented along with specifically referenced sources of that data in this Appendix. The sources given are not an exhaustive listing of the literature consulted, but rather a selected list. Data is given for each class of material considered within COMADE: semicrystalline thermoplastics, amorphous thermoplastics, blended thermoplastics, epoxies, polyesters, phenolics, bismaleimides, and polyimides. Data for the various fiber materials considered is also presented. Data presented in the following tables complements the information given in Appendices B and C.

Table 31: Referenced data for semicrystalline thermoplastics in COMADE.

Matrix	Tmelt (°C)	Tglass (°C)	HDT* (°C)	UL (°C)	Use Temp¥ (°C)	Tensile Str† (MPa)	Flexural Mod† (GPa)	Selected References
PBT	234	29	68	120	120	54	2.3	[Carlsson, 1991], p.172
	240	17	50	120				[ASM International, 1987], p. 142
	228	40	85					BASF Ultradur® Design Guide
PA66	266	48	83	130	140	82.7	2.8	[Carlsson, 1991], pp.75, 172
	267	45	75	130				BASF Ultramid® Design Guide
	265	50	90					DuPont ZYTEL Design Guide
PET	268	73	41	140	162	55.2	2.41	[Strong, 1989], p.35
	270	69	41	140				[Carlsson, 1991], pp.75, 172
	265	70						[ASM International, 1987], p. 142
		80						GE Plastics VALOX Design Guide
PPO	285	85	100	N/A	180	53.8	2.48	[Carlsson, 1991], p.75
								GE Plastics NORYL Design Guide
PPS	282	89	135		181	78	3.5	[Carlsson, 1991]
	288	85	135	N/A		78		pp. 38-41, 75, 171-172, 174
	285	90				90.3		[Phillips, 1989], p. 116
	273	93				65		[ASM International, 1987]
		88						pp. 101, 142
PEEK	339	144	156	250	250	91	3.8	[Carlsson, 1991]
	340	143	156	250				pp. 38-41, 75, 171-172, 174
	335	144				66 @ 100°C		[Phillips, 1989], p. 116
	343					35 @ 150°C		[ASM International, 1987]
								pp. 101, 138, 142

* At 1.82 MPa.

¥ As encoded within COMADE.

† At room temperature unless indicated otherwise.

Table 32: Referenced data for amorphous thermoplastics in COMADE.

Matrix	Tglass (°C)	HDT* (°C)	UL (°C)	Use Temp¥ (°C)	Tensile Str† (MPa)	Flexural Mod† (GPa)	Selected References
ABS	104	106		104	42.1	2.21	GE Plastics CYCOLAC Design Guide
	104	106	N/A				Monsanto LUSTRAN Design Guide
PC	147	132		147	62.7	2.28	[Carlsson, 1991], p. 172
	147	132	N/A				Dow Chemical Calibre Design Guide
PSU	190	175	204	204	76	3.25	[Strong, 1989], p. 35
	190 (dry)	175	204				BASF Ultrason® Design Guide
PAS	213			215 (dry)	100	3.24	[Carlsson, 1991]
	215 (dry)	N/A	N/A	210 (wet)			pp. 38-41, 171, 174
	210 (wet)						[ASM International, 1987]
							p. 101, 138
PEI	213	203	170	225 (dry)	104	3.4	[Carlsson, 1991]
	225 (dry)	200	170	205 (wet)	41.4 @170°C		pp. 38-41, 171-172, 174
	205 (wet)	206					[ASM International, 1987]
	210						pp. 101, 142
PAI	274	274		274	146.7	3.7	[Carlsson, 1991]
	275	270	N/A		186	4.8	pp. 38-41, 171-172, 174
	243	278			166	3.6	[ASM International, 1987]
	290				145	2.8	pp. 101, 138
	288				89.6		Amoco Torlon Design Guide
					124 @150°C		

* At 1.82 MPa.

¥ As encoded within COMADE.

† At room temperature unless indicated otherwise.

Table 33: Referenced data for blended thermoplastics in COMADE.

Matrix	Class	HDT* (°C)	Use Temp¥ (°C)	Tensile Str† (MPa)	Flexural Mod† (GPa)	Selected References
ABS/PBT	Eng.	74	110	41.4	1.86	GE Plastics Cycoloy Design Guide Dow Chemical Pulse Design Guide
PC/PBT	Eng.	91	140	54.5	1.97	GE Plastics Xenoy Design Guide GE Plastics Valox Design Guide
PC/ABS	Eng.	220	220	49.6	2.1	Bayblend Miles GP T65 MN Design Guide

* At 1.82 MPa.
¥ As encoded within COMADE.
† At room temperature unless indicated otherwise.

Table 34: Referenced data for epoxies in COMADE.

Chemical Agent	HDT* (°C)	Use Temp¥ (°C)	Tensile Str† (MPa)	Flexural Mod† (GPa)	Selected References
MPDA	150	155	55	2.76	[ASM International, 1987] pp. 72, 74-76, 134, 137-138 399, 401-403 [Phillips, 1989] pp. 75, 106-107, 159 [Strong, 1989] pp. 17-21 [Bruins, 1968] pp. 35, 38, 43, 53-54, 65, 73, 77
	155				
DDS	190	200	56		
	175				
Dicyandiamide	135	130			
DEAPA	100	95	57	3.2	
	78		49		
	94		65		
Chlorendic Anyhdride	197	190	83	3.59	
	200				
	145				
	190				

* At 1.82 MPa.

¥ As encoded within COMADE.

† At room temperature unless indicated otherwise.

Table 35: Referenced data for polyesters in COMADE.

Matrix	HDT* (°C)	Use Temp¥ (°C)	Tensile Str† (MPa)	Flexural Mod† (GPa)	Selected References
Orthophthalic	80	80	55	3.45	[ASM International, 1987]
Isophthalic	90	90	75	3.59	pp. 90-95, 133, 138
BPA Fumarate	130	130	40	3.38	[Strong, 1989], p. 16
Chlorendic	140	140	20	3.93	[Phillips, 1989], p. 75
Vinyl Ester	100	100	80	3.72	

* At 1.82 MPa.

¥ As encoded within COMADE.

† At room temperature unless indicated otherwise.

Table 36: Referenced data for phenolics in COMADE.

Matrix	HDT* (°C)	UL (°C)	Use Temp¥ (°C)	Tensile Str† (MPa)	Flexural Mod† (GPa)	Selected References
Novolac	120 315	150 230	150	55	7.58	[Phillips, 1989] pp. 109-110
Resole		177	150	62	6.1	[ASM International, 1987] pp. 381-391

* At 1.82 MPa.
¥ As encoded within COMADE.
† At room temperature unless indicated otherwise.

Table 37: Referenced data for bismaleimides in COMADE.

Chemical Agent	Tglass (°C)	Use Temp¥ (°C)	Tensile Str† (MPa)	Flexural Mod† (GPa)	Selected References
m-aminobenzoic acid	290	290	89	5.5	[Phillips, 1989], pp. 107-109
	250			4.6	[ASM International, 1987], pp. 33, 78, 88
0,0'-diallyl bisphenol A	273	273	82 (dry)	4	[Phillips, 1989], pp. 107-109
	295 (dry)		66 (wet)		[ASM International, 1987]
	305 (wet)		40 @ 204°C		pp. 33, 78, 86
BMI-1,4-tolyl	290	290	63.4	5.6	[Phillips, 1989], pp. 107-109
	285		43 @ 200°C	3.4 @ 250°C	[ASM International, 1987], pp. 33, 78, 86-87
dicyanate bisphenol A	252	252	60	3.4	[Phillips, 1989], pp. 107-109
	257		85	4.8	[ASM International, 1987], pp. 33, 78, 87

¥ As encoded within COMADE.

† At room temperature unless indicated otherwise.

Table 38: Referenced data for polyimides in COMADE.

Matrix	Tglass (°C)	Use Temp¥ (°C)	Tensile Str† (MPa)	Flexural Mod† (GPa)	Selected References
PMR 15	340	316	38.6	4	[ASM International, 1987], pp. 78, 89
				1.9 @ 316°C	[Phillips, 1989], pp. 110-113
Thermid 600	335	288	83	4.5	[ASM International, 1987] pp. 78, 83, 89

¥ As encoded within COMADE.

† At room temperature unless indicated otherwise.

Table 39: Referenced data for fibers in COMADE.

Fiber	Length	Tensile Str† (MPa)	Tensile Mod† (GPa)	Selected References
Aramid				[Carlsson, 1991]
Kevlar 29	long fibers	3600	83	pp. 180-181
Kevlar 49	long fibers	3600	124	[ASM International, 1987]
Kevlar 149	long fibers	3400	186	pp. 114-115, 362
Specialty				
SiC	whisker	103000	483	[Phillips, 1989]
		207000		pp. 5, 23, 77
SiC	chopped fiber	24100	165	[Carlsson, 1991]
	long fiber	3900	400	p. 105
Boron	long fiber	3180	389.5	[ASM International, 1987]
		3600	400	pp. 58-63
	long fiber	2760	379	
Glass				
Quartz	long fiber	3450	69	[Carlsson, 1991]
		3450	69	pp. 105, 177, 180-181
C-glass	long fiber	3310	79.9	[ASM International, 1987]
S-glass	short fiber	48000	83	pp. 46-47, 62, 360
	long fiber	4580	86.8	[Strong, 1989], p. 51
E-glass	short fiber	34000	76	[Phillips, 1989]
	long fiber	3440	81.3	pp. 5, 23, 77
Carbon				[ASM International, 1987]
AS-4	PAN-based	4000	231	pp. 49-52, 113, 362
P-55	pitch-based	1730	379	[Carlsson, 1991]
P-100	pitch-based	2240	724	pp. 179-80

† At room temperature.

Appendix E

COMADE & CBR Details

There were many detailed studies done during the investigation of designing composite material systems with CBR. The material presented in Chapter Ten, while representative of the conclusions of these studies, does not convey the massive amount of information behind those conclusions. This Appendix is an attempt to give a better representation of the major aspects of these studies.

The details of the cases referred to in Chapter Ten are presented in the following pages. In Chapter Ten, the case memory hierarchy figure that was presented did not include specific indications of the cases associated with the tip nodes. Such connections are also presented herein. Several alternative case memory structures considered before the final case hierarchy was determined are presented. All of this information concerns the 27 cases that were specifically generated to convey a comprehensive coverage of the possible COMADE material domain.

Alternatively, studies were also done with randomly generated case libraries. A library of random cases is presented, along with the resulting case memory hierarchy that was generated. This used the same indexing nomenclature and navigation techniques as did the structured cases above. A comparison of the two case retrieval processes found the structured case library to be more efficient than the random library.

Table 40: Input variables for the 27 structured cases.

FR	HE		CE	UT	RFM	RTS		CE	UT	RFM	RTS		CE	UT	RFM	RTS		CE	UT	RFM	RTS
SE			none					cons					incr					cycl			
yes	yes	case1	none	-20	10	2600	case6	hc	280	170	2100	case3	wb	160	250	1100	case8	wa	100	250	1100
							case27	sb	40	10	100										
yes	no	case9	wa	40	90	2100	case14	os	-20	90	1600	case11	sb	220	170	1600	case16	sa	160	10	2600
		case17	wb	100	330	600	case22	none	40	250	600	case19	hc	100	330	100	case24	wb	280	170	100
		case26	sa	160	90	1600															
no	yes	case13	sb	160	250	100	case10	wa	100	330	1100	case15	none	-20	410	600	case12	sb	40	90	2100
		case21	hc	220	170	1600	case18	sa	160	410	100	case23	wa	40	10	2600	case20	hc	280	90	1600
																	case25	os	220	410	1100
no	no	case5	os	280	410	1100	case2	wb	220	10	2600	case7	sa	100	90	2100	case4	none	-20	330	600

Table 41: Composite material systems for the 27 structured cases.

	Matrix Material	Chemical Agent	Fiber Type	Fiber Length
1	ABS	None (TP Matrix)	S-Glass	Continuous
2	PC/ABS	None (TP Matrix)	S-Glass	Continuous
3	PEEK	None (TP Matrix)	P-100 Graphite	Continuous
4	TGETPE	Dicyandiamide	P-100 Graphite	Continuous
5	4,4'-MDA-BMI	MABA	P-100 Graphite	Continuous
6	4,4'-MDA-BMI	1,4-Tolyl-BMI	Boron	Continuous
7	BPA Fumarate Resin	Vinyl Toluene	AS-4 Carbon	Continuous
8	PMR 15 Monomers	MeOH	P-100 Graphite	Continuous
9	Phenolic Novolac Resin	HMTA	AS-4 Carbon	Continuous
10	ABS/PBT	None (TP Matrix)	P-100 Graphite	Continuous
11	PAI	None (TP Matrix)	SiC	Continuous
12	PA66	None (TP Matrix)	AS-4 Carbon	Continuous
13	PPO	None (TP Matrix)	P-100 Graphite	Continuous
14	TGMDA	Chlorendic Anhydride	Kevlar-149	Continuous
15	TGETPE	MPDA	P-100 Graphite	Continuous
16	DGEBA	Chlorendic Anhydride	S-Glass	Continuous
17	Chlorendic Resin	Styrene	P-100 Graphite	Continuous
18	PET	None (TP Matrix)	P-100 Graphite	Continuous
19	Epoxidized Phenolic Novolac	Chlorendic Anhydride	P-100 Graphite	Continuous
20	4,4'-MDA-BMI	1,4-Tolyl-BMI	Kevlar-149	Continuous
21	Thermid 600 Oligomers	EtOH	SiC	Continuous
22	Phenolic Resole Resin	None (Thermal cure only)	P-100 Graphite	Continuous
23	Isophthalic Resin	Styrene	S-Glass	Continuous
24	4,4'-MDA-BMI	MABA	P-55 Graphite	Continuous
25	4,4'-MDA-BMI	Dicyanate BPA	P-100 Graphite	Continuous
26	TGETPE	Chlorendic Anhydride	AS-4 Carbon	Continuous
27	PAS	None (TP Matrix)	AS-4 Carbon	Chopped
28	PMR 15 Monomers	MeOH	P-100 Graphite	Continuous

TP/TS	MClass	Matrix	CA	FClass	Fiber	Length	Case
		PA66	None (TP Matrix)	Carbon	AS-4 Carbon	Continuous	12
		PET	None (TP Matrix)	Carbon	P-100 Graphite	Continuous	18
	SXTP						
		PPO	None (TP Matrix)	Carbon	P-100 Graphite	Continuous	13
		PEEK	None (TP Matrix)	Carbon	P-100 Graphite	Continuous	3
Thermoplastic		ABS	None (TP Matrix)	Glass	S-Glass	Continuous	1
	ATP	PAI	None (TP Matrix)	Specialty	SiC	Continuous	11
		PAS	None (TP Matrix)	Carbon	AS-4 Carbon	Chopped	27
		PC/ABS	None (TP Matrix)	Glass	S-Glass	Continuous	2
	BTP						
		ABS/PBT	None (TP Matrix)	Carbon	P-100 Graphite	Continuous	10

Figure 76. Top half of the case memory hierarchy for the 27 structured cases.

TP/TS	MClass	Matrix	CA	FClass	Fiber	Length	Case
			Dicyandiamide	Carbon	P-100 Graphite	Continuous	4
		TGETPE	Chlorendic Anhydride	Carbon	AS-4 Carbon	Continuous	26
			MPDA	Carbon	P-100 Graphite	Continuous	15
	Epoxy	TGMDA	Chlorendic Anhydride	Aramid	Kevlar-149	Continuous	14
		DGEBA	Chlorendic Anhydride	Glass	S-Glass	Continuous	16
		Epoxidized Phenolic Novolac	Chlorendic Anhydride	Carbon	P-100 Graphite	Continuous	19
		BPA Fumarate Resin	Vinyl Toluene	Carbon	AS-4 Carbon	Continuous	7
	Polyester	Chlorendic Resin	Styrene	Carbon	P-100 Graphite	Continuous	17
Thermoset		Isophthalic Resin	Styrene	Glass	S-Glass	Continuous	23
		Phenolic Novolac Resin	HMTA	Carbon	AS-4 Carbon	Continuous	9
	Phenolic	Phenolic Resole Resin	None (Thermal cure only)	Carbon	P-100 Graphite	Continuous	22
					P-55 Graphite	Continuous	24
			MABA	Carbon			
					P-100 Graphite	Continuous	5
	BMI	4,4'-MDA-BMI	Dicyanate BPA	Carbon	P-100 Graphite	Continuous	25
				Specialty	Boron	Continuous	6
			1,4-Tolyl-BMI				
				Aramid	Kevlar-149	Continuous	20
		PMR 15 Monomers	MeOH	Carbon	P-100 Graphite	Continuous	8
	Polyimide						
		Thermid 600 Oligomers	EtOH	Specialty	SiC	Continuous	21

Figure 77. Bottom half of the case memory hierarchy for the 27 structured cases.

UT	CE	RFM	RTS	FR	HE	SE	
<=10	none	<=50	<=3200	yes	yes	none	case1
<=10	none	<=370	<=850	no	no	cycl	case4
<=10	none	<=500	<=850	no	yes	incr	case15
<=10	os	<=130	<=1850	yes	no	cons	case14
<=70	none	<=290	<=850	yes	no	cons	case22
<=70	sb	<=50	<=350	yes	yes	cons	case27
<=70	sb	<=130	<=2350	no	yes	cycl	case12
<=70	wa	<=50	<=3200	no	yes	incr	case23
<=70	wa	<=130	<=2350	yes	no	none	case9
<=130	hc	<=370	<=350	yes	no	incr	case19
<=130	sa	<=130	<=2350	no	no	incr	case7
<=130	wa	<=290	<=1350	yes	yes	cycl	case8
<=130	wa	<=370	<=1350	no	yes	cons	case10
<=130	wb	<=370	<=850	yes	no	none	case17
<=170	sa	<=50	<=3200	yes	no	cycl	case16
<=170	sa	<=130	<=1850	yyes	no	none	case26
<=170	sa	<=500	<=350	no	yes	cons	case18
<=170	sb	<=290	<=350	no	yes	none	case13
<=170	wb	<=290	<=1350	yes	yes	incr	case3
<=250	hc	<=210	<=1850	no	yes	none	case21
<=250	os	<=500	<=1350	no	yes	cycl	case25
<=250	sb	<=210	<=1850	yes	no	incr	case11
<=250	wb	<=50	<=3200	no	no	cons	case2
<=300	hc	<=130	<=1850	no	yes	cycl	case20
<=300	hc	<=210	<=2350	yes	yes	cons	case6
<=300	os	<=500	<=1350	no	no	none	case5
<=300	wb	<=210	<=350	yes	no	cycl	case24

Figure 78. A syntactic redesign-influenced case hierarchy for the 27 cases.

FR	HE	SE	RTS	RFM	CE	UT	
yes	yes	cons	<=350	<=50	sb	<=70	case27
yes	yes	cons	<=2350	<=210	hc	<=300	case6
yes	yes	cycl	<=1350	<=290	wa	<=130	case8
yes	yes	incr	<=1350	<=290	wb	<=170	case3
yes	yes	none	<=3200	<=50	none	<=10	case1
yes	no	cons	<=1850	<=130	os	<=10	case14
yes	no	cons	<=850	<=290	none	<=70	case22
yes	no	cycl	<=3200	<=50	sa	<=170	case16
yes	no	cycl	<=350	<=210	wb	<=300	case24
yes	no	incr	<=350	<=370	hc	<=130	case19
yes	no	incr	<=1850	<=210	sb	<=250	case11
yes	no	none	<=2350	<=130	wa	<=70	case9
yes	no	none	<=850	<=370	wb	<=130	case17
yes	no	none	<=1850	<=130	sa	<=170	case26
no	yes	cons	<=1350	<=370	wa	<=130	case10
no	yes	cons	<=350	<=500	sa	<=170	case18
no	yes	cycl	<=2350	<=130	sb	<=70	case12
no	yes	cycl	<=1350	<=500	os	<=250	case25
no	yes	cycl	<=1850	<=130	hc	<=300	case20
no	yes	incr	<=850	<=500	none	<=10	case15
no	yes	incr	<=3200	<=50	wa	<=70	case23
no	yes	none	<=350	<=290	sb	<=170	case13
no	yes	none	<=1850	<=210	hc	<=250	case21
no	no	cons	<=3200	<=50	wb	<=250	case2
no	no	cycl	<=850	<=370	none	<=10	case4
no	no	incr	<=2350	<=130	sa	<=130	case7
no	no	incr	<=850	<=290	os	<=250	case28
no	no	none	<=1350	<=500	os	<=300	case5

Figure 79. An inverse syntactic redesign-influenced case hierarchy for the 27 cases.

Table 42: Input variables for the randomly generated cases.

r10case1	**5 designs**
ReqTensileStrength	1251
FlameRetardance	yes
StressEnvironment	constant
ChemicalEnvironment	strong base
HumidEnvironment	no
UseTemperature	184
ReqFlexuralModulus	215
r10case2	**1 design**
ReqTensileStrength	2753
FlameRetardance	no
StressEnvironment	none
ChemicalEnvironment	organic solvent
HumidEnvironment	no
UseTemperature	268
ReqFlexuralModulus	40
r10case3	**17 designs**
ReqTensileStrength	1027
FlameRetardance	yes
StressEnvironment	constant
ChemicalEnvironment	weak acid
HumidEnvironment	yes
UseTemperature	48
ReqFlexuralModulus	244
r10case5	**17 designs**
ReqTensileStrength	413
FlameRetardance	no
StressEnvironment	none
ChemicalEnvironment	weak base
HumidEnvironment	no
UseTemperature	52
ReqFlexuralModulus	396

Table 43: Composite material systems for the randomly generated cases.

r10-1a	**r10-1b**	**r10-1c**	**r10-1d**	**r10-1e**
4,4'-MDA-BMI	4,4'-MDA-BMI	PEEK	PC/ABS	PSU
Dicyanate BPA	1,4-Tolyl BMI	None (TP Matrix)	None (TP Matrix)	None (TP Matrix)
P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite
Continuous	Continuous	Continuous	Continuous	Continuous
r10-2a	**r10-3a**	**r10-3b**	**r10-3c**	**r10-3d**
4,4'-MDA-BMI	4,4'-MDA-BMI	4,4'-MDA-BMI	ABS	ABS/PBT
1,4-Tolyl BMI	1,4-Tolyl-BMI	Dicyanate BPA	None (TP Matrix)	None (TP Matrix)
S-Glass	P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite
Continuous	Continuous	Continuous	Continuous	Continuous
r10-3e	**r10-3f**	**r10-3g**	**r10-3h**	**r10-3i**
Chlorendic Resin	PA66	PBT	PC	PC/ABS
Styrene	None (TP Matrix)	None (TP Matrix)	None (TP Matrix)	None (TP Matrix)
P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite
Continuous	Continuous	Continuous	Continuous	Continuous
r10-3j	**r10-3k**	**r10-3l**	**r10-3m**	**r10-3n**
PC/PBT	PET	Phenolic Resole Resin	PMR 15 Monomers	PPO
None (TP Matrix)	None (TP Matrix)	None (Thermal cure only)	MeOH	None (TP Matrix)
P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite
Continuous	Continuous	Continuous	Continuous	Continuous
r10-3o	**r10-3p**	**r10-3q**	**r10-5a**	**r10-5b**
PSU	TGETPE	Thermid 600 Oligomers	4,4'-MDA-BMI	4,4'-MDA-BMI
None (TP Matrix)	Chlorendic Anhydride	EtOH	1,4-Tolyl-BMI	Dicyanate BPA
P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite
Continuous	Continuous	Continuous	Continuous	Continuous
r10-5c	**r10-5d**	**r10-5e**	**r10-5f**	**r10-5g**
ABS	Chlorendic Resin	PA66	PAI	PAS
None (TP Matrix)	Styrene	None (TP Matrix)	None (TP Matrix)	None (TP Matrix)
P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite
Continuous	Continuous	Continuous	Continuous	Continuous
r10-5h	**r10-5i**	**r10-5j**	**r10-5k**	**r10-5l**
PBT	PC	PEEK	PET	PMR 15 Monomers
None (TP Matrix)	None (TP Matrix)	None (TP Matrix)	None (TP Matrix)	MeOH
P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite
Continuous	Continuous	Continuous	Continuous	Continuous
r10-5m	**r10-5n**	**r10-5o**	**r10-5p**	**r10-5q**
PPO	PPS	PSU	TGETPE	Thermid 600 Oligomers
None (TP Matrix)	None (TP Matrix)	None (TP Matrix)	Chlorendic Anhydride	EtOH
P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite	P-100 Graphite
Continuous	Continuous	Continuous	Continuous	Continuous

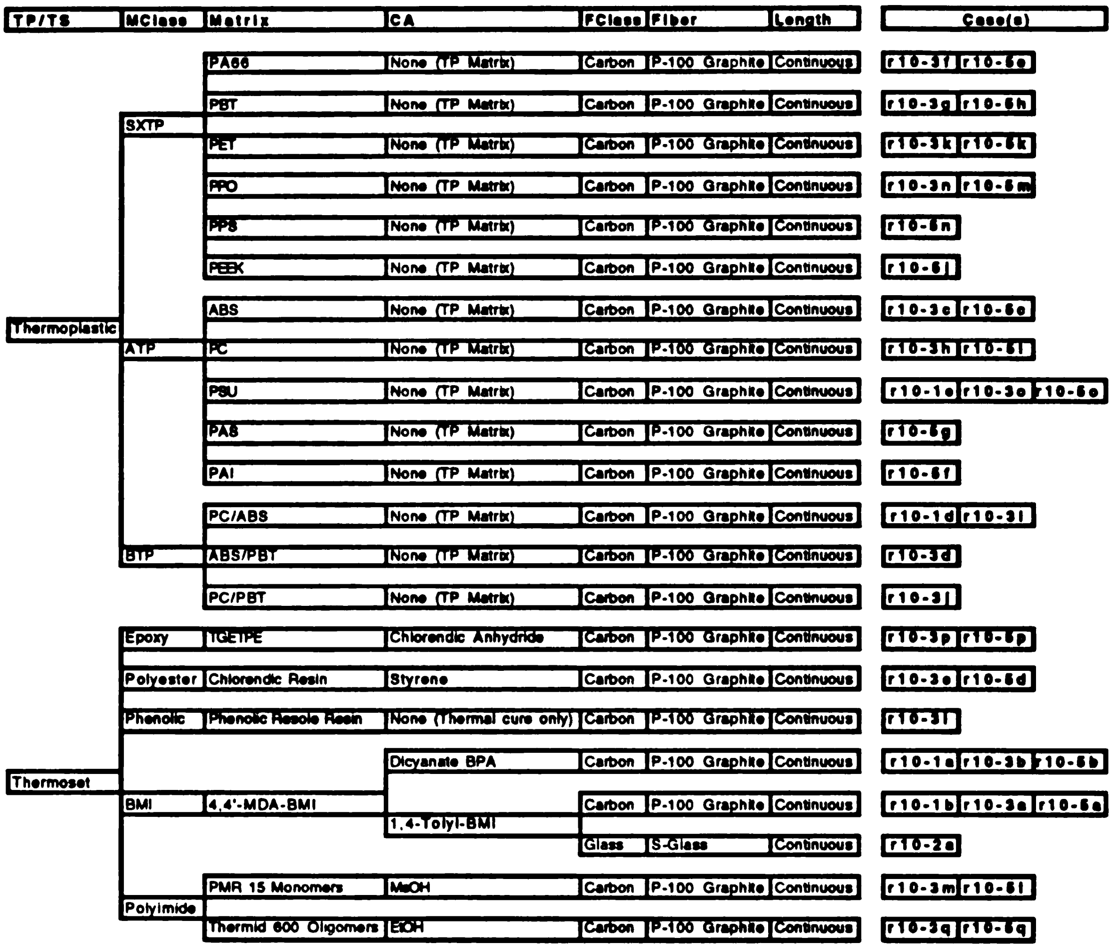

Figure 80. Case memory hierarchy for the randomly generated cases.

BIBLIOGRAPHY

BIBLIOGRAPHY

[Aamodt & Plaza, 1994]
Aamodt, A., & E. Plaza. (1994). Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. Artificial Intelligence Communications, 7(1).

[Allemang, 1994]
Allemang, D. (1994). Combining Case-Based Reasoning and Task-Specific Architectures. IEEE Expert, 9(5), 24-34.

[Allemang, 1995]
Allemang, D. (1995, February 2nd). Personal communication (email).

[ASM International, 1987]
ASM International (Ed.). (1987). Engineered Materials Handbook: Composites. (Vol. 1). Metals Park, OH: ASM International.

[Bardasz & Zeid, 1993]
Bardasz, T., & I. Zeid. (1993). DEJAVU: Case-based Reasoning for Mechanical Design. AI EDAM, 7(2), 111-124.

[Bareiss, 1989]
Bareiss, R. (1989). Exemplar-Based Knowledge Acquisition: A Unified Approach to Concept Representation, Classification, and Learning: Academic Press.

[Berger, 1989]
Berger, J. (1989). ROENTGEN: A Case-Based Approach to Radiation Therapy Planning. Proceedings of the Case-Based Reasoning Workshop (DARPA), Pensacola Beach, Florida. 218-223.

[Berger, 1994]
Berger, J. (1994). ROENTGEN: Radiation Therapy and Case-Based Reasoning. Proceedings of the 10th Conference on Artificial Intelligence for Applications, San Antonio, TX. 171-177.

[Brown, 1987]
Brown, D. C. (1987). Routine Design Problem Solving. In J. Gero (Ed.), Knowledge Based Systems in Engineering and Architecture, Addison-Wesley.

[Brown & Chandrasekaran, 1989]
Brown, D., & B. Chandrasekaran. (1989). Design Problem Solving: Knowledge Structures and Control Strategies. London: Pitman.

[Bruins, 1968]
Bruins, P. F. (Ed.). (1968). Polymer Engineering and Technology: Epoxy Resin Technology. New York: Interscience.

[Buchanan & Lederberg, 1971]
Buchanan, B. G., & J. Lederberg. (1971). The heuristic DENDRAL program for explaining empirical data. Proceedings of the International Federation for Information Processing. 179-188.

[Bylander & Mittal, 1986]
Bylander, T., & S. Mittal. (1986). CSRL: A Language for Classificatory Problem Solving and Uncertainty Handling. AI Magazine, 7(2), 66-77.

[Bylander & Chandrasekaran, 1987]
Bylander, T., & B. Chandrasekaran. (1987). Generic Tasks for Knowledge-Based Reasoning: the "Right" Level of Abstraction for Knowledge Acquisition. International Journal of Man-Machine Studies, 26, 231-243.

[Bylander *et al.*, 1991]
Bylander, T., T. R. Johnson, & A. Goel. (1991). Structured Matching: A Task-Specific Technique for Making Decisions. Knowledge Acquisition, 3, 1-20.

[Carbonell, 1983]
Carbonell, J. G. (1983). Learning by Analogy: Formulating and Generalizing Plans from Past Experience. Machine Learning, 1.

[Carbonell, 1986]
Carbonell, J. G. (1986). Derivational Analogy: A Theory of Reconstructive Problem Solving and Expertise Acquisition. Machine Learning, 2.

[Carlsson, 1991]
Carlsson, L. A. (Ed.). (1991). Thermoplastic Composite Materials. (Vol. 7). Amsterdam: Elsevier.

[Chandrasekaran, 1990]
Chandrasekaran, B. (1990). Design Problem Solving: A Task Analysis. AI Magazine, 11(4), 59-71.

[Davis *et al.*, 1977]
Davis, R., B. G. Buchanan, & E. H. Shortliffe. (1977). Production Rules as a Representation for a Knowledge-Based Consultation System. Artificial Intelligence, 8, 15-45.

[Douglas, 1988]
Douglas, J. M. (1988). Conceptual Design of Chemical Processes. McGraw-Hill, Inc.

[Duda *et al.*, 1979]
Duda, R. O., P. E. Hart, K. Konolige, & R. Reboh. (1979). A Computer-Based Consultant for Mineral Exploration (Technical report). SRI International.

[Dvorak, 1988]
Dvorak, D. (1988). CL-Protos Users Guide. Artificial Intelligence Laboratory, Department of Computer Science, University of Texas at Austin.

[Dym & Levitt, 1991]
Dym, C. L., & R. E. Levitt. (1991). Knowledge-Based Systems in Engineering. (New York:) McGraw-Hill Inc.

[Dym, 1994]
Dym, C. L. (1994). Engineering Design: A Synthesis of Views. Cambridge University Press.

[Edwards *et al.*, 1994]
Edwards, K. L., P. M. Sargent, & D. O. Ige. (1994). Designing with Composite Materials - an Expert System Approach. Proceedings of the The American Society of Mechanical Engineers 2nd Biennial European Conference on Engineering Systems Design & Analysis, London.

[Fisher, 1988]
Fisher, D. H. (1988). A Computational Account of Basic Level and Typicality Effects. Proceedings of the Seventh International Conference on Artificial Intelligence, Minneapolis, MN.

[Gentner, 1983]
Gentner, D. (1983). Structure-mapping: A Theoretical Framework for Analogy. Cognitive Science, 7(2).

[Glegg, 1969]
Glegg, G. L. (1969). The Design of Design. (London:) Cambridge University Press.

[Goel, 1989]
Goel, A. K. (1989). Integration of Case-Based Reasoning and Model-Based Reasoning for Adaptive Design Problem Solving. Ph.D. dissertation, The Ohio State University.

[Golding & Rosenbloom, 1994]
Golding, A. R., & P. S. Rosenbloom. (1994). The Evaluation of ANAPRON: A Case Study in Evaluating a Case-Based System. Seattle, WA: 12th National Conference on Artificial Intelligence.

[Hammond, 1989]
Hammond, K. (1989). Case-Based Planning: Viewing Planning as a Memory Task. San Diego: Academic Press.

[Hanney, 1996]
Hanney, K. (1996). Learning Adaptation Rules from Cases. MSc Thesis, Trinity College Dublin, Ireland.

[Hart *et al.*, 1978]
Hart, P. E., R. O. Duda, & M. T. Einaudi. (1978). A Computer-Based Consultation System for Mineral Exploration (Technical report). SRI International.

[Heness, 1996a]
Heness, G. (1996, October 31). Personal communication (email).

[Heness, 1996b]
Heness, G. (1996, November 15). Personal communication (email).

[Hennessy & Hinkle, 1992]
Hennessy, D. H., & D. Hinkle. (1992). Applying Case-Based Reasoning to Autoclave Loading. IEEE Expert, 7(5), 21-26.

[Hinrichs, 1992]
Hinrichs, T. R. (1992). Problem Solving in Open Worlds: A Case Study in Design. Northvale, NJ: Erlbaum.

[Hofmann, 1996]
Hofmann, H. F. (1996, October 9). Personal communication (email).

[Hua & Faltings, 1993]
Hua, K., & B. Faltings. (1993). Exploring Case-based Building Design--CADRE. AI EDAM, 7(2), 135-143.

[Hulthage *et al.*, 1990]
Hulthage, I. A. E., M. S. Fox, M. D. Rychener, & M. L. Farinacci. (1990). The Architecture of ALADIN: A Knowledge-Based Approach to Alloy Design. IEEE Expert, 5(4), 56-73.

[Kamel & Sticklen, 1990]
Kamel, A., & J. Sticklen. (1990). An Artificial Intelligence-Based Design Tool for Thin Film Composite Materials. Proceedings of the The Fifth Technical Conference of The American Society for Composites, East Lansing, Michigan. 375-381.

[Kamel, 1994]
Kamel, A. (1994). Generating Multiple Design Alternatives of Composite Materials using a Generic-Task Approach. Ph.D., Michigan State University.

[Kamel *et al.*, 1994]
Kamel, A., J. K. McDowell, & J. Sticklen. (1994). Multiple Design: An Extension of Routine Design for Generating Multiple Design Alternatives. Proceedings of the Artificial Intelligence in Design '94, Lausanne, Switzerland. 275-292.

[Kamel *et al.*, 1996]
Kamel, A., J. Sticklen, & T. J. Lenz. (1996). Leveraging Task Specific Frameworks to Modify Existing Designs for Composite Materials (Technical Report): Intelligent Systems Laboratory, Michigan State University.

[Kolodner, 1983a]
Kolodner, J. (1983). Maintaining organization in a dynamic long-term memory. Cognitive Science, 7, 243-280.

[Kolodner, 1983b]
Kolodner, J. (1983). Reconstructive memory, a computer model. Cognitive Science, 7, 281-328.

[Kolodner, 1984]
Kolodner, J. L. (1984). Retrieval and Organization Strategies in Conceptual Memory: A Computer Model. Northvale, NJ: Erlbaum.

[Kolodner, 1993]
Kolodner, J. (1993). Case-Based Reasoning. San Mateo: Morgan Kaufmann.

[Koton, 1989]
Koton, P. (1989). Evaluating Case-Based Problem Solving. Proceedings of the Proceedings of the Case-Based Reasoning Workshop (DARPA), Pensacola, Florida. 173-175.

[Laird *et al.*, 1987]
Laird, J. E., A. Newell, & P. S. Rosenbloom. (1987). SOAR: An Architecture for General Intelligence. Artificial Intelligence, 33, 1-64.

[Leake, 1995]
Leake, D. B. (1995). Case-Based Reasoning: Issues, Methods, and Technology. Sesimbra, Portugal: First International Conference on Case-Based Reasoning. Available on-line from David's web site at http://www.cs.indiana.edu/hyplan/leake/cbr/tutorial_slides.ps.Z.

[Lenz *et al.*, 1994]
Lenz, T., J. K. McDowell, B. Moy, J. Sticklen, & M. C. Hawley. (1994). Intelligent Decision Support for Polymer Composite Material Design in an Integrated Design Environment. Proceedings of the American Society of Composites 9th Technical Conference on Composite Materials, Newark, Delaware. 685-691.

[Lenz *et al.*, 1995a]
Lenz, T., J. K. McDowell, A. Kamel, J. Sticklen, & M. C. Hawley. (1995). A Decision Support Architecture for Polymer Composites Design: Implementations and Evolution. Proceedings of the Eleventh IEEE Conference on Artificial Intelligence Applications, Los Angeles, CA. 102-108.

[Lenz *et al.*, 1995b]
Lenz, T., J. K. McDowell, J. Sticklen, & M. C. Hawley. (1995, October). An Integrated Approach to the Design and Fabrication of a Composite Seeker Casing. Composites Processor, The Newsletter for the NSF Center for Low Cost, High Speed Polymer Composites Processing at Michigan State University, 2, 2-3.

[Lenz *et al.*, 1996a]
Lenz, T. J., J. K. McDowell, B. Moy, J. Sticklen, & M. C. Hawley. (1996). Intelligent Decision Support Systems in Polymer Composites Design: Industrial Focus by Application Segment and Product Line. Proceedings of the 41st International SAMPE Symposium, Anaheim, CA. 1259-1268.

[Lenz *et al.*, 1996b]
Lenz, T., J. K. McDowell, A. Kamel, J. Sticklen, & M. C. Hawley. (1996). The Evolution of a Decision Support Architecture for Polymer Composites Design. IEEE Expert, 11(5), 77-83.

[Maher & Zhang, 1993]
Maher, M. L., & D. M. Zhang. (1993). CADSYN: A Case-based Design Process Model. AI EDAM, 7(2), 97-110.

[McDermott, 1982]
McDermott, J. (1982). R1: A rule-based configurer of computer systems. Artificial Intelligence, 19(1), 39-88.

[McDermott, 1984]
McDermott, J. (1984). R1 revisited: Four years in the trenches. AI Magazine, 5(3).

[McDowell *et al.*, 1993]
McDowell, J. K., A. Kamel, J. Sticklen, & M. C. Hawley. (1993). Domain Structure and Inference Strategies in Knowledge-based Design of Polymer Composites. Proceedings of the SAE Technical Conference, Detroit, MI. #930167.

[McDowell *et al.*, 1996]
McDowell, J. K., A. Kamel, J. Sticklen, & M. C. Hawley. (1996). Integrating Material/Part/Process Design for Polymer Composites: A Knowledge-Based Problem-Solving Approach. Journal of Thermoplastic Composite Materials, 9(3), 218-238.

[Michaeli & Biswas, 1994]
Michaeli, W., & A. Biswas. (1994). Knowledge- and Case-based Computer Support for the Design of Composite Parts. Proceedings of the ANTEC '94, San Francisco. 2347-2350.

[Mitchell *et al.*, 1986]
Mitchell, T. M., R. M. Keller, & S. T. Kedar-Cabelli. (1986). Explanation-based Generalization: A Unifying View. Machine Learning, 1(1), 47-80.

[Moy *et al.*, 1994]
Moy, B., J. K. McDowell, T. Lenz, J. Sticklen, & M. C. Hawley. (1994). Expansion of an Intelligent Decision Support System for Process Selection and Process Design in Polymer Composites. Proceedings of the 26th International SAMPE Technical Conference, Atlanta, Georgia. 162-171.

[Netten *et al.*, 1995]
Netten, B. D., R. A. Vingerhoeds, & H. Kippelaar. (1995). Expert Assisted Conceptual Design: An Application to Fibre Reinforced Composite Panels. In R. M. Oxman, M. F. T. Bax, & H. H. Achten (Eds.), Design Research in The Netherlands, (pp. 125-139). Eindhoven: Faculty of Architecture, Planning, and Building Science. Eindhoven University of Technology.

[Netten, 1996]
Netten, B. (1996, July 19). Personal communication (email).

[Netten & Vingerhoeds, 1996a]
Netten, B. D., & R. A. Vingerhoeds. (1996). Adaptation for Conceptual Design in EADOCS. 12th European Conference on Artificial Intelligence.

[Netten & Vingerhoeds, 1996b]
Netten, B. D., & R. A. Vingerhoeds. (1996). EADOCS: Conceptual Design in 3 Steps. An Application to Fibre Reinforced Composite Panels (Technical report): Delft University of Technology.

[Newell, 1982]
Newell, A. (1982). The Knowledge Level. Artificial Intelligence, 18, 87-127.

[Petrie *et al.*, 1994]
Petrie, C., J. Wagner, & Others. (1994). The MADEFAST Project (Web site): www.madefast.org.

[Petroski, 1991]
Petroski, H. (1991). On the Backs of Envelopes. American Scientist, 79, 15-17.

[Petroski, 1994]
Petroski, H. (1994). Design Paradigms: Case Histories of Error and Judgement in Engineering. Cambridge University Press.

[Phillips, 1989]
Phillips, L. N. (Ed.). (1989). Design with Advanced Composite Materials. London: The Design Council.

[Pitchumani *et al.*, 1993]
Pitchumani, R., P. A. Schwenk, V. M. Karbhari, J. F. Ramsay, & T. D. Claar. (1993). A Knowledge-based Decision Support System for the Manufacture of Composite Preforms. Proceedings of the 25th SAMPE Technical Conference, Philadelphia, PA.

[Pitchumani & Karbhari, 1994]
Pitchumani, R., & V. M. Karbhari. (1994). Use of Knowledge-Based Decision Support Systems for Rapid, Efficient Production Planning, with Applications to Ceramic-Matrix Composites Fabrication. Proceedings of the Annual American Ceramics Society Conference, Cocoa Beach.

[Pitchumani *et al.*, 1994]
Pitchumani, R., P. A. Schwenk, A. K. Kordon, V. M. Karbhari, B. R. Rossing, & T. D. Claar. (1994). An Expert System Approach to Manufacturing Preforms for Infiltration-Processing of Ceramic- and Metal-Matrix Composites. Processing of Advanced Materials, 4(3), 155-165.

[Principe *et al.*, 1987]
Principe, F. S., M. M. Monib, & D. R. Linsenmann. (1987). Design Requirements. In ASM International (Ed.), Engineered Materials Handbook: Composites, (Vol. 1, pp. 181-184). Metals Park, OH: ASM International.

[Punch, 1989]
Punch, W. F. (1989). A Diagnostic System Using A Task Integrated Problem Solver Architecture (TIPS). Ph.D. dissertation, The Ohio State University.

[Rich & Knight, 1991]
Rich, E., & K. Knight. (1991). Artificial Intelligence. (2 ed.): McGraw-Hill, Inc.

[Riesbeck & Schank, 1989]
Riesbeck, C., & R. Schank. (1989). Inside Case-Based Reasoning. Erlbaum.

[Rissland *et al.*, 1989]
Rissland, E. L., J. L. Kolodner, & D. Waltz. (1989). Case-Based Reasoning from DARPA: Machine Learning Program Plan. Proceedings of the Case-Based Reasoning Workshop (DARPA), Pensacola, Florida. 1-13.

[Roderman & Tsatsoulis, 1993]
Roderman, S., & C. Tsatsoulis. (1993). PANDA: A Case-based System to Aid Novice Designers. AI EDAM, 7(2), 125-133.

[Samuel, 1963]
Samuel, A. L. (1963). Some studies in machine learning using the game of checkers. In E. A. Feigenbaum & J. Feldman (Eds.), Computers and Thought. New York: McGraw-Hill.

[Schank, 1982]
Schank, R. C. (1982). Dynamic Memory: A Theory of Learning in Computers and People: Cambridge University Press.

[Schroeder, 1995]
Schroeder, K. (1995). Integration Documentation: Knowledge Engineer (Technical Report): Intelligent Systems Laboratory, Michigan State University.

[Smithers, 1996]
Smithers, T. (1996). On Knowledge Level Theories of Design Process. Artificial Intelligence in Design '96, Palo Alto, CA. 561-579.

[Steier *et al.*, 1993]
Steier, D. M., R. L. Lewis, J. F. Lehman, & A. L. Zacherl. (1993). Combining Multiple Knowledge Sources in an Integrated Intelligent System. IEEE Expert, 8(3), 35-44.

[Sticklen, 1989]
Sticklen, J. (1989). Problem Solving Architectures at the Knowledge Level. Journal of Experimental and Theoretical Artificial Intelligence, 1(1), 1-52.

[Sticklen *et al.*, 1992a]
Sticklen, J., A. Kamel, M. Hawley, & V. Adegbite. (1992). Fabricating Composite Materials: A Comprehensive Problem Solving Architecture Based on a Generic Task Viewpoint. IEEE Expert, 7(2), 43-53.

[Sticklen *et al.*, 1992b]
Sticklen, J., A. Kamel, M. Hawley, & J. Delong. (1992). An Artificial Intelligence-Based Design Tool for Thin Film Composite Materials. Applied Artificial Intelligence, 6(6), 303-313.

[Strong, 1989]
Strong, A. B. (1989). Fundamentals of Composites Manufacturing: Materials, Methods, and Applications. (Dearborn, MI:) Society of Manufacturing Engineers.

[Turner, 1988]
Turner, R. M. (1988). Opportunistic Use of Schemata for Medical Diagnosis. Proceedings of the 10th Annual Conference of the Cognitive Science Society.

[Turner, 1989]
Turner, R. M. (1989). Case-Based and Schema-Based Reasoning for Problem Solving. Proceedings of the Case Based Reasoning Workshop (DARPA), Pensacola, Florida. 341-344.

[Wallingford, 1992]
Wallingford, V. E., II. (1992). Conceptual retrieval from case memory based on problem-solving roles: A generic task architecture with application to justificatory reasoning in tax law. Ph.D. dissertation, Michigan State University.

[Wilenski *et al.*, 1995]
Wilenski, M. S., T. J. Lenz, J. K. McDowell, J. Sticklen, & L. T. Drzal. (1995). A Knowledge Based Approach to Interphase Design and Composite Selection. Advanced Composites Conference and Exhibition (ACCE), Dearborn, MI.

[Zumsteg *et al.*, 1985]
Zumsteg, J. R., D. Pecora, & V. J. Pecora. (1985). A Prototype Expert System for the Design and Analysis of Composite Material Structures. Proceedings of the ASME International Conference on Computers in Engineering. 137-143.