DEVELOPMENT OF AN EXPERT SYSTEM BASED EXPERIMENTAL FRAME FOR MODELING OF MANUFACTURING SYSTEMS

By

DURSUN DELEN

Bachelor of Science
Istanbul Technical University
Istanbul, Turkey
1986

Master of Science
Yildiz University
Istanbul, Turkey
1988

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
May, 1997

DEVELOPMENT OF AN EXPERT SYSTEM BASED EXPERIMENTAL FRAME FOR MODELING OF MANUFACTURING SYSTEMS

Thesis Approved:

David B. Pratt

Thesis Advisor

Manjunath Kamath

T. J. Greene

Michael H. Branson

Thomas C. Collins

Dean of the Graduate College

ACKNOWLEDGMENTS

In the name of Allah, most gracious, most merciful. All praises and thanks be to Allah, the cherisher and the sustainer of the worlds. As a Muslim, my life and death belongs to Allah and so does my first thanks. I thank Allah for granting me perseverance and strength I needed to complete this study.

I wish to express my sincere appreciation to my major advisor, Dr. David B. Pratt for his intelligent supervision, constructive guidance, inspiration, and friendship. I thank him for being not only a great teacher and a true researcher in the field of Industrial Engineering but also a close friend and an excellent role model for his students including myself. My sincere appreciation extends to my other committee members, Dr. Timothy Greene, Dr. Manjunath Kamath, Dr. Mike Branson, and Dr. Adedeji B. Badiru, whose guidance, assistance, encouragement, and friendship are also invaluable.

More over, I wish to express my sincere gratitude to those who have provided the necessary knowledge and experience required for this study: Dr. Joe H. Mize, Dr. Ghassan Abdelnour, Dr. Steve Rogers, Dr. Simin Pulat, Dr. Mustafa Pulat, Dr. Bobbie Foote, Dr. Deborah J. Seifert, Dr. Phil Wolfe, Dr. Terrence G. Beaumeriage, Dr. Stan Settles, Dr. Ying Tat Leung, Dr. Devanath Tirupati, and Dr. Ronald G. Askin. Thank you.

To my friends, the colleagues in the Center for Computer Integrated Manufacturing, thank you for making me feel like a member of the family, the CIM Team. I will always remember and cherish our friendship.

I would also like to give my special appreciation to my best friend and wife, Handan, for her strong encouragement at times of difficulty, love and understanding throughout this whole process. Special thanks also go to my parents, Süleyman and Ayse Delen, for their support and encouragement.

Finally, I would like to thank the Center for Computer Integrated Manufacturing and the School of Industrial Engineering and Management at Oklahoma State University for providing financial support during this study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

CHAPTER I

INTRODUCTION

Motivation For The Research

Advanced manufacturing technology, without a doubt, is a major corporate advantage in today's global market. Strategic application of such technology can markedly improve a manufacturer's product quality, responsiveness to customers, process control, process flexibility, and flexibility of capital investment which are all determinants of global manufacturing competitiveness.

The stochastic nature of manufacturing systems with ever changing market behavior, makes it difficult for manufacturing engineers to analyze and design/redesign a complete discrete part manufacturing system. The most common way of analyzing stochastic systems, such as discrete part manufacturing systems, is to conduct experiments on them. Conducting experiments on such systems can be done in one of two forms: experimenting with the actual system or experimenting with a model of the system. If it is possible and cost effective to alter the system physically and allow it operate under the new conditions, it is probably desirable to do so, for in this case there is little question about whether the results of the experiment are valid. However, it is rarely feasible to do this because such experiments are often too costly or too disruptive to the system. In many cases, the "system" might not even exist, as in the case of initially designing a discrete part manufacturing system. Nevertheless, we may want to study the system in various proposed alternative configurations to see how it should be built in the first place. For these reasons, it is usually necessary to build a model as a representation of the system and conduct the analyses on this model.

Models can be physical (cockpits disconnected from their airplanes to be used in pilot training or a manufacturing laboratory with machines and material handlers, etc.) or conceptual (representing a system in terms of logical and quantitative relationships that are then manipulated and changed to see how the model reacts). Figure 1 illustrates a taxonomy of ways to study a system which is similar to the one proposed by Pritsker [1986].

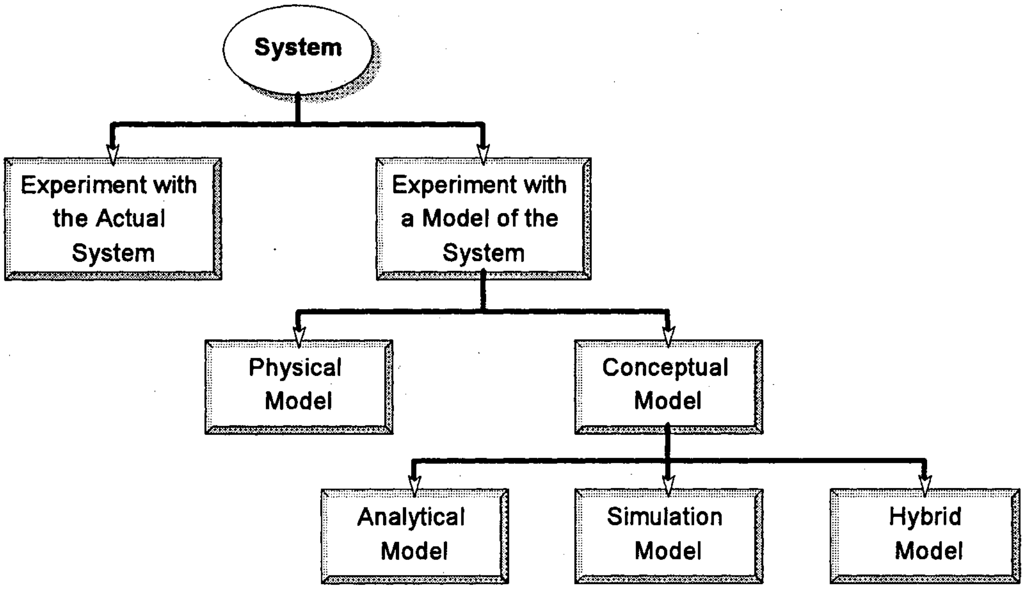


Figure 1. The Ways to Study a System

Once a conceptual model is built, it must then be examined to see how it can be used to answer the questions of interest about the system it represents. If an analytical solution to the conceptual model is available and is computationally efficient, it is usually desirable to study the model in this way rather than via simulation. However, many systems are highly complex, so that valid mathematical models of them are themselves complex. In this case a model may be studied by means of simulation, i.e., numerically exercising the model for the inputs in question to see how they affect the output measures of performance, or by means of hybrid model which combines analytical solution with simulation [Shanthikumar and Sargent 1983].

It has been demonstrated many times that modeling, especially computer modeling (analytical and/or simulation), is vital for the design of complex manufacturing systems [Leung and Suri 1990; ElMaraghy and Ravi 1992; Mize et al. 1992; Suri and De Treville 1993]. Modeling is perhaps the only method for analyzing the stochastic behavior of such systems under various scenarios. Nevertheless, modeling of such complex systems is not without its disadvantages. First, models have traditionally been viewed as single purpose, throw-away efforts. A model is built from scratch to address a particular problem or question, and then it is often discarded with no thought given to additional use. This single-use, throw-away mentality of modeling is very expensive, time consuming and wasteful. Second, lack of access by non-modeling specialists to models limits their usage and value.

The primary motivation for this research is the author's desire to contribute to the advancement of modeling of manufacturing systems by addressing some of the issues related to the disadvantages of traditional modeling methodologies mentioned above.

Overview of the Dissertation

The remainder of this dissertation is presented in seven chapters plus a bibliography and appendixes. Chapter II develops the problem statement in detail, within which, many of the points made above are explored more fully. Chapter III reviews the literature on modeling of manufacturing systems, expert system applications in modeling, expert system applications in manufacturing, and finally, expert systems in modeling of manufacturing systems. This review chapter is not intended to be a comprehensive review of all the related literature. Rather, only recent literature relevant to this study is cited. In Chapter IV expert systems and knowledge engineering processes are summarized. Research goals and research objectives are defined in Chapter V along with the scope and limitations of this research. In Chapter VI the details of the research methodology are discussed. Research results, with supporting example scenarios, are presented in Chapter VII. Chapter VIII is the summary and conclusion chapter that synopsizes the results of this effort and suggests directions that appear fruitful for additional investigation.

CHAPTER II

STATEMENT OF THE PROBLEM

Introduction

A system is a collection of objects working together toward a common goal. A manufacturing setting is an excellent example of a system. It is a collection of objects such as people, physical objects (machines, material handlers, etc.), information objects (bills of material, routing of parts, etc.), and control objects (logic for which part to remove from a specific buffer, logic for how to determine which material handler to use, etc.). All of these objects work together toward a common goal: to manufacture products with desired quality and quantity specifications.

A model, in the simplest sense, is a representation of a system. If the model is expressed mathematically, as a set of logical and functional relationships, it is referred to as an abstract model. If the model is a collection of physical objects that has one-to-one mapping to the real system then it is referred to as a physical model. For the purposes of this research, computer modeling of manufacturing systems is defined to be an abstract model implemented on a computer upon which experiments are conducted for the purpose of generating information useful in making decisions.

Traditional approaches to the modeling of complex manufacturing systems have two major disadvantages:

1. Models have been viewed as single purpose, throw-away efforts. A model is built from scratch to address a particular problem or question, and then it is often discarded. When a new problem is encountered, a new model is generated from scratch even though it may include elements contained in earlier models. This single-use, throw-away mentality of modeling is obviously very expensive, time consuming and wasteful [Mize et al. 1992].
2. Lack of access by non-modeling specialists. As stated by Youngblood [1991], "... direct use of a model is usually limited to a few experts, people who have spent a great deal of time learning about the model and how it works. Anyone wishing to obtain results from the model must ask for help from one of the experts to input the required parameters into the model, run the model, and then interpret the results."

One approach to address these issues lies in taking advantage of recent developments in several related areas. These areas include Object Oriented Programming (OOP), Object Oriented Modeling, Artificial Intelligence (AI)/Expert Systems, Knowledge Engineering, Software Engineering, and Modeling Formalisms. OOP and Modeling Formalisms make it possible to build highly reusable, general purpose software components whereas AI/Expert Systems and Knowledge Engineering help us to build readily accessible and user friendly software environments.

Reusable and Plug-Compatible Modeling

Object oriented programming, a paradigm in which all program variables are represented as objects which communicate by means of message passing, appears to be a significant advancement toward the development of multiple use, general purpose, and plug-compatible models [Zeigler 1990]. OOP, in order to achieve this advancement, possesses five key concepts: encapsulation, message passing, late binding, polymorphism, and inheritance. The differences between software developed in procedural languages and OOP languages are due to these five characteristics.

Software developers find the OOP approach superior for a variety of reasons [Tello 1989]. To one programmer, it might be the possibility of eliminating redundant code through inheritance that is most appealing. To another, it might be the protection that objects have through encapsulation which prevents objects from being invaded by code in other parts of the program. To still another, it might be the time saving involved in being able to build programs from standard programming components that communicate with one another, rather than having to start writing code from scratch.

The concepts underlying OOP can be extended to modeling, especially simulation modeling [Adiga 1989; Mizė et al. 1992; Narayanan et al. 1992; LeFrancois and Montreuil 1994]. Using these concepts an object oriented modeling (OOM) environment is under development within Oklahoma State University's Center for Computer Integrated Manufacturing. This environment specifically targets reusability and plug-compatibility as key development factors. As a consequence of the reusability emphasis, the bottom-up modeling strategy is employed to create modeling constructs for the lowest level physical, informational, and control components of a real world manufacturing system [Pratt, Mize, and Kamath 1993]. These modeling constructs, comprising both generic elements and company-specific elements, are referred to as modeling primitives. These primitives, after being validated, become part of a manufacturing modeling library.

Another key consequence of the reusability emphasis is the implementation of the separation concept [Mize et al. 1992; Pratt et al. 1995]. The implementation of separation involves the creation of separate and distinct modeling primitives for physical elements, information flow, and control decisions. Traditional simulation languages do not provide natural constructs for separately and distinctly modeling physical, informational, and control elements. According to Pratt et al. [1995], the constructs provided for information and control are frequently hard coded and dispersed into the model. This results in code that is hard to modify and difficult to use for multiple purposes.

Designing for reusability involves identification of behaviors that are useful in more than one context. In general, this implies a system design which adheres rather strictly to the "one-component-one function" doctrine. If a component performs more than one of the three basic functions (i.e., physical, information, and/or control), its usage becomes limited to situations in which all of its functions are required. On the other hand if a strict one-to-one functionality is maintained between component and function, then the components truly become "building blocks" from which a total system model can be constructed.

Another advantage of the separation of physical, information, and control objects is that it allows the system modeler to think of these elements independently during model development. This provides a more natural model development environment. In other words, when developing the physical model, the model builder need not be concerned with information or control aspects. The process involves selecting the appropriate physical components without being constrained by concerns regarding how to model information flow. Similarly, information flow is considered without regard to physical objects. This independence facilitates the creation of models with a higher degree of integrity and greater flexibility relative to experimentation with the models.

Base Model Concept

A base model is an abstraction of a real world manufacturing system in the richest possible way [Duse et al. 1993]. In the creation of a base model, as depicted in Figure 2, a library of manufacturing modeling primitives can be used [Delen, Pratt, and Kamath 1996a]. These primitives are stored in a library and, as previously stated, can be classified into three types: physical primitives, information primitives, and control primitives. This library of manufacturing modeling primitives is assumed to include all necessary manufacturing objects in their abstract forms. A user (model builder) can construct the manufacturing system specific base model by simply selecting appropriate modeling primitives and assembling them into a base model using a windows driven software environment.

A base model is never complete. It evolves with the organization, and it is persistent over time. It is maintained as an ongoing activity just as a company database is created and updated on an ongoing basis. Thus, it is a modeling activity for the sake of modeling and not pursued with an immediate specific purpose in mind, as is done in the traditional modeling approach which is purpose driven and tool specific. Given the existence of such a base model, one can derive tool-specific models, called execution models, using configurators and translators which reside between the base model and the solvers (Figure 3).

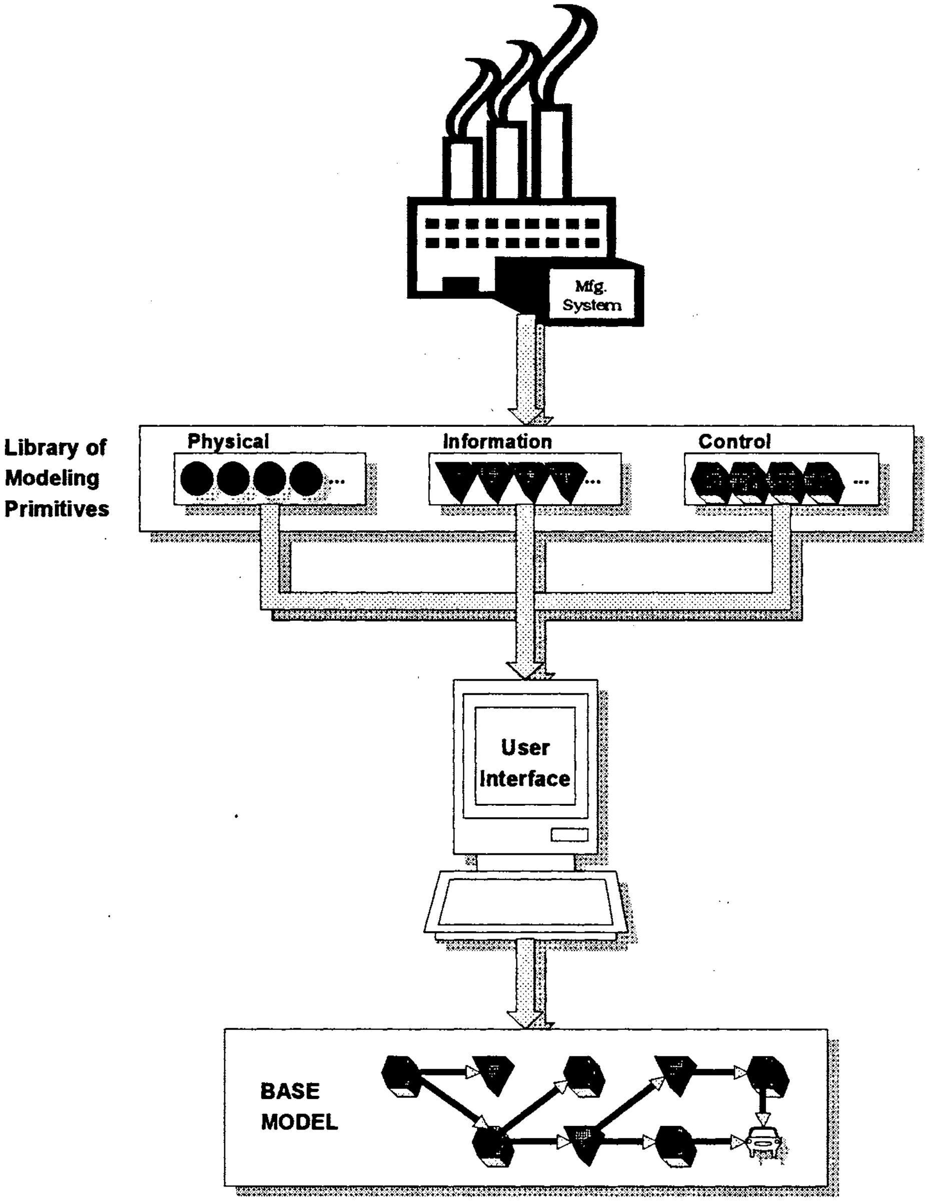


Figure 2. The Process of Constructing the Base Model

A modeling environment should provide a variety of appropriate analysis tools for solving problems which lie in different domains. For instance, rough-cut system design problems require aggregate estimates of performance measures and one usually employs analytical methodologies (e.g., queuing network models and mathematical programming models) for their solution [Suri and De Treville 1993]. On the other hand, for determining detailed estimates of performance measures, one usually employs a simulation based approach. It is assumed that the base model, which is a tool independent representation of a specific manufacturing system, has all the information necessary for any analysis tool in the available set of tools. Once the analysis tool is determined based on the specific experimental circumstances, the next step is to configure the base model into an execution model of the selected analysis tool. Such a process is illustrated in Figure 3.

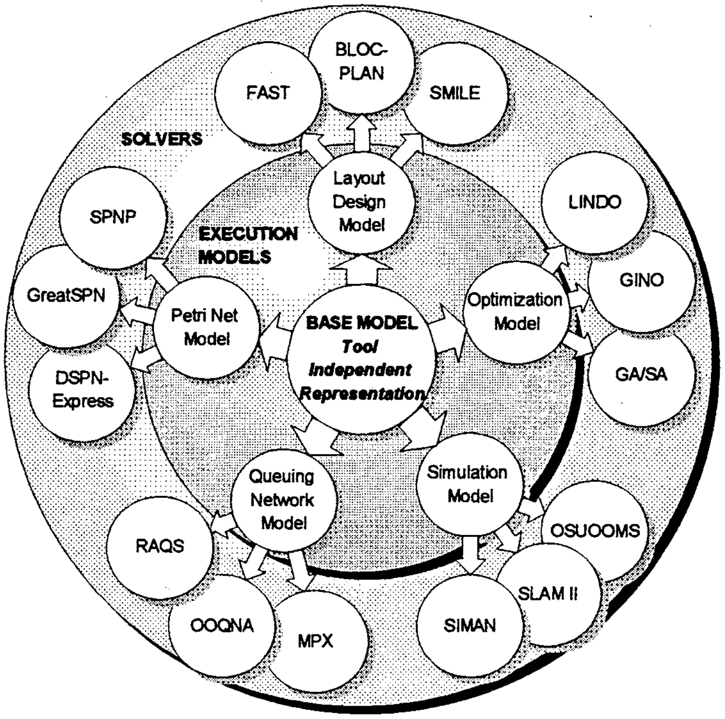


Figure 3. Tool Independent Model Representation

Research underway within Oklahoma State University's Center for Computer Integrated Manufacturing presents a new modeling framework (Figure 4) [Kamath, Pratt and Mize 1995]. The *Manufacturing System* (1) block in Figure 4 represents the real world manufacturing system. A *Model Builder* can construct the tool independent, generic, persistent *Base Model* (3) by using the *Base Model Configurator* (2). Once the base model is constructed a *Decision Maker* can seek answers to the questions related to the manufacturing system through an *Experimental Frame* (4). Based on the nature of the problem to be solved and the complexities of the manufacturing system under study an appropriate *Tool Specification* (5) can be determined.

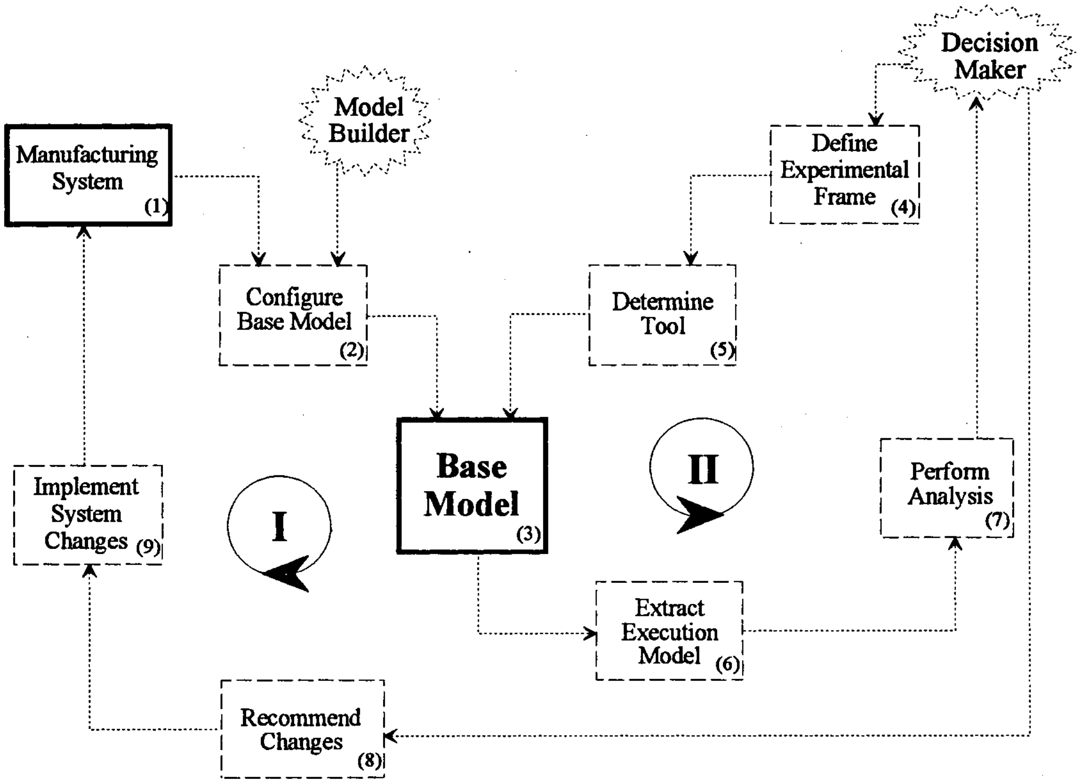


Figure 4. A New Framework for Manufacturing Modeling

With the tool specification in hand, a tool dependent configurator extracts the *Execution Model* (6) from the base model. *Performing Analysis* (7) will lead to either another experimental frame definition or *Recommended Changes* (8) for the system. As *Changes are Implemented* (9) within the manufacturing system, it is necessary to update the base model to reflect the changes.

The first disadvantage related to the traditional modeling of manufacturing systems, namely reusability and plug-compatibility issues mentioned early in this chapter, has been addressed by OSU's CCIM research group in their advanced modeling project [Kamath, Pratt and Mize 1995; Kamath et al. 1996]. The second disadvantage, namely lack of access to models by non-modeling specialists, is addressed by this research.

Experimental Frame Expert System

In the modeling framework presented in Figure 4, there are two primary processes to be considered. The first process is the *Model Building and Maintenance* process and is illustrated with feedback loop I. This process includes not only the one-time effort of initial base model development (path 1-2-3 in Figure 4), but also, the incremental on-going effort to maintain the currency and accuracy of the base model representation as the manufacturing system evolves (path 7-8-9-1 in Figure 4). The second process is the *Model Utilization* process and is illustrated in Figure 4 with feedback loop II. This process (path 4-5-3-6-7-4 in Figure 4) includes the usage of the base model to assist decision makers in designing and reconfiguring the manufacturing system. As Figure 4 indicates, analysis may lead to improving the system through implementation of changes where the utilization loop feeds into the model building and maintenance loop by which the base model is incrementally updated [Kamath, Pratt and Mize 1995].

The model utilization process can also be characterized as an experimentation process through which a decision maker can interact with the modeling environment in order to obtain answers to the questions related to the manufacturing system during a decision making process [Delen, Pratt and Kamath 1996a]. Such an experimentation process concept gives rise to two important issues, namely problem domain and tool domain. Within problem domain the concerns are focused on the expertise that exists relative to various questions and problems that are typically found within a discrete part manufacturing system. This expertise allows the decision maker to derive a structured problem from a set of symptoms and unstructured questions to be used as inputs to the tool selection process. Tool domain refers to expertise relative to the capabilities and limitations of a given set of tools with respect to the problems that are typically found within a discrete part manufacturing system.

Before justifying an AI (specifically expert systems) approach to the model utilization process, a few definitions, with regard to the structured problem, symptom, and experimental frame, are needed.

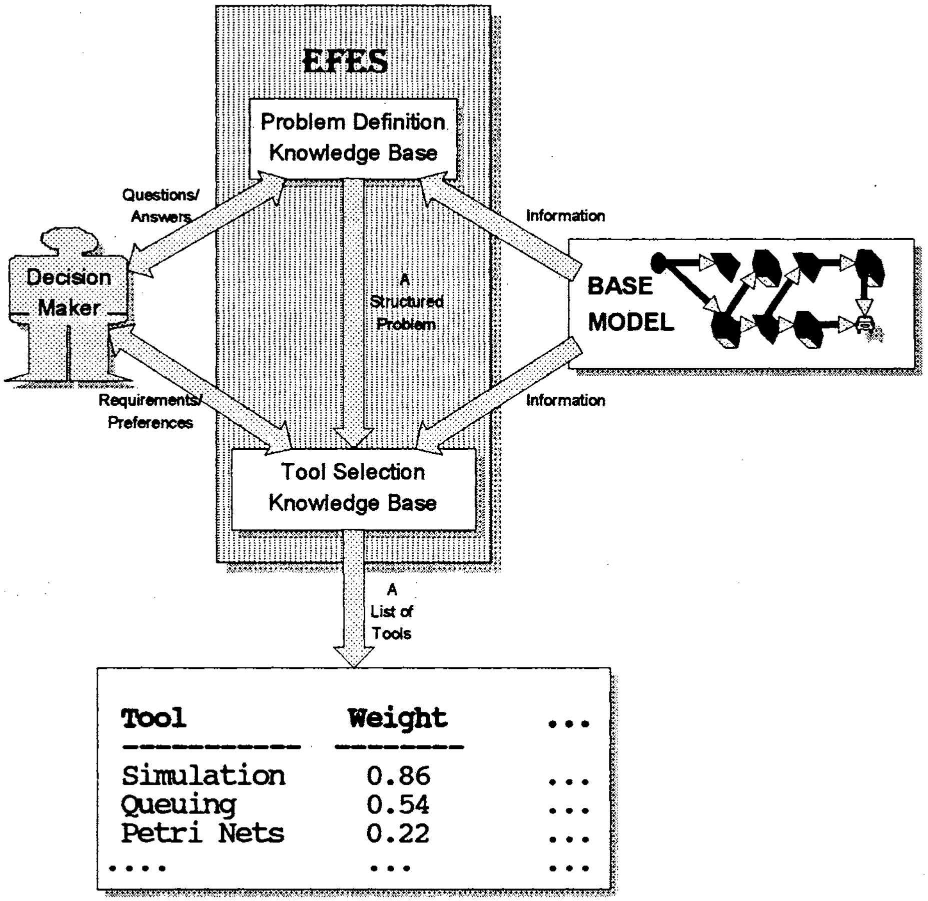
To the best of the author's knowledge, there is no clear definition for "structured problem" in the context of discrete part manufacturing systems, in the published literature. For the purposes of this research, the term structured problem will be used in a general context. It is a concise description of a situation that implies "need for investigation". Not only the problems (dissatisfying situations), but also the what-if questions (scenario analyses that are generated from the need or the curiosity of the decision maker) fall into this definition. A symptom, for the purposes of this research, is defined as an observable (or perceivable) fact that indicates the possible existence of a structured problem. In some cases a symptom can also be a structured problem at the same time (e.g., low product quality). The experimental frame has been defined by Tretheway, Hunt and Court [1995] as "... a framework for a component of a comprehensive modeling and analysis workstation." This definition does not provide the necessary depth required for this research, therefore, in the context of this research the author prefers to adopt the definition of experimental frame as "a methodology through which a decision maker can make use of a variety of modeling tools in his/her decision making processes." In other words, an experimental frame is a step-by-step process of helping decision makers to gain insight into the system being considered under various system configurations. Such a process starts with a need posed by the decision maker and ends with a list of modeling tools that can be used to address the need.

An expert system can be defined as " an interactive computer-based decision tool that uses both facts and heuristics to solve decision problems based on knowledge acquired from an expert" [Badiru 1992] (a more detailed tutorial and summary on expert systems is presented in Chapter IV). Expert systems are known as powerful tools to solve problems when (1) the solution asks for expert heuristics, (2) the data to be used is somewhat "noisy", (3) human expertise is scarce, (4) there is not a concise formula or a logical flow to the solution, and (5) fuzziness comes into play [Badiru 1992].

Based on these characteristics of expert systems, the experimental frame definition and tool selection steps of the previously described modeling process make two excellent candidates for the application of expert systems. They both require expertise in their own domain. The experimental frame definition deals with mapping a set of symptoms and unstructured questions to an appropriate structured problem in the context of the problem domain of discrete part manufacturing systems. The tool selection deals with selecting the most suitable tool for a given structured problem along with the user preferences and/or requirements in the context of the tool domain. Neither the experimental frame definition nor the tool selection problems have concise solution procedures to follow. They both deal with heuristics that evolve from an expert's knowledge and experience.

Once the two knowledge bases are embedded in the experimental frame module of the advanced modeling environment, a decision maker can seek answers to his/her questions with very little or no assistance from modeling specialists. Such a process is shown in Figure 5. In this figure the consultation starts with the decision maker posing questions and/or identifying symptoms to the problem definition knowledge base. This includes a selection of all applicable symptoms from a given list. Based on the selections made by the decision maker, the problem definition knowledge base asks more questions to acquire additional details in the process of making a decision on the specification of a structured problem. Such a question/answer session will continue until the problem definition part of the expert system collects enough information to make recommendations or determine that the problem lies outside its domain. Given that the structured problem is defined, another knowledge base, namely the tool selection knowledge base, takes over the consultation with the decision maker for the purpose of choosing the most appropriate analysis tool from a given set.

The tool selection knowledge base uses three main information sources as inputs to the decision making process: (1) a structured problem determined by the problem definition knowledge base, (2) information from the base model (number of machines, number of products, nature of the material handling system, etc.), (3) requirements and/or preferences of the decision maker.



Tool	Weight	...
Simulation	0.86	...
Queuing	0.54	...
Petri Nets	0.22	...
....	...	...

Figure 5. Interactions Between the two Knowledge Bases and Their Environment

The result of this consultation will be a list of available, appropriate tools ranked from most appealing to least appealing with numerical weights (preferences) attached. Diversification of these numerical weights indicates the level of fitness among the listed analysis tools with respect to the given situation. For instance, in Figure 5, simulation has been recommended as the most suited tool to study the system for the given situation. Queueing follows simulation in the preference ranking, and Petri nets is the least preferred among the three tools to study the given system.

When this Experimental Frame Expert System (EFES) is embedded into the previously presented modeling framework (Figure 4), the result is a more advanced modeling environment which can address not only reusability issues but also issues that deal with limited-access to models by non-modeling specialists (Figure 6).

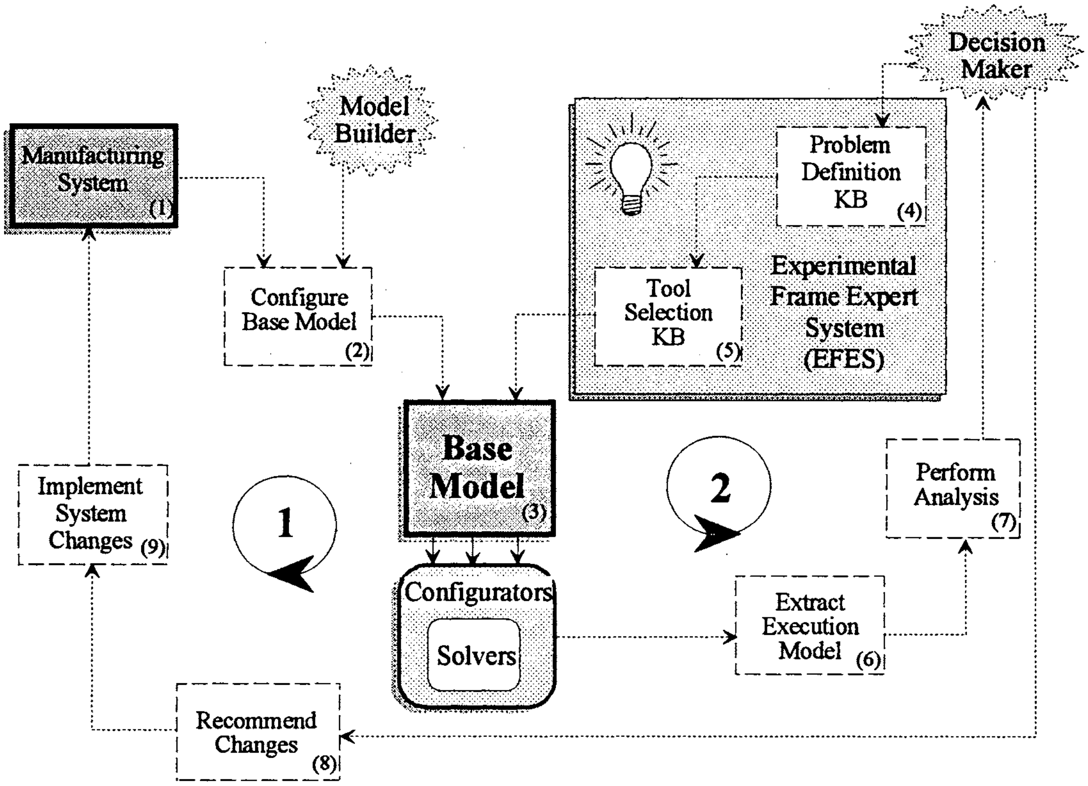


Figure 6. A New Framework with EFES Extension

If Figure 4 is compared to Figure 6, it can be seen that the left side of Figure 6, namely the model building and maintenance process, is identical to the left side of Figure 4. The difference between these two figures comes from their model utilization parts (the right sides of the figures). This part of Figure 6 is designed to supplement and possibly eliminate human modeling experts by EFES in order to allow decision makers to use the modeling environment more readily.

As mentioned earlier, the result of a consultation session with EFES is a list of tools that can be used to address a specific structured problem (Figure 5). While this list is ranked in preferred order, the decision maker is free to choose any tool from this prioritized list. Once a tool is selected, the environment creates an execution model of the selected tool, runs the execution model and returns the modeling results to the decision maker (Figure 6). Based on the output obtained from the consultation session the decision maker might conduct more experimentation until a suitable decision can be made. If this process leads to changes in the manufacturing system, the base model should also be updated to reflect such modifications.

Problem Statement

The objective of this research is *not to* develop another expert system application *but to* make significant contribution to the evolving field of manufacturing systems modeling by conceptualizing and implementing a new framework within which expert systems, the object oriented paradigm, and new modeling methodologies, which include the separation concept and the base model concept, are being utilized. The outcome of this research effort is expected to make modeling a more attractive tool for use by manufacturing managers in their decision making processes by addressing one of the most important shortcomings of the traditional modeling methodologies, namely lack of access to models by non-modeling specialists. Thus, the problem statement of this research can be summarized as:

"Availability of modeling experts is frequently a limiting factor in the use of models to solve manufacturing system configuration/reconfiguration problems."

Unanswered Questions

The above development leaves many unanswered questions regarding potential contributions and applicability of expert systems in computer modeling environments for manufacturing systems. Among the unanswered questions are:

- Can a knowledge base supplement or replace the expertise requirement in the process of deriving a structured problem from a set of symptoms? Here the knowledge base is focused on manufacturing systems engineering expertise.
- Can a knowledge base supplement or replace the expertise requirements in the process of suggesting the best analysis tool to use for a given set of conditions? Here the knowledge base is focused on system analysis expertise.
- Can a modeling environment for manufacturing systems be developed for decision makers to use without needing help from one or more modeling experts? Here the modeling environment for manufacturing systems incorporates the two knowledge bases mentioned above.

This research effort seeks to address these questions and gain insight into a methodology for creating and using such a modeling environment.

CHAPTER III

REVIEW OF PREVIOUS WORK

Introduction

This chapter reviews the literature in the domain of intersections between expert systems, modeling, and manufacturing system (Figure 7).

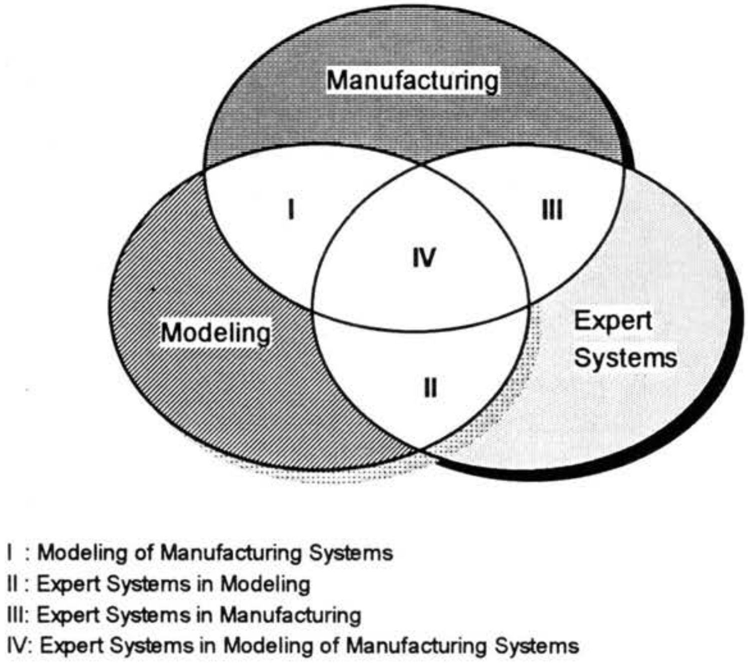

Manufacturing
Modeling
Expert Systems
I
II
III
IV
I : Modeling of Manufacturing Systems
II : Expert Systems in Modeling
III: Expert Systems in Manufacturing
IV: Expert Systems in Modeling of Manufacturing Systems

Figure 7. Sections of Literature Review in a Venn Diagram

The review is divided into four sections. The first section presents a review of literature in the area of modeling of manufacturing systems (intersection I in Figure 7).

Since this domain includes a vast amount of literature, only a few of the most relevant and most recent research efforts will be reviewed. The second section reviews the literature related to expert system applications in modeling (intersection II in Figure 7). This section not only discusses applications of simulation modeling in manufacturing but also applications of modeling in general. The third section reviews literature in the area of expert system applications in manufacturing. This section includes expert system applications in many manufacturing functional areas (intersection III in Figure 7), for instance, scheduling, quality control, process design, facility layout, etc. The fourth, and the last, section reviews the related literature in the domain of expert system applications in modeling of manufacturing systems (intersection IV in Figure 7).

Literature Review of Modeling of Manufacturing Systems

Various modeling techniques are used to design and evaluate the performance of manufacturing systems [ElMaraghy and Ravi 1992]. Figure 8 shows these modeling techniques in a graphical perspective.

Physical models are operational scaled models of the actual system. They use hardware devices whose characteristics are similar to those used in the actual system. More information about physical models, or physical simulators as they are often called, can be found in Deisenroth, Nof and Meier [1980] and Diesch and Malstrom [1985].

Analytical models are abstract representations of actual systems. They use mathematical variables and expressions to represent the physical quantities and behavior of the actual system. Two types of operations research techniques, namely queuing networks and mathematical programming, have been used extensively to develop analytical models.

In the queuing networks modeling approach, a manufacturing system is modeled either as an open queuing network or a closed queuing network. Open queuing network applications to model and evaluate the performance of manufacturing systems can be found in Shanthikumar and Buzacott [1981], Buzacott and Shanthikumar [1985], and Shanthikumar and Stecke [1986] whereas closed queuing network applications to model and evaluate the performance of manufacturing systems can be found in Solberg [1977].

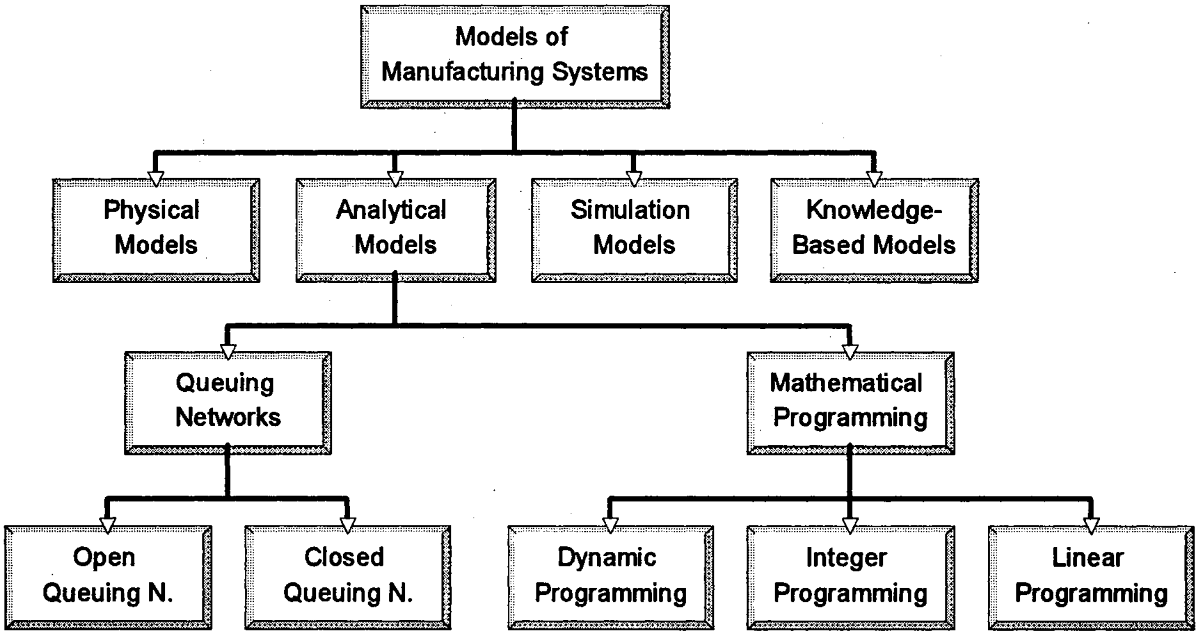


Figure 8. Manufacturing Systems Modeling Techniques

Mathematical programming deals with the optimal allocation of a limited set of resources to competing activities subject to a number of constraints [ElMaraghy and Ravi 1992]. Mathematical programming techniques such as linear programming, integer programming, and dynamic programming have been used to formulate and solve manufacturing system planning and scheduling problems. A variety of examples can be found in Kimemia and Gershwin [1983], Kusiak [1983], and Stecke [1983].

In discrete event simulation, a system is viewed as a collection of entities that interact with one another causing dependent performance variables to change discretely at specified points in simulated time referred to as event times. This kind of simulation model has been used to address design issues as well as operational issues of manufacturing systems. Some of the applications of simulation modeling to manufacturing systems can be found in Stecke and Solberg [1981] and ElMaraghy [1982].

Recent advances in the field of artificial intelligence and expert systems have opened up the possibility of a new generation of simulation systems, namely, knowledge-based simulation systems. Since most of the literature related to this area will be reviewed in the following section, expert systems in modeling, it will not be repeated here.

Even though simulation has been the most widely used modeling technique of all the techniques listed above [Pritsker 1986], it has its limitations and problems. One of these problems, maybe the most important one, mentioned by researchers [Narayanan et al. 1993] is the difficulty of using the abstractions of the simulation model to describe the system being analyzed. Recently, there has been a growing interest in object-oriented programming in simulation modeling of manufacturing systems [Adiga and Glassey 1991; Mize et al. 1992; Narayanan, Bodner and Mitchell 1992; Ulgen, Thomasma and Mao 1989; Zeigler 1990]. There are several reasons why the OOP paradigm has tremendous appeal for manufacturing systems simulation.

- OOP provides the possibility of having a one-to-one mapping between objects in the manufacturing systems being modeled and their abstractions in the simulation model [Narayanan, Bodner and Mitchell 1992].
- The OOP paradigm facilitates modular design and software reusability of simulation models [Mize et al. 1992; Narayanan, Bodner and Mitchell 1992]. Features such as encapsulation, inheritance, and polymorphism, provided by OOP, when exploited fully, facilitate code reuse and programming efficiency. For building simulation models, the idea is to design reusable classes and store them in a software library or model base, thus facilitating rapid model development [Mize et al. 1992; Kamath et al. 1996].
- There is compatibility between the OOP paradigm and the discrete-event world view formalism [Zeigler 1987].
- OOP provides a natural environment for graphical user interface development [Ulgen, Thomasma and Mao 1989].

There are numerous instances of OOP-based simulation studies of manufacturing applications. Here, we will focus on some of the well known large-scale persistent research efforts aimed at developing generic OOP-based architectures for the modeling and evaluation of manufacturing systems.

BLOCS [Adiga 1989; Glassey and Adiga 1989 and 1990; Adiga and Glassey 1991; Adiga, Petrakian and Shabe 1992] was the first manufacturing specific object-oriented simulation architecture. It was developed in Objective-C, and uses ICPac201, a software utilities library for interfaces. BLOCS exploits the software reusability feature of OOP. The design goal was to develop a library of reusable software modules to assemble special-purpose simulation models for manufacturing. The BLOCS library has been primarily applied to modeling semiconductor fabrication.

DEVS [Zeigler 1987; Zeigler 1990; Zeigler 1991] is both a methodology and a software implementation designed for autonomous systems modeling. It was developed using SCOOPS, a super set of PC-scheme on IBM PCs and TI Explorers. The constructs in DEVS are generic and not restricted to manufacturing systems simulation. DEVS explores the compatibility between the OOP paradigm and the discrete-event world view formalism and capitalizes on the reusability of OOP to develop simulation models. The focus of DEVS is on developing hierarchical and reusable model bases, in combining simulation modeling and artificial intelligence techniques, and exploring distributed simulation models and architectures. DEVS applications in manufacturing have been in autonomous robot simulation and in developing a printed circuit board test architecture.

The modeling approach developed by the Laval research group [Mayrand, LeFrancois and Montreuil 1993; Montreuil, LeFrancois and Harvey 1993] was implemented in Smalltalk-80. The major applications have been in modeling rolling mills. Simulation modeling and artificial intelligence techniques are integrated in this research. The focus of the Laval research is on developing abstractions for manufacturing decision making.

OOSIM [Narayanan et al. 1993; Govindaraj et al. 1993] was developed in C++ with a graphical interface implemented in Motif widgets and the X windowing system under the UNIX operating system. OOSIM uses the natural mapping between objects in the manufacturing world and the software objects provided by OOP to develop real-time, interactive simulations for discrete manufacturing. The primary applications of the research have been in IC fabrication and modeling printed circuit card assembly machines and lines.

SmartSim [Thomasma, Mao and Ulgen 1988; Ulgen, Thomasma and Mao 1989; Ulgen and Thomasma 1990] was developed in Smalltalk and has been applied to simulate robotic machine cells. The focus of the research is to utilize the natural mapping between real world entities and the software in OOP to develop an icon-based simulation program generator for manufacturing systems.

Research underway within Oklahoma State University's (OSU) Center for Computer Integrated Manufacturing (CCIM) has as its primary goal the development of a new, comprehensive, cohesive, and tool independent modeling environment for aiding manufacturing engineers in the detailed design and rapid reconfiguration of discrete part manufacturing systems [Mize and Pratt 1991; Mize et al. 1992; Bhuskute et al. 1992; Duse et al. 1993; Pratt, Mize and Kamath 1993; Pratt et al. 1995; Kamath, Pratt and Mize 1995; Kamath et al. 1996; Delen, Pratt and Kamath 1996a and 1996b]. The prototype implementation has been developed in the VisualWorks 2.0 [ParcPlace 1994] development environment which is based on the Smalltalk-80 programming language. At present the user of the modeling environment can construct, save, and reuse a base model of a manufacturing system and analyze manufacturing system performance using one of a set of analysis tools (discrete-event simulation, queuing networks, or Petri nets) that are configured automatically by the environment using the base model.

According to the research team of the Center for CIM at OSU, a rapidly reconfigurable modeling environment of manufacturing system should consist of a collection of primitive elements which are of three fundamental types [Pratt et al. 1995]:

- *physical elements* - work stations, assembly stations, material handlers, storage devices, etc.;
- *control elements* - system wide controls (e.g., push vs. pull philosophy), workcenter controllers (e.g., order release algorithms), and low level controllers (e.g., machine sequencing rules);
- *information elements* - data packets (e.g., bills of material), current status values (e.g., queue lengths, machine conditions), and system performance measures.

The modeling framework resulting from the separate modeling constructs for physical, information, and control elements coupled with OOP concepts forms the basis for a library of reusable objects for modeling discrete part manufacturing systems.

Literature Review of Expert Systems in Modeling

Because development of an expert system is both development of a software system and a type of modeling, implementation of such system exhibits properties of both information systems and operations research [Duchessi and O'Keefe 1995]. An optimum convergence of both disciplines is an essential requirement in creating successful applications. Martin, Subramanian and Yaverbaum [1996] present an explanatory investigation of benefits from expert systems whereas Duchessi and O'Keefe [1995] summarize a study of the factors that lead to successful (or conversely less successful) expert system applications. In the following part of this section, selected applications of expert systems (from recent literature) to modeling in general and simulation modeling in particular will be reviewed.

Applications of Expert Systems Specific to Simulation Modeling

Although Artificial Intelligence/Expert Systems (AI/ES) and simulation are two disciplines which matured independently, they have developed a common domain; that of problem solving. The problem solving paradigm in simulation modeling is mainly a search; thus, it parallels the problem solving paradigm in AI/ES. Table I shows some of these similarities [Rolston 1988].

TABLE I. Similarities Between AI and Simulation Modeling

Artificial Intelligence/Expert Systems	**Simulation Modeling**
• Define a problem environment as a collection of states	• Define a system in terms of objects and the objects' characteristics
• Define start states within the space to represent initial problem conditions	• Establish a set of input parameters as the initial conditions of experiment
• Define goal states that lead to acceptable solutions	• Define the state space for each output variable so as to satisfy objectives
• Define a set of operators to guide the changes from one state to another	• Build a digital model of the system

Joseph [1989] presents a rule-based expert system advisor which aims to reduce the up-front time spent prior to conducting a simulation study. This up-front study, called a pre-simulation study, if planned and conducted properly, can greatly reduce delays. Time spent during this phase of the study is typically devoted to gaining a good understanding of the issues with respect to the objectives, constraints and interactions, by considering various factors involved in a manufacturing system. Also, in most cases, the individual responsible for the actual simulation model development has to extract information from members of the team familiar with various design aspects of the system. A prototype implementation of this system has been developed in the TI PC-PLUS expert system programming environment.

Montan and Reddy [1989] outline an approach which interprets a goal for an existing base model, instruments the model based on the goal, designs one or more scenarios of the model, executes each scenario using an existing discrete event simulation engine, and evaluates the results. Once a scenario is found which satisfies the user's goals, the changes from the base model are reported to the user. According to Montan and Reddy [1989], there are four main tasks a simulation expert performs which can be emulated by an automated simulation system with expert systems. The first task is designing an appropriate simulation model which would reveal the most about the system of interest. After the model is constructed it needs to be instrumented in such a way that any information which will be needed to properly evaluate the model is recorded while the model is running. Once the model is executed, the results are evaluated by comparing them with the goals for the system. Finally, there must be some way to detect relationships between parameters of the model and the outputs of the model.

In the past, the solution to the simulation of manufacturing systems was to first design and construct the model, second verify and validate the model, and third analyze the results. In an ideal expert simulation system, the system interprets and understands the user's request and then determines what is needed in terms of data input, techniques to process the information and the type of information to be output [Shannon, Mayer and Adelsberger 1985]. Ford and Schroer [1987] present a system that couples an expert system with a commercial simulation language for simulating an electronics manufacturing plant. The whole system can be divided into four sub-modules: the transformer and the understander, the simulation writer, the simulation analyzer, and the simulation language. The goals of such a system are to develop a simulation capability for an electronics facility, provide a natural language interface so the decision maker will not have to learn the simulation language, and embed in the system an expert system to assist the decision maker.

ROSS [Klahr, Faught and Martin 1980] is a rule oriented simulation system developed by the RAND corporation. It is an interactive system implemented using LISP. ROSS was developed specifically for war gaming. Real world systems are modeled as objects. Messages are passed between objects, and IF-THEN rules describe the behavior of the objects. The user may halt the simulation at any time, modify the model, and continue the simulation. KBS [Fox and Reddy 1982] is a knowledge-based simulation system developed at Carnegie-Mellon. Like ROSS, it incorporates object oriented programming to describe the real world. Unlike ROSS, it allows goals describing the performance criteria of model components to be attached to objects, and it informs the user whether the goals were met.

Centeno and Standridge [1992] present a survey of the ways in which expert system tools and databases have been used to develop simulation modeling methodologies.

To the authors, simulation modeling is a powerful modeling technique that requires various kinds of knowledge and tools. Formal simulation modeling methodologies and efficient and smart tools require merging formalisms and techniques from artificial intelligence and databases.

Oren and Zeigler [1987] present research directions for exploring the role of artificial intelligence in modeling and simulation. The authors also emphasize the importance of quality assurance concepts and techniques that require advanced knowledge-processing paradigms (such as artificial intelligence, expert systems, or knowledge-based systems).

Applications of Expert Systems in Modeling

Even though a majority of papers written about expert system applications in modeling refer to simulation modeling, there are a few exceptions. Following is a set of paper reviews that do not directly relate to expert system applications in simulation modeling but expert system applications in modeling in general.

Several of the modeling approaches which follow are based in IDEF. IDEF is one of the available structured methods for modeling of manufacturing systems. It was developed by the US Air Force to describe manufacturing organizations in a structured graphical form. It provides users with a powerful means of analysis and development. A broader definition of IDEF can be found in Bravoco and Yadav [1985a and 1985b].

Lingzhi et al. [1996] present a conceptual framework called KBIDEF (knowledge-based IDEF) for the integration of the information model (IDEF1) with the functional model (IDEF0). On the basis of the IDEF0 and IDEF1 concepts, the principle for the integration of IDEF1 with IDEF0 is analyzed. In the proposed framework, the KMIDEF system consists of two databases: object-oriented IDEF0 and IDEF1 databases; three libraries: Entity Class Library, Relation Class Library, and Domain Relation Class Library; and two knowledge bases: Relation Analysis Knowledge Base and Domain Knowledge Base in addition to the main flow. The authors claim that such a KBIDEF will overcome the major problems of manual generation of the IDEF1 models. Thus, KBIDEF makes it easier, faster, and more convenient for users to build information models for CIM systems and help to further enhance the popularity of IDEF1 models in the industrial community and therefore accelerate the process of CIM design and implementation. A prototype KBIDEF for CIM information design has been built at the GINTIC Institute of Manufacturing Technologies, Singapore.

Ang, Luo, and Gay [1994] present a knowledge-based manufacturing modeling system for the automatic generation of models. The authors claim that the proposed system will not only greatly reduce the time of IDEF0 modeling but also eliminate the inconsistency problem of conventional IDEF0 modeling systems. The paper explains the knowledge-based approach and identifies the types of domain knowledge that are required for the construction of the knowledge-based manufacturing modeling system.

Standridge and Centeno [1991] present a production modeling system (PMS) that is implemented through the integration of multiple data analysis techniques such as databases and knowledge bases. The PMS is a computer-based modeling environment for developing and applying computer models to the production system design process. The PMS has several key characteristics including a single model representation of the system which enables multiple analysis types, adaptability of PMS functionality to the system under study, transparent information transfer between analysis, and gateways to external information sources. In addition, the PMS supports multiple user types including those that make decisions, those that use models to evaluate alternatives, those that construct models, and those that build the software that provides the functionality of the PMS.

According to Ruiz-Mier and Talavage [1987], the most advanced network modeling environments lack explicit concepts for the representation of complex behavior such as decision making. Artificial intelligence research, because of its emphasis on knowledge representation, provides several methodologies which can be successfully applied to the modeling of decision making behavior. The authors propose an approach to modeling complex behavior based on a hybrid methodology unifying the concepts of object-oriented programming, logic programming, and the discrete event approach to system modeling. They present SIMYON, an experimental network modeling environment, which provides explicit constructs for the representation of complex behavior of real-world systems. SIMYON is implemented by defining a library of logic objects in the object-oriented, logic programming environment CAYENE. These objects, which are analogous to the nodes of network modeling languages, are the main building blocks.

Kovacs et al. [1994] present an application of expert systems to assist in quality control, modeling and control of FMSs. Well known traditional simulation (Cinema/SIMAN) and networking (MAP) packages are combined with high performance expert systems (ALL-EX, G2) to provide the modeling environment.

Lirov et al. [1988] propose a functional approach for the design of expert systems that perform model generation. Using this approach, first the procedures that the system should perform are identified, then the representation issues are resolved. Next, the problem-solving paradigm is chosen and finally the system is implemented. Cooperation between the control systems is achieved through a goal hierarchy. The applications of this expert system are primarily in the areas of air combat games and navigation of mobile robots.

Literature Review of Expert Systems in Manufacturing

Manufacturing has been one of the most promising and successfully utilized fields of application for AI and expert systems. In fact, a contest, called Innovative Applications of Artificial Intelligence and Expert Systems sponsored by the American Association for Artificial Intelligence, indicated that ten out of sixteen winning applications involved some field of manufacturing. Due to the length and the level of relevance (secondary as opposed to direct) to this research activity, a selected set of literature reviews regarding expert system applications in manufacturing is presented in Appendix A.

Literature Review of Expert Systems in Modeling of Manufacturing

Hubner [1988] presents FLOPAS (Flow Production Analysis System) which is a knowledge-based software tool (written in COMMON LISP) for the analysis and performance evaluation of production flow lines. FLOPAS may be used as a stand-alone tool for flow line designers as well as part of a knowledge-based CIM controller which includes various expert systems. The analysis techniques used are Markov-chain analysis and knowledge-based discrete event simulation. The analysis module of FLOPAS allows the determination of workstation availability based on an application of Markov-chain theory. It offers an easy method to investigate the influence of the existence of buffers on the overall line throughput. A knowledge-based discrete event simulation (including a graphical monitor with explanation facilities for bottleneck determination) verifies the analysis results and gives further information about the influence of buffer sizes as well as other statistical results. Hubner's research is significant in terms of being one of the earliest attempts to incorporate the advantages of expert systems into the modeling of manufacturing systems.

Luong, Chan and Sessomboon [1994] describe the development and implementation of a system which integrates simulation and expert systems as a tool for FMS design. In their approach, knowledge and experience associated with FMS design are stored in a knowledge base which is used to interpret simulation outputs and suggest suitable modifications to a proposed FMS design in order to meet its manufacturing and financial objectives. The system is implemented using SIMAN and VP-EXPERT software tools, and runs on personal computers. A case study is also presented to illustrate the usefulness and potential of this system for FMS design.

Weinroth, Madey and Shah [1992] present a research effort that uses artificial intelligence to enhance simulation modeling for support of JIT management. A model is constructed to achieve optimal combinations of kanbans and daily work rates. Following experimentation with the model, an expert system is embedded in the simulation code to model the learning process. Verification of the expert system is achieved through observing that it produced the correct search strategies and solutions for the training set and a test set.

Tibbitts [1993] presents a rule-based simulator for a semiconductor manufacturing line. The simulator is written in a rule-based declarative style that uses a single-rule template to move thousands of product lots through various steps. Since line or product changes require only reading new data from a database, without reprogramming, this provides a modeling environments that is simple, flexible, and maintainable. The model is implemented in ECLPS (Enhanced Common Lisp Production System), which is also known as a knowledge-based or expert system language.

Summary of Literature Review

The use of expert systems in manufacturing is one of the most promising areas in the development of AI. Many applications of expert systems have been successfully implemented in the field of manufacturing. Nevertheless, when it comes to modeling of manufacturing systems (intersection IV in Figure 7) there has not been much work done so far. Only a few researchers have tried to address expert system applications in the context of modeling. This author believes that the modeling of manufacturing systems requires a vast amount of expert knowledge in every step of the modeling effort. Due to the inherent complex nature of the manufacturing systems, for a model to be created and used properly, a number of experts, who have spent considerable amount of time with the system, must be involved in the process. Most of the time this dependency on modeling experts leads to lack of usage for such models by decision makers. Expert systems are known to be a powerful tool to represent a specific domain expertise in the form of IF *condition* THEN *action* rules. The author also believes that such necessary expert knowledge can be incorporated into the modeling environment itself through the use of expert systems. Thus, this research aims to contribute to the body of knowledge in the field of building more advanced and user-friendly modeling environments for manufacturing systems through the application of expert systems to the problem definition and tool selection problems within an overall modeling process.

CHAPTER IV

BASICS OF EXPERT SYSTEMS

Introduction

This chapter provides a brief tutorial on expert systems for readers who have limited knowledge about expert systems. Readers who have a good understanding in this area may prefer to skip this chapter.

What exactly is an expert system? The British Computer Society's Specialist Group on the subject has proposed a formal definition [Forsyth 1984]:

> "An expert system is regarded as the embodiment within a computer of a knowledge-based component, from an expert skill, in such a form that the system can offer intelligent advice or take an intelligent decision about a processing function. A desirable additional characteristic, which many would consider fundamental, is the capability of the system, on demand, to justify its own line of reasoning in a manner directly intelligible to the inquirer. The style adopted to attain these characteristics is rule-based programming."

Another less formal definition is: "An expert system is a piece of software that encapsulates specialized knowledge about a particular domain of expertise and is capable of making intelligent decisions within that domain" [Durkin 1994].

The following features are useful in understanding expert systems [Forsyth 1984]:

- An expert system is limited to a specific domain of expertise.
- It can reason with uncertain data.
- It can explain its train of reasoning in a comprehensible way.
- Facts and inference mechanism are clearly separated.
- It is designed to grow incrementally.
- It is typically rule-based.
- It delivers advice as its output - not tables of figures, nor pretty video screens.

When to Use an Expert System?

The answer to the question of 'do I need an expert system?' depends on the kind of problem you want to solve. The following table presents a checklist of features that effect the suitability of an expert system approach [Forsyth 1987; Badiru 1992].

TABLE II. Checklist for Determining Whether an Expert System is Suitable

Suitable	**vs.**	**Unsuitable**
□ Diagnostic		□ Calculative
□ No established history		□ Magic formula exists
□ Human expertise scarce		□ Human experts are two-a-penny
□ Data is 'noisy'		□ Facts are known precisely
□ Requires a lot of data		□ Does not require a lot of data
□ Requires large working memory		□ Does not require large working memory

If the intended application falls more on the left side of the above table than on the right side, an expert system should seriously be considered.

Language Issues in Expert Systems

LISP and PROLOG were originally the primary languages for writing expert systems [Winston 1992]. Since the introduction of the object-oriented paradigm there has been a shift from conventional LISP and PROLOG to either their object-oriented versions or other object-oriented languages such as Smalltalk or C++. Forsyth proposes in his book "Expert Systems" [Forsyth 1987] that the best plan is to use the language you know on the machine you have. An alternative solution, maybe the most popular, is to purchase an expert system shell and use it to develop expert systems applications.

Components of an Expert System

There are four essential components of a complete expert system (Figure 9):

(1) the knowledge base,

(2) the working memory,

(3) the inference engine, and

(4) the user interface.

All four components are critical and while a knowledge-based system may lack one or two of them a true "expert system" should not. The following section briefly introduces each of these four components.

The Knowledge Base

The knowledge base contains the domain knowledge. A knowledge engineer obtains the knowledge from expert(s) and printed materials and codes it in the knowledge base using one of several knowledge representation techniques (discussed later in this chapter). One typical way of representing knowledge in an expert system is *production* *rules.* A rule is an IF/THEN structure that logically relates information contained in the IF part to information contained in the THEN part.

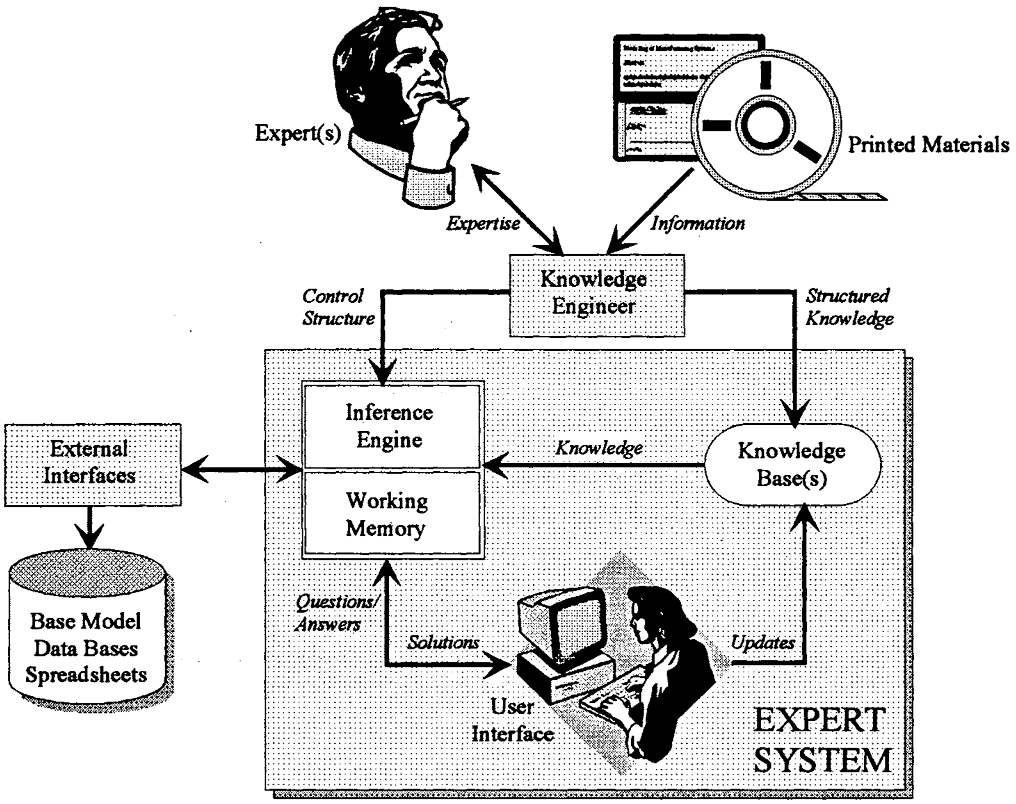


Figure 9. Components of a Complete Expert System

The Working Memory

The working memory contains the facts about a problem that are discovered during a consultation. During a consultation with an expert system, the user enters information on a current problem into the working memory. The system matches this information with knowledge contained in the knowledge base to infer new facts. The system then enters these new facts into the working memory and the matching process continues. The system usually reaches some conclusion that it also enters into the working memory. Thus, the working memory contains all the information about the problem that is either supplied by the user or inferred by the system.

The Inference Engine

The expert system models the process of human reasoning with a module known as the inference engine. The inference engine works with the facts contained in the working memory and the domain knowledge contained in the knowledge base to derive new information. It searches the rules (in a rule-based expert system) for a match between their premises and information contained in the working memory. When the inference engine finds a match, it adds the rule's conclusion to the working memory and continues to scan the rules looking for new matches.

Forward chaining and backward chaining are the two techniques used by inference engines for reasoning. Broadly speaking, forward chaining involves reasoning from data to hypotheses, while backward chaining attempts to find data to prove, or disprove, a hypothesis. Pure forward chaining leads to unfocused questioning in a dialog-mode system, whereas pure backward chaining tends to be rather relentless in its goal-directed questioning [Forsyth 1987]. Most successful systems use a mixture of both.

The User Interface

The interaction between an expert system and user is conducted in a natural language style that is highly interactive and follows closely the dialog exchanged between humans. To conduct this process in a manner that is acceptable to the user places special demands on expert system builders when designing the user interface.

A basic design requirement of the interface is to ask questions and feedback the recommendations in a manner that can easily be understood by the user. To obtain reliable information from the user, an expert system builder needs to pay particular attention to the design and presentation of questions. This may lead the builder to design the interfaces using menus, selection lists, graphics, and action buttons.

Knowledge Acquisition

Knowledge acquisition is the collection and analysis of information from one or more domain experts and other sources, such as documents, journals, and books, that form the basis of a functioning knowledge-base [Greenwell 1988]. The knowledge acquisition process is depicted in Figure 10.

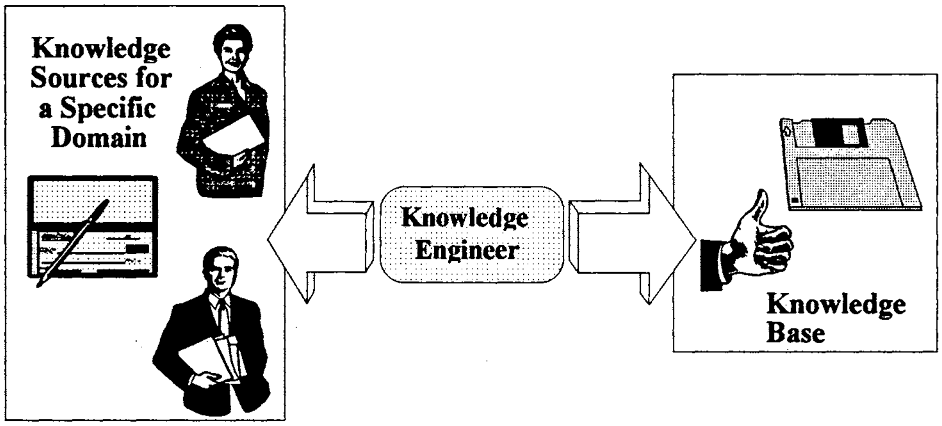


Figure 10. Knowledge Acquisition Process

Acquiring knowledge from the domain expert(s) is distinguished from the more general term knowledge acquisition and is called *knowledge elicitation.* The recommendations provided by an expert system are only as good as the knowledge contained in the knowledge base. In essence, the analogy commonly used for conventional programming "garbage in, garbage out" also applies here.

Most expert system developers have come to realize that knowledge acquisition is not a simple task. In fact, they have found the task to be the most difficult part of designing the expert system. Duda and Shortliffe [1983] express their concern about this issue by stating "... The identification and encoding of knowledge is one of the most complex and arduous tasks encountered in the construction of an expert system... Thus the process of building a knowledge base has usually required a time-consuming collaboration between a domain expert and an AI researcher...". Hayes-Roth, Waterman and Lenatt [1983] used the term *bottleneck* to describe the difficulty in knowledge acquisition "... Knowledge acquisition is a bottleneck in the construction of expert systems...".

The characteristics of knowledge are often dependent on the source of the knowledge. According to Bȧdiru [1992], typical sources of knowledge are:

- direct consultation with human experts;
- printed materials such as books and journals;
- direct task observation;
- direct task performance; and
- third-party account of expert procedures.

Of all the available sources listed above, direct consultation with human experts poses the greatest difficulty but offers the highest level of reliability. Books and other printed materials are particularly suitable as stable sources of knowledge.

Knowledge Acquisition from Multiple Experts

Many existing expert systems could be described as "single-expert" systems. That is, their knowledge bases primarily reflect input from a single expert combined with other knowledge sources. Some circumstances preclude the exclusive use of a single domain expert. For instance, a group of experts is required for solving problems where the domain is so broad that no one individual's expertise spans the entire domain.

Having several experts with diversified expertise is viewed as a significant resource for expert systems [Medsker, Tan and Turban 1995] (see Table III).

TABLE III. Advantages of Working with Multiple Experts

⇒ Better understanding of knowledge domain
⇒ Improved knowledge base. In terms of ♦ validity ♦ consistency ♦ completeness ♦ accuracy ♦ relevancy
⇒ Better productivity
⇒ Easier identification of incorrect results
⇒ Ability to address broader domains
⇒ Ability to deal with more complex problems

The acquired knowledge has more validity and a better understanding of the domain knowledge is obtained from consensus across experts. The understanding of the domain knowledge is enhanced through group discussion and clarification, and group productivity is quantitatively superior to that of an average individual. Groups can usually recognize and reject incorrect solutions/suggestions to a higher degree than individuals. Addressing broader domains and/or complex problems naturally suggests multiple experts.

Despite the significant benefits of using multiple experts, their use is sometimes avoided due to implementation difficulties [Medsker, Tan and Turban 1995] (see Table IV). Acquiring knowledge from multiple experts can be a very difficult task because agreement may not be obtained among the experts about the knowledge domain. Experts use different mental models of the task domain. Also, experts may be geographically dispersed. The performance of groups, in terms of quality of decisions arrived at and the time it takes, may be inferior to that of a competent individual, especially if the decision making process is not properly managed. Some experts may resent working in a group, and many feel that it is a waste of time or that a few group members may dominate the process. Experts may have disrespect for other experts because they use different methodologies, or mental models, or because their own track record is superior.

TABLE IV. Disadvantages of Working with Multiple Experts

⇒ Experts may not agree and may have different mental models
⇒ Experts may be geographically dispersed
⇒ Group reasoning difficult to see
⇒ Lower productivity if process is flawed
⇒ Negative view of committee work
⇒ Socializing and politics
⇒ Personality and professional conflicts
⇒ Domination by one or a few members

The most common methods for facilitating knowledge acquisition from multiple experts are given in Table V [Medsker, Tan and Turban 1995].

TABLE V. A Summary List of Techniques Commonly Used in Knowledge Acquisition from Multiple Experts

Technique	Comments
Brainstorming	Encourages idea generation and expansion beyond expert's usual approach; delays problem solving until innovative ideas are identified.
Delphi method	Structured sharing for gaining consensus; for assimilation of knowledge, opinions, communication, and resolution of diverse ideas.
Consensus decision making	Uses conventional group dynamics techniques to enhance the knowledge acquisition process.
Nominal group technique	Organizes experts as a nominal group functioning independently; may include computer support.
Analytical approach	Useful when numerical values (weights, probabilities) are used.
Computer facilitated collaborative work	Enhancement via computer technology (e.g., Group Decision Support Systems [GDSS]); useful for displaying information and ideas for other experts; display of data base and progress on knowledge base development.
Distributed artificial intelligence (DAI)	Uses blackboard or other technique to represent and store the ideas of the different experts; allows several knowledge bases with competing ideas; provides mechanism for organizing, manipulating, and reconciling multiple expertise and beliefs.

Knowledge Representation

Before presenting the details of knowledge representation, it is necessary to define the word knowledge. Knowledge is an abstract term that attempts to capture an individual's understanding of a given subject area [Durkin 1994]. It can be classified into different types: *procedural knowledge* (describes how a problem is solved), *declarative knowledge* (describes what is known about a problem), *meta-knowledge* (describes knowledge about knowledge), *heuristic knowledge* (describes a rule-of-thumb that guides the reasoning process), and *structural knowledge* (describes knowledge structures).

For an expert system to be effective, knowledge acquired from the knowledge sources should be properly represented in the expert system's knowledge base. Different knowledge representation techniques are available. Each representation technique emphasizes certain information about a problem while ignoring other information. Each technique also has advantages and disadvantages for capturing efficiently the different types of knowledge mentioned in the previous paragraph. Choosing the correct representation for a given application produces a structure that supports effective problem solving. The following is a brief review of the most common representation techniques used in the development of an expert system.

Object-Attribute-Value Triplets

In the Object-Attribute-Value (O-A-V) triplets scheme, objects may be physical entities such as door or a transistor, or they may be conceptual entities such as a logic gate, a bank loan, or a sales episode. Attributes are general characteristics or properties associated with objects. Size, shape, and color are typical attributes for physical objects. Interest rate is an attribute for a bank loan, setting may be an attribute for a sales episode. The final member of the triplet is the value of an attribute. The value specifies the specific nature of an attribute in a particular situation. An apple's color may be red, for example, or the interest rate for a bank loan may be 12 percent.

For most problems addressed by an expert system, objects have more than one important feature. In these instances, multiple attributes are defined for the object, with corresponding attribute values. O-A-V representation is commonly used within the framework of other representation techniques. For example, MYCIN (a classic expert system developed to diagnose infectious blood diseases and help a doctor recommend the appropriate treatment) uses this scheme with rules.

Rules

User-supplied facts are important to the operation of an expert system. They allow the system to understand the current state of the world. However, the system must have additional knowledge that allows it to work intelligently with these facts to solve a given problem. One knowledge structure commonly used in the design of an expert system that provides this additional knowledge is a rule. A rule is a form of procedural knowledge. It associates given information to some action. This action may be the assertion of new information or some procedure to perform. In this sense, a rule describes how to solve a problem.

The rules structure logically connects one or more antecedents (a.k.a. premises) contained in the IF part, to one or more consequences (a.k.a. conclusions) contained in the THEN part. For example: “IF the car’s color is red THEN she likes the car.”

Semantic Networks

One of the earliest attempts in AI to represent knowledge in a computer relied on a semantic network. A semantic network provides a graphical view of a problem’s important objects, properties and relationships. It contains nodes and arcs that connect the nodes. The nodes can represent objects, object properties or property values. The arcs represent the relationship between the nodes. Both the nodes and arcs have labels that clearly describe the objects represented and their relationships. A node for example might have the name "Bird" or "Jack." Arcs are commonly labeled with terms such as "IS-A," "HAS," etc., that clearly define the relationships between connected nodes. Figure 11 illustrates a simple semantic net which describes a bird.

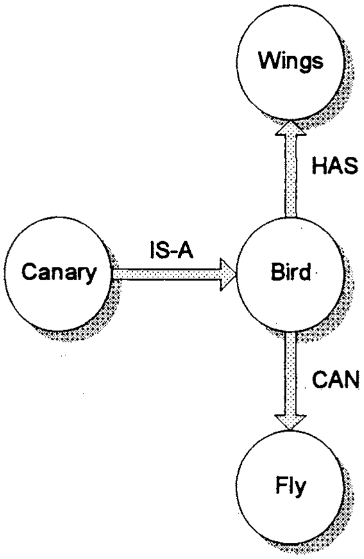


Figure 11. Semantic Network of Bird

Frames

A natural extension of the semantic network is a frame (or often called schema). A frame is a unit that contains typical knowledge about some concept or object, and includes both declarative and procedural knowledge. For example, the frame bird might include knowledge that it has wings and legs, and how it hunts for food. A frame holds stereotypical information about a concept that can be applied to a specific situation for study. Frames are used for representing declarative knowledge. As briefly defined in the beginning of this section, declarative knowledge is knowledge that cannot be immediately executed but can be stored and retrieved as needed to provide information for solving a problem.

Frames have slots, like attributes, which store values. Slots may also contain default values, pointers to other frames, sets or rules, or procedures by which values may be obtained. The inclusion of these additional features make frames different from O-A-V triplets. From one perspective, frames allow for richer representations of knowledge. From another, they are more complex and more difficult to develop than simpler O-A-V triplets.

Logic

The oldest form of knowledge representation in a computer is logic. Over the years, several logic representation techniques have been suggested and studied. The ones most often linked with intelligent systems have been propositional logic and predicate logic. Propositional logic represents and reasons with propositions; statements that are either true or false. Predicate logic, also called predicate calculus, is an extension of propositional logic that provides a finer representation of the knowledge. Both techniques use symbols to represent knowledge and operators applied to the symbols to produce logical reasoning. They offer a well-founded approach to knowledge representation and reasoning. Though expert system designers rarely use a classical logical knowledge representation approach, it forms the basis upon which most AI programming languages and shells are built. PROLOG for example is one of the key AI programming languages and is based on the predicate calculus. Predicate calculus is like the assembly language of knowledge representation. An understanding of it provides insight into higher level representation techniques.

Hybrids

Each knowledge representation technique has its advantages and disadvantages. For example, rules are especially useful for representing procedural knowledge (methods for accomplishing goals). Semantic networks are good for representing relationships among objects. Predicate logic provides a means for explicitly expressing different types of knowledge. Early expert systems tended to use one technique or another exclusively. More recently the tendency has been to combine different representation techniques, so as to take advantage of the capabilities of each technique within the context of the prevailing problem [Badiru 1992]. An expert system, for example, might use rules to define procedures for discovering attributes of objects, semantic networks to define the relationships among the objects referenced in the rules, and frames to describe the objects' typical attributes.

Dealing with Uncertainty

One problem faced by all experts, whether human or inhuman, is that nothing in life is certain except death, and even that may arrive unexpectedly. Real-life problem solving demands an acceptance of uncertainty, in order to minimize the difficulties it poses [Durkin 1994]. Various schemes have been tried, some quite successfully, which allow the use of fragmentary and uncertain information to reach an estimate of the truth. In this section, three of the most common schemes, namely Bayes' Theorem, certainty factors, and fuzzy logic, will be summarized.

Bayesian Approach to Inexact Reasoning

Since the roots of Bayes' Theorem come from probability theory, it is necessary to review some of the important concepts in probability theory before introducing Bayesian theory. Probability theory proposes the existence of a number *P(E)* called the probability, which is the likelihood of some event *E* occurring from a random experiment. That is, if we perform some experiment a large number of times, then we can be almost certain that the relative frequency of the event *E* is approximately equal to *P(E)*. The set of all possible outcomes of an experiment is called the sample space, and is denoted as *S*. Each possible outcome of the experiment is an event *E* and is part of the sample space.

One of the most important concepts to revisit before getting into Bayesian theory is conditional probability. The probability of an event *A* occurring, given that an event *B* has already occurred, is called the conditional probability and is given as

$$P(\mathrm{A}|\mathrm{B}) = P(\mathrm{A}\cap\mathrm{B}) \,/\, P(\mathrm{B})$$

where:

$P(\mathrm{A}|\mathrm{B})$: Occurrence probability of event *A*, given that event *B* has occurred

$P(\mathrm{A}\cap\mathrm{B})$: Occurrence probability of event *A* and event *B* at the same time

$P(\mathrm{B})$: Occurrence probability of event *B*

The conditional probability $P(\mathrm{A}|\mathrm{B})$ permits us to obtain the probability of event *A* given that event *B* has occurred. In many problems we are concerned with the reverse situation: What is the probability of an earlier event given that some later one has occurred? This is often referred to as the posterior probability. A typical problem where this situation would be present is in machine diagnostics. For example, we can observe the later events in terms of machine fault symptoms, but diagnosis is concerned with the earlier events that caused the fault symptoms. In general, the conditional probability is forward in time, while the posterior probability is backward in time.

The solution to this problem was found by 18th-century British mathematician Thomas Bayes, and is known as the Bayesian Theorem. The formal definition of this theorem provides the probability of the truth of some hypothesis *H* given some evidence *E*, and is presented as

$$P(\mathrm{H}|\mathrm{E}) = [P(\mathrm{H}) * P(\mathrm{E}|\mathrm{H})] / P(\mathrm{E})$$

where:

$P(\mathrm{H}|\mathrm{E})$: Probability that *H* is true given evidence *E*

$P(\mathrm{H})$: Probability that *H* is true

$P(\mathrm{E}|\mathrm{H})$: Probability of observing evidence *E* when *H* is true

$P(\mathrm{E})$: Probability of *E*

Bayes' Theorem relies on knowing prior probabilities of an event, which can be used to interpret the present situation. If a rich source of prior statistical information is available, then we can determine the likelihood of some hypothesis being true, given some evidence about the problem. The way Bayes' Theorem comes into play in the design of expert systems is based on the structure of a typical rule "IF *E* THEN *H*". The equation given above is used to provide the probability of the hypothesis *H* given the evidence *E*.

Certainty Theory

A popular alternative to probability theory for inexact reasoning in expert systems is certainty theory [Shortliffe 1976]. This theory grew out of the work on MYCIN and relies on defining judgmental measures of belief rather than adhering to strict probability estimates. This work led to the development of the certainty model, which offers practical techniques for performing inexact reasoning in many expert system applications. What follows is a brief review of this theory.

In certainty theory, the degree of belief or disbelief is to be represented by a number, commonly called the certainty factor (CF), ranging from -1 (definitely false) to +1 (definitely true). A positive value indicates a degree of belief, while a negative value indicates a degree of disbelief. Certainty factors are used to represent uncertain rules

IF E_1 AND E_2 ... THEN H WITH CF = CF_i

where E_i represent the available evidence, H the conclusion, and CF_i the level of belief in H given the evidence.

For rules with more than one premise, the certainty factor for the rule's conclusion is calculated as follows:

1. For conjunctive rules, the approach used in the certainty model is as follows:

   IF E_1 AND E_2 AND ... THEN H WITH CF = CF_i

   $\mathrm{CF}(H, E_1 \text{ AND } E_2 \text{ AND } \ldots) = \min \{\mathrm{CF}(E_i)\} * \mathrm{CF}_i$

2. For disjunctive rules, the approach used in the certainty model is as follows:

   IF E_1 OR E_2 OR ... THEN H WITH CF = CF_i

   $\mathrm{CF}(H, E_1 \text{ OR } E_2 \text{ OR } \ldots) = \max \{\mathrm{CF}(E_i|\mathrm{E}`)\} * \mathrm{CF}_i$

3. For rules that conclude to the same hypothesis, the combined degree of belief can be determined as follows:

$$\begin{aligned} CF_{COMBINE}(CF_1, CF_2) &= CF_1 + CF_2 * (1 - CF_1) && \text{both} > 0 \\ &= (CF_1 + CF_2) / 1 - \min\{ |CF_1|, |CF_2|\} && \text{one} < 0 \\ &= CF_1 + CF_2 * (1 + CF_1) && \text{both} < 0 \end{aligned}$$

Here, CF_1 represents the confidence in *H* established by one rule and CF_2 represents the confidence in *H* established by another rule.

Fuzzy Logic

Another approach to inexact reasoning is the concept of fuzzy logic. Fuzzy logic was introduce by Zadeh and Fukanaka [1975]. The objective of fuzzy logic is to represent ambiguous terms commonly found in natural languages. For example, consider the statement "The person is tall." This statement is ambiguous because of the use of the word "tall." Humans have little difficulty in interpreting and reasoning with ambiguous terms, however, computers need some help. Such help is provided by the field known as fuzzy logic. Fuzzy logic provides methods for both representing and reasoning with ambiguous terms. Ambiguous terms are represented in fuzzy sets, which capture quantitatively the human interpretation of the terms. For example, Figure 12 shows fuzzy sets that represent different adjectives that describe a person's height [adapted from Durkin 1994]. This figure shows three fuzzy sets mapping the domain of height into a number called the membership value. The membership value is a number between 0 and 1 that reflects the level of belief that a given height belongs to a given fuzzy set. For example, an individual of a height of 5.5 feet would be said to a member of "medium" persons with a membership value of 1, and at the same time, a member of "short" and "tall" persons with a value of 0.30.

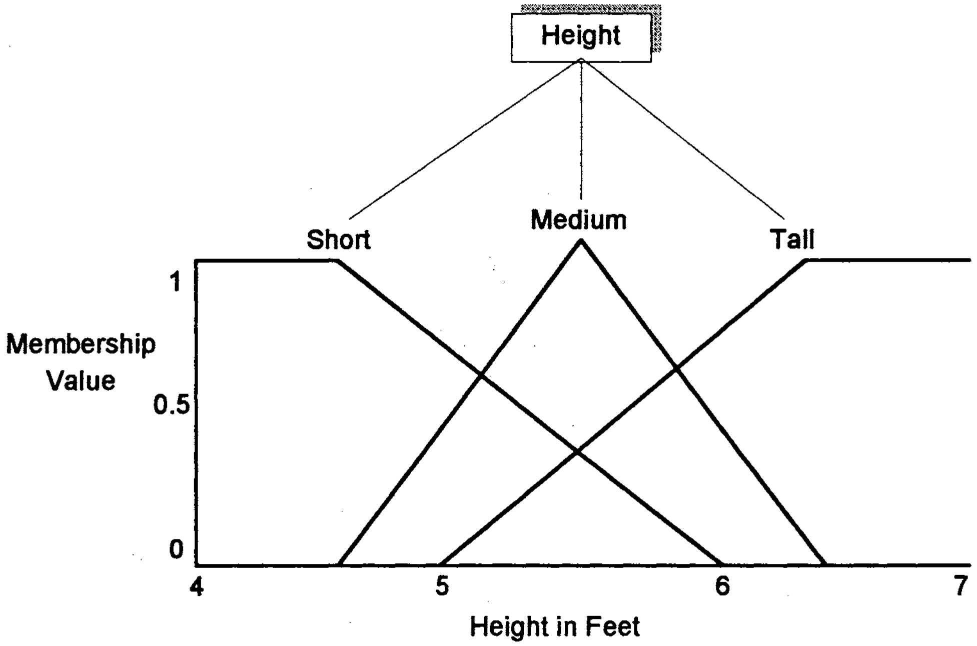


Figure 12. Fuzzy Sets on Height

In addition to fuzzy sets, fuzzy logic permits one to write fuzzy rules. A fuzzy rule contains fuzzy sets in both its IF and THEN parts. Consider the following example:

IF The person's height is *tall*

THEN The person's weight is *heavy*

By first forming a belief of membership value in the rule's premise, this rule can infer the corresponding value in the fuzzy set heavy defined on the domain weight. In effect, a fuzzy rule maps fuzzy sets to fuzzy sets.

In this chapter a brief tutorial on expert systems is provided for readers who have limited knowledge about expert systems. The concepts and the techniques given in this chapter will be used as references in the discussion presented in Chapter VI for justifying the selected research methodology.

CHAPTER V

RESEARCH STATEMENT

Research Goal

There are two major categories of research. One is the "classical" research which is driven by research hypotheses, and the other is the "developmental" research which is driven by a research goal and research objectives. This research can best be classified as "developmental" research.

The overall goal of this research is to develop a methodology that utilizes expert systems in the experimental frame module of an advanced modeling environment for manufacturing systems in order to provide timely assistance to the problem definition and tool selection decision making processes. Such an experimental frame expert system will include (1) a problem definition part which will be capable of deriving a structured problem from a set of symptoms, and (2) a tool selection part which will be able to recommend the most suitable analysis tools to address the defined problem.

Research Objectives

To accomplish the research goal, the following research objectives are addressed:

OBJECTIVE 1 - Acquire the Necessary Knowledge for the Problem Domain

Develop a conceptual framework that derives a structured problem from a set of symptoms. This objective can be partitioned into the following sub-objectives:

1. Decide on the definition of "who is an expert?" in this domain. Additionally there is a need to determine a set of potential experts that satisfy this definition.
2. Acquire the knowledge from the experts by utilizing appropriate knowledge acquisition techniques. Here, experts are expected to provide problems and symptoms that can be found within the domain of discrete part manufacturing systems.
3. Structure the acquired knowledge. Using multiple experts and other knowledge sources in the process of knowledge acquisition is expected to result in somewhat conflicting facts and certainty factors. Resolving the conflicts and establishing the relationships between the symptoms and the problems is the theme of this sub-objective.

OBJECTIVE 2 - Development of the Problem Domain Knowledge Base

Develop a problem definition knowledge base using the framework established in Objective 1. Such a knowledge base is designed not only to accommodate the results of the Objective 1, but also to interact with the decision maker and the base model in the process of searching for the best suited structured problem statement for the given set of symptoms. This interaction process is depicted in Figure 13.

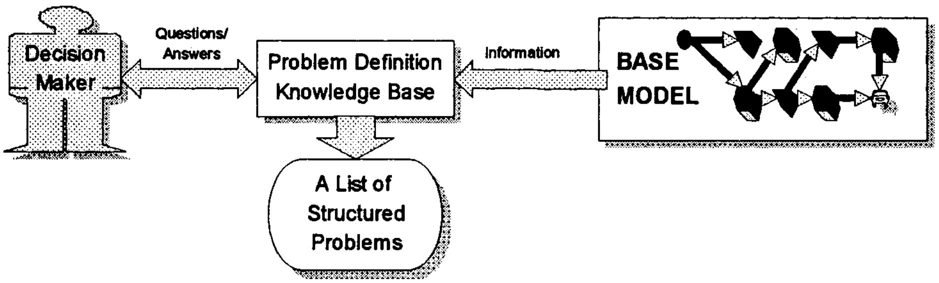

Decision Maker
Questions/ Answers
Problem Definition Knowledge Base
Information
BASE MODEL
A List of Structured Problems

Figure 13. Interaction Process of Problem Definition Knowledge Base

The interaction between the decision maker and the knowledge base is in the form of questions and answers whereas the interaction between the base model and the knowledge base is in the form of information flow.

OBJECTIVE 3 - Acquire the Necessary Knowledge for the Tool Domain

Develop a conceptual framework that specifies analysis tools with their capabilities and limitations. This objective can be partitioned into the following sub-objectives:

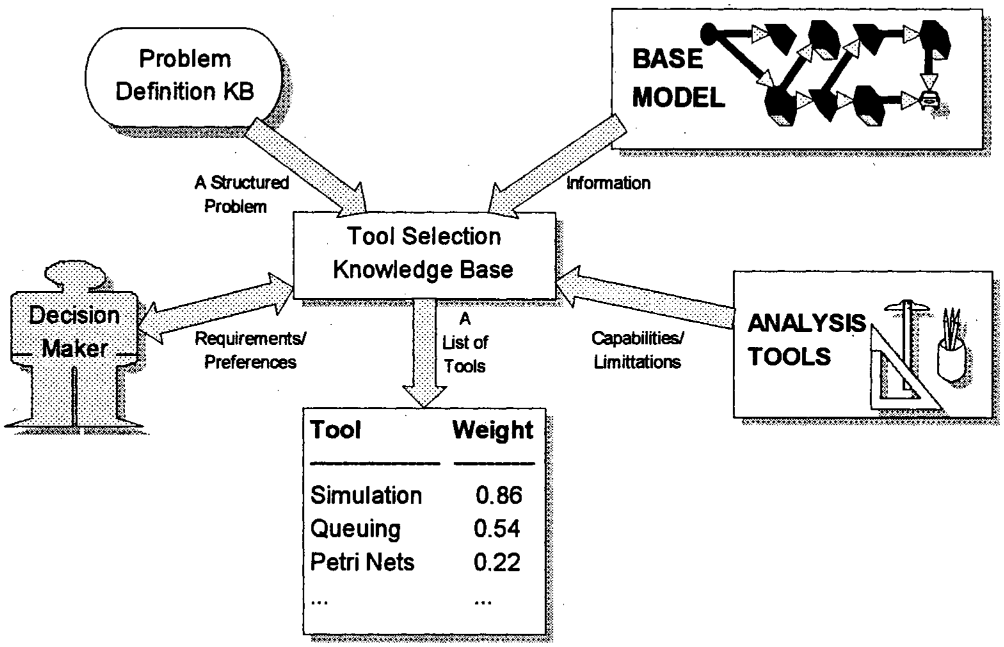
1. Decide on the tool set. To facilitate the collection of expert knowledge in a timely and concise manner, the set of tools to be used in this knowledge acquisition process, will be limited to a manageable size.
2. Decide on the definition of "who is an expert?" with regard to the selected tool set. Additionally there is a need to determine a set of potential experts that satisfy this definition.
3. Acquire the knowledge from the experts by utilizing knowledge acquisition techniques. Here, experts are expected to provide expertise relative to the applicability of tools to various problems.
4. Structure the acquired knowledge. Using multiple experts and other knowledge sources in the process of knowledge acquisition is expected to result in conflicting facts and certainty factors. Resolving the conflicts and establishing the relationships between the problems and the tools is the theme of this sub-objective.

OBJECTIVE 4 - Development of the Tool Domain Knowledge Base

Develop the tool selection knowledge base by utilizing the framework determined in Objective 3. The tool selection knowledge base is to be designed in a way that leads to a list of recommended analysis tools from a given set of tools based on not only the structured problem from Objective 2, but also other inputs that come from the user and the base model. The input set for the tool selection knowledge base is as follows:

1. A structured problem (the output obtained in Objective 2);
2. The base model (the abstract representation of the manufacturing system);
3. Tool domain knowledge (the framework established in the Objective 3);
4. User requirements and preferences.

The output of this knowledge base will be a set of recommended analysis tools ranked in a most preferred to less preferred priority list. This process is illustrated in Figure 14.



Tool	Weight
Simulation	0.86
Queuing	0.54
Petri Nets	0.22
...	...

Figure 14. Interaction Process of the Tool Selection Knowledge Base

OBJECTIVE 5 - Proof of Concept Implementation of EFES

A prototype version of the results obtained from Objective 1 through Objective 4 will be implemented in a way that demonstrates proof of concept. This implementation will be accomplished using the modeling environment under development in the Center for CIM at OSU. In order to incorporate the intelligent advisor (EFES) into the environment, the existing structure will be modified from both the interfaces and class hierarchy standpoints. Verification and validation processes, for this implementation, will also be conducted.

OBJECTIVE 6 - Further Research

Conceptualize a framework for carrying out further research in order to expand the capabilities of the prototype implementation used to demonstrate proof of concept. After the completion of this research effort, much work will yet need to be done to generalize the prototype implementation into a more robust environment. Knowledge gained from this research effort is expected to be used as a foundation for additional research. There are additional potential areas for expert systems to improve a modeling environment for manufacturing systems outside the experimental frame module. Each of these areas can be thought of as an opportunity for further research.

Research Assumptions and Limitations

The following assumptions and limitations are pertinent to this research.

- The research will be limited to the modeling of job shop type discrete part manufacturing systems. Flow lines and continuous manufacturing systems are outside the scope of this research. Non-manufacturing applications are also outside the scope of this research.

- The problem space for the intelligent advisor will be limited to the ones that are collected from the experts. Since the purpose of this research is proof of concept and not to develop a comprehensive problem definition knowledge base for discrete part manufacturing systems, the number of problems considered in the knowledge base will be limited to a manageable size (in the neighborhood of 50).
- The research will make use of the modeling environment under development in the Center for Computer Integrated Manufacturing (CCIM) at Oklahoma State University (OSU) as a starting point for the proof of concept implementation.
- The knowledge bases for the proof of concept implementation will be created using the expert system shell HUMBLE [XEROX 1994], which is written in Smalltalk-80 [Goldberg and Robson 1989]. The primary reason for choosing HUMBLE, as the expert system shell in the prototype implementation of this research, is its platform compatibility with the advanced modeling environment. Both are written in Smalltalk-80. A brief summary of HUMBLE is presented in Appendix B.
- The set of analysis tools will be limited to a discrete event simulation, a queuing network analyzer, a Petri net tool, queueing followed by simulation, and search based optimization using simulation. Even though the modeling environment, under development in the Center for CIM at OSU, does not currently support the last two tools, the intelligent advisor can still consider the limitations and capabilities of those tools and includes them in the recommendations.

Research Contributions

The major contribution anticipated from this research is the conceptualization and validation of a methodology that utilizes expert systems in the experimental frame module of a modeling environment for manufacturing systems in order to provide timely assistance to the problem definition and tool selection decision making processes. For decision makers, having a modeling environment that facilitates easy and timely model construction/reconstruction capabilities that allow non-modeling specialists to access such models is a tremendous advantage for making improvement decisions with regard to the manufacturing system. Such a modeling methodology is potentially expected to eliminate the disadvantages (i.e., reusability and accessibility) found in traditional modeling methodologies.

Other contributions anticipated from this research include:

√ A list of problems, symptoms, and what-if questions collected from the experts. A list of this sort has not been previously published to the author's knowledge.

√ Mappings of symptom(s) to the problem(s). Mappings of this sort has not been previously published to the author's knowledge.

√ Capabilities and limitations of the most commonly used tools with respect to the problems found in discrete part manufacturing systems.

√ Mapping of tool(s) to the problem(s). The conflicts discovered in the data collection phase of this research indicates that further efforts to clarify the issues in this area warrants additional consideration.

CHAPTER VI

RESEARCH METHODOLOGY

Introduction

This chapter starts by examining and justifying the techniques used for knowledge acquisition, knowledge representation, and dealing with uncertain data. Next, issues related to validation and verification are summarized. The last section of this chapter lists the phases in which the research was performed to achieve the goal and the objectives outlined in Chapter V.

Selection of the Knowledge Engineering Techniques

Selection of the Knowledge Acquisition Technique

Knowledge acquisition, a vital task when developing an expert system, is the process of extracting, structuring, and organizing knowledge from several knowledge sources of which, typically, human experts are considered the most important. As Forsyth [1984] states, the power of an expert system derives from the knowledge it possesses, not from the particular formalisms and inference schemes it employs. One or more experts may become involved in an expert system development project depending on who posses the required expertise. Most expert system developers prefer a single expert, but there are many cases in which it is necessary to utilize multiple experts. McGraw and Harbison-Briggs [1989] identify four primary problems with knowledge acquisition from a single expert: (1) difficulty in allocating adequate time by a key individual; (2) possible bias; (3) limitation to a single line of reasoning; and (4) incomplete domain expertise.

The knowledge domains (problem domain and tool domain) in this research require key individuals with extensive knowledge and experience in solving manufacturing systems problems by utilizing analysis and optimization tools and techniques. Such individuals, either from industry or from academia, are usually the busiest people and have little time to spare for a research effort. Additionally, based on specific knowledge and experience, individuals posses their own biases towards the problems situations and the tools to analyze those problems. The expression "to one who has a hammer as a tool, everything looks like a nail" is a well known proverb related to this issue. In addition, no one individual can reasonably be expected to have complete knowledge and experience in these broad and complex domains. It is for these reasons that the knowledge acquisition process in this research utilizes multiple experts from academia and from industry.

The first step in knowledge acquisition is the identification of the experts from which the knowledge is to be acquired. One of the most significant issues to be addressed in the process of expert selection is the identification of minimal requirements for the experts. In other words, who is to be considered an expert for a given domain. Table VI lists the minimum requirements for the experts in the domains of problem definition and tool selection that are utilized in this research. Other factors that are critical to the expert selection process are the availability and the willingness of the experts to contribute and the ability to explaining their expertise [McGraw and Harbison-Briggs 1989].

TABLE VI. Minimum Requirements for the Experts

	Expertise in Problem Domain *Symptoms and Problems in Manufacturing*	**Expertise in Tool Domain** *Simulation* | *Queueing* | *Petri Nets* | *Queueing + Simulation* | *Search-Based Optimization*
Experts Found in Industry	• Should have a minimum of 5 years of professional experience in manufacturing system design and control problems. • Consultants should have a minimum of 5 consulting contacts.	• Should have at least 5 years of professional experience in application of listed analysis tools in analyzing manufacturing system design and control problems.
Experts Found in Academia	• Should have published at least 4 papers or taught at least 4 courses related to manufacturing system design and control.	• Should have published at least 4 papers or taught at least 4 courses related to application of listed analysis tools in analyzing manufacturing system design and control problems.

Once the experts are identified, one or more knowledge acquisition techniques are used to extract the necessary knowledge from the experts. An overview of knowledge acquisition techniques was provided in Chapter IV. This research utilizes questionnaires and surveys as the primary knowledge acquisition technique and follow-up interviews as the secondary (backup) knowledge acquisition technique. It is known that the most common and probably the most effective knowledge acquisition technique for multiple experts is to conduct group discussions and/or individual interviews with the experts. Due to the geographical dispersion and the limited time availability of the experts the author chose to utilize questionnaires and surveys.

Selection of the Knowledge Representation Technique

Once the knowledge is acquired from the experts and other sources, the next step is to represent the knowledge. The purpose of knowledge representation is to organize the knowledge into a form such that the expert system can readily access it for decision-making purposes. Different types of knowledge and different types of knowledge representation techniques were summarized in Chapter IV. As stated therein, the nature of the knowledge acquired generally determines the best knowledge representation technique to represent it. There is not a single best knowledge representation technique for each and every different type of knowledge. The knowledge that is being used in this research can be classified as procedural knowledge that utilizes expert heuristics. Procedural knowledge describes *how* a problem is solved. As mentioned in Chapter IV, the best way to represent procedural knowledge is to use production rules or more commonly, rules.

Selection of the Technique to Deal with Uncertain Data

One of the most important characteristics of expert systems, when compared to traditional programming, is the ability to deal with uncertain/inexact data. Several techniques have been developed to handle uncertainty in decision making. Three common techniques, namely the Bayesian approach, certainty theory, and fuzzy logic, were briefly summarized in Chapter IV. As indicated therein, because the Bayesian approach relies on knowing prior probabilities of the events being considered and fuzzy logic requires extensive calculation procedures, these two technique were judged inappropriate for this research. Certainty theory is the most common technique used in recently developed rule-based expert system applications [Durkin 1994]. Its popularity comes from not only the ease of implementation of the production rules but also the accuracy of the results obtained. Hence, the experimental frame expert system utilizes certainty theory to cope with uncertainty.

Research Plan

In order to satisfy the research objectives listed in Chapter IV, the research was conducted in the following phases:

PHASE 1. Problem Domain Determination

As anticipated, this phase proved to be the most challenging and most time consuming part of the research. The purpose was to collect a comprehensive set of information relative to the problems encountered during decision making processes for a discrete part manufacturing system in the form of unstructured questions and symptoms, and map these into a fixed set of structured problems. This phase can be better described in two main tasks:

Task 1. Collection of information in the form of a comprehensive list of symptoms, a list of unstructured questions, and a list of structured problems. The source of this information was:

- Printed materials: books and journal articles related to the subject.
- Human experts: people from academia (e.g., Dr. Joe Mize, Dr. David Pratt, Dr. Manjunath Kamath, Dr. Timothy Greene, Dr. Adedeji Badiru, and Dr.

Bobbie Foote) and people from industry (e.g., Dr. Mustafa Pulat from Lucent Technologies, Inc., Dr. Ghassan Abdelnour from Seagate Technologies, Inc., and Dr. Deborah J. Seifert from Allied Signal Aerospace, Inc.) who have expert knowledge in discrete part manufacturing systems.

Task 2. Determination of the information with respect to the listed structured problems by having experts map one or more symptoms to one or more structured problems with a certain levels of belief (certainty). As stated in Chapter II, a structured problem can be in the form of either a problem or a what-if question.

Such a mapping process can be conceptualized as shown in Figure 15.

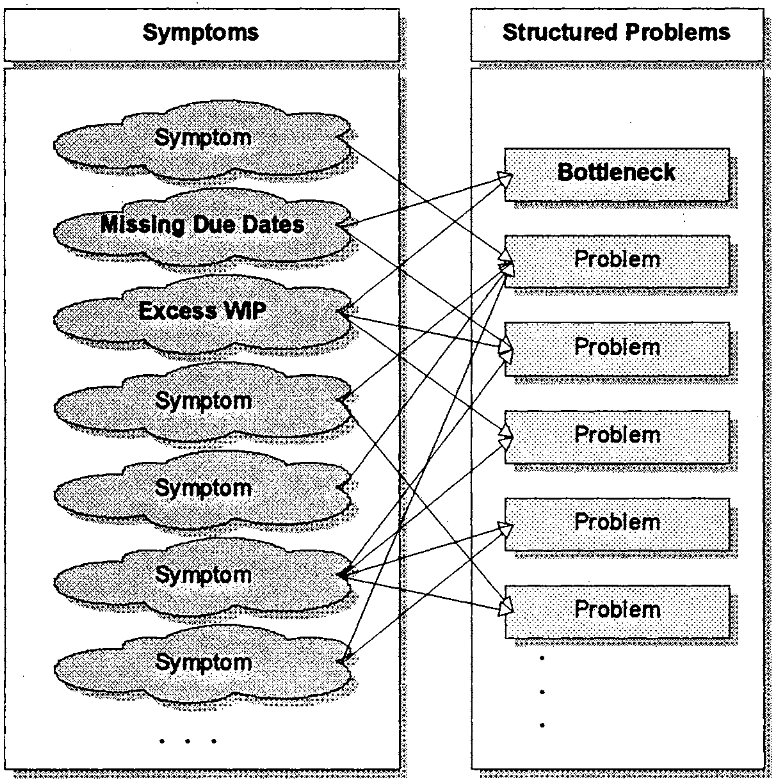


Figure 15. Mapping Questions and Symptoms to the Problems

PHASE 2. Problem Domain Knowledge Base

This phase was designed to organize the knowledge obtained in Phase 1 into a rule-based knowledge base. Rules, as described in Chapter IV, can be thought of as an extension of the simple IF *condition* THEN *action* format. A schematic example for a rule would be:

```
IF      Symptom #1
        AND  Symptom #2
        AND  Symptom #5
             OR    NOT   Symptom #12
THEN    Problem is Problem #3 with certainty of 80%
        Problem is Problem #13 with certainty of 60%.
```

And a specific example would be:

```
IF      Due dates are being missed
        AND  There is a lot of WIP
THEN    Problem is Bottleneck with certainty of 80%
        Problem is Poor Production Planning with certainty of 60%.
```

PHASE 3. Tool Domain Determination

This phase included collection and organization of information relative to the preference level of a set of analysis tools in the context of problems found in discrete part manufacturing systems. The set of analysis tools for this research consisted of simulation, queuing networks, Petri nets, queueing followed by simulation, and search-based optimization with simulation. Based on the structured problem determined in Phase 1 and

Phase 2, the set of appropriate tools can be put into a preferred order list. Given that the tool selection process is completed, a set of secondary factors are used to adjust the preference levels of the specified tools for a given structured problem. The secondary factors are the time to the solution, the level of detail required in output (e.g., averages, distributional data), and the complexity of the system under study. Consideration of these secondary factors can result in an adjusted preference order in which the ranking might or might not change. In this phase there are two main tasks:

Task 1. Determination of a list of tools that can be used to address a structured problem. Such information, which is collected from the domain experts, should not only specify the set of appropriate tools but also include the level of belief attached to each tool for the given structured problem. The source of this information was:

- Printed materials: books and journal articles related to the subject.
- Human experts: people from academia (e.g., Dr. Joe Mize, Dr. David Pratt, Dr. Manjunath Kamath, Dr. Timothy Greene, and Dr. Bobbie Foote) and people from industry (e.g., Dr. Mustafa Pulat from Lucent Technologies, Inc., Dr. Ghassan Abdelnour from Seagate Technologies) who have expert knowledge in different tools that can address problems found in discrete part manufacturing systems.

Task 2. Determination of a list of characteristics by which a set of analysis tools can be prioritized in terms of the user's preferences and/or requirements. These characteristics are the manufacturing system complexity, time to solution, and level of detail required from the output. Once the characteristics are determined, the next step is to prioritize the set of tools for each and every characteristic with a degree of confidence. The degree of confidence values are collected from the domain experts and subject to change from expert to expert.

PHASE 4. Tool Domain Knowledge Base

This phase focuses on designing the tool selection knowledge base given a structured problem, manufacturing system specific base model, capabilities and limitations of a given set of analysis tools in the context of a set of characteristics of interest, and requirements and preferences of the decision maker. This is also a rule-based knowledge base. A schematic example for a rule would be:

```
IF      Problem is #1
        OR    Problem is #4
THEN    Analysis tool is Q with certainty of 90%
        Analysis tool is P with certainty of 70%.
```

And a specific example would be:

```
IF      Problem is Excessive Machine Capacity
        OR    Problem is Excessive material handler capacity
THEN    Analysis tool is Simulation with certainty of 90%
        Analysis tool is Queueing with certainty of 70%
        Analysis tool is Petri nets with certainty of 40%.
```

PHASE 5. Proof of Concept Implementation

This phase can be described with the following tasks:

Task 1. Revise the existing object oriented modeling environment under development in the Center for CIM to accommodate the two knowledge bases in terms of providing the necessary information which resides in the base model into the knowledge bases in a concise format.

Task 2. Design and implement a user friendly set of interfaces to allow the communication between the knowledge bases and the decision maker.

Task 3. Revise the existing user interfaces, developed for the advanced modeling project, to accommodate the experimental frame expert system in a logical manner.

PHASE 6. Summarize Results and Prepare Final Format

Summarize and justify the system developed. This phase represents the culmination of the research activities and the presentation of results in final form.

PHASE 7. Further Research

Developing the long term framework providing directions for future research in this area.

Verification and Validation

The nature of the study requires that it utilize qualitative performance measures such as ease of use, completeness, user friendliness, and accuracy. Verification and validation are more important for expert systems than for conventional programs due to the inherent nature of artificial intelligence technology. Conventional programs are premised on proven scientific facts and numerical algorithms whereas expert systems attempt to implement heuristics that may not have been subjected to intense analysis.

Verification is essentially the determination of whether the system is functioning the way it was intended to function. Selection of appropriate tools, debugging of the graphical user interface, and stabilizing the run-time operation are a few factors that are included in verification of an expert system. The verification process of EFES is conducted considering these factors. More discussion of the verification process is given in Chapter VII.

Validation is concerned with how closely the expert system's solution matches the human expert's solution [Badiru 1992]. A valid expert system should offer solutions, recommendations, and conclusions that can be substituted for those of a human expert. The objective of validation is to ensure that, for correct input to the system, correct output is obtained. The following is a list of factors that are involved in expert system validation [Badiru 1992].

1. *Consistency.* The system should provide similar results for similar problem scenarios.
2. *Completeness.* Completeness concerns whether the system can effectively address all of the desired problems within the problem domain.
3. *Efficiency.* Efficiency reflects how well the expert system uses the knowledge bases, data, software, and computational power available.
4. *Soundness.* Soundness considers how solid the basis is on which the reasoning is premised. This is typically dependent on the quality of the knowledge used.
5. *Maintainability.* Maintainability involves how well the integrity of the system can be preserved even when operating conditions change.

6. *Precision.* Precision refers to the level of certainty or reliability associated with the recommendations provided by the system.
7. *Reliability.* Under reliability evaluation, the system is expected to perform satisfactorily whenever it is consulted. It should not be subject to erratic performance and results.
8. *Usability.* Usability considers such factors as easily understandable questions, ease of data input, and interaction capabilities.
9. *Accommodating.* Ideally, expert systems should be very forgiving of minor errors in data inputs. The user should be informed of incorrect data input. This requires an in-process consistency check procedure.
10. *Quality.* Quality depends on the level of knowledge in the knowledge base and the problem solving strategies used.
11. *Justification.* A key factor of validation involves justification. An expert system should be justified in terms of cost requirements, operating characteristics and user requests.
12. *Clarity.* Clarity refers to how well the system presents its prompts to avoid ambiguities in the input/output process.

A group of the experts who have provided the necessary knowledge to create EFES along with a few individuals who have the necessary knowledge in the problem domain and/or the tool domain but did not participate in the knowledge acquisition process were asked to validate the system. Results of the validation process are presented in Chapter VII.

CHAPTER VII

PRESENTATION AND ANALYSIS OF RESEARCH RESULTS

Introduction

The execution of an interaction with EFES can be summarized in a three-step process. This process is illustrated in Figure 16. As shown therein, the first step is to determine a structured problem. The second step is to determine a list of tools that can be used to address the structured problem. The third and last step in this process is to apply the secondary factors to the list of tools generated in the previous step. In the rest of this chapter, Figure 16 will be used as a reference in the discussions for conveying the overall framework.

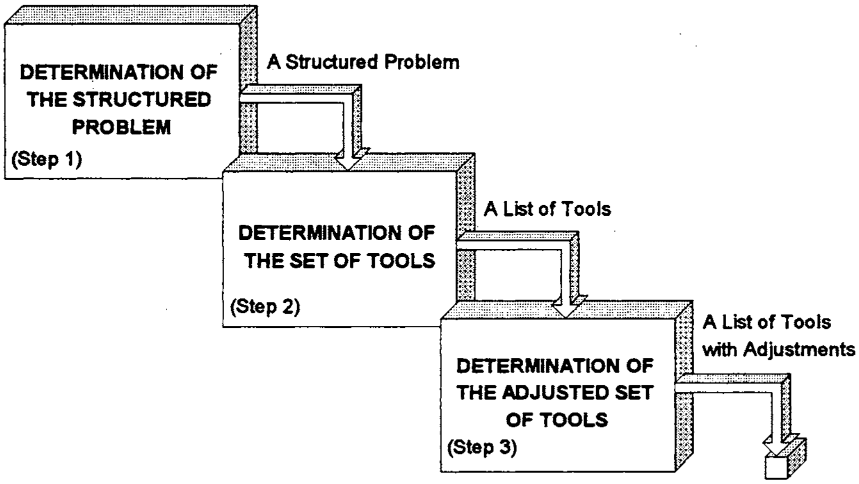

DETERMINATION OF THE STRUCTURED PROBLEM
(Step 1)
A Structured Problem
DETERMINATION OF THE SET OF TOOLS
(Step 2)
A List of Tools
DETERMINATION OF THE ADJUSTED SET OF TOOLS
(Step 3)
A List of Tools with Adjustments

Figure 16. Three Step Process in EFES Execution

This chapter presents the results of the research within the framework shown in Figure 16. The second section of this chapter presents the results of the knowledge acquisition process for the problem definition knowledge base. The most common symptoms and problems found in a discrete part manufacturing system, which were collected from the experts, are presented in a concise format. Mappings of symptom(s) to problem(s) are also given in this section. In the third section a similar presentation is made for the results of the knowledge acquisition process for the tool selection knowledge base. The forth section summarizes the calculations and application of the secondary factors to the tool selection process. Section five presents the verification and validation results of EFES. The concluding section discusses several implementation issues and presents an illustrative example of EFES and its use within the advanced modeling environment.

Problem Definition Knowledge Base

As presented in Chapter IV, the creation of a knowledge base starts with the assessment of the problem. Such an assessment process should include, but not be limited to, the justification of an expert system solution to the given problem. The next step is to determine the sources, including human experts as well as printed materials, from which the necessary knowledge can be acquired. Once the knowledge sources are determined, the human experts who have the required knowledge need to be identified. In this identification process, the knowledge engineer should specify criteria by which the definition of an "expert" in the given domain can be made. Then, the knowledge acquisition (often called knowledge elicitation if the knowledge is acquired from human experts) technique(s), through which the necessary knowledge will be collected, should be selected. Next, by utilizing the selected knowledge acquisition technique(s), the necessary knowledge is collected. After the collection process, the knowledge must be represented in a form that enables it to be usable by the expert system.

The purpose of the problem definition knowledge base is to derive a structured problem from a given set of symptoms and/or questions within the domain of discrete part manufacturing systems. This part of the overall process is illustrated by the Step 1 box in Figure 16. Based on the characteristics of expert systems summarized in Chapter IV, such a derivation process proves itself to be a good application for a knowledge based solution. Given that the problem domain is defined, the knowledge sources, among which the human experts are the most important, need to be determined. The identification of human experts who possess the necessary knowledge and/or experience is made according to the terms and conditions stated in Chapter VI. After the determination of the potential experts, the knowledge acquisition process is performed according to the techniques (surveys and interviews) listed in Chapter IV. Initial contact letters were sent to 42 experts who were qualified according to the criteria listed in Chapter IV. The final survey forms were sent to 32 of the 42 experts who agreed to participate in the study. From those 32 forms, 15 responses were received back from the experts. Out of these 15 experts who have responded to the survey, 4 have industrial, 4 have academic, and 7 have industrial (or consulting) and academic backgrounds with an average of 11 years of experience. All of the 15 experts have Ph.D. degrees. From these 15 experts, 8 reside in Oklahoma and 7 reside outside of Oklahoma. Several of the remaining 17 experts claimed not to have either the time or the knowledge to fill out the survey form after they had received it. Samples of this contact letters and survey forms are presented in Appendix C.

Collection of the domain knowledge was followed by an extensive organizing and structuring effort. As noted in Chapter IV, acquiring knowledge from multiple experts, almost always, introduces some level of conflicting results with respect to the information collected. One way of dealing with these conflicts, as stated in Chapter VI, is to establish a sub-set of the experts and have them come to a consensus on those issues. During the process of organizing the acquired knowledge for the problem definition knowledge base, no major conflicts were encountered. Lists of symptoms, problems, and what-if questions were generated by merging and rearranging the lists provided by the individual experts. In the survey data where the experts matched the symptoms to the structured problems, some minor conflicts were encountered. These minor differences, such as assigning different confidence levels to the same symptom-problem mapping, were resolved by taking the averages of the confidence levels. Since the differences were minor (± 10% of the average) averaging seemed to be a reasonable approach to resolve the differences.

What follows in this section are partial lists of symptoms, problems, what-if questions, and the mappings of symptom(s) to the problem(s). Because of the length of these lists, only partial lists are presented in this chapter. The full versions of these lists can be found in Appendix D.

Table VII is a partial list of the symptoms, which were collected from those experts who returned the survey form. A unique number (s1, s2, s3, etc.) is assigned to each symptom listed in Table VII. Based on the characteristics of the acquired knowledge, four categories were established for the symptoms (Inventory related, Customer related, Employee related, and Production related).

TABLE VII. Partial List of Symptoms

Number	Symptom Definition
	Inventory related
s1	Inaccurate inventory records (mismatch between actual and inventory numbers)
s2	Stockout of some RM items and/or high level of some other RM items
s3	High finished goods inventory on some items and stockout on others
s4	High WIP inventory (overall)
s5	High WIP in selected work centers
s6	Frequently running out of storage space
s7	Perishable items going bad
s8	Late deliveries from vendors
s9	Low quality RM from vendors
	Customer related
s10	Low customer evaluation
...	...

Table VIII is a partial list of the problems. A unique number (p1, p2, p3, etc.) is assigned to each problem listed in Table VIII.

TABLE VIII. Partial List of Problems

Number	Problem Definition
p1	Poor employee discipline
p2	Lack of training, know-how
p3	Lack of employees with required level of education
p4	Inflexible work force
p5	Inadequate reward/incentive system
...	...
p18	Low RM quality
p19	Bottleneck
p20	Deadlock
p21	Highly unreliable machines
p22	Inadequate maintenance
p23	Improper batch sizes (too small or too large)
...	...

Table IX is a partial list of the what-if questions. A unique number (w1, w2, w3, etc.) is assigned to each what-if question listed in the Table IX.

TABLE IX. Partial List of What-If Questions

Number	What-If Question
	Related to the Physical Objects
w1	Add/remove a work station
w2	Add/remove an assembly station
w3	Add/remove a material handler
w4	Add/remove a stock room
w5	Add/remove a product from the product-mix
w6	Change plant layout (add/remove an aisle)
w7	Add/remove a manufacturing line
w8	Add/remove tools, fixtures, pallets, etc.
w9	Change equipment reliability parameters (add/remove a maintenance crew)
w10	Add/remove operators
w11	Change operator capabilities (cross-training)
w12	Change worker assignments
...	...

Table X is a partial list of the mappings between the symptoms and the problems. The first column in Table X shows the unique numbers of the symptoms. The second column shows the mappings of the symptom(s) (underlined) to the problem(s) (indented) with the appropriate confidence levels (column 3).

TABLE X. Partial List of Mappings of Symptoms to Problems

Sy.	Matching Symptoms to the Problems	Confidence
#	*Symptoms are underlined, problems are indented*	Level (0-100)
s1	**Inaccurate inventory records**	
	Insufficient work-force	50
	Theft	70
	Poor storage design	80
	Poor record keeping	90
	Poor information system	50
s2	**Shortage of some RM and/or overstock of some others**	
	Theft	70
	Poor inventory policies	90
	Poor demand forecasting	70
	Low RM quality	40
	Incorrect BOMs	40
	Highly variable demand on product mix	60
	Poor record keeping	70
	Poor information system	50
...	...	...

Tool Selection Knowledge Base

The purpose of this knowledge base is to derive a set of the most appropriate tools for a given structured problem (a problem or a what-if question) in the domain of discrete part manufacturing systems. This part of the overall process is illustrated by the Step 2 box in Figure 16. Based on the characteristics of expert systems summarized in Chapter IV, such a derivation process can be accomplished through the application of a knowledge based solution. Given that the problem domain is defined, the knowledge sources, of which the human experts are the most important, need to be determined. Since no literature has been found in this subject area, human experts became even more important knowledge sources. The determination of the human experts who possess the necessary knowledge and/or experience is made according to the terms and conditions stated in Chapter IV. After the determination of the potential experts, the knowledge acquisition process was performed according to the techniques (surveys and interviews) listed in Chapter IV. Initial contact letters were sent to 26 experts who were qualified according to the criteria stated in Chapter VI. The final survey forms were sent to 17 of the 26 experts who agreed to participate. From these 17 forms, 9 responses were returned. Out of these 9 experts who have returned the survey, 2 have industrial, 2 have academic, and 5 have industrial (or consulting) and academic backgrounds with an average of 12 years of experience. All of the 9 experts have Ph.D. degrees. From these 9 experts, 4 reside in Oklahoma and 5 reside outside of Oklahoma. Several of the remaining 8 experts, who did not return the survey, claimed not to have either the time or the knowledge to complete it. Samples of the contact letters and survey forms are included in Appendix C.

Collection of the domain knowledge was followed by an extensive organizing and structuring effort. During the process of organizing the acquired knowledge for the tool selection knowledge base, no major conflicts were encountered. In the mapping of structured problems to the most promising tools, some minor differences were encountered among the results obtained from the experts. Some of these minor differences, such as assigning different confidence levels to the same structured problem to tool mapping, were resolved by taking the averages of those confidence levels. Others were resolved through meetings and discussions involving a small sub-set of the experts. An assumption made during the compilation of the knowledge acquired was that each piece of information is accurate regardless of the number of opposing opinions. For instance, if one expert indicates that "deadlock" problem can be addressed by "Petri net" tool with a degree of belief 0.90 and five other experts indicate that the same problem can be addressed by "Simulation" tool with an average degree of belief 0.70, the compiled results will include "Petri nets" as the first tool (with certainty level of 0.90) and "Simulation' as the second tool (with certainty level of 0.70) to address the same problem.

In the rest of this section, the compiled results, mappings of structured problems (problems and what-if questions) to the tools, which are collected from the experts, are presented in a table format. In Table XI, which is an illustration of mapping the problems to the appropriate tools, the first column represents the unique numbers of the problems. The second column, shows the mappings of the problem(s) (underlined) to the appropriate tools (indented). The third column, gives the confidence level (1 to 100) in mapping a problem to a tool. A complete version of this list (partially shown in Table XI) is presented in Appendix D.

TABLE XI. Partial List of Mappings of Structured Problems to Tools

Pr. #	**Matching Problems to the Tools** *Problems are underlined, tools are indented*	**Confidence Level** (0-100)
p1	**Poor employee discipline**	
	Non-modeling approach (e.g., observation)	90
p2	**Lack of training, know-how**	
	Non-modeling approach (e.g., observation)	90
...	**...**	...
p19	**Bottleneck**	
	Simulation	90
	Queueing	70
	Search-based optimization with simulation	70
p20	**Deadlock**	
	Petri nets	90
	Simulation	60
p21	**Highly unreliable machines**	
	Simulation	90
	Queueing	60
	Search-based optimization with simulation	50
p22	**Inadequate maintenance**	
	Simulation	90
	Queueing	60
...	**...**	...

Table XII is a partial list of the mappings of what-if questions to the tools in a similar manner to that outlined for the previous table.

TABLE XII. Partial List of Mappings of What-If Questions to Tools.

W. #	**Matching What-if Questions to the Tools** *What-if questions are underlined, tools are indented*	**Confidence Level** (0-100)
	With regard to the Physical Objects	
w1	**Add/remove a work station**	
	Simulation	90
	Queueing	80
	Queueing + Simulation	90
w2	**Add/remove an assembly station**	
	Simulation	90
	Queueing	80
	Queueing + Simulation	60
w3	**Add/remove a material handler**	
	Simulation	90
	Queueing	80
	Queueing + Simulation	60
...	...	...

Collecting, organizing, and structuring the necessary knowledge in the form of tables (presented above) followed by representing this knowledge in *rules* such that the knowledge can be used by a rule-based expert system. Two sample rules that are generated by using the previously presented tables are illustrated in Figure 17. The first rule (RULE 1) is an example of mapping the symptom(s) to the structured problem(s). The second rule (RULE 2) is an example of mapping the structured problem(s) to the tool(s). A complete listing of the rules generated for the problem definition knowledge base and the tool selection knowledge base are presented in Appendix E.

```
RULE 1 (taken from problem definition knowledge base)

ExcessiveOvertime
        "Ties the symptom 'Excessive overtime' to the possible structured problems"

        if: (anyOf: Symptoms have: [Name = 'Excessive overtime'])
                then: [
                StructuredProblem is: 'Insufficient work force' withCertainty: 0.7.
                StructuredProblem is: 'Inflexible work force' withCertainty: 0.4.
                StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.5.
                StructuredProblem is: 'Highly unreliable machines' withCertainty: 0.4.
                StructuredProblem is: 'Highly variable demand' withCertainty: 0.7.
                StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.6.
                StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.8]

RULE 2 (taken from tool selection knowledge base)

InsufficientCapacityOfResources
        " Ties the structured problem 'Insufficient capacity of resources' to the possible tools"

        if: (selectedProblem = 'Insufficient capacity of resources')
                then: [
                Tool is: 'Simulation' withCertainty: 0.9.
                Tool is: 'Queueing' withCertainty: 0.7.
                Tool is: 'Queueing + Simulation' withCertainty: 0.7]
```

Figure 17. Two Sample Rules from the EFES Knowledge Bases

This concludes the presentation of the results for the knowledge acquisition process. The next section presents the calculation and application of the secondary factors to the results of the tool selection knowledge base.

Consideration of Secondary Factors

In creating the prioritized list of preferred tools, the tool selection knowledge base considers the structured problem as the only determinant. As previously mentioned, in the process of tool selection the manufacturing system complexity and the user preferences and/or user requirements also need to be taken into account.

In this section, the methodology for including (calculating and incorporating) these additional factors (manufacturing system complexity, user preferences and/or requirements), which are considered to be secondary, is explained. The expected outcome from this process is a set of numbers (percentages) to adjust the confidence levels obtained from the tool selection knowledge base which considered only the primary factor (structured problem).

In the following sub-sections, these secondary factors are defined and the methods for calculating and incorporating these factors into the adjustments are explained.

Manufacturing System Complexity

This factor considers the complexity level of the manufacturing system being modeled. According to Dietrich [1991] the complexity of a manufacturing system is determined largely by the number of resources that exist in the system. In the context of this research, the complexity level of a discrete part manufacturing system is represented by a numerical value which can be determined by considering not only the number of resources that exist in the system but also the number of end-products along with their bills of material and routing structures. Following is a list of the five sub-factors, which are determined through the consultation with two of the local experts, that are to be included in the calculation process:

(1) number of stations in the system (work stations and assembly stations),

(2) number of material handlers,

(3) number of product types,

(4) structure of the bills of material, and

(5) routing structures.

The first three factors, which are summarized in Table XIII, are "number of stations", "number of material handlers", and "number of end-products". The Number of Stations is an arbitrary scaling factor value between 1 and 10 that can be mapped to the number of stations (work stations and assembly stations) that exist in the manufacturing system under consideration. The Number of Material Handlers is an arbitrary scaling factor value between 1 and 10 that can be mapped to the number of material handlers that exist in the manufacturing system under consideration. The Number of Products is an arbitrary scaling factor value between 1 and 10 that can be mapped to the number of end-products produced in the manufacturing system under consideration.

These sub-factors and the factor values are summarized in Table XIII. The break-points in Table XIII, that are to be used in mapping the number of resources (or end-products) to the factor values, were determined through the consultations with several local experts.

TABLE XIII. Factor Equivalents for Number of Stations, Number of MHs, and Number of Products

Number of Stations	**Factor (1-10)**	**Number of Mat. Hand.**	**Factor (1-10)**	**Number of Products**	**Factor (1-10)**
up to 5	1	1	1	1	1
6 to 10	2	2	2	2	2
11 to 15	3	3	3	3	3
16 to 20	4	4	4	4	4
21 to 25	5	5	5	5	5
26 to 30	6	6	6	6	6
31 to 35	7	7	7	7	7
36 to 40	8	8	8	8	8
41 to 45	9	9	9	9	9
over 45	10	over 9	10	over 9	10

Bills of Material (BOM) Structure is also an arbitrary scaling factor value between 1 and 10 that can be mapped to the BOM complexity of the end-products produced in the manufacturing system under consideration. BOM complexity of an end-product can be estimated by considering its depth and breadth. The depth of a product is a number that represents the maximum number of levels that exist in the BOM structure. The breadth of a product is a number that represents the maximum number of sub-parts that exist in a single level of the BOM structure. Given the depth and the breadth of an end product, the BOM structure can be calculated by the following equation which considers the effect of depth and breadth as being *additive* + *multiplicative* rather than simply *additive*.

$$\text{BOM Structure}_{01} = \text{Depth}_{01} + \text{Breadth}_{01} + \text{Depth}_{01} * \text{Breadth}_{01}$$

Figure 18 illustrates the concept of depth and breadth of the BOM of an arbitrary end product. For the BOM structure given in Figure 18, the depth for Product-01 is 5 and the breadth of the Product-01 is 4. Thus, the BOM structure of Product-01 yields 5 + 4 + 5*4 = 29. By applying this calculation process repeatedly to each and every end-product, a collection of values across the manufacturing system's products can be obtained. Let us assume this collection for a system is {29, 17, 26, 31, 22}. Given such a collection, a single representative number that corresponds to the BOM structure of the given manufacturing system can be approximated in a manner similar to the estimation procedure used in PERT networks [Bussey and Eschenbach 1992]:

Estimate Value = [Maximum + Minimum + 4 * Average] / 6

For the hypothetical example given above the BOM structure value is:

BOM Str. value = [31 + 17 + 4 * 25] / 6 = 24.67

Once the BOM structure value is calculated, the factor equivalent can be mapped from Table XIV. Again, the break-points in Table XIV were determined through the consultations with several local experts.

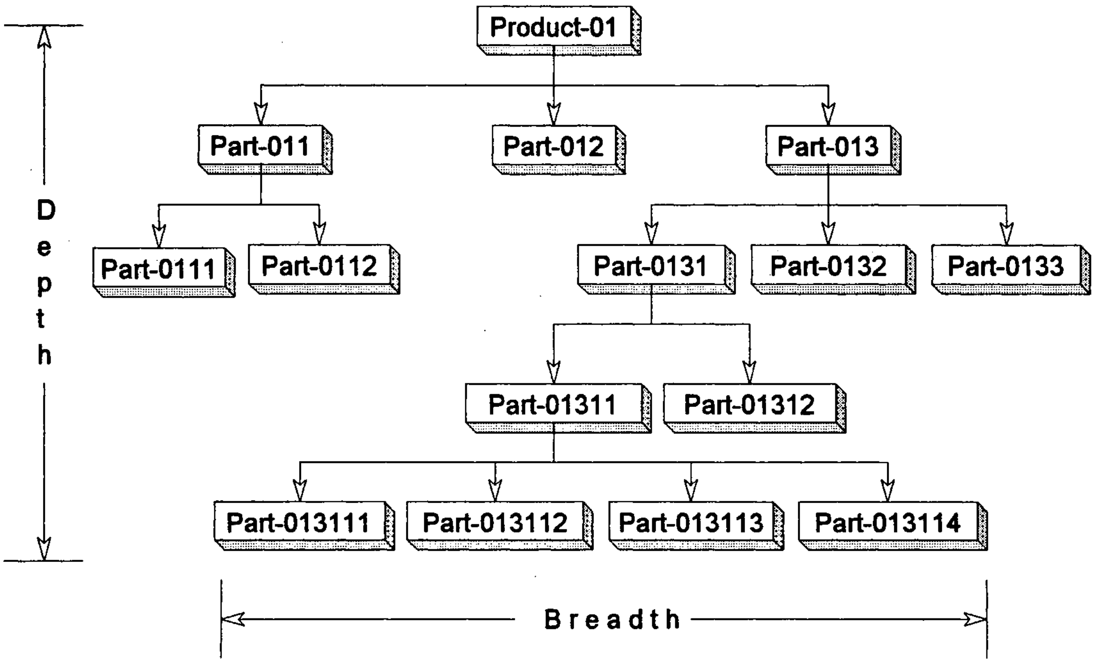


Figure 18. Illustration of Depth and Breadth of a Product

Routing Structure is, also, an arbitrary scaling factor value between 1 and 10 that can be mapped to the routing complexity of the end-products as well as the intermediate parts processed in the manufacturing system under consideration. Routing complexity of a part is, simply, the number of operations required for processing the part. If the modeling environment allows alternate routing, a breadth value can also be assigned to each part and can be included in calculation of routing complexity. For this research, due to the limitation of the modeling environment being used, breadth calculations are excluded from the calculation process. The routing complexities for each part in the system can be determined and put into a collection. Given such a collection, a single representative number that corresponds to the routing structure value of the given manufacturing system can be approximated through the following formulation:

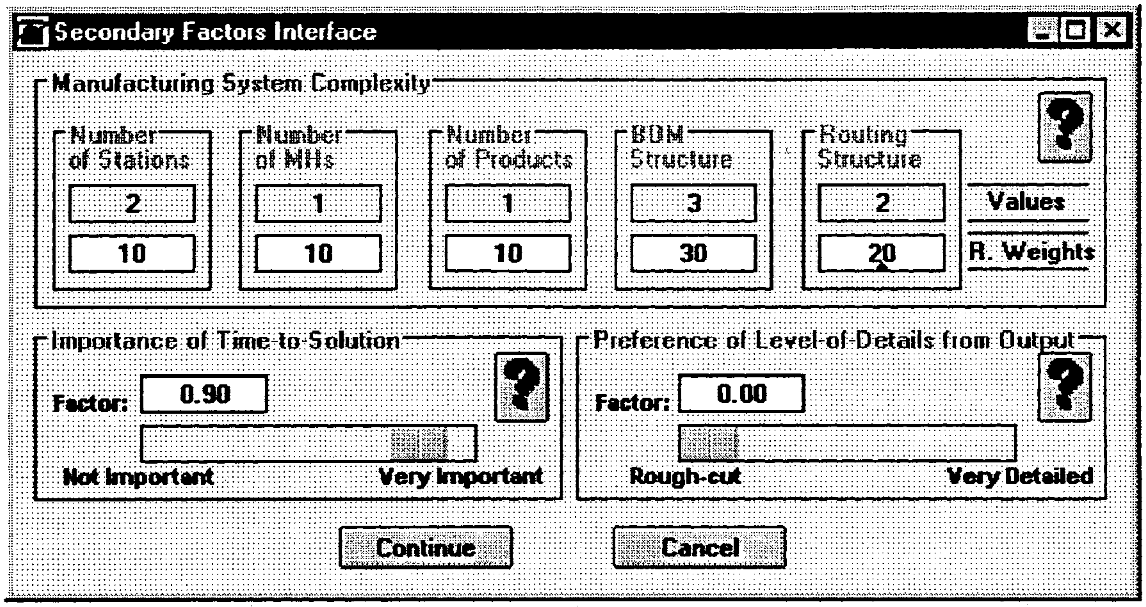
$$\text{Routing Str. Value} = [\text{Maximum} + \text{Minimum} + 4 * \text{Average}] / 6$$

Once the Routing structure value is calculated the factor equivalent can be mapped from Table XIV. The break-points for routing structure factor values were determined through the consultations with several local experts.

TABLE XIV. Factor Equivalents for BOM and Routing Structures

BOM Structure Value	**Factor (1-10)**
up to 8	1
9 to 15	2
16 to 24	3
25 to 35	4
36 to 48	5
49 to 63	6
64 to 80	7
81 to 99	8
100 to 120	9
over 120	10

Routing Structure Value	**Factor (1-10)**
up to 2	1
3 to 4	2
5 to 6	3
7 to 8	4
9 to 10	5
11 to 12	6
13 to 14	7
15 to 16	8
17 to 18	9
over 18	10



Secondary Factors Interface

Manufacturing System Complexity

	Number of Stations	Number of MHs	Number of Products	BOM Structure	Routing Structure
Values	2	1	1	3	2
R. Weights	10	10	10	30	20

Importance of Time-to-Solution

Factor: 0.90

Preference of Level-of-Details from Output

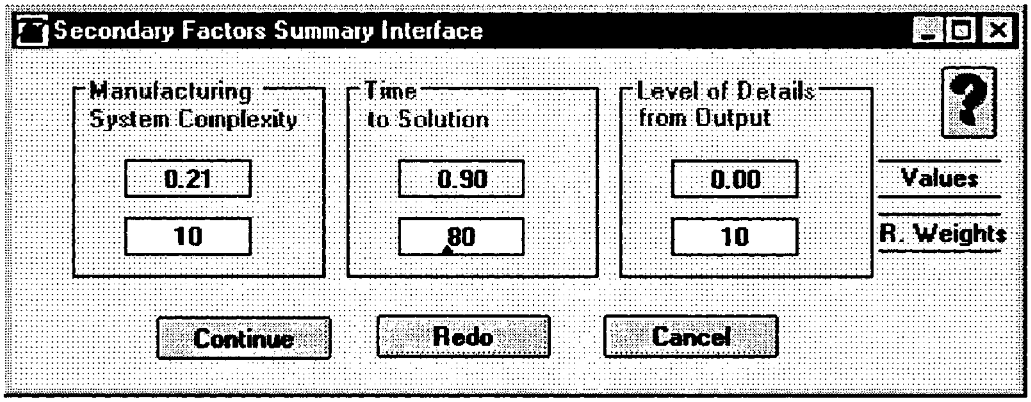
Factor: 0.00

Figure 19. User Preferences/Requirements Interface

User Preferences/User Requirements

In this research effort, the user preferences/user requirements category of secondary factors include time-to-solution and level-of-detail-from-output as the only two sub-factors. Given the set of tools being considered, these two sub-factors seem to have the greatest effect on assessing the tradeoffs among the tools. Time-to-Solution is a scaling factor value between 0 and 1 that specifies the user's preference or requirement on the time to solution. As illustrated in Figure 19, the user can specify this value through an "interface slider." Sliding the indicator towards "very important" (implying that a short time to solution is very important) will increase the value of this sub-factor. A user who wants to better understand this factor, can view an explanation through the help feature by pressing the question mark button in the time to solution box. Level-of-Detail-from-Output is also a scaling factor value between 0 and 1 that specifies the user's preference or requirement with respect to the level of detail of the output. As it is illustrated in Figure 19, the user can specify this value through an "interface slider". Sliding the indicator towards "very detailed" (implying that detailed distributional data is required rather than averages) will increase the value of this sub-factor. A user who wants to better understand this factor, can view an explanation through the help feature, incorporated into the interface, by pressing the question mark button in the level of detail from output box.

Once each of these secondary factor values (Manufacturing-System-Complexity, Time-to-Solution, and Level-of-Detail) are determined, the user can adjust the weighting associated with each of these factors through the interface illustrated in Figure 20.



Secondary Factors Summary Interface

	Manufacturing System Complexity	Time to Solution	Level of Details from Output
Values	0.21	0.90	0.00
R. Weights	10	80	10

Figure 20. Secondary Factors Weight Adjustment Interface

Once all three factors are determined and appropriate weights are specified, the next step is to determine the total effect of these secondary factors on the predetermined confidence levels of the recommended tools. To complete this determination, mappings between these secondary factors and the given set of tools are required, so that the changes to tool preference values can be reflected in the confidence levels of the tools.

Only a group of experts who have knowledge and experience in the given tool set can effectively estimate the sensitivity level of the preferences of the given tools for a given change in the secondary factor values. A group of the local experts were utilized in this knowledge acquisition process. Experts were given a survey (a copy of this survey document is given in Appendix C), which explains the nature of the knowledge required from them, and asks them to fill out the form shown in Table XV.

TABLE XV. Adjustment Levels Between Secondary Factors and the Tools

	FACTORS		
TOOLS	**Mfg-Sys-Compl.**	**Time-to-Solution**	**Level-of-Detail**
Simulation	∅	↓ 2 (-0.20)	↑ 1 (+0.30)
Queueing	↓ 2 (-0.20)	↑ 1 (+0.30)	↓ 2 (-0.20)
Petri Nets	↓ 1 (-0.30)	↓ 3 (-0.10)	↓ 1 (-0.30)
Queueing + Sim.	↑ 1 (+0.30)	∅	↑ 3 (+0.10)
SB Opt. w/Sim.	∅	↓ 1 (-0.30)	↑ 2 (+0.20)

While providing data for this table, experts are asked to consider each sub-factor independently from the others. For each column, experts first determine for each tool (row) whether the factor (column) results in an increase or decrease in the preference level of the tool. These are specified by an up arrow representing increased preference and a down arrow representing decreased preference. Next, the experts rank the same direction arrows in a column among themselves from most influential to least influential. The experts then repeat this process for the next column and continue until all columns have been evaluated. The results from this data collection process are summarized in Table

XV. The symbol "∅", found in some boxes in Table XV, means that no value could be assigned to that intersection of the tools and factors through the limited survey and the follow-up consultation with the experts. An important point that comes out of this is the fact that there is not a single answer to the preferability of the tools under the consideration of the listed factors.

As an example of the interpretation of Table XV, if the time to solution preference and/or requirement increases (Time-to-Solution column), preference for the simulation tool decreases (Simulation row), preference towards the queueing tool increases (Queueing row), preference towards the Petri nets tool decreases (Petri nets row), preference towards queueing + simulation does not change (Queueing + Sim. row), and finally preference towards the search-based optimization with simulation tool decreases (SB Opt. w/Sim. row). Among the decreasing preferences, search-based optimization with simulation decreases most (specified by assigning 1), then simulation (specified by assigning 2), and then Petri nets (specified by assigning 3). In this particular column the only increase is shown in preference towards the queueing tool.

Now that all component pieces of data are determined, the final adjusting factor for a given tool can be computed by using the following equation which was developed for this research:

$$AFTool_i = F_{msc} * MSC_i + F_{tts} * TTS_i + F_{lod} * LOD_i$$

where,

i : 1 for simulation tool
2 for queueing tool
3 for Petri nets tool
4 for queueing + simulation tool
5 for search-based optimization with simulation tool

$AFTool_i$: Final adjusting factor for tool i
F_{msc} : Factor value for manufacturing system complexity
MSC_i : Adjustment weight for manufacturing system complexity for tool i
F_{tts} : Factor value for time to solution
TTS_i : Adjustment weight for time to solution for tool i
F_{lod} : Factor value for desired level of detail from output
LOD_i : Adjustment weight for desired level of detail from output for tool i

The final step is to apply these final adjusting factors to the confidence levels of the corresponding tools in the recommended tools list. At this point, each recommended tool will have two confidence levels. One is determined through the tool selection knowledge base which considers only the structured problem. The other is determined through the application of the secondary factors. In some cases, the second set of confidence levels might be similar (in ranking and magnitude) to the first set of the confidence levels, and in other cases the second set might have the tools in a preference order that is different from the first set. The final output at this stage reflects both the original preferences and the user adjusted preferences. The final determination of the tool to use is left to the user.

Verification and Validation of EFES

Verification, as it has defined in Chapter VI, is the determination of whether the system is functioning the way it was intended to function. Selection of appropriate tools, debugging of the graphical user interface, and stabilizing the run-time operation are a few factors that should be included in verification of an expert system. In EFES the verification process included all of these factors. The HUMBLE [XEROX 1994] expert systems shell was selected for the proof of concept implementation. The major reason for choosing HUMBLE was the fact that HUMBLE is developed under Smalltalk-80 which is the development platform for the advanced modeling environment. Having the expert system shell and the modeling environment under the same development platform eases the interfacing between the two application modules. In addition, HUMBLE has facilities to trace the execution of the program through which an extensive debugging and verification can easily be accomplished. Once the trace function of HUMBLE is turned on, expert system developer can choose to trace the sequence of goals (or conclusions) reached or can trace the sequence of rules fired. A short trace output, generated by HUMBLE tracer, is given in Figure 21. Since the trace is a knowledge base dependent activity (specific to a single knowledge base), in the verification of EFES multiple trace outputs were generated for both problem definition and tool selection knowledge bases.

```
Partial trace on GOALS reached

Goal - Name
Finished - Name
Goal - StructuredProblem
Goal - Name
Finished - Name
Finished - StructuredProblem
...
Partial trace on CONCLUSIONS reached

Rule ExcessiveAmountOfBackorders concludes Problem_Scenario-1 - StructuredProblem -
        'Highly variable demand' (0.7)
Rule ExcessiveAmountOfBackorders concludes Problem_Scenario-1 - StructuredProblem -
        'Obsolete technology' (0.7)
...
Partial trace on RULES fired

firing... TooManyUnplannedActivities
firing... TooManyJobsAtTopPriority
firing... MissingProductionWorkOrders
firing... PoorFlowOfMaterials
...
```

Figure 21. Partial Trace Outputs Generated by EFES

The objective of validation, as mentioned in Chapter VI, is to measure how closely the expert system's solution matches the human expert's solution. The best way of measuring this closeness is to have the experts evaluate the system based on a given set of factors. In the evaluation of EFES a group of local experts who have taken part in the knowledge acquisition process, along with a few individuals who have the necessary knowledge in the problem domain and/or the tool domain but did not take part in the knowledge acquisition process, were asked individually to evaluate the EFES based on its ease of use and user friendliness (e.g., help features, input/output interaction with user, appearance of the interfaces, etc.). Based on the limited evaluation made by 10 experts, EFES was found to be a "good" application for a proof of concept implementation.

A full (commercially viable/professional) version of EFES should be subjected to a more rigorous validation process that should include: (1) consideration of all the 12 factors listed in Chapter VI, (2) participation of a majority of the experts who have provided the knowledge along with a large number of individuals who have the necessary knowledge and experience in the problem domain and the tool domain but did not participate in the knowledge acquisition process, and (3) utilization of a rich set of problem scenarios in the evaluation process. Results of such validation could be summarized in a form of a table (Table XVI). Cells in Table XVI would be filled in with numerical values specifying the evaluation results obtained from the experts for each and every factor (row). Past experience by other expert system developers indicated that this process takes weeks (or perhaps months) of interaction between multiple experts and EFES. For this reason, full validation of EFES was judged to be beyond the scope of this effort (whose purpose was proof of concept).

TABLE XVI. Results Table for a Complete Validation

FACTOR DESCRIPTIONS		**FACTOR VALUE** **(1: very poor - 5: very good)**				
		Expert 1	**Expert 2**	**Expert 3**	**...**	**AVERAGE**
1	***Consistency***					
2	***Completeness***					
3	***Efficiency***					
4	***Soundness***					
5	***Usability***					
6	***Accommodating***					
7	***Quality***					
8	***Maintainability***					
9	***Precision***					
10	***Reliability***					
11	***Justification***					
12	***Clarity***					

An Example EFES Session

The consultation session between the user and EFES can be illustrated in a flowchart (Figure 22). Once a consultation session is initiated by pressing the expert button (human picture) in the environment launcher interface (illustrated in Figure 23a on page 107), the user is asked to specify the study type (Figure 23b). There are two choices; (1) a problematic scenario and (2) a what-if scenario. If the user chooses the what-if scenario, a list of what-if questions is given to the user (Figure 23c), and the user is expected to select one from this list. If the user chooses the problematic scenario from the study types, then a second question is posed by EFES. The user is asked to specify if he/she knows what the problem is (Figure 23d). If so, the user is asked to choose from a given list of problems (Figure 23e). Not finding a problem in this list might mean, either the problem falls outside the scope of EFES or the definition (wording) of the problem is different from what EFES provides.

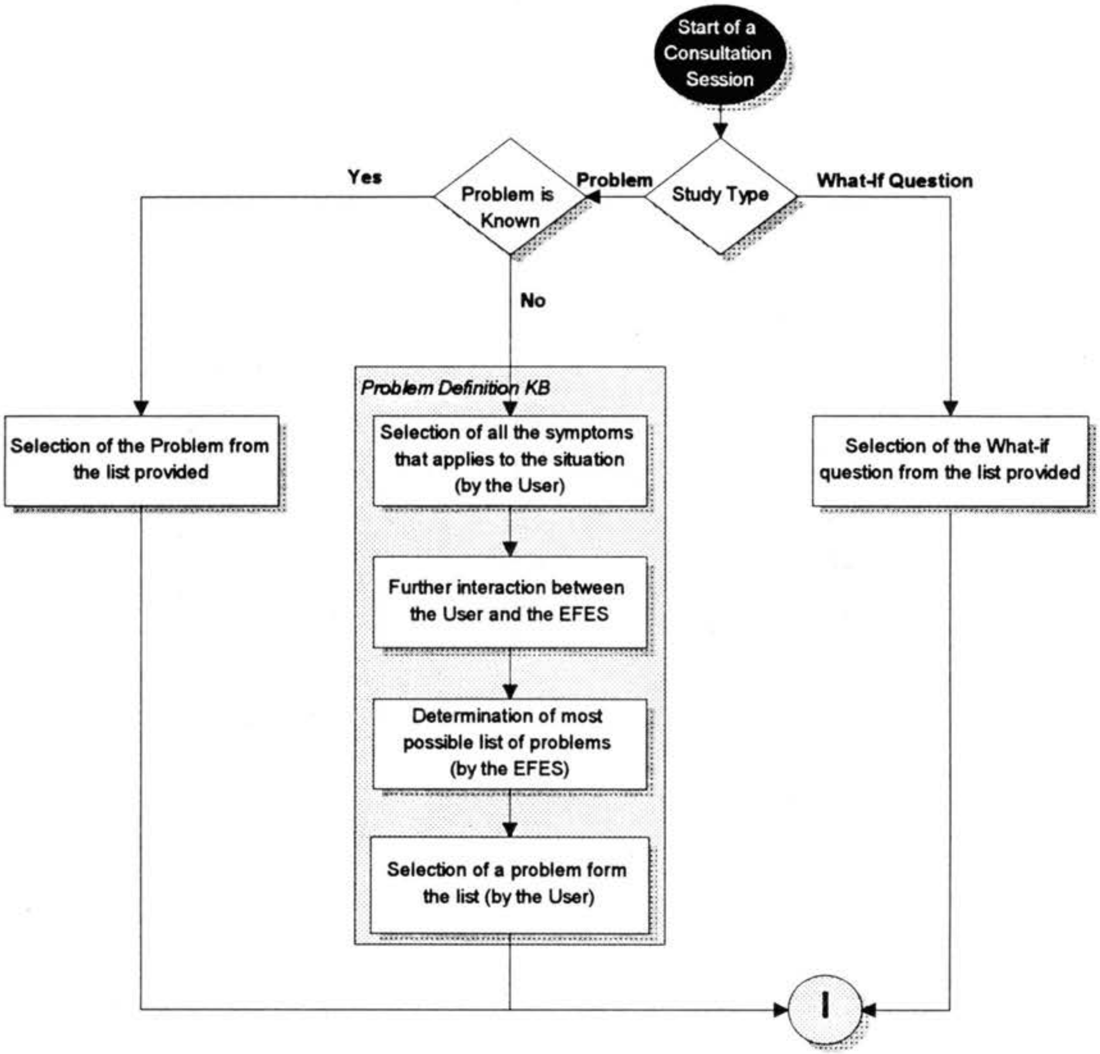

Start of a Consultation Session
Study Type
Problem
What-If Question
Problem is Known
Yes
No
Problem Definition KB
Selection of the Problem from the list provided
Selection of all the symptoms that applies to the situation (by the User)
Further interaction between the User and the EFES
Determination of most possible list of problems (by the EFES)
Selection of a problem form the list (by the User)
Selection of the What-if question from the list provided
I

Figure 22. Creation of the Structured Problem

If the user does not now what the problem is, then EFES makes use of the problem definition knowledge base and provides the user with a list of symptoms from which the user is expected to select as many symptoms as apply to the situation (Figures 23f, 23g, and 23h). Let us assume that the user selects two symptoms from the provided list: "High

WIP inventory (overall)" and "Excessive overtime." Selection of these two symptoms will lead the inference engine of EFES to fire corresponding rules (ExcessiveOvertime and HighWIPInventoryOverall) from the problem definition knowledge base (these two rules are presented in Appendix E). As it can be seen from these two rules, the conclusion sets (structured problems) are not mutually exclusive. In other words, they both conclude in the same type of structured problems with different degrees of belief (a.k.a. confidence factors). For instance, one of the conclusions in ExcessiveOvertime is "Insufficient capacity of resources" with a degree of belief 0.70 which, at the same time, is one of the conclusions in HighWIPInventoryOverall with a degree of belief 0.80. These two positive conclusions' degrees of belief are to be combined by EFES before presenting the conclusion as one of the recommendations. The way of combining these degrees of belief for these two rules, which conclude to the same hypothesis, is accomplished by the calculations presented in Chapter IV. For the example on hand, the following calculations apply:

$$CF_{COMBINE}(CF_1, CF_2) = CF_1 + CF_2 * (1 - CF_1) \quad \text{if both} > 0$$

$$CF_{COMBINE}(0.70, 0.80) = 0.70 + 0.80 * (1 - 0.70)$$

$$CF_{COMBINE}(CF_1, CF_2) = 0.70 + 0.24 = 0.94$$

As a result, one of the structured problem in the recommendation list will have a name "Insufficient capacity of resources" with the degree of belief 0.94. Let us assume that the user selects "Insufficient capacity of resources" as the structured problem from the recommendation list and continues on the consultation (Figure 23i).

Once the structured problem is defined, the EFES uses the tool selection knowledge base (fires the corresponding rule(s)) in order to recommend the most suitable tools in a prioritized order. This part of the process is depicted in the flowchart given in Figure 24. Since the structured problem in this example is assumed to be "Insufficient capacity of resources", EFES fires the corresponding rule named InsufficientCapacityOfResources (presented in Appendix E) from the tool selection knowledge base. As it can be seen from this rule, the structured problem is tied to three tools with different degrees of belief: Simulation with a degree of belief 0.90, Queueing with a degree of belief 0.70, and Queueing + Simulation with a degree of belief 0.70. The next step to be taken by EFES is to determine (Figure 19) and apply the secondary factors to these tools and the degrees of belief. Manufacturing system complexity, which is the first of three secondary factors considered in this study, is consist of five sub-factors (number of stations (work stations and assembly stations), number of material handlers, number of end-products, BOM structure, and routing structure) and is determined by evaluating the information which resides in the current base model. The process of determining the sub-factors and calculating the manufacturing system complexity is explained early in this chapter. For the example on hand, based on the information obtained from the current base model, the following factor values are determined: number of stations = 2, number of MHs = 1, number of products = 1, BOM structure = 3, and routing structure: 2. Let us assume that the relative weights for these five sub-factors are specified by the user as 10, 10, 10, 30, and 20 respectively. Given these factor values and the relative weights, the manufacturing system complexity can be calculated as follows:

$$MSC = [2*(10/80) + 1*(10/80) + 1*(10/80) + 3*(30/80) + 2*(20/80)]/10$$

$$MSC \approx 0.21$$

Let us also assume that the user specifies the time-to-solution as 0.90 and the level-of-detail-from-output as 0.00, and continues on the consultation. The summary of these three secondary factors as well as the relative weights are presented in the next interface (previously illustrated in Figure 20). Let us assume that the user adjusts the defaulted relative weights for the secondary factors (manufacturing system complexity, time-to-solution, and level-of-detail-from-output) as 10, 80, and 10 respectively. Based on the factor values, weights, and knowledge acquired from the experts (summarized in Table XV) with respect to the capabilities and/or limitations of the tools, the adjustment factors can be calculated as presented earlier in this chapter. Following are the calculations for the example on hand. For the tools, which are determined by utilizing the tool selection knowledge base (Simulation withCertainty 0.90, Queueing withCertainty 0.70, and Queueing + Simulation withCertainty 0.70), the following calculations apply:

$AFTool_i = F_{msc} * MSC_i + F_{tts} * TTS_i + F_{lod} * LOD_i$

for i = 1 (Simulation tool)

$AFTool_1 = F_{msc} * MSC_1 + F_{tts} * TTS_1 + F_{lod} * LOD_1$

$AFTool_1 = 0.21*(0*(10/100)) + 0.90*(-0.20*(80/100)) + 0*(0.30*(10/100))$

$AFTool_1 = -0.144$

for i = 2 (Queueing tool)

$AFTool_2 = F_{msc} * MSC_2 + F_{tts} * TTS_2 + F_{lod} * LOD_2$

$AFTool_2 = 0.21*(-0.20*(10/100)) + 0.90*(0.30*(80/100)) + 0*(-0.20*(10/100))$

$AFTool_2 = 0.212$

for i = 3 (Queueing + Simulation tool)

$AFTool_3 = F_{msc} * MSC_3 + F_{tts} * TTS_3 + F_{lod} * LOD_3$

$AFTool_3 = 0.21*(0.30*(10/100)) + 0.90*(0*(80/100)) + 0*(0.10*(10/100))$

$AFTool_3 = 0.006$

Next, these adjustment factors are applied to the degrees of belief (a.k.a. weights) of the recommended tools. Following shows these calculations:

⇒ for Simulation tool

Un-Adjusted Weight = 0.90

Adjusted Weight = 0.90*(1-0.144) = 0.770

⇒ for Queueing tool

Un-Adjusted Weight = 0.70

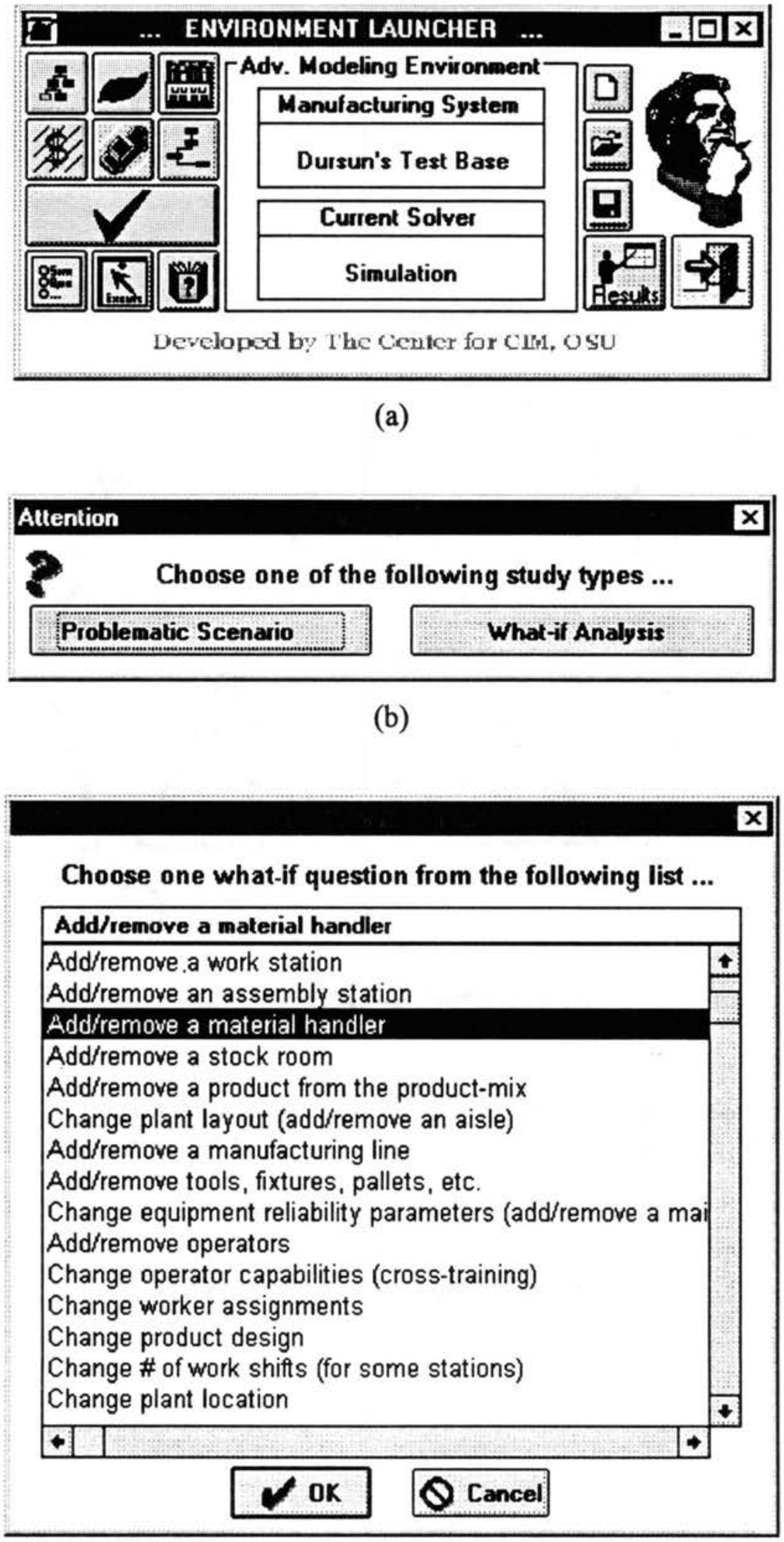
Adjusted Weight = 0.90*(1+0.212) = 0.848

⇒ for Queueing + Simulation tool

Un-Adjusted Weight = 0.70

Adjusted Weight = 0.90*(1+0.006) = 0.704

Application of these adjustment factors to the original confidence factors will result in creating a second set of confidence factors which are called adjusted confidence factors (weights). A results interface which shows these two sets of confidence factors (weights) is illustrated in Figure 25. The reader should notice that the application of the secondary factors taken into account in this hypothetical example resulted in changing the most preferred tool from Simulation to Queueing. Such a change in the prioritized order is expressed by a warning box in the results interface. The user is free to choose any tool that is in the recommended list. Once a tool is selected, the advanced modeling environment takes over, and performs the analysis.



... ENVIRONMENT LAUNCHER ...

Adv. Modeling Environment

Manufacturing System

Dursun's Test Base

Current Solver

Simulation

Developed by The Center for CIM, OSU

(a)

Attention

Choose one of the following study types ...

Problematic Scenario

What-if Analysis

(b)

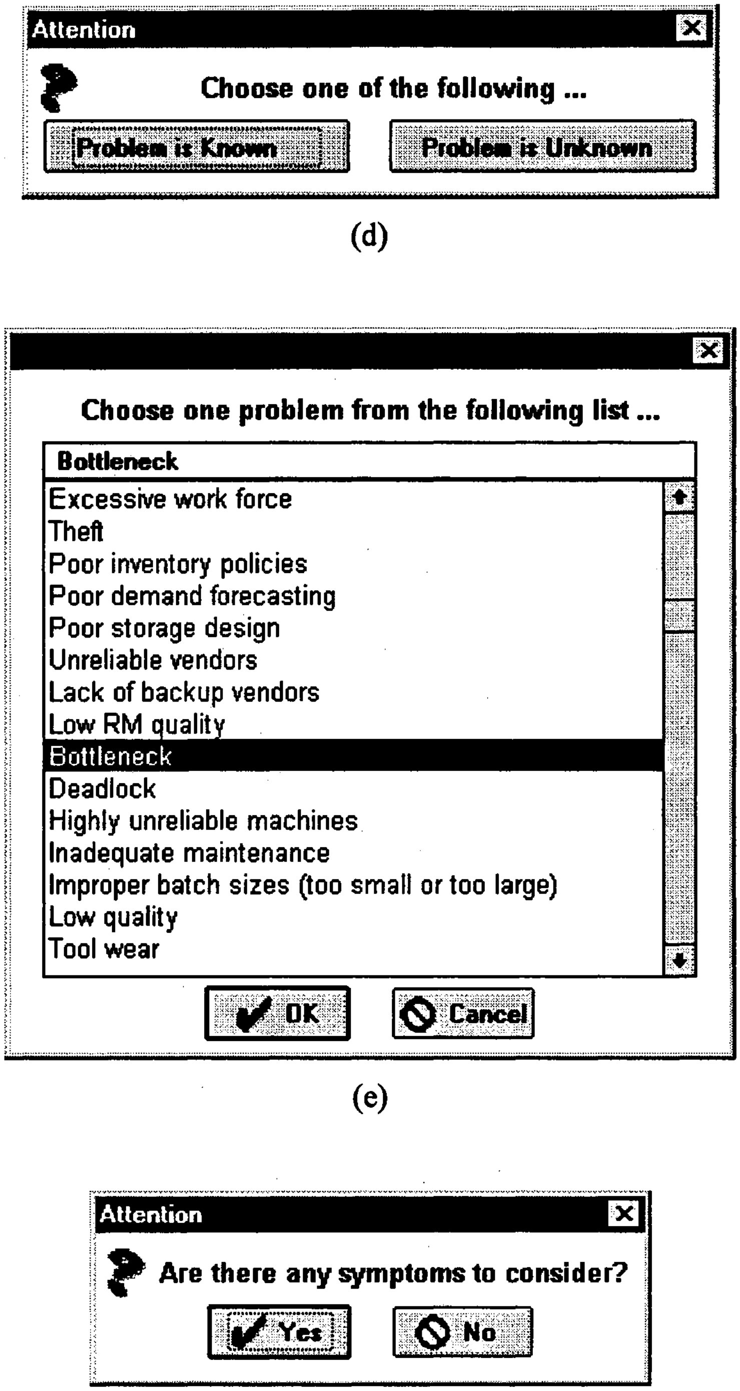
Choose one what-if question from the following list ...

Add/remove a material handler

Add/remove.a work station
Add/remove an assembly station
Add/remove a material handler
Add/remove a stock room
Add/remove a product from the product-mix
Change plant layout (add/remove an aisle)
Add/remove a manufacturing line
Add/remove tools, fixtures, pallets, etc.
Change equipment reliability parameters (add/remove a mai
Add/remove operators
Change operator capabilities (cross-training)
Change worker assignments
Change product design
Change # of work shifts (for some stations)
Change plant location

(c)

Figure 23. EFES Interfaces



Attention

Choose one of the following ...

(d)

Choose one problem from the following list ...

Bottleneck

- Excessive work force
- Theft
- Poor inventory policies
- Poor demand forecasting
- Poor storage design
- Unreliable vendors
- Lack of backup vendors
- Low RM quality
- Bottleneck
- Deadlock
- Highly unreliable machines
- Inadequate maintenance
- Improper batch sizes (too small or too large)
- Low quality
- Tool wear

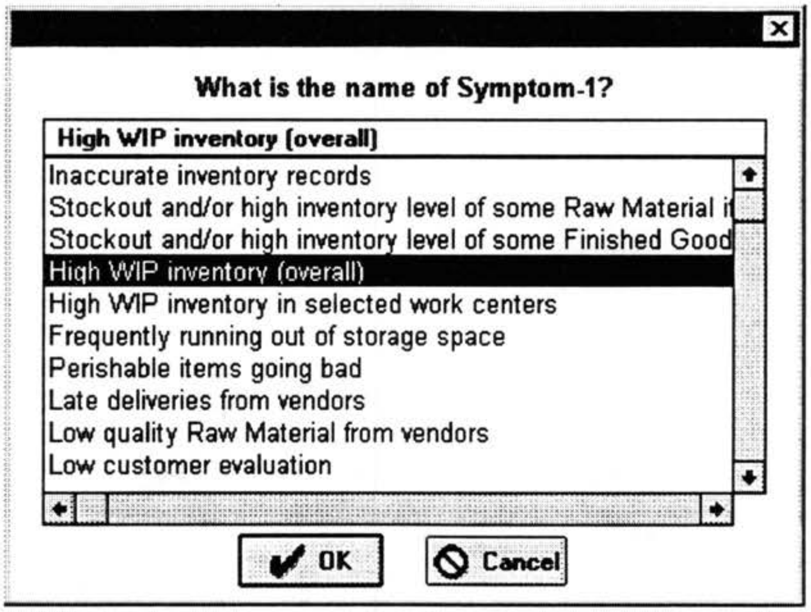
(e)

Attention

Are there any symptoms to consider?

(f)

Figure 23 (Continued). EFES Interfaces

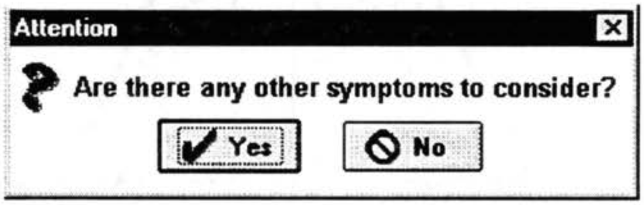


What is the name of Symptom-1?

High WIP inventory (overall)

Inaccurate inventory records
Stockout and/or high inventory level of some Raw Material i
Stockout and/or high inventory level of some Finished Good
High WIP inventory (overall)
High WIP inventory in selected work centers
Frequently running out of storage space
Perishable items going bad
Late deliveries from vendors
Low quality Raw Material from vendors
Low customer evaluation

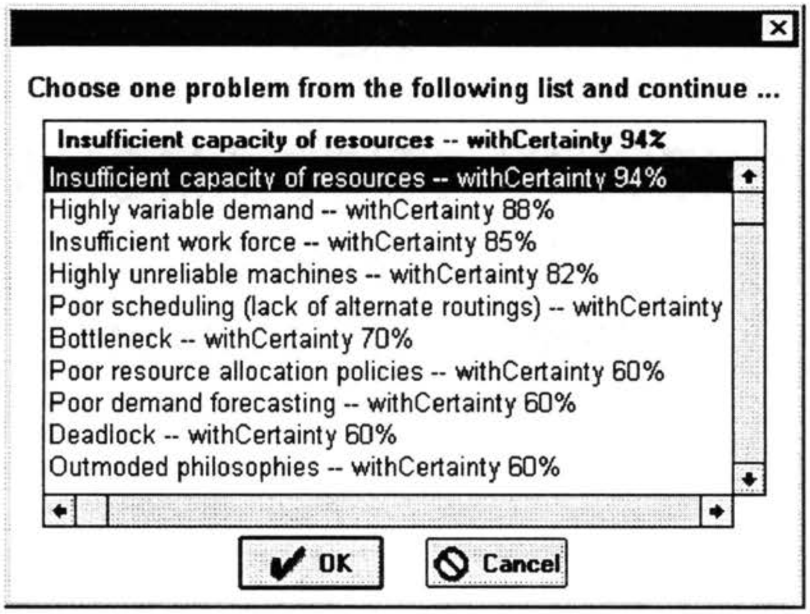
(g)



Attention

Are there any other symptoms to consider?

(h)



Choose one problem from the following list and continue ...

Insufficient capacity of resources -- withCertainty 94%

Insufficient capacity of resources -- withCertainty 94%
Highly variable demand -- withCertainty 88%
Insufficient work force -- withCertainty 85%
Highly unreliable machines -- withCertainty 82%
Poor scheduling (lack of alternate routings) -- withCertainty
Bottleneck -- withCertainty 70%
Poor resource allocation policies -- withCertainty 60%
Poor demand forecasting -- withCertainty 60%
Deadlock -- withCertainty 60%
Outmoded philosophies -- withCertainty 60%

(i)

Figure 23 (Continued). EFES Interfaces

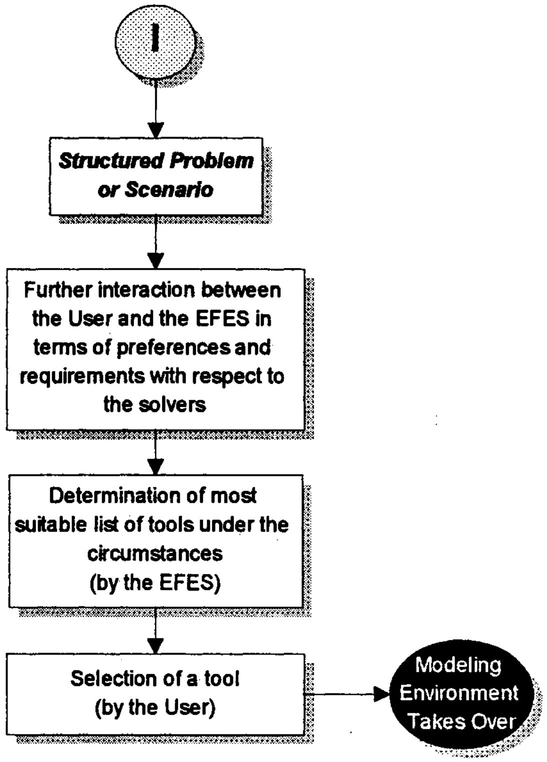
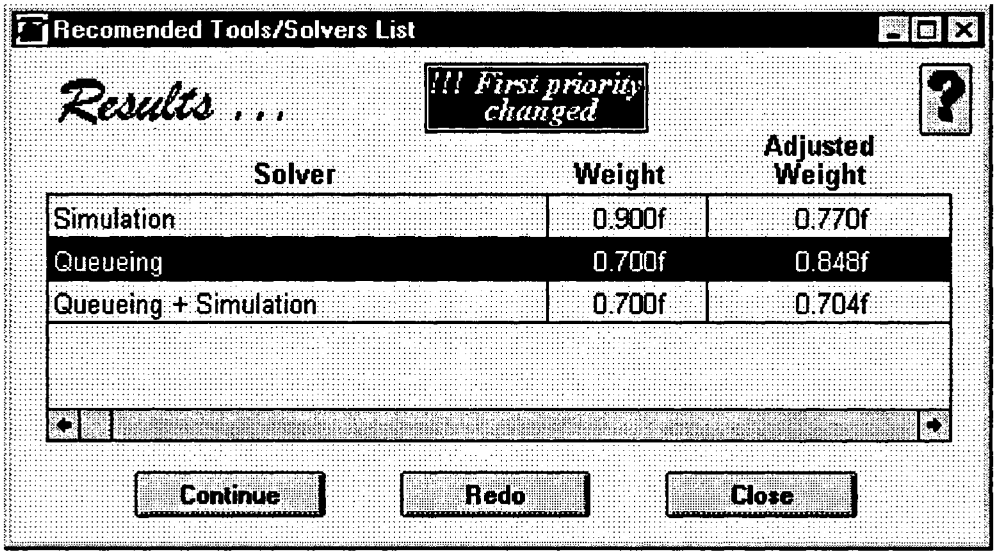

1
Structured Problem or Scenario
Further interaction between the User and the EFES in terms of preferences and requirements with respect to the solvers
Determination of most suitable list of tools under the circumstances (by the EFES)
Selection of a tool (by the User)
Modeling Environment Takes Over

Figure 24. Determination of the Tools



Recomended Tools/Solvers List

Results ...

!!! First priority changed

Solver	Weight	Adjusted Weight
Simulation	0.900f	0.770f
Queueing	0.700f	0.848f
Queueing + Simulation	0.700f	0.704f

(j)

Figure 25. EFES Results Interface

CHAPTER VIII

SUMMARY, CONCLUSIONS, AND RECOMMENDATIONS

Introduction

Chapter VI and Chapter VII have presented the methodology and the outcomes of this research effort. The first section of this chapter summarizes the research results and links the research outcomes to the research objectives defined in Chapter V. The contributions of this research to the field of Industrial Engineering, specifically to the modeling of manufacturing systems, are also identified in this chapter. The final section of this chapter outlines possible directions for further research.

Research Summary

The goal of this research was to develop a methodology that utilizes expert systems in the experimental frame module of an advanced modeling environment for manufacturing systems in order to provide timely assistance to the problem definition and tool selection decision making processes. The goal was systematically addressed and successfully achieved through the accomplishment of six research objectives, defined in Chapter V, which served as the driving force for the completion of this research effort. The potential generalizability of these research results is discussed later in this chapter. The following sub-sections present the highlights of the research outcomes and identify their contributions to the various research objectives.

Acquire the Necessary Knowledge for the Problem Domain

The first objective was to acquire the necessary knowledge for the problem domain. The methodology for accomplishing this objective was defined and justified in Chapter VI. As stated in Chapter V, the process of acquiring the necessary knowledge for the problem domain required the following sub-objectives to be accomplished in the given order:

- ◊ deciding on the definition of "who is an expert?" in this domain,
- ◊ determining a number of experts who can satisfy this definition,
- ◊ deciding on the knowledge acquisition method to be used,
- ◊ acquiring the knowledge from the experts by utilizing the most suited knowledge acquisition technique(s), and
- ◊ organizing and structuring the acquired knowledge.

One of the outcomes of this acquisition process was a collection of symptoms, what-if questions, and structured problems commonly found in discrete part manufacturing systems (refer to Table X, Table XI, and Table XII in the previous chapter and Appendix D). Another outcome was the mappings of those symptoms and questions to the structured problems (refer to Table XIII in the previous chapter and Appendix D).

Development of the Problem Definition Knowledge Base

This objective required the development of the problem definition knowledge base using the acquired knowledge from the previous objective. This knowledge base was designed such that it interacts with the user to collect enough information to recommend a list of structured problems in a prioritized order.

Acquire the Necessary Knowledge for the Tool Domain

This objective was to develop a conceptual framework that specified the capabilities and limitations of the given set of tools. Based on the characteristics of the problem to be addressed, the preferability of the various tools is determined. Following were the sub-objectives that could be extracted from this objective:

- ◊ deciding on the tool set by limiting the analysis tools to a manageable size was required for the sake of timely completion of the knowledge acquisition process of this part of the research;
- ◊ deciding on the definition of "who is an expert?" with regard to the selected tool set;
- ◊ determining potential experts who can satisfy this definition;
- ◊ acquiring the knowledge from the experts by utilizing knowledge acquisition techniques; and
- ◊ structuring the acquired knowledge.

The result was a list of mappings through which the structured problems are tied to the selected tools. Resolving the conflicts and mapping the problems to the tools was the theme of this objective. The completion of this objective resulted in a set of mappings between the structured problems and the selected tools (refer to Table XIV in the previous chapter and Appendix D).

Development of the Tool Selection Knowledge Base

This objective required the development of the tool selection knowledge base using the knowledge acquired as a result of the previous objective. This knowledge base is designed such that it recommends a list of tools in a prioritized order for a given structured problem. Following this, the secondary factors are applied to the results of the tool selection knowledge base. As stated earlier, this process creates a set of adjusted confidence factors in addition to the original factors.

Development of the Proof of Concept Implementation of EFES

A prototype version of the results obtained from Objective 1 through Objective 4 was implemented in a way that demonstrated proof of concept. This implementation was accomplished using the advanced modeling environment (AME) under development in the Center for Computer Integrated Manufacturing at Oklahoma State University. One of the major challenges faced in developing the proof of concept implementation was to interface EFES with the rest of AME. First, the existing structure of AME was modified from both the interfaces and class hierarchy standpoints. Second, a new class called "Interrogator" was created to play the role of an intermediator between EFES and AME, such that data passing from EFES to AME and from AME to EFES can easily be accomplished. Such a software architecture, from a conceptual standpoint, is depicted in Figure 26. The interrogator class not only facilitates the interactions between EFES and AME (in the form of receiving and passing arguments) but also governs the change of control (logic flow) from AME to EFES and vise versa during a consultation.

The evaluation of EFES, which was conducted by the experts who contributed to the development of the expert system, gave a very satisfactory result for a proof of concept implementation. The relevant portions of new and/or modified Smalltalk-80 code (classes and methods) are provided in Appendix F for interested readers.

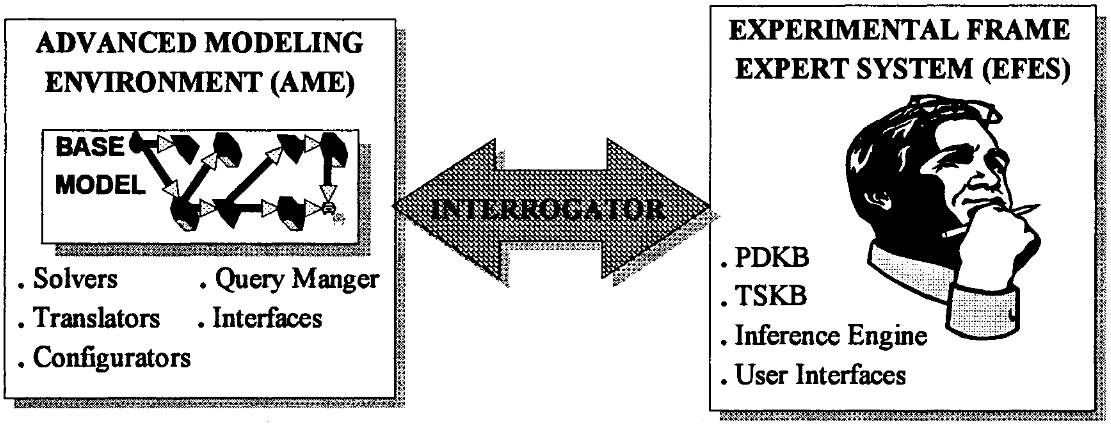


Figure 26. Conceptual Representation of High Level Software Modules

Further Research

The final objective was the identification of further research directions in this area. Many areas of potential research exist within the broad context of AI applications to the modeling of manufacturing systems. Published results to date in this area barely scratch the surface of what might be accomplished via the application of AI techniques in the field of modeling of manufacturing systems. The final section of this chapter presents ideas on further research that grew out of this research effort.

Research Contributions

The primary motivation for this research, as it is stated in Chapter I, is the author's desire to contribute to the advancement of modeling of complex manufacturing systems by addressing some of the issues related to the disadvantages of traditional modeling methodologies including the lack of reusability and lack of accessibility. At the highest level, the contribution of this research effort to the field of modeling of manufacturing systems is a new framework within which AI techniques are combined with the

OOP/OOM paradigm to maximize the accessibility of models by non-modeling specialists. The specific contributions of this research to the field of Industrial Engineering, especially modeling of manufacturing systems, are as follows.

- ◊ Through the knowledge acquisition processes conducted with the domain experts, lists of symptoms, problems, and what-if questions which can be commonly found in discrete part manufacturing systems are collected, organized, and presented. Though these lists are not exhaustive, they are a good start for further investigation. To the best of the author's knowledge no list as comprehensive as the one established in this work exists in the literature.
- ◊ Mappings of collected symptoms to structured problems and structured problems to the given tools are collected, organized, and presented. To the best of the author's knowledge no list of mappings as comprehensive as the one established in this work exists in the literature.
- ◊ Very little has been done in the area of multi-tool/multi-use modeling of manufacturing systems in the past, because single purpose manufacturing modeling tools were not robust enough to handle the wide variety of problems typically encountered in the manufacturing setting. With the introduction of new, more advanced modeling environments, the evolution of a structured problem from a set of symptoms and the selection of an appropriate tool by a non-modeling specialist becomes a more visible issue. This research has taken a first step towards defining, clarifying, and resolving this issue.

◊ A proof of concept implementation of the proposed framework has been developed in the Smalltalk-80 and integrated into the ongoing research environment of the Center for CIM at OSU. From the software architecture standpoint, the integration of EFES to the rest of the modeling environment has been accomplished through the application of an Interrogator class which resides between EFES and the advanced modeling environment and facilitates not only the information passing between the modules but also the control of the logic flow during a consultation. While much remains to be done in this area before a commercially viable methodology is achieved, this research has established the validity and feasibility of the approach.

◊ During validation, one expert stated that "in many ways the goals of EFES are consistent with the goals of an IE education - to understand problem solving tools (their capabilities and limitations) and capture the knowledge of how and when to apply which tools to what kind of problems". In this regard, the framework proposed under EFES can be used as a partial self assessment of an Industrial Engineering curriculum.

◊ During the execution of this research effort the following knowledge has been discovered:

- Knowledge acquired from the experts with regard to the symptoms, problems, and mappings of the symptoms to the problems were surprisingly consistent.

- Knowledge acquired from the experts with regard to the mappings of the structured problems to the tools were relatively consistent.
- Knowledge acquired from the experts with regard to the capabilities and the limitations of the given set of tools (with regard to the complexity of the manufacturing system under study, time to solution, and the level of details required from the output) were surprisingly inconsistent (diverse). The availability of multi-tool modeling elevates the resolution of this issue to a higher level. This issue most certainly deserves additional research effort.

◊ The extensible use of the advanced modeling environment under development in the Center for CIM at OSU has been demonstrated.

Recommendations for Further Research

Further research directions which are identified during this research effort can be classifies into following five categories:

1. Extensions to the EFES framework.
2. Extension of the knowledge bases.
3. Substitution of expert systems with neural networks.
4. Application of the framework to other fields.
5. Further understanding and documenting problem definition and tool selection decision making processes.

Extension to the EFES Framework

In addition to the problem definition and the tool selection knowledge bases, two more knowledge bases could be added to EFES to enhance the advisory role. These two additional knowledge bases would be an *experiment generator* knowledge base and a *results interpreter* knowledge base. Figure 27 illustrates this new EFES structure.

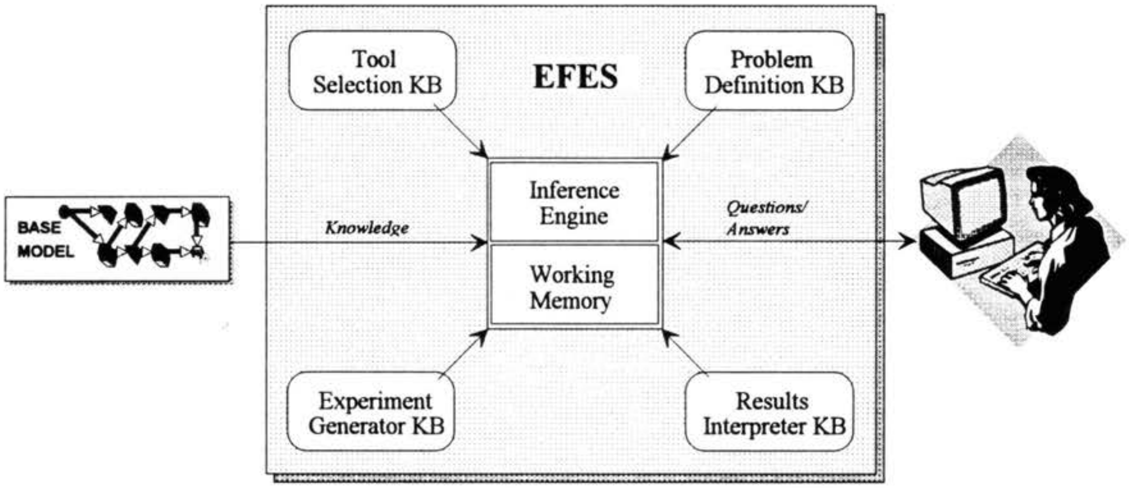


Figure 27. Extended EFES with Its 4 Knowledge Bases

The results interpreter knowledge base would be designed in a manner such that it takes the output generated from the analysis and converts it into a format that can be understood by the experiment generator knowledge base. Then the experiment generator knowledge base, by considering the results supplied by the results interpreter knowledge base, decides on new/additional experimentation based on an objective function determined by the user before the start of the analysis.

Once the results satisfy the objective function, the experiment generator signals back to the results interpreter knowledge base with the indication of the completion of the experiments. At this point, the results interpreter knowledge base converts/configures the compiled results into textual and graphical outputs in a manner that can easily be understood by the user (decision maker). Figure 28 illustrates the fitting of this new EFES extension in the previously presented modeling framework (Figure 6).

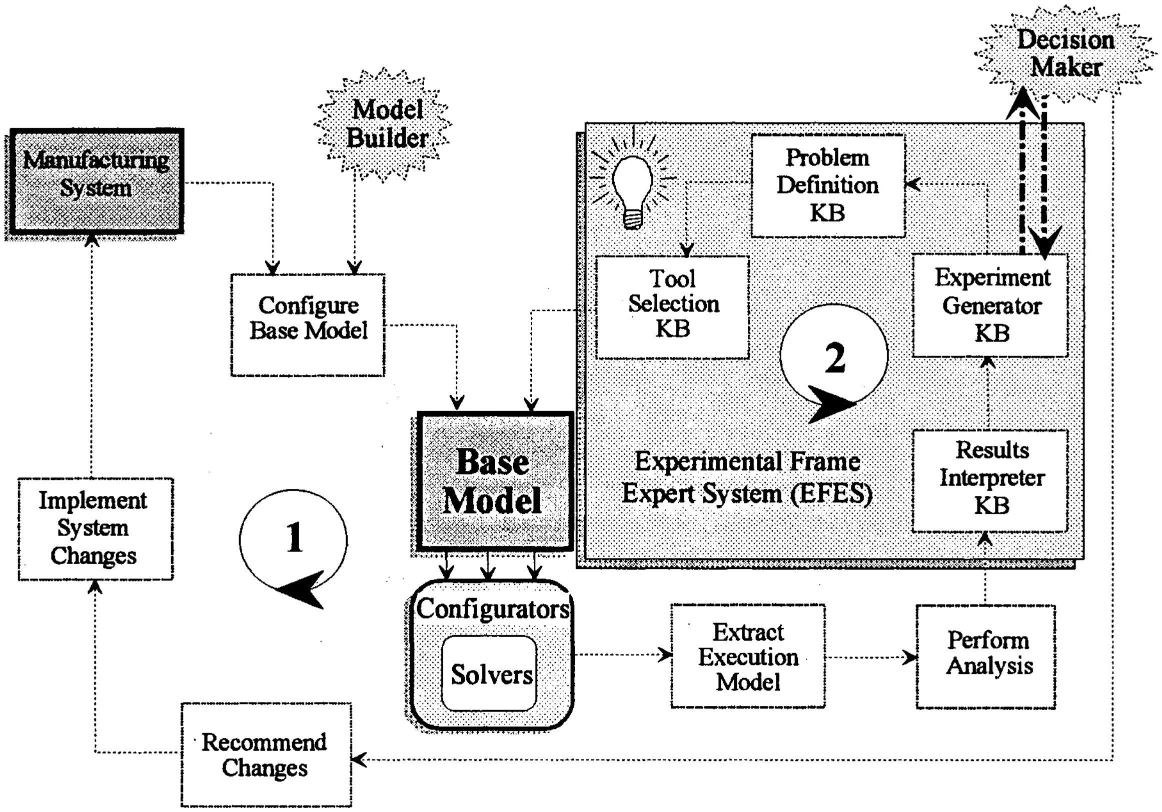


Figure 28. A New Framework with More Complete EFES Extension

As shown in Figure 28, the decision maker poses the question and the objective function to the experiment generator. After experimentation, output, which can be understood by the decision maker, and that satisfies the user defined objective function, is produced by EFES.

Extension of the Knowledge Bases

As part of this research, two knowledge bases have been created. Due to the nature of the study, the lists of the symptoms, problems, and what-if questions are constructed to apply to discrete part manufacturing systems. A commercially viable version of these knowledge bases can be created by considering the following issues:

- ◊ Problem domain can be narrowed down to a specific application field (e.g., small size automobile manufacturing) in order to have a better coverage of the domain in the representative knowledge bases. The lists presented in this study can be taken as a starting point for the creation of such knowledge bases.
- ◊ In the *selection of experts*, one of the main objectives should be to create a diverse, and at the same time, balanced poll of experts. Such diversity and balance should be representative of the real world.
- ◊ In the *knowledge acquisition from the experts*, multiple contacts to the experts should be made. Though this is a somewhat time consuming process, for the sake of completeness and accuracy of the knowledge bases, it is a necessary task. If possible, interviews should be used as the primary data collection technique, and surveys should be used as secondary data collection technique for only the cases in which interviews are not viable. Suitability of other advanced techniques for knowledge acquisition from multiple experts, listed in Chapter IV, should be investigated (the Delphi method is suggested).
- ◊ In the *knowledge representation*, an alternative approach would be to incorporate the secondary factors into the tool selection knowledge base so

that the whole tool recommendation process can be handled in a single step (as opposed to having problem to tool and application of secondary factors as two separate steps). In this case, the rules in the problem definition knowledge base will include not only the mappings of structured problems to tools but also the capabilities and limitations of those tools with respect to the manufacturing system complexity, time to solution, and level of detail required from output.

◊ In *resolving the conflicts*, which are commonly inherent in the knowledge acquired from multiple experts, no commonly accepted technique is exist in the current literature (at least based on the best of author's knowledge). Development of such a technique warrants further investigation.

◊ In the *validation of the knowledge bases*, a fixed number of problem scenarios can be used. Both, experts who have contributed to the knowledge acquisition process and experts who have not been utilized in the knowledge acquisition process should be asked to respond to these pre-determined set of scenarios. Then, the results obtained from the expert system and the results obtained from the experts should be compared. A valid expert system application is expected to generate close results to that obtained from the human experts.

Substitution of Expert Systems with Neural Networks

It is a well known fact in the field of AI that the bottleneck in the process of creating an expert system is the acquisition of the domain knowledge from a variety of sources of which human experts are the most important. Neural networks are another well known and highly promising AI technique. The most attractive characteristic of neural networks, as opposed to expert systems, is their capability of learning from examples and not being dependent on an exhaustive knowledge acquisition process. This advanced feature of neural networks might make them a promising alternative to the expert system in the EFES framework.

Application of the framework to other fields

The general framework of EFES can be thought of independently from the application field, and can be applied to fields outside discrete part manufacturing systems. For example, the same framework might easily be applied to the health industries (i.e., a hospital). The parts in the discrete manufacturing system would be replaced by patients in the hospital, the resources such as machines, operators, material handlers, pallets, etc. in the discrete manufacturing system would be replaced by emergency rooms, doctors, wheel chairs, beds, etc. in the hospital. Problem definition and tool selection processes should apply to the modeling of health industries the same way they apply to the modeling of discrete part manufacturing systems.

Further understanding and documenting problem definition and tool selection decision making processes

During the data collection phase of this research a need for answering the following questions emerged.

- What do we mean by a structured problem?
- How do we distinguish a structured problem from an "unstructured" one?
- What is the meaning of an experimental frame?

- How do we define problem definition and tool selection decision making processes within a discrete part manufacturing system?

An initial attempt was made towards answering these questions in this research effort. A further understanding and documenting of these issues warrants additional consideration.

BIBLIOGRAPHY

Adiga, S. (1989). "Software Modeling of Manufacturing Systems: A Case for an Object-Oriented Programming Approach." *Annals of Operations Research* **17**: 363-378.

Adiga, S. and C. R. Glassey (1991). "Object-Oriented Programming in Manufacturing Systems." *International Journal of Production Research* **29**(12): 2529-2542.

Adiga, S., R. G. Petrakian, and P. M. Shabe (1992). "A Knowledge-Based Support System for Prototyping Simulation with a Reusable Software Library." *Information and Decision Technologies* **18**(3): 185-194.

Akao, Y. (1990). *Quality Function Deployment*. Productivity Press, Cambridge, MA.

Ang, C. L., M. Luo, and R. K. L. Gay (1994). "Development of a Knowledge-Based Manufacturing Modelling System Based on IDEF0 for the Metal Cutting Industry." *International Journal of Production Economics* **34**: 267-281.

Attia, F., O. Hosny, S. Ramu, and N. Chawla (1992a). "EMHES: An Expert System for Material Handling Equipment Selection." *Expersys-92*. IITT International: 407-412.

Attia, F., O. Hosny, G. Sundarraj, and S. Ramu (1992b). "An Expert System Model for Process Planning in Computer Integrated Manufacturing." *Expersys-92*. IITT International: 396-411.

Aytug, H., G. J. Koehler, and J. L. Snowdon (1994). "Genetic Learning of Dynamic Scheduling Within a Simulation Environment." *Computers and Operations Research* **21**(8): 909-925.

Badiru, A. B. (1992). *Expert Systems Applications in Engineering and Manufacturing*. Prentice Hall, New Jersey.

Bansana, E., G. Bel, and D. Dubois (1988). "OPAL: A Multi-Knowledge-Based System for Industrial Job-Shop Scheduling." *International Journal of Production Research* **26**(5): 795-819.

Basnet, C. and J. H. Mize (1994). "Scheduling and Control of Flexible Manufacturing Systems: A Critical Review." *International Journal of CIM* **7**(6): 340-355.

Bhuskute, H. C., M. N. Duse, J. Gharpure, M. Kamath, D. B. Pratt, and J. H. Mize (1992). "Design and Implementation of a Highly Reusable Modeling and Simulation Framework for Discrete Part Manufacturing Systems." *Proceedings of the 1992 Winter Simulation Conference*. IEEE: 680-688.

Billatos, S. B. and P. C. Tseng (1991). "Knowledge-Based Optimization for Intelligent Machining." *Journal of Manufacturing Systems* **10**(6): 464-475.

Bird, S. (1991). "Object-Oriented Expert System Architectures for Manufacturing Quality Management." *Journal of Manufacturing Systems* **11**(1): 50-60.

Bose, P. P. (1988). "Expert Systems for Maintenance." *American Machinist & Automated Manufacturing* (June): 52-54.

Bravoco, R. R. and S. B. Yadav (1985a). "A Methodology to Model the Functional Structure of an Organization." *Computers in Industry* **6**: 345-361.

Bravoco, R. R. and S. B. Yadav (1985b). "Requirement Definition Architecture - An Overview." *Computers in Industry* **6**: 237-251.

Bussey, L. E., T. G. Eschenbach (1992). *The Economic Analysis of Industrial Projects*. Prentice Hall, New Jersey.

Buzacott, J. A. and J. G. Shanthikumar (1985). "Approximate Queueing Models of Dynamic Job Shop." *Management Science* **31**: 870-887.

Centeno, M. A. and C. R. Standridge (1992). "Databases and Artificial Intelligence: Enabling Technologies for Simulation Modeling." *Proceedings of the 1992 Winter Simulation Conference*. IEEE: 428-434.

Cesarone, J. (1991). "QEX: An In-Process Quality Control Expert System." *Robotics and CIM* **8**(4): 257-264.

Dagli, C. H. (1988). "Expert System for Selecting Quality Control Charts." *Expert Systems: Strategies and Solutions in Manufacturing Design and Planning*. A. Kusiak. Dearborn, MI, SME: 325-343.

Day, W. B. and M. J. Rostosky (1994). "Diagnostic Expert Systems for PLC Controlled Manufacturing Equipment." *International Journal of CIM* **7**(2): 116-122.

Deisenroth, M. P., S. Nof, and W. L. Meier (1980). "Computerized Physical Simulators are Developed to Solve IE Problems." *Industrial Engineering* **12**(10): 70-75.

Delen, D., D. Pratt, and M. Kamath (1996a). "A New Paradigm for Modeling of Complex Manufacturing Systems", *Proceedings of the First Turkish Symposium on Intelligent Manufacturing Systems*, Adapazari, Turkey. Sakarya University Press: 27-37.

Delen, D., D. Pratt, and M. Kamath (1996b). "A New Paradigm for Manufacturing Enterprise Modeling: Reusable, Multi-Tool Modeling." *Proceedings of the 1996 Winter Simulation Conference*, San Diego, CA. IEEE: 985-992.

Diesch, K. H. and E. M. Malstrom (1985). "Physical Simulator Analyzes Performance of Flexible Manufacturing System." *Industrial Engineering* **17**(6): 66-67.

Dietrich, B. L. (1991). "A Taxonomy of Discrete Manufacturing Systems." *Operations Research* **39**(6): 886-902.

Duchessi, P. and R. M. O'Keefe (1995). "Understanding Expert Systems Success and Failure." *Expert Systems with Applications* **9**(2): 123-133.

Duda, R. O. and E. H. Shortliffe (1983). "Expert System Research." *Science* **220**: 261-268.

Durkin, J. (1994). *Expert Systems: Design and Development*. Prentice Hall, Englewood Cliffs, New Jersey.

Duse, M., J. Gharpure, H. Bhuskute, M. Kamath, D. B. Pratt, and J. H. Mize (1993). "Tool-Independent Model Representation." *Proceedings of the 2nd Industrial Engineering Research Conference*. IIE: 700-704.

ElMaraghy, H. A. (1982). "Simulation and Graphical Animation of Advanced Manufacturing Systems." *Journal of Manufacturing Systems* **1**(1): 53-63.

ElMaraghy, H. A. and T. Ravi (1992). "Modern Tools for the Design, Modelling, Evaluation of Flexible Manufacturing Systems." *Robotics & Computer Integrated Manufacturing* **9**(4/5): 335-340.

Farhoodi, F. (1990). "A Knowledge Based Approach to Job-Shop Scheduling." *International Journal of Integrated Manufacturing* **3**(2): 84-95.

Fisher, E. L. and S. Y. Nof (1984). "FADES: Knowledge-Based Facility Design." *Proceedings of the 1984 Annual International Industrial Engineering Conference*. IIE: 154-162.

Ford, D. R. and B. J. Schroer (1987). "An Expert Manufacturing Simulation System." *Simulation* **48**(5): 193-200.

Forsyth, R. (1984). *Expert Systems: Principles and Case Studies*. Chapman and Hall Computing, London, NY.

Fox, M. and Y. Reddy (1982). "Knowledge Representation in Organizational Modeling and Simulation: Definition and Interpretation." *13th Annual Pittsburgh Conference on Modeling and Simulation*, Pittsburgh, PA.: 133-142.

Fox, M. S. and S. F. Smith (1984). "ISIS: A Knowledge-Based System for Factory Scheduling." *Expert Systems* **1**(1): 25-49.

Glassey, C. R. and S. Adiga (1989). "Conceptual Design of a Software Object Library for the Simulation of Semiconductor Manufacturing Systems." *Journal of Object-Oriented Programming* (Sept./Oct.): 39-43.

Glassey, C. R. and S. Adiga (1990). "Berkeley Library of Objects for Control and Simulation of Manufacturing (BLOCS/M)." *Applications of Object-Oriented Programming*. L. J. Pinson and R. S. Weiner. Reading, MA, Addison Wesley: 1-27.

Greenwell, M. (1988). *Knowledge Engineering for Expert Systems*. Halsted Press: A Division of John Wiley & Sons, New York, NY.

Goldberg, A. and D. Robson (1989). *Smalltalk-80: The Language*. Addison-Wesley, Reading, MA.

Govindaraj, T., L. F. McGinnis, C. M. Mitchell, D. A. Bodner, S. Narayanan, and U. Sreekanth (1993). "OOSIM: a Tool for Simulating Modern Manufacturing Systems." *Proceedings of the 1993 NSF Design and Manufacturing Systems Conference*. SME: 1055-1062.

Gu, P. (1992). "PML: Product Modelling Language." *Computers in Industry* **18**: 265-277.

Gupta, T. and B. K. Ghosh (1989). "A Survey of Expert Systems in Manufacturing and Process Planning." *Computers in Industry* **11**: 195-204.

Hadavi, K., M. S. Shahraray, and K. Voigt (1990). "ReDS - A Dynamic Planning, Scheduling, and Control System for Manufacturing." *Journal of Manufacturing Systems* **9**(4): 332-344.

Hadavi, K., W. Hsu, T. Chen, and C. Lee (1992). "An Architecture for Real-Time Distributed Scheduling." *AI Magazine* (Fall): 46-56.

Hayes-Roth, F., D. A. Waterman, and D. B. Lenatt (1983). *Building Expert Systems*. Addison Wesley, Reading, MA.

Hofer-Alfeis, J. J. (1992). "Knowledge-Based Systems and Knowledge Management in Product Design and Manufacturing: Experiences and Future Directions." *IFAC Intelligent Manufacturing Systems Workshop* (October): 42-45.

Hubner, M. (1988). "Knowledge-Based Configuration of Flow Lines." *International Journal of CIM* 1(4): 210-221.

Joseph, T. K. (1989). "An Expert System Advisor to Aid Goal Definition for Manufacturing System Simulation." *Advances in AI and Simulation*, San Diego, CA. SCS: 203-207.

Juran, J. M. (1988). *Juran on Planning for Quality*. Free Press, New York.

Kamath, M., D. B. Pratt, and J. H. Mize (1995). "A Comprehensive Modeling and Analysis Environment for Manufacturing Systems." *Proceedings of the 4th Industrial Engineering Research Conference*. IIE: 759-768.

Kamath M., D. Pratt, D. Delen, S. Sivaramakrishnan, B. Krishnamoorthy, and R. Fernandes (1996). "An Advanced Modeling Environment to Support Rapid Analysis and Reconfiguration of Agile Manufacturing Systems." Submitted to *IIE Transactions* special issue on Agile Manufacturing.

Kim, Y-J. (1995). "A Framework for an On-Line Diagnostic Expert System for Intelligent Manufacturing." *Expert Systems with Applications* 9(1): 55-61.

Kimemia, J. G. and S. B. Gershwin (1983). "An Algorithm for the Computer Control of Production in Flexible Manufacturing Systems." *IIE Transactions* 15(4): 353-362.

Klahr, P., W. S. Faught, and G. R. Martin (1980). "Rule Oriented Simulation." *Proceedings of the 1980 International Conference on Cybernetics and Society*. IEEE: 213-221.

Kovacs, G., I. Mezgar, and S. Kopacsi (1991). "Concurrent Design of Automated Manufacturing Systems Using Knowledge Processing Technology." *Computers in Industry* 17: 257-267.

Kovacs, G. L., I. Mezgar, S. Kopacsi, D. Gavalcova, and J. Nacsa (1994). "Application of Artificial Intelligence to Problems in Advanced Manufacturing Systems." *Computer Integrated Manufacturing Systems* 7(3): 153-160.

Kusiak, A. (1983). "Loading Models in Flexible Manufacturing Systems." *Proceedings of the VIII International Conference on Production Research*, Windsor, Canada.: 98-105.

Kusiak, A. (1987). "Group Technology." *Computers in Industry* 9: 83-91.

Kusiak, A. (1988). "Knowledge-Based Group Technology." *Expert Systems: Strategies and Solutions in Manufacturing Design and Planning*. A. Kusiak. Dearborn, MI, SME: 259-299.

Kusiak, A. and M. Chen (1988). "Expert Systems for Planning and Scheduling Manufacturing Systems." *European Journal of Operational Research* **34**: 113-130.

Kusiak, A. and S. S. Heragu (1988). "Knowledge-Based System Guides Machine Layout in FMS." *Industrial Engineering* (November): 48-53.

LeFrancois, P. and B. Montreuil (1994). "An Object-Oriented Knowledge Representation for Intelligent Control of Manufacturing Workstations." *IIE Transactions* **26**(1): 11-26.

Leung, Y. and R. Suri (1990). "Performance Evaluation of Discrete Manufacturing Systems." *IEEE Control Systems Magazine* (June): 77-86.

Liberatore, M. J. and A. C. Stylianou (1995). "Expert Support Systems for New Product Development Decision Making: A Modeling Framework and Applications." *Management Science* **41**(8): 1296-1316.

Lingzhi, L., A. C. Leong, and R. K. L. Gay (1996). "Integration of Information Model (IDEF1) with Functional Model (IDEF0) for CIM Information Systems Design." *Expert Systems with Applications* **10**(3/4): 373-380.

Lirov, Y., E. Y. Rodin, B. G. McElhaney, and L. W. Wilbur (1988). "Artificial Intelligence Modelling of Control Systems." *Simulation* **50**(1): 12-24.

Luong, L. H. S. (1992). "Applications of Expert Systems in Manufacturing : A Case Study." *Computers in Industry* **18**: 193-198.

Luong, L. H. S., F. T. S. Chan, and W. Sessomboon (1994). "Integration of Simulation and Expert Systems for the Design of Flexible Manufacturing Systems." *WORKSIM '94* (Nov. 9-11): 133-138.

Mahallingam, S. and E. C. Dunzinski (1988). "CSRL-A Tool for Building Diagnostic Expert Systems." *Manufacturing Engineering* (July): 79-82.

Majstorovic, V. D. (1990). "Expert Systems for Diagnosis and Maintenance: The State-of-the-Art." *Computers in Industry* **15**: 43-68.

Majstorovic, V. D. and V. R. Milacic (1990). "Expert Systems for Maintenance." *Computers in Industry* **15**: 83-93.

Majstorovic, V. D. and V. R. Milacic (1991). "Learning in an Expert System for Maintenance in Flexible Manufacturing Systems." *Computers in Industry* **17**: 279-285.

Maley, J. G., S. Ruiz-Mier, and J. J. Solberg (1988). "Dynamic Control in Automated Manufacturing: A Knowledge-Integrated Approach." *International Journal of Production Research* **26**: 1739-1748.

Martin, B., G. H. Subramanian, and G. J. Yaverbaum (1996). "Benefits from Expert Systems: An Explanatory Investigation." *Expert Systems with Applications* **11**(1): 53-58.

Mayrand, E., P. Lefrancois, and B. Montreuil (1993). "An Agent-Oriented Simulation Model of Manufacturing Activities." To appear in *Heuristics.*

McGraw, K. L. and K. Harbison-Briggs (1989). *Knowledge Acquisition: Principles and Guidelines.* Prentice Hall, Englewood Cliffs, New Jersey.

Medsker, L., M. Tan, and E. Turban (1995). "Knowledge Acquisition from Multiple Experts: Problems and Issues." *Expert Systems with Applications* **9**(1): 35-40.

Mellichamp, J. M. and A. F. A. Wahab (1987). "An Expert System for FMS Design." *Simulation* (May): 201-208.

Meyer, R. J. (1987). "AI and Expert Systems: In Pursuit of CIM." *CIM Technology* (February): 15-18.

Mize, J. H. and D. B. Pratt (1991). "Modeling of Integrated Manufacturing Systems Using an Object-Oriented Approach." *Proceedings of the Joint International Conference on Factory Automation and Information Management*, Limerick, Ireland. CRC Press Inc.: 250-257.

Mize, J. H., H. C. Bhuskute, D. B. Pratt, and M. Kamath (1992). "Modeling of Integrated Manufacturing Systems Using an Object-Oriented Approach." *IIE Transactions* **24**(3): 14-26.

Monostori, L., P. Bartal, and L. Zsoldos (1990). "Concept of a Knowledge Based Diagnostic System for Machine Tools and Manufacturing Cells." *Computers in Industry* **15**: 95-102.

Monostori, L., D. Barschdorff (1992). "Artificial Neural Networks in Intelligent Manufacturing." *Robotics and Computer Integrated Manufacturing* **9**: 421-437.

Montan, V. and Y. V. Reddy (1989). "An Expert System Approach to the Analysis of Discrete Event Simulations." *Advances in AI and Simulation*, San Diego, CA. SCS: 189-193.

Montreuil, B., P. Lefrancois, and S. Harvey (1993). "Organism-Oriented Models of Manufacturing Systems." To appear in *Journal of Intelligent Manufacturing*.

Narayanan, S., D. A. Bodner, and C. M. Mitchell (1992). "Object-Oriented Simulation to Support Modeling and Control of Automated Manufacturing Systems. " *Proceedings of the 1992 Western Multiconference*, San Diego, CA. SCS: 113-121.

Narayanan, S., D. A. Bodner, U. Sreekanth, T. Govindaraj, L. F. McGinnis, and C. M. Mitchell (1993). "Research in Object-Oriented Manufacturing Simulations: An Assessment of the State of the Art." *Working Paper, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA.*

Nasr, H. (1987). "A Prototype Knowledge-Based System for Material Handling Equipment Selection." *Working paper, Honeywell, Inc., Minneapolis, Minnesota.*

Oren, T. I. and B. P. Zeigler (1987). "Artificial Intelligence in Modeling and Simulation: Directions to Explore." *Simulation* (April): 131-134.

ParcPlace (1994). *VisualWorks Release 2.0 User's Guide*. ParcPlace Systems, Inc., CA.

Pratt, D. B., J. H. Mize, and M. Kamath (1993). "A Case for Bottom-Up Modeling." *Proceedings of the 2nd Industrial Engineering Research Conference.* IIE: 430-434.

Pratt, D. B., P. Farrington, C. Basnet, H. Bhuskute, M. Kamath, and J. H. Mize (1995), "The Separation of Physical, Information, and Control Elements for Facilitating Reusability in Simulation Modeling," *International Journal in Computer Simulation* **4**: 327-342.

Pritsker, A. A. B. (1986). *Introduction to Simulation and SLAM II.* Systems Publishing Corporation, New York.

Rao, H. A. and P. Gu (1995). "Expert Self-Organizing Neural Network for the Design of Cellular Manufacturing Systems." *Journal of Manufacturing Systems* **13**(5): 346-358.

Robertson, J. (1987). "Knowledge-Based CAD/CAM Puts Manufacturing Smarts to Work." *Tooling & Production* (May): 44-45.

Rolston, D. W. (1988). *Principles of Artificial Intelligence and Expert Systems Development*. McGraw-Hill Book Company, New Jersey.

Ruiz-Mier, S. and J. Talavage (1987). "A Hybrid Paradigm for Modeling of Complex Systems." *Simulation* **48**(4): 135-141.

Shannon, R., R. Mayer, and H. Adelsberger (1985). "Expert Systems and Simulation." *Simulation* **44**(6): 275-284.

Shanthikumar, J. G. and J. A. Buzacott (1981). "Open Queueing Network Models of Dynamic Job-Shops." *International Journal of Production Research* **19**: 255-266.

Shanthikumar, J. G. and R. G. Sargent (1983). "A Unifying View of Hybrid Simulation / Analytical Models and Modeling." *Operations Research* **31**(6): 1031-1052.

Shanthikumar, J. G. and K. E. Stecke (1986). "Reducing Work-In-Process Inventory in Certain Classes of Flexible Manufacturing Systems." *European Journal of Operational Research* **26**: 266-271.

Shirhatti, G. M. and M. Kamath (1995). "Real-Time Control of Manufacturing Systems: A Review of Recent Research and a New Perspective." *Technical Report #CIM-TRS-95-GS1, School of Industrial Engineering and Management, Stillwater, OK.*

Shortliffe, E. (1976). *Computer-Based Medical Consultation: MICIN.* Elsevier, New York, NY.

Sim, S. K., K. T. Yeo, and W. H. Lee (1994). "An Expert Neural Network for Dynamic Job-Shop Scheduling." *International Journal of Production Research* **32**(8): 1759-1773.

Sitte, R. and H. B. Harrison (1991). "Use of an Expert System for Quality Improvement in Flexible Manufacturing of Integrated Circuits." *Computers in Industry* **17**: 35-38.

Smith, S. F., P. S. Ow, N. Muscettola, and D. C. Matthys (1990). "An Integrated Framework for Generating and Revising Factory Schedules." *Journal of the Operational Research Society* **41**(6): 539-552.

Solberg, J. J. (1977). "A Mathematical Model of Computerized Manufacturing Systems." *Proceedings of the IV International Conference on Production Research*, Tokyo, Japan.: 102-109.

Standridge, C. R. and M. A. Centeno (1991). "Concepts for Production Modeling Systems Based on Multiple User Types." *Proceedings of the 1991 Winter Simulation Conference*. IEEE: 428-434.

Stecke, K. E. (1983). "Formulation and Solution of Nonlinear Integer Production Planning Problems for Flexible Manufacturing Systems." *Management Science* **29**(3): 273-288.

Stecke, K. E. and J. J. Solberg (1981). "Loading and Control Policies for a Flexible Manufacturing System." *International Journal of Production Research* **19**(5): 481-490.

Steffen, M. S. (1986). "A Survey of Artificial Intelligence-Based Scheduling Systems." *Proceedings of the Fall Industrial Engineering Conference*, Norcross, GA. IIE: 179-187.

Suri, R. and S. De Treville (1993). "Rapid Modeling: The Use of Queueing Models to Support Time-Based Competitive Manufacturing." *Operations Research in Production Planning and Control.* Springer-Verlag: 21-30.

Tayanithi, P., S. Manivannan, and J. Banks (1991). "A Knowledge-Based Simulation Architecture to Analyze Interruptions in a Flexible Manufacturing Systems." *Journal of Manufacturing Systems* **11**(3): 195-202.

Tello, E. R. (1989). *Object-Oriented Programming for Artificial Intelligence.* Addison-Wesley Publishing Company, Inc., Reading, MA.

Thomasma, T., Y. Mao, and O. M. Ulgen (1988). "Hierarchical, Modular Simulation Modeling in Icon-Based Simulation Program Generators of Manufacturing." *Proceedings of the 1988 Winter Simulation Conference*. IEEE: 723-730.

Tibbitts, B. R. (1993). "Flexible Simulation of a Complex Semiconductor Manufacturing Line Using a Rule-Based System." *IBM Journal of Research & Development* **37**(July): 507-521.

Tirupatikumara, S. R., R. L. Kashyap, and C. L. Moodie (1975). "Artificial Intelligence Techniques in Facility Layout Planning: The Development of an Expert System." *Technical Report #TR-ERC 86-1, School of Industrial Engineering, Purdue University, West Lafayette, IN.*

Tretheway, S. J., C. S. Hunt, and M. Court (1995). "The Experimental Frame: Conceptual Structure for Developing a Comprehensive Manufacturing Analysis Tool." *Proceedings of the 5th Industrial Engineering Research Conference*. IIE: 251-256.

Ulgen, O. M., T. Thomasma, and Y. Mao (1989). "Object Oriented Toolkits for Simulation Program Generators." *Proceedings of the 1989 Winter Simulation Conference*, Washington, DC. IEEE: 593-600.

Ulgen, O. M. and T. Thomasma (1990). "SmartSim: An Object Oriented Simulation Program Generator for Manufacturing Systems." *International Journal of Production Research* **28**(9): 1713-1730.

Wadley, H. N. G. and J. W. E. Echkart (1989). "The Intelligent Processing of Materials for Design and Manufacturing." *Journal of Operations Management* (October): 10-19.

Weinroth, J., G. Madey, and V. Shah (1992). "Enhancing Simulation for Support of JIT Management." *Expersys-92*. IITT International: 420-428.

Welgama, P. S. and P. R. Gibson (1995). "A Hybrid Knowledge Based Optimization System for Automated Selection of Materials Handling System." *Computers & Industrial Engineering* **28**(2): 205-217.

Winston, P. H. (1992). *Artificial Intelligence*. Addison-Wesley Publishing Company, Reading, MA.

XEROX (1994). *Xerox Humble Reference Manual*. Xerox Corporation, CA.

Youngblood, S. M. (1991). "Increasing Simulation and Model Utility Via Improved Interface Techniques." *Proceedings of the 1991 Summer Computer Simulation Conference*. SCS: 57-62.

Zadeh, L. and K. Fukanaka (1975). *Fuzzy Sets and Their Applications to Cognitive Decision Processes*. Academic Press, New York, NY.

Zeigler, B. P. (1987). "Hierarchical, Modular Discrete-Event Modeling in an Object-Oriented Environment." *Simulation* **49**(5): 219-230.

Zeigler, B. P. (1990). *Object-Oriented Simulation with Hierarchical, Modular Models: Intelligent Agents and Endomorphic Systems*. Academic Press, San Diego, CA.

Zeigler, B. P. (1991). "Object-Oriented Modeling and Discrete-Event Simulation." *Advances in Computers* **33**: 67-114.

Zhijun, H. and L. Kai (1990). "A New Control Method and System for FMS Job Scheduling." *International Journal of Production Research* **27**: 1603-1623.

APPENDIXES

APPENDIX A

LITERATURE REVIEW OF EXPERT SYSTEMS IN MANUFACTURING

Introduction

This appendix contains a selected set of literature reviews regarding expert system applications in manufacturing. Only the most relevant and the most recent literature cited in eight categories based on their application areas.

Scheduling of Manufacturing Systems

Scheduling has been one of the most difficult and challenging tasks for manufacturing management. Despite the advances in scheduling theory, many actual scheduling problems are still too complex to yield analytical solutions. Much of the difficulty stems from having to deal with a large set of diverse constraints and multiple objectives that are often conflicting and ill-defined. Moreover, the dynamic and stochastic nature of the environment further contributes to the complexity of the task. Since the early 1980s the AI community has investigated factory scheduling in depth and various paradigms have been presented. ISIS [Fox and Reddy 1982 and Fox and Smith 1984], a knowledge based system to schedule production, is one of them. Its main focus is on the constraints of the production system being modeled. ISIS is constraint directed in the sense that constraints are used to identify the next state and are also used to evaluate the current state. One other well known production scheduling system is called OPAL [Bansana, Bel and Dubois 1988] which is also based on constraint directed search.

Maley, Ruiz-Mier and Soolberg [1988] present a closed loop control structure for the scheduling and control of CIM systems. Real time feedback from the physical system monitors the performance of the current scheduling decisions and updates a historical knowledge base used to make future decisions by providing initial starting solutions and guiding the search efforts. Another approach to scheduling and control is reported by Zhijun and Kai [1990]. It divides the scheduling task into four sub-tasks. They are system input control, which determines the time each part enters the system, work piece routing control, which directs the parts along multiple possible routings, workstation input control, which decides the sequence in which stations process the parts in their respective buffers, and vehicle control, which determines the service and routes of automated guide vehicles. The authors believe that the control of each of these sub-tasks is an event sequence control task and cannot be managed by traditional control theory.

Aytug, Koehler and Snowdon [1994] present a dynamic scheduling system which uses a knowledge base to make decisions. The system also incorporates a learning element that is implemented using a genetic algorithm (GA). The authors run simulation experiments to test their approach and compare their approach against simple dispatching rules such as Earliest Due Date and First Come First Serve based on the "average time spent in the system" performance measure. They report a performance improvement of 12-19%.

Farhoodi [1990] presents an interactive knowledge based approach to job shop scheduling. The system, which is a hybrid knowledge based system with composite inference engines involving optimization, consists of the several functional components, such as schedule generation, user interface manager, schedule evaluation, schedule improvement, schedule repair, disturbance monitoring, and database management system. The author asserts that a hybrid approach involving artificial intelligence and heuristic algorithms is likely to provide the only realistic solution to the production scheduling.

Hadavi, Shahraray and Voigt [1990] discuss an architecture for real-time control of a manufacturing system called REDS (REquirements Driven Scheduling) in which both predictive and reactive scheduling are incorporated. REDS uses both artificial intelligence and operations research techniques. The basic architecture of the system is a recursive structure of constraint pools, where each constraint pool represents the physical constraints over an arbitrary period of time. Each period unravels into smaller periods at the next level of recursion. The system consists of four modules, the Preprocessor, Feasibility Analysis, Detailed Scheduler, and Sequencer. Hadavi et al. [1992] discuss a second version of REDS, called REDS^2 (REal-time Distributed Scheduling). The modules in REDS^2 are implemented as Planning Agents (PA), where a PA is a module that has two components; a predictive element and a reactive element. Each PA operates independently of the others. The various modules in REDS^2 are the Order Watcher, the Capacity Watcher, the Data Collector, the Shop Floor Controller, the Order Entry Handler, and the Event Handler, which coordinates all the other modules. The authors report implementations of REDS/REDS^2 in a VLSI pilot line and a PCB assembly plant.

Smith et al. [1990] present an AI based scheduler that they call OPIS (OPportunistic Intelligent Scheduler). The authors characterize their problem solving approach as "opportunistic reasoning", where each activity is consistently directed towards those actions that appear most promising in terms of furthering the current problem solving state. In the case of OPIS, this refers to an incremental scheduling methodology where characteristics of current solution constraints, such as resource contention, schedule conflicts, etc., are used to dynamically focus attention on the most critical decisions that need to be made or revised. OPIS is implemented using a blackboard architecture, wherein a set of knowledge sources (KSs) are selectively employed to generate, revise, or analyze specific components of the overall schedule.

Sim, Yeo and Lee [1994] present a neural network based expert system for dynamic job shop scheduling. They use neural networks to learn and store in the network structure the interwoven relative factors that influence the various considerations for dynamic job shop scheduling. Given sufficient realistic examples, the network can be trained to recognize how these considerations cooperate or compete in the assignment of jobs as they mutually reinforce or nullify their influences on meeting the scheduling objective. According to the authors, production demands are often cyclic in nature and if the patterns of these demands can be recognized and the system adapts to seasonal, and even sudden changes, the scheduling will be adaptive and reactive in its capability. The artificial neural network (ANN) has the ability to recognize and learn new patterns, update patterns learned and store these patterns for retrieval and problem solving. One major disadvantage of ANN is its inability to explain the factors or decisions made in arriving at the solution. Another constraint is that the high degree of interconnection among neurons in a network requires a considerable amount of time for training in the network. To overcome these weaknesses of ANNs, the authors combine neural networks with an expert system. The expert system reduces the amount of time required for training ANN. The authors carried out simulation experiments to test these concepts. The results show that for job lateness related measures the schedules generated by the expert neural network based scheduler perform better or match the performance of the dispatching rules corresponding to the scheduling factors used to train the neural network for a range of arrival rates. Monostori and Barschdorff [1992] provide an overview of the application of ANNs to scheduling of CIM systems.

Good surveys of AI based scheduling systems can be found in Steffen [1986], Kusiak and Chen [1988], Basnet and Mize [1994], and Shirhatti and Kamath [1995].

Diagnosis and Maintenance

The diagnosis and maintenance of complex manufacturing systems represents a difficult interdisciplinary engineering task. Expert systems are a good choice of software for this purpose, enabling the creation of expertise for specific maintenance efforts. Majstorovic [1990] classifies expert system applications for diagnosis into five categories:

1) Failure diagnosis and corrective maintenance technology on different classes of mechanical systems.
2) Diagnosis of electronic components and systems.
3) Expert systems for the supervision and monitoring of the condition of mechanical systems during continuous processes.
4) Expert systems for maintenance planning and control of complex mechanical systems.
5) Expert systems for integrated maintenance modeling of mechanical systems.

EXMAS [Majstorovic 1990; Majstorovic and Milacic 1990; and Majstorovic and Milacic 1991] is a typical representative of the fifth group listed above. According to the authors EXMAS is a software product that satisfies the need of complex engineering systems with a very complex structure from the aspect of diagnosis and maintenance.

Mahallingam and Duzinski [1988] present a tool called CSRL for building diagnostic expert systems. CSRL is a task specific tool for building expert systems for the generic task of classification. It provides programming constructs to organize knowledge and to encode control strategies for the expert system to perform its classification problem solving. Weldex, Romad, and Corex are three examples of diagnosis expert systems created using CSRL.

Monostori, Bartal and Zsoldos [1990] present a hierarchical monitoring diagnostic expert system structure for manufacturing cells and systems. This kind of structure is to be gradually implemented and is expected to be a contribution to making this complex material and data processing system more understandable and manageable for human beings.

Luong [1992] presents his experiences related to the development of an expert system shell and a knowledge base for fault diagnosis in numerically controlled machine tools. Particular emphasis is placed on the implementation strategy and the difficulties encountered during the course of the project.

Kim [1995] presents a knowledge based framework for performing sensor validation for diagnosis in manufacturing processes. The proposed sensor validation system consists of both algorithmic and heuristic modules, including both qualitative and quantitative approaches. The author believes that the architecture and the techniques of the study can be applied to any system in which the degree of validity of sensor readings is a major factor in determining the accuracy of the diagnosis and the usefulness of the resulting corrective recommendations.

Although manufacturing productivity could benefit from implementation of diagnostic and maintenance expert systems, the costs of developing these systems limit their widespread application. To alleviate this problem, Day and Rostosky [1994] developed a method that automatically generates rules from a PLC (Programmable Logic Controllers) program and facts from a PLC data table.

Expert systems not only help companies diagnose equipment failures, often predicting them, but also recommend repair strategies. WaXpert, developed with Texas Instruments' Personal Consultant expert system, is one that diagnoses and suggests remedies for a variety of wax-casting problems [Bose 1988].

Expert systems also lend themselves to preventive maintenance systems in which system uptime and availability are crucial. Computer manufacturer NCR Corporation, for example, has designed and developed the ESPm expert system to monitor mainframes in the field, analyze error logs, and suggest preventive maintenance procedures before a computer fails [Bose 1988]. Billatos and Tseng [1991] present a knowledge-based optimization structure for intelligent machining in which tool failures are anticipated through on-line direct measurement capability. Tayanithi, Manivanan and Banks [1991] present a knowledge-based simulation architecture to analyze interruptions in a flexible manufacturing system. This architecture reduces the disruptions due to interruptions by finding the best policy from the feasible alternative set in a reasonable amount of time. This is accomplished by novel features such as nested databases and dual blackboards.

Facility Design

Conventional methods for facility design are often challenged by the increased complexity and timeliness needs of decision making in a flexible, automated manufacturing environment. Although these methods can still be most useful, a new approach needs to be developed in which these methods can be combined with needed data and decision rules by a higher level system.

FADES [Fisher and Nof 1984] is an expert system which has been designed for solving the facility design problem, selecting equipment, and performing economic analysis. It consists of a knowledge base, a PROLOG interpreter, and a database management system. The database consists of economic models, algorithms, and rules for selecting equipment, developing a relationship rating between facilities, selecting and invoking the appropriate algorithm. The knowledge is represented using first-order predicate logic.

Kusiak and Heragu [1988] present a knowledge-based system for machine layout (KBML) in an automated manufacturing environment. KBML combines the optimization and expert system approaches and considers quantitative as well as qualitative factors while solving a machine layout problem. The knowledge base in KBML consists of 35 rules. The system is coded in Common LISP and implemented on a mid-size mainframe machine.

IFLAPS [Tirupatikumara, Kashyap and Moodie 1975] consists of two basic modules; an expert system module and a syntactic pattern recognition module. Both modules can generate solutions for the facility layout problem. The expert system module uses three types of assignment rules to assign facilities to their respective sites. The rules of the first type assign a facility *i* to a site *j*, if the resource required by facility *i* is available at site *j*. The rules of the second type assign facilities with a high flow value between them to adjacent sites. The rules of the third type assign facilities which should not be located adjacently to non-adjacent sites. The pattern recognition module consists of production rules which determine the facility to be assigned first in the floor plan. The other facilities are added to sites in the floor plan such that hazardous facilities are assigned to their corresponding sites and non-hazardous facilities are assigned based on their interactions with previously assigned facilities.

Quality Control

Though quality control is generally not an explicit development goal for expert systems in manufacturing, a number of manufacturers have realized quality improvements from the use of expert systems. Bird [1991] proposes an object-based expert system architecture to take a proactive orientation toward quality. He developed the system from the principles of two proven quality management methods -- Quality Function Deployment [Akao 1990] and Juran Quality Planning [Jurran 1988]. According to the author, the utilization of object-based techniques for implementing these methods has a number of advantages. First, the object-based approach allows the independence of multiple knowledge bases to be maintained, but still provides for integration of the separate knowledge bases into a seamless system. Second, and potentially more important in the long run, an object-based approach offers integration with object-based approaches to creating integrated manufacturing environments.

Cesarone [1991] presents a research effort directed toward one of the links required for fully automated manufacturing systems, that of quality control judgments of ongoing processes. According to Cesarone, the use of expert system programming, combined with in-process metrology and system integration, allows the factory to be more fully automated and computer-integrated, resulting in higher process precision and lower production errors. QES, an in-process quality control expert system, is an intermediate result of this research effort.

CHIMES [Sitte and Harrison 1991] is an expert system developed to aid in the recovery actions of a flexible manufacturing system for integrated circuit wafer production. The underlying philosophy is to provide a mechanism, whereby potential faults, which are introduced in one manufacturing step on the wafers, are corrected by a compensating action in a later manufacturing step. It is the purpose of the expert system to recommend which recovery action is necessary under the given conditions of the history of wafer manufacturing.

Dagli [1988] presents the structure of a prototype expert system which is based on an integrated expert systems/operations research approach in the quality control area. The expert system provides support to the process or the quality control engineer in the selection of the proper type of control charts to use in tracing the state of the process. The functionality and operating procedures of this system are also demonstrated in the paper.

Manufacturing System Design

Flexible manufacturing systems (FMS) built up from smaller, complex units, i.e., from cells (FMC), are becoming typical examples of flexible systems [Kovacs, Mezgar and Kopacsi 1991]. The design and the operation of these cells require new methods to capture all the embedded benefits of the sophisticated and expensive elements installed for production purposes.

Kovacs, Mezgar and Kopacsi [1991] present a prototype design system that makes use of different knowledge based tools and techniques to configure, or reconfigure manufacturing cells taking into consideration technological plans. The design of a manufacturing cell is supported by the CS-PROLOG based ALL-EX expert system shell. The results of the design are forwarded to an AutoCAD based layout design program, which results in a proposed layout of all equipment in the cell. The authors claim that the evaluation of the early experimental results shows promise in leading toward the final product called the COOPERATOR system. Mellichamp and Wahab [1987] present an expert system which can analyze the output from an FMS simulation model, determine whether operational and financial objectives were met, identify design deficiencies, and propose designs which could overcome identified deficiencies.

Rao and Gu [1995] present a multi-layered hybrid neural network to incorporate some of the constraints and objectives required in the design of manufacturing systems into the design process. The neural network, which is constraint-bound, is structured to include practical limitations such as duplicate machine availability and machine capacity. The expert system, which is interactive, takes its input from the neural network and uses alternate process plans to reassign any exceptional parts that may occur as a result of the constraint imposition during the initial design. Thus the hybrid neural net-expert system technique gives an added flexibility to the design approach by facilitating the incorporation of multiple constraints and objectives.

Group technology takes advantage of similarities of parts and machines in a manufacturing system. Kusiak [1987] presents several classification and clustering approaches to group technology in manufacturing systems. In addition to his presentation of mathematical programming formulations for the clustering problem, applications of expert systems to group technology are also discussed. KBGT [Kusiak 1988] is a knowledge-based system for solving the group technology problem. KBGT considers alternative process plans and multiple machines and takes advantage of the developments in expert systems and optimization.

Meyer [1987] discusses the issues related to the potential benefits of expert systems in pursuit of CIM. Robertson [1987] places extra emphasis on the necessity of the integration of CAD/CAM through knowledge-based systems. Wadley and Echkart [1989] present the intelligent processing of materials (IPM) which is a novel method of improving the processing and quality control of advanced materials and products through knowledge-based manufacturing system design.

Material Handling Equipment Selection

Equipment selection in material handling is a very complex and tedious task, due to the fact that there are so many types of equipment to select from and there are so many attributes that guide the selection procedure. These attributes include the characteristics of the material to be handled, level of manufacturing automation, and space and time constraints. Human expertise and intuition play a major role in selecting the proper equipment. The expertise required for selecting material handling equipment can be augmented by developing an expert system which provides the link between the broad scope of available equipment and the specific manufacturing requirements.

EMHES [Attia et al. 1992a] is an expert system prototype which incorporates knowledge based systems technology and database management techniques to assist the planning engineer in selecting cost effective material handling equipment. Through a structured and comprehensive examination of the characteristics of the required handling tasks and the available equipment selection options, EMHES provides a basis for material handler equipment selection decisions in manufacturing.

Welgama and Gibson [1995] present a system that selects the optimum material handling equipment and assigns optimum moves, extending previous concepts in analytical methods and expert systems. The optimization model considers minimization of both the cost and total aisle space as objectives. The knowledge base consists of rules to determine feasibility of using particular material handling equipment for a given move. The methodology employs the systems approach in selecting the optimum material handling system for a given set of moves while considering many practical aspects associated with the material handling system selection process.

Nasr [1987] presents a prototype knowledge-based system, called SEMH, for material handling equipment selection. This system departs from the classical nature of database systems, it relies heavily on artificial intelligence problem solving methods.

According to the author, the equipment selection problem could vary from one material handling domain to another, with this in mind, the proposed prototype is designed to be task-independent.

Process Planning

Process planning is a function that establishes which machining processes and parameters are to be used to convert a piece part from its initial form to a finished product as predetermined from an engineering drawing. Traditionally, this task has been performed manually and requires a considerable amount of human expertise. Thus it has been one of the most potentially suitable areas for expert system applications.

According to Gupta and Ghosh [1989], one important reason for the need for an automated process planning system is the ultimate goal of completely constructing an automated factory. In order to make this dream come true, an integration between the design and the manufacturing system is needed. A process planning system can serve as a bridge between these two systems.

It is evident that the use of AI techniques in process planning is actually a two step process: (1) integration of computer-aided design (CAD) and computer-aided manufacturing (CAM), and (2) development of an expert process planning system which takes full advantage of the integration between CAD and CAM.

Attia at al. [1992b] outline the application of expert systems and decision support systems to the development of a process planning model. Concepts from knowledge based systems and artificial intelligence based planning techniques appear to be particularly suitable for generating process plans. A framework that integrates expert systems technology with database management systems is proposed to assist manufacturing engineers in planning and organizing the stages and activities of process planning for a CIM environment.

A review of several expert system applications (GARI, TOM, PROPLAN, EXCAP, SIPP, PWA, Intelligent Process Planner, HI-MAPP) can be found in Gupta and Ghosh [1989].

Product Design

Another application area for expert systems in manufacturing is product design. Liberatore and Stylianou [1995] present a modeling framework that merges knowledge-based expert systems and decision support systems with management science methods for project evaluation in new product design/development. According to the authors, one of the most important contributions of this research is the demonstration of successfully merging management science techniques with knowledge based and decision support systems. In addition to a series of related case studies, that have successfully applied the proposed framework. Several potential further research directions are also identified.

Gu [1992] presents the development of a design modeling language called product modeling language based on analysis of CAD systems, modeling, and expert systems. The language is a unique representation of product, solid and features, and an effective communication method for linking design and various manufacturing activities. The language has been implemented and integrated with AutoSolid and an expert system for part assignments.

Hofer-Alfeis [1992] presents his experiences and future directions in the fields of knowledge-based systems and knowledge management in product design. The application of knowledge-based systems in the product generation process within SIEMENS Corp. in Germany is described with examples of running systems and of work in progress. According to the author, the increase of knowledge-based systems applications has slowed down. One of the reason is the difficulty of knowledge acquisition.

APPENDIX B

SUMMARY OF HUMBLE

Introduction

This appendix contains a summary of an object oriented expert system shell called HUMBLE which is used to develop EFES in this research.

HUMBLE is an object-oriented expert system shell written in Smalltalk that is commercially available from Xerox Special Information Systems. It is provided in several formats for most implementation of Smalltalk-80. HUMBLE combines rules in both forward and backward chaining with object representation, message passing, and reasoning about objects. The rule syntax used in HUMBLE is a modified version of the Smalltalk syntax. Unlike many rule languages, the If *condition* THAN *action* ELSE *action* construct of procedural languages is retained rather than just the If *condition* THAN *action* form.

In HUMBLE, rules operate on entities. Entities are an important type of object that have a specific representation. In applications, they are categorized into a number of different types which are defined by the developer or knowledge engineer. Entities differ from one another by their parameters or slots. Perhaps even more significantly, entities can be composite such that they can hold other entities within themselves.

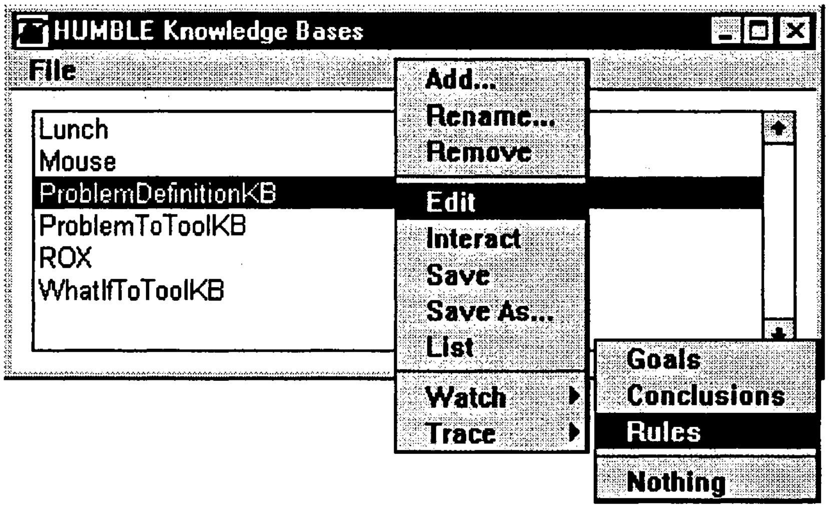
The structure of an entity tree is important to how rules are written because rules written for a given type of entity have a restriction in their access to parameters of other entities in the hierarchy. They are only permitted access to entities of that type or those higher up in the tree. This amounts to a kind of inheritance up entity trees that is distinct from inheritance by types of entities.

One of the superior features that exists in HUMBLE is the class called Interrogator. Interrogator objects are not just used for posing questions to the user. A specialization of this class can be created for seeking required information from any type of external source.

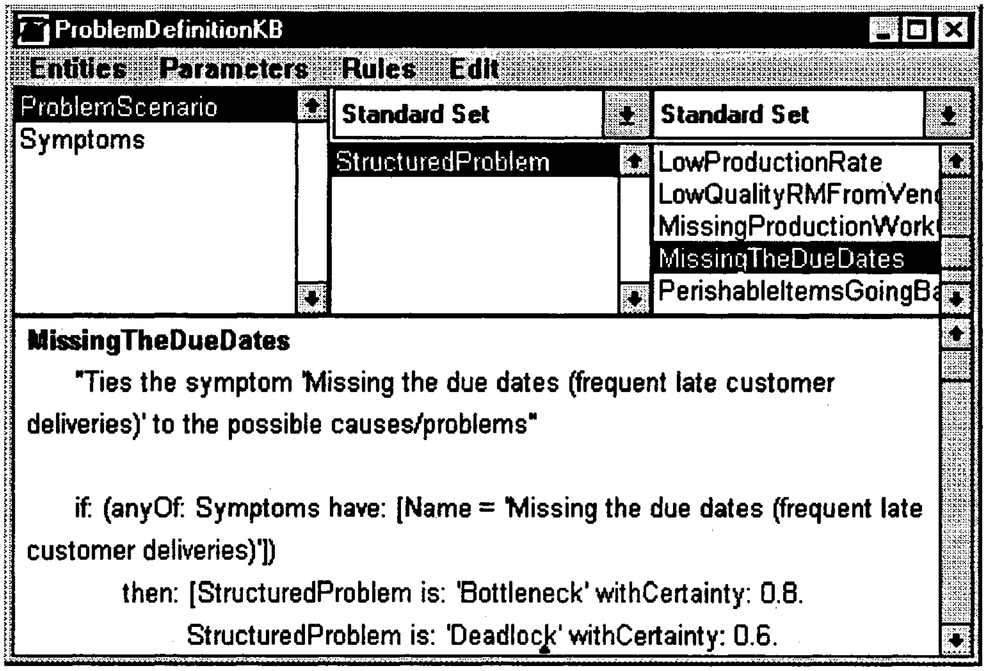
In short, HUMBLE is an expert system shell that takes maximum advantage of the object-oriented paradigm. The use of entity trees provides a built-in facility for composite objects. The fact that Smalltalk methods can be called easily by rules means that very sophisticated expert systems can be constructed. Because the procedural part of applications is written in an object-oriented language like Smalltalk, most user interfaces and other routines can be written generically. In this way the amount of code that needs to be written to suite a particular application to its installed site is kept to an absolute minimum.

Screen Printouts

In this section a few HUMBLE screen printouts are provided.



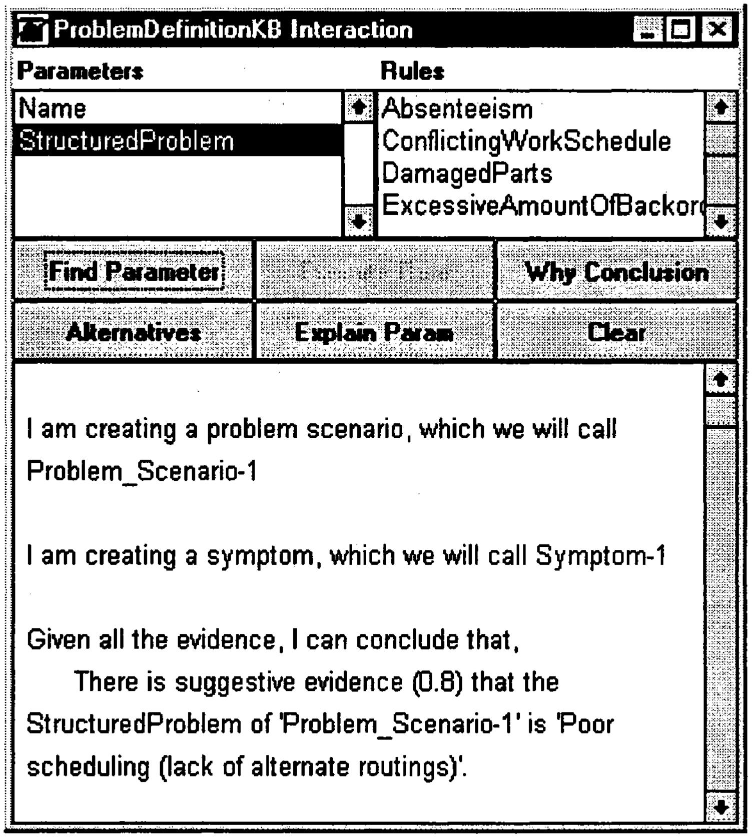
HUMBLE Manager Interface



```
MissingTheDueDates
    "Ties the symptom 'Missing the due dates (frequent late customer
deliveries)' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Missing the due dates (frequent late
customer deliveries)'])
        then: [StructuredProblem is: 'Bottleneck' withCertainty: 0.8.
            StructuredProblem is: 'Deadlock' withCertainty: 0.6.
```

HUMBLE Editor Interface



I am creating a problem scenario, which we will call Problem_Scenario-1

I am creating a symptom, which we will call Symptom-1

Given all the evidence, I can conclude that,
    There is suggestive evidence (0.8) that the StructuredProblem of 'Problem_Scenario-1' is 'Poor scheduling (lack of alternate routings)'.

HUMBLE Interaction Interface

APPENDIX C

CONTACT LETTERS AND SURVEY FORMS

Introduction

This appendix contains the letters and the survey forms used in the knowledge acquisition process of this research. Following is a list of contact letter(s) and survey forms sent to the experts and presented in this appendix.

- Initial contact letter (or email message) sent to the potential experts.
- Knowledge bases data collection survey form.
- Secondary factors data collection survey form.
- Consent form.

Initial Contact Letter (or Email Message) to the Potential Contributors (Experts)

Dear ....

Dr. _______________ recommended I contact you. I would like to request some information from you based on your experience in the field of manufacturing systems modeling.

My name is Dursun Delen. I am a Ph.D. candidate in the School of Industrial Engineering and Management at Oklahoma State University. I am also a research assistant in the Center for Computer Integrated Manufacturing working on an NSF funded research project titled "A Modeling Environment to Support Rapid Reconfiguration of Manufacturing Systems". My dissertation deals with developing an advanced, readily accessible modeling environment for manufacturing systems through the application of expert systems to the problem definition and solver selection procedures within an overall modeling process. Such a modeling environment is expected to eliminate/minimize the constraints related to reusability and accessibility found in traditional modeling paradigms.

Constructing an expert system requires collection of expert knowledge from a variety of sources of which human experts are the most important. You are an expert in the field of manufacturing systems modeling.

I would appreciate it if you would agree to contribute to my research through sharing your expert knowledge in manufacturing system modeling. Information provided is not tied directly to your job or company but rather to your overall experience within manufacturing systems. All data collected will remain completely anonymous. If you agree to contribute, you will receive a written document summarizing the nature of the problem domain for which I am seeking your expertise as well as a few definitions related to expert systems. Thank you for your consideration.

Sincerely,

Dursun Delen
School of Industrial Engineering & Management, OSU
322 Engineering North
Stillwater OK 74078
Tel: (405) 744-7202 or (405) 744-5968 (office)
(405) 744-2059 (home)
Email: delen@ceatlabs.okstate.edu

Research Advisor
Dr. David B. Pratt
Tel: (405) 744-6055
Email: dpratt@okway.okstate.edu

Survey Form (Sent to the Experts)

Dear Dr. ....:

I have talked to you briefly on the phone about my research and my need to acquire knowledge on problem domain and tool domain in a manufacturing context.

My research deals with conceptualizing, designing, and implementing a new framework for modeling of manufacturing systems. This new framework is expected to eliminate/minimize some of the shortcomings found in the traditional modeling paradigms by improving reusability and accessibility. Eliminating/minimizing these shortcomings is expected to increase the usage of modeling by manufacturing managers in their decision making processes [Figure 1]. Following is a brief explanations of these two shortcomings:

1. Lack of Reusability: Models have been viewed as single purpose, throw-away efforts. A model is built from scratch to address a particular problem or question, and then it is often discarded. When a new problem is encountered, a new model is generated from scratch even though it may include elements contained in earlier models. This single-use, throw-away mentality of modeling is obviously very expensive, time consuming and wasteful.
2. Lack of Accessibility: Access to models by non-modeling specialists is typically limited. Direct use of a model is usually limited to a few experts, people who have spent a great deal of time learning about the model and how it works. Anyone wishing to obtain results from the model must ask for help from one of the experts to input the required parameters into the model, run the model, and then interpret the results.

The solution to these shortcomings lies in taking advantage of recent developments in several related areas. These areas include Object Oriented Programming, Object Oriented Modeling, AI/Expert Systems, Knowledge Engineering, Software Engineering, and Modeling Formalisms. OOP and Modeling Formalisms make it possible to build highly reusable, general purpose software components whereas AI/Expert Systems and Knowledge Engineering help to build more user friendly (readily accessible) software environments.

Previous research in the Center for Computer Integrated Manufacturing (CCIM) at Oklahoma State University concentrated on the first shortcoming, namely reusability. CCIM researchers conceptualized and developed a new modeling environment through which a tool independent model (base model) of a specific manufacturing system can be constructed by using highly reusable, plug compatible modeling primitives. The *base model* is an abstraction of a real world manufacturing system in the richest possible way. Once a base model is constructed a tool specific execution model (such as simulation or queueing) can be extracted from it using specifically developed translators. Execution models, through which analysis can be performed, are solver specific representations of the base model.

My research aims to create the framework to capture the necessary expertise to define a structured problem from a set of symptoms (observable facts) in a manufacturing setting and then to select the most suitable analysis tool to study the problem. This framework will be captured in an expert system called Experimental Frame Expert System (EFES) [Figure 2]. There are two main categories of expertise required to accomplish this task. First, expertise is required to define a structured problem, and second, expertise is required to select a solver.

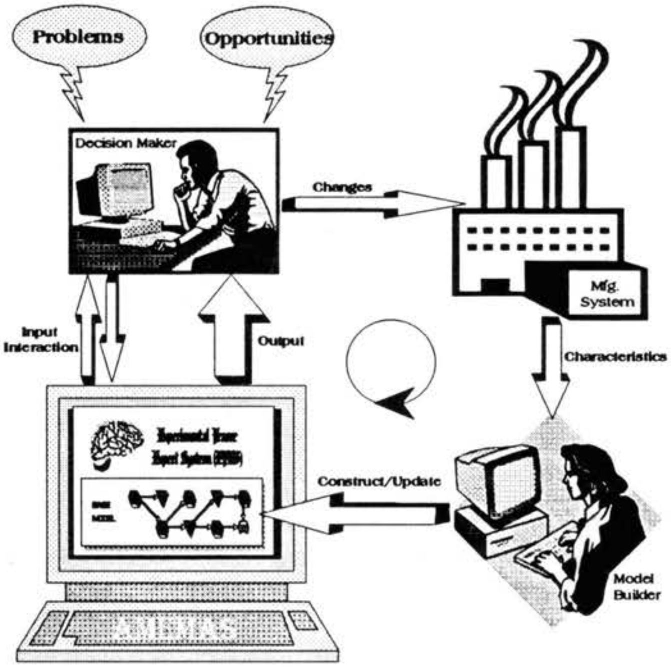


Figure 1. Role of AMEMAS (Advanced Modeling Environment for MAnufacturing Systems) in decision making processes for manufacturing systems.

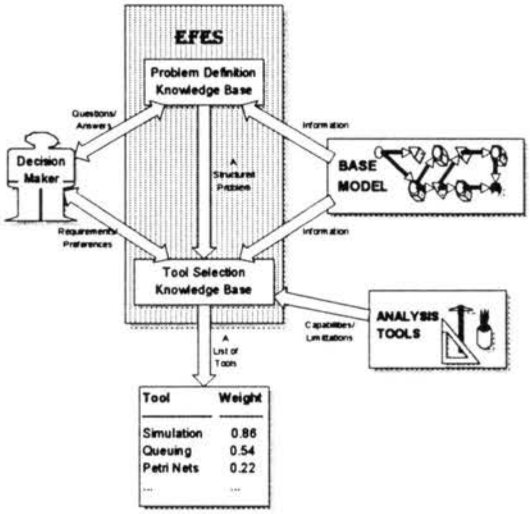


Figure 2. Consultation and information usage processes in EFES.

Following is a survey to be filled in by you - an expert in the fields of manufacturing systems and modeling. *You are expected to base your responses on your own personal experiences and biases - there are no right or wrong answers.*

The survey is divided into five sections. The first section asks experts to list as many problems and symptoms as possible. The second section asks experts to map (match) these symptoms to the problems. The third section asks experts to map (match) the problems to the analysis tools. The fourth section requires experts to list a number of what-if questions whereas the fifth section asks experts to map (match) these what-if questions to the given set of tools. Experts are to fill as much information as possible within their level of expertise. In other words, if the expertise is specifically limited to manufacturing problems, then the expert should fill in the first, second, forth, and fifth sections.

Your contribution will help to build the problem domain and tool domain knowledge bases. Once you have completed the questionnaire please mail it to the address listed below in the return envelope provided. I will contact you after I received the questionnaire to acknowledge its receipt and ask follow-up questions.

The last page of this package is a consent form. The university requires me to ask you sign this form before I use the information you provided to me.

If you have any questions on any aspects of this package please feel free to call me or send me an email. I will telephone you after a few days from the date I mail this survey package to make sure you received it and to answer any questions you might have at that time. A quick response to this survey will be greatly appreciated.

Sincerely,

Dursun Delen
School of Industrial Engineering and Management
Oklahoma State University
322 Engineering North
Stillwater, OK 74078
Tel: (405) 744-5968 or (405) 744-7202 (Office)
(405) 744-2059 (Home)
Email: delen@ceatlabs.okstate.edu

Research Advisor: Dr. David B. Pratt
School of Industrial Engineering and Management
Oklahoma State University
322 Engineering North
Stillwater, OK 74078
Tel: (405) 744-9140 or (405) 744-6055
Email: dpratt@okway.okstate.edu

Section 1. Fill in the following table with symptoms and problems. *Symptoms* can be thought of "observable facts" which are not necessarily structured such as "missing due dates" whereas *problems* are more structured such as "bottleneck machine". The possibility of a symptom being a problem at the same time is viable.

Symptoms	Problems
s1. High Work-In-Process Inventory	p1. Bottleneck
s2. Low Production Rate	p2. Deadlock
s3. Missing Due Dates	p3.
s4.	p4.
s5.	p5.
s6.	p6.
s7.	p7.
s8.	p8.
s9.	p9.
s10.	p10.
s11.	p11.
s12.	p12.
s13.	p13.
s14.	p14.
s15.	p15.
s16.	p16.
s17.	p17.
s18.	p18.
s19.	p19.
s20.	p20.
s21.	p21.
s22.	p22.
s23.	p23.
s24.	p24.
s25.	p25.
s26.	p26.
s27.	p27.
s28.	p28.
s29.	p29.
s30.	p30.
s31.	p31.
s32.	p32.
s33.	p33.
s34.	p34.
s35.	p35.

Section 2. Map (match) the symptoms to the problem in the following table (you can map one or more symptoms to one or more problems) . A symptom can be an input to more than one problem. Experts also need to estimate a confidence parameter in mapping symptoms to the problems. For example if you are certain that a certain set of symptoms are an indication to a certain problem then your confidence level for this instance might be close to 100 where 100 means full confidence. If you make such mapping most of the time but not always then your confidence level for this instance might be around 80s.

Example: The first line (given as an example) in the following table means: symptom *s1* and *s3* indicates the Problem *p2* with confidence level of 80%.

Symptoms	**Problem**	**Confidence Level (%)**
s1, s3	p1	80

Section 3. Map (match) previously listed problems to the given sample set of tools with appropriate confidence levels. Fell free to add individual tools or tool combinations based on your experiences.

Tool set:

t1. Simulation analysis
t2. Queueing analysis
t3. Petri net analysis (known to be superior to simulation and queueing for qualitative measures such as deadlock detection and reachability analysis)
t4. Queueing analysis followed by simulation
t5. Search-base optimization through simulation (multiple simulation runs based on a designed experiment)

The first line (given as an example) in the following table means: problem *p1* can be addressed with the analysis tool *t1* with confidence level of 70%.

Problem	**Tool**	**Confidence Level (%)**
p1	t1	70

Section 4. Fill in the following table with what-if questions. What-if questions are grouped into three categories: physical, informational, and control/decisional. The following table contains an initial lists of what-if questions. Experts are asked to expand and modify the given lists. If you are not sure about which category should a specific what-if question belongs to, then list it in the table given under the heading of "what if it does not fit well in these three category".

Physical	Informational	Control/Decisional
ph1. Add/remove a work station	in1. Change bill of material (add a new part/component)	co1. Change plant inventory policy
ph2. Add/remove an assembly station	in2. Change routing (add/remove an alternate routing)	co2. Change material handler allocation logic
ph3. Add/remove a material handler	in3.	co3. Change primary part selection logic
ph4. Add/remove a stock room or buffer	in4.	co4.
ph5. Change plant layout	in5.	co5.
ph6.	in6.	co6.
ph7.	in7.	co7.
ph8.	in8.	co8.
ph9.	in9.	co9.
ph10.	in10.	co10.
ph11.	in11.	co11.
ph12.	in12.	co12.
ph13.	in13.	co13.
ph14.	in14.	co14.
ph15.	in15.	co15.

What if it does not fit well in these three category

wh1.	wh9.
wh2.	wh10.
wh3.	wh11.
wh4.	wh12.
wh5.	wh13.
wh6.	wh14.
wh7.	wh15.
wh8.	wh16.

Section 5. Map (match) a what-if question to a tool (tool set given in section 3) with appropriate confidence level in the following table.

Example: The first line (given as an example) in the following table means what-if question *ph1* can be addresses by using tool *t1* with confidence level of 90%.

What-If Question	Tool	Confidence Level (%)
ph1	t1	90
ph1	t2	50

Secondary Factors Data Collection Survey Form

Dear Expert:

Following is a table to be filled in by you based on your expertise (knowledge and/or experience) in the field of modeling of manufacturing systems. When filling this table, you are expected to consider each sub-factor independent from the others. For each column, first determine the tools that has an increase or decrease on the preference level as the factor value increases. Such increase and decrease can be symbolized by up arrow or down arrow respectively. The next step is to rank the same direction arrows among themselves. The final result should look like the example table given at the bottom of this page.

Table to be filled in by the expert.

	FACTORS		
TOOLS	**Mgf-Sys-Compl.**	**Time-to-Solution**	**Level-of-Detail**
Simulation			
Queueing			
Petri Nets			
Queueing + Sim.			
SB Opt. w/Sim.			

Table with a hypothetical example.

	FACTORS		
TOOLS	**Mgf-Sys-Compl.**	**Time-to-Solution**	**Level-of-Detail**
Simulation	↑ 1	↓ 2	↑ 2
Queueing	↓ 1	↑ 1	↓ 1
Petri Nets	↓ 2	↑ 2	↓ 2
Queueing + Sim.	↓ 3	↑ 3	↑ 3
SB Opt. w/Sim.	↑ 2	↓ 1	↑ 1

Sincerely,

Dursun Delen

CONSENT FORM

"I, ________________________________, hereby authorize Dursun Delen to use the information collected from interactions with me as part of his data collection process for his dissertation research titled "Role of Expert Systems in an Advanced Modeling Environment for Manufacturing Systems."

The purpose of the data collection is to construct an expert system that can assist model users (decision makers) in their problem definition and solver selection processes in an overall modeling environment for manufacturing systems. None of the experts will be identified with their specific responses. However, names of experts will be mentioned as sources of information used in construction of the expert system.

I may contact Dursun Delen at telephone numbers *(405) 744-7202* or *(405) 744-2059* or email address *delen@master.ceat.okstate.edu* if I have any questions. I may also contact Jennifer Moore at telephone number *(405) 744-5700* regarding the confidentiality of my rights.

I have read and fully understand the consent form. I sign it freely and voluntarily. A copy has been given to me.

Date: ____________________ Time: ____________

Signed: ____

Signature of Expert

I certified that I have personally explained all elements of this form as well as the purpose and procedures for the data collection process to the expert before requesting him/her to sign this form.

Signed: ____

Signature of the Project Director

APPENDIX D

RESULTS OF THE KNOWLEDGE ACQUISITION

Introduction

This appendix contains the results of the knowledge acquisition process in the form of tables. In specific, this appendix contains the complete versions of the following tables.

- List of symptoms.
- List of problems.
- List of what-if questions.
- List of mappings of the symptoms to the problems.
- List of mappings of the structured problems to the tools.
- List of mappings of the what-if questions to the tools.

LIST OF SYMPTOMS

Number	Symptom Definition
	Inventory related
s1	Inaccurate inventory records (mismatch between actual and inventory numbers)
s2	Stockout of some RM items and/or high level of some other RM items
s3	High finished goods inventory on some items and stockout on others
s4	High WIP inventory (overall)
s5	High WIP in selected work centers
s6	Frequently running out of storage space
s7	Perishable items going bad
s8	Late deliveries from vendors
s9	Low quality RM from vendors
	Customer related
s10	Low customer evaluation
s11	Excessive number of customer returns (warranty redemption)
s12	Frequent customer order cancellations
s13	Frequent changes on customer orders
s14	Excessive customer complaints
s15	Loss of demand
s16	Excessive amount of backorders
	Employee related
s17	Absenteeism
s18	Worker tardiness
s19	High workmen compensation
s20	High level of injuries and deaths
s21	High level of frustration
s22	Slow learning curve
s23	High level of demotivation towards the work and the company
	Production related
s24	Excessive waiting time for secondary resources (tools, pallets, fixtures, etc.)
s25	Long waiting times for material handlers
s26	Long waiting times for machine tools
s27	Missing the due dates (frequent late customer deliveries)
s28	Low production rate
s29	Long production flow times
s30	Excessive overtime
s31	Jobs with excessive tardiness
s32	High reject/scrap rate
s33	High rework rate
s34	Damaged parts
s35	High variable work loads
s36	High equipment idle times
s37	Frequent breakdowns/malfunctions
s38	Excessive setup times
s39	Excessive material movement
s40	Excessive sub-contracting
s41	Excessive reliance on expediting (too many "hot" jobs)
s42	Incomplete production orders

Table continues on the next page...

Continued from the previous page...

s43	Production time ≠ Predicted time
s44	Excessive changes to MPS
s45	Long customer order response time
s46	Conflicting work schedule
s47	Resource conflicts
s48	Poor flow of materials
s49	Missing production work orders
s50	Too many jobs at top priority
s51	Too many unplanned activities (due to poor planning)

LIST OF PROBLEMS

Number	Problem Definition
p1	Poor employee discipline
p2	Lack of training, know-how
p3	Lack of employees with required level of education
p4	Inflexible work force
p5	Inadequate reward/incentive system
p6	Failing to comply with OSHA, EPA regulations
p7	Poor safety measures
p8	Poor corporate culture
p9	Poor (bad) ergonomics
p10	Insufficient work force
p11	Excessive work force
p12	Theft
p13	Poor inventory policies
p14	Poor demand forecasting
p15	Poor storage design
p16	Unreliable vendors
p17	Lack of backup vendors
p18	Low RM quality
p19	Bottleneck
p20	Deadlock
p21	Highly unreliable machines
p22	Inadequate maintenance
p23	Improper batch sizes (too small or too large)
p24	Low quality
p25	Tool wear
p26	Low customer service
p27	Inadequate value/price relation
p28	Long lead times
p29	Inadequate MH allocation policies
p30	Insufficient MH capacity
p31	Outmoded MH equipment (in terms of speed, capacity, and quality)
p32	Poor layout design
p33	Insufficient capacity of resources
p34	Poor resource allocation policies (strategies)
p35	Poor due date settings procedures
p36	Poor production planning
p37	Inaccurate time standards
p38	Inaccurate/missing process plans
p39	Lack of backup plans
p40	Lack of alternate plans
p41	Poor scheduling (lack of alternate routings)
p42	Usage of inappropriate tools and fixtures
p43	Excessive machine capacity
p44	Excessive material handler capacity
p45	Highly variable demand
p46	Lack of demand
p47	Shrinking market (overall)

Table continues on the next page...

Continued from the previous page...

p48	Outmoded philosophies Push mentality Sequential design process Focus on labor efficiency and machine utilization Management with objectives Vertical organizational charts etc.
p49	Inaccurate blueprints
p50	Poor part design (complex products)
p51	Poor job instructions
p52	Obsolete Technology
p53	Slow decision making
p54	Wrong marketing strategy
p55	Poor record keeping
p56	Poor benchmarking
p57	Incorrect BOMs
p58	Inaccurate routings
p59	Improper performance measurement system
p60	Poor information management system
p61	Poor planing by customers

LIST OF WHAT-IF QUESTIONS

Number	What-If Question
	Related to the Physical Objects
w1	Add/remove a work station
w2	Add/remove an assembly station
w3	Add/remove a material handler
w4	Add/remove a stock room
w5	Add/remove a product from the product-mix
w6	Change plant layout (add/remove an aisle)
w7	Add/remove a manufacturing line
w8	Add/remove tools, fixtures, pallets, etc.
w9	Change equipment reliability parameters (add/remove a maintenance crew)
w10	Add/remove operators
w11	Change operator capabilities (cross-training)
w12	Change worker assignments
w13	Change product design
w14	Change # of work shifts (for some stations)
w15	Change plant location
w16	Change buffer capacities
	Related to the Informational Objects
w17	Change bill of materials (add/remove a component)
w18	Change routing (add/remove an alternate routing)
w19	Combine operations
w20	Change demand for the product mix
w21	Change lot sizes
w22	Change product versions
w23	Change sampling rates
w24	Change operation sequence
w25	Change setup/processing times
w26	Change defect/scrap rate
	Related to the Control/Decisional Objects
w27	Change production philosophy (MRP, JIT, Heuristics)
w28	Change plant inventory policy (for RM, FG, etc.)
w29	Change material handler allocation policy
w30	Change primary/secondary part selection policies from input queues of resources
w31	Change control domain (span of control)
w32	Change order acceptance and/or order release policies
w33	Change operator assignment policies
w34	Change material flow control policies (push vs. pull)
w35	Change Kanban size (if applicable)
w36	Change overtime policy

LIST OF MAPPINGS OF THE SYMPTOMS TO THE PROBLEMS

Sy. #	**Matching Symptoms to the Problems** *Symptoms are underlined, problems are indented*	**Confidence Level (0-100)**
s1	**Inaccurate inventory records**	
	Insufficient work-force	50
	Theft	70
	Poor storage design	80
	Poor record keeping	90
	Poor information system	50
s2	**Shortage of some RM and/or overstock of some others**	
	Theft	70
	Poor inventory policies	90
	Poor demand forecasting	70
	Low RM quality	40
	Incorrect BOMs	40
	High variable demand on product mix	60
	Poor record keeping	70
	Poor information system	50
s3	**Shortage of some FG and/or overstock of some others**	
	Theft	40
	Poor inventory policies	60
	Poor demand forecasting	80
	Insufficient capacity of resources	40
	Poor production planning	70
	High variable demand on product mix	70
	Outmoded philosophies (Push mentality, Sequential design process, Focus on labor efficiency and machine utilization, Management with objectives, Vertical organizations, etc.)	50
	Poor record keeping	60
	Poor information system	50
s4	**High WIP inventory (overall)**	
	Inflexible work-force	30
	Insufficient work-force	50
	Bottleneck	70
	Deadlock	60
	Highly unreliable machines	70
	Inadequate MH allocation policies	40
	Insufficient MH capacity	50
	Outmoded MH equipment	50
	Poor layout design	60
	Insufficient capacity of resources	70
	Poor resource allocation policies	60
	Lack of alternate plans	40
	Poor scheduling (lack of alternate routings)	80
	Outmoded philosophies (Push mentality, Sequential design process, Focus on labor efficiency and machine utilization, Management with objectives, Vertical organizations, etc.)	60
	Inaccurate blueprints	40

Table continues on the next page...

Continued from the previous page...

	Incorrect BOMs	50
	Incorrect routings	50
	High variable demand on product mix	60
s5	**High WIP inventory on selected work centers**	
	Inflexible work-force	50
	Low RM quality	60
	Bottleneck	90
	Deadlock	60
	Highly unreliable machines	80
	Tool wear	50
	Poor layout design	70
	Insufficient capacity of resources	50
	Poor scheduling (lack of alternate routings)	40
	Incorrect BOMs	30
	Incorrect routings	40
s6	**Frequently running out of storage space**	
	Lack of employee discipline	50
	Lack of training, know-how	40
	Poor inventory policies	70
	Poor demand forecasting	60
	Poor storage design	90
	Unreliable vendors	50
	High variable demand on product mix	60
	Poor information system	40
s7	**Perishables items going bad**	
	Poor inventory policies	80
	Poor demand forecasting	80
	Poor storage design	60
	High variable demand on product mix	60
	Slow decision making	30
	Poor record keeping	30
	Poor information system	40
s8	**Late RM deliveries from the vendor(s)**	
	Unreliable vendors	80
	Lack of backup/alternate vendors	70
	High variable demand on product mix	60
	Poor information system	50
s9	**Low quality RM from the vendor(s)**	
	Unreliable vendors	80
	Lack of backup/alternate vendors	70
	High variable demand on product mix	60
s10	**Low customer evaluation**	
	Low quality	80
	Inadequate value/price matching	70
	Low customer service	60
	Long lead times	80
	Poor due date setting procedures (too short)	50
	Wrong marketing strategies	70
s11	**Excessive number of customer returns**	
	Low quality	90
	Inadequate value/price matching	70
s12	**Frequent customer order cancellations**	
	Long lead times	80

Table continues on the next page...

Continued from the previous page...

	Low customer service	60
	Inadequate value/price matching	80
	Wrong marketing strategies	70
	Poor benchmarking	60
	Poor planing by customers	70
s13	**Frequent changes on customer orders**	
	Shrinking market (overall)	70
	Poor benchmarking	60
	Wrong marketing strategies	70
	Poor planing by customers	80
s14	**Excessive customer complaints**	
	Low quality	80
	Inadequate value/price matching	70
	Low customer service	60
	Long lead times	80
	Poor due date setting procedures (too short)	50
	Wrong marketing strategies	70
s15	**Loss of demand**	
	Low quality	80
	Low customer service	70
	Inadequate value/price matching	70
	Shrinking market (overall)	70
	Long lead times	80
	Wrong marketing strategies	80
s16	**Excessive amount of backorders**	
	Insufficient work-force	70
	Poor demand forecasting	80
	Long lead times	70
	Insufficient capacity of resources	80
	High variable demand on product mix	70
	Obsolete technology	70
s17	**Absenteeism**	
	Lack of employee discipline	50
	Inadequate reward/incentive system	70
	Poor corporate culture	60
	Improper performance measurement system	50
	Poor job instructions	30
s18	**Worker tardiness (inefficiency)**	
	Lack of training, know-how	90
	Lack of required level of education	70
	Inadequate reward/incentive system	70
	Poor corporate culture	50
	Poor job instructions	70
	Improper performance measurement system	30
s19	**High workmen compensation**	
	Lack of training, know-how	60
	Lack of required level of education	60
	Failing to comply with OSHA, EPA regulations	90
	Poor safety measures	80
	Poor (bad) ergonomics	70
	Poor job instructions	40
s20	**High level of injuries**	
	Poor safety measures	90

Table continues on the next page...

Continued from the previous page...

	Poor (bad) ergonomics	70
	Lack of employee discipline	50
	Lack of training, know-how	80
	Lack of required level of education	50
s21	**High level of employee frustration**	
	Lack of training, know-how	90
	Inadequate reward/incentive system	70
	Poor job instructions(80),	80
	Improper performance measurement system	70
	Poor corporate culture	50
	Poor safety measures	50
	Insufficient work-force	60
	Outmoded philosophies (Push mentality, Sequential design process, Focus on labor efficiency and machine utilization, Management with objectives, Vertical organizations, etc.)	40
s22	**Slow learning curve**	
	Lack of required level of education	90
	Lack of employee discipline	50
	Poor (bad) ergonomics	80
	Inadequate reward/incentive system	80
	Poor job instructions	60
s23	**High level of demotivation towards the work/company**	
	Inadequate reward/incentive system	90
	Improper performance measurement system	80
	Poor safety measures	70
	Lack of training, know-how	50
	Outmoded philosophies (Push mentality, Sequential design process, Focus on labor efficiency and machine utilization, Management with objectives, Vertical organizations, etc.)	40
	Poor job instructions	50
	Poor corporate culture	50
	Poor (bad) ergonomics	60
s24	**Excessive waiting times for resources (tools, fixtures, etc.)**	
	Insufficient capacity of resources	90
	Poor resource allocation policies	70
	Usage of outmoded resources (tools and fixtures)	60
	Slow decision making	30
s25	**Long waiting times for material movement**	
	Inadequate MH allocation policies	80
	Insufficient MH capacity	90
	Outmoded MH equipment	70
	Poor layout design	50
	Insufficient capacity of resources	30
	Poor resource allocation policies	30
	Inflexible work-force	20
	Insufficient work-force	40
s26	**Long waiting times for machine tools**	
	Insufficient capacity of resources	90
	Poor resource allocation policies	80
	Inflexible work-force	50
	Insufficient work-force	70
s27	**Missing due dates (frequent late customer deliveries)**	
	Bottleneck	80

Table continues on the next page...

Continued from the previous page...

	Deadlock	60
	Highly unreliable machines	60
	Improper batch sizes (too small or too large)	50
	Poor due date setting procedures (too short)	90
	Inaccurate time standards	60
	Poor production planning	70
s28	**Low production rate**	
	Bottleneck	80
	Deadlock	60
	Highly unreliable machines	60
	Improper batch sizes (too small or too large)	50
	Long lead times	90
	Poor due date setting procedures (too short)	80
	Inaccurate time standards	60
	Poor production planning	70
s29	**Long production flow times**	
	Bottleneck	80
	Deadlock	60
	Highly unreliable machines	60
	Improper batch sizes (too small or too large)	50
	Long lead times	90
	Poor due date setting procedures (too short)	80
	Inaccurate time standards	60
	Poor production planning	70
s30	**Excessive overtime**	
	Insufficient work-force	70
	Inflexible work-force	40
	Lack of training, know-how	50
	Highly unreliable machines	40
	High variable demand on product mix	70
	Poor demand forecasting	60
	Insufficient capacity of resources	80
s31	**Jobs with excessive tardiness**	
	Theft	40
	Low RM quality	50
	Deadlock	70
	Poor due date setting procedures	80
	Inaccurate time standards	60
	Lack of alternate plans	40
	Inaccurate blueprints	70
	Poor job instructions	50
	Slow decision making	30
	Incorrect BOMs	60
	Incorrect routings	60
s32	**High reject/scrap rate**	
	Highly unreliable machines	80
	Lack of employee discipline	60
	Lack of training, know-how	70
	Poor (bad) ergonomics	40
	Lack of Inadequate maintenance	70
	Tool wear	70
	Usage of outmoded resources (tools and fixtures)	50
	Poor part design (unnecessarily complex)	50

Table continues on the next page...

Continued from the previous page...

	Poor job instructions	40
s33	**High rework rate**	
	Highly unreliable machines	80
	Lack of employee discipline	60
	Lack of training, know-how	70
	Poor (bad) ergonomics	40
	Lack of Inadequate maintenance	70
	Tool wear	70
	Usage of outmoded resources (tools and fixtures)	50
	Poor part design (unnecessarily complex)	50
	Poor job instructions	40
s34	**Damaged parts**	
	Outmoded MH equipment	70
	Poor part design (unnecessarily complex)	50
s35	**High variable work loads**	
	High variable demand on product mix	80
	Poor demand forecasting	80
	Poor production planning	70
	Outmoded philosophies (Push mentality, Sequential design process, Focus on labor efficiency and machine utilization, Management with objectives, Vertical organizations, etc.)	50
s36	**High equipment idle time**	
	Excessive machine capacity	80
	Excessive material handler capacity	60
	Lack of demand	70
	Shrinking market (overall)	50
	Insufficient work-force	60
	Poor demand forecasting	70
	Poor scheduling (lack of alternate routings)	60
s37	**Frequent breakdowns / malfunctions**	
	Lack of required level of education	50
	Lack of training, know-how	60
	Low RM quality	70
	Highly unreliable machines	80
	Lack of Inadequate maintenance	80
	Tool wear	60
	Insufficient capacity of resources	70
s38	**Excessive setup times**	
	Improper batch sizes (too small or too large)	80
	Insufficient work-force	60
	Insufficient capacity of resources	70
	Usage of inappropriate tools and fixtures	70
s39	**Excessive material movement**	
	Poor layout design	80
	Insufficient capacity of resources	50
	Poor production planning	60
	Poor scheduling (lack of alternate routings)	60
	Poor job instructions	40
	Obsolete technology	50
s40	**Excessive sub-contracting**	
	Insufficient work-force(70),	70
	Lack of required level of education(60),	60
	Inflexible work-force(40),	40

Table continues on the next page...

Continued from the previous page...

	Insufficient capacity of resources(80),	80
	High variable demand on product mix(60),	60
	Poor demand forecasting(50)	50
s41	**Excessive amount of expediting (too many "hot" jobs)**	
	Poor demand forecasting	70
	Unreliable vendors	40
	Lack of backup/alternate vendors	40
	Poor due date setting procedures	70
	High variable demand on product mix	80
	Poor production planning	80
s42	**Incomplete production orders**	
	Theft	60
	Poor inventory policies	60
	Deadlock	40
	Lack of alternate plans	40
	Poor scheduling (lack of alternate routings)	50
	Inaccurate blueprints	70
	Poor job instructions	50
	Slow decision making	50
	Poor information system	60
	Incorrect BOMs	60
	Incorrect routings	60
	Poor production planning	80
s43	**Production times are not matching the predicted times**	
	Inaccurate time standards	80
	Lack of training, know-how	60
	Inadequate reward/incentive system	50
	Low RM quality	40
	Highly unreliable machines	60
	Insufficient capacity of resources	50
	Poor production planning	80
s44	**Excessive changes to MPS**	
	Poor demand forecasting	80
	High variable demand on product mix	70
	Poor corporate culture	40
	Poor inventory policies	50
	Unreliable vendors	40
	Lack of alternate plans	50
	Poor production planning	70
	Poor information system	40
s45	**Long customer order response time**	
	Long lead times	80
	Poor due date setting procedures (too short)	80
	Outmoded philosophies (Push mentality, Sequential design process, Focus on labor efficiency and machine utilization, Management with objectives, Vertical organizations, etc.)	60
s46	**Conflicting work schedule**	
	Poor resource allocation policies	60
	Poor production planning	80
	Inaccurate time standards	70
	Inaccurate/missing process plans	60
	Inaccurate blueprints	60

Table continues on the next page...

Continued from the previous page...

	Poor job instructions	40
	Poor information system	50
s47	**Resource conflicts**	
	Poor resource allocation policies	80
	Insufficient capacity of resources	40
	Poor production planning	70
	Inaccurate time standards	60
	Inaccurate/missing process plans	50
	Poor scheduling (lack of alternate routings)	70
	High variable demand on product mix	40
	Inaccurate blueprints	50
	Poor information system	50
	Poor job instructions	50
s48	**Poor flow of materials**	
	Poor production planning	80
	Inadequate MH allocation policies	60
	Insufficient MH capacity	60
	Outmoded MH equipment	60
	Poor layout design	70
	Outmoded philosophies (Push mentality, Sequential design process, Focus on labor efficiency and machine utilization, Management with objectives, Vertical organizations, etc.)	40
	Poor production planning	70
	Poor scheduling (lack of alternate routings)	60
s49	**Missing production work orders**	
	Theft	60
	Lack of required level of education	60
	Poor job instructions	70
	Inaccurate blueprints	80
	Poor record keeping	40
	Incorrect routings	60
	Poor information system	50
s50	**Too many jobs at top priority**	
	High variable demand on product mix	80
	Poor demand forecasting	70
	Poor production planning	70
	Wrong marketing strategies	50
	Poor information system	60
s51	**Too many unplanned activities**	
	High variable demand on product mix	80
	Poor demand forecasting	70
	Poor production planning	90
	Wrong marketing strategies	50
	Poor information system	60

LIST OF MAPPINGS OF THE STRUCTURED PROBLEMS TO THE TOOLS

Pr .#	**Matching Problems to the Tools** *Problems are underlined, tools are indented*	**Confidence Level** (0-100)
p1	**Poor employee discipline**	
	Non-modeling approach (e.g., observation)	90
p2	**Lack of training, know-how**	
	Non-modeling approach (e.g., observation)	90
p3	**Lack of employees with required level of education**	
	Non-modeling approach (e.g., observation)	90
p4	**Inflexible work force**	
	Non-modeling approach (e.g., observation)	90
p5	**Inadequate reward/incentive system**	
	Non-modeling approach (e.g., observation)	90
p6	**Failing to comply with OSHA, EPA regulations**	
	Non-modeling approach (e.g., observation)	90
p7	**Poor safety measures**	
	Non-modeling approach (e.g., observation)	90
p8	**Poor corporate culture**	
	Non-modeling approach (e.g., observation)	90
p9	**Poor (bad) ergonomics**	
	Non-modeling approach (e.g., observation)	90
p10	**Insufficient work force**	
	Simulation	80
	Other (e.g., time series, layout modeling)	70
	Petri nets	40
p11	**Excessive work force**	
	Simulation	80
	Other (e.g., time series, layout modeling)	70
	Petri nets	40
p12	**Theft**	
	Non-modeling approach (e.g., observation)	90
p13	**Poor inventory policies**	
	Search-based optimization with simulation	80
	Queueing + Simulation	70
	Simulation	60
	Queueing	40
p14	**Poor demand forecasting**	
	Other (e.g., time series, layout modeling)	90
	Simulation	30
p15	**Poor storage design**	
	Other (e.g., time series, layout modeling)	90
p16	**Unreliable vendors**	
	Other (e.g., time series, layout modeling)	80
	Simulation	50
p17	**Lack of backup vendors**	
	Other (e.g., time series, layout modeling)	90
	Simulation	30
p18	**Low RM quality**	
	Non-modeling approach (e.g., observation)	90

Table continues on the next page...

Continued from the previous page...

p19	**Bottleneck**	
	Simulation	90
	Queueing	70
	Search-based optimization with simulation	70
p20	**Deadlock**	
	Petri nets	90
	Simulation	60
p21	**Highly unreliable machines**	
	Simulation	90
	Queueing	60
	Search-based optimization with simulation	50
p22	**Inadequate maintenance**	
	Simulation	90
	Queueing	60
p23	**Improper batch sizes (too small or too large)**	
	Simulation	70
	Queueing	60
	Queueing + Simulation	80
	Search-based optimization with simulation	90
p24	**Low quality**	
	Other (e.g., time series, layout modeling)	90
	Simulation	60
p25	**Tool wear**	
	Other (e.g., time series, layout modeling)	90
	Simulation	60
p26	**Low customer service**	
	Other (e.g., time series, layout modeling)	90
	Simulation	90
	Search-based optimization with simulation	50
p27	**Inadequate value/price relation**	
	Other (e.g., time series, layout modeling)	90
p28	**Long lead times**	
	Simulation	80
	Queueing	70
	Petri nets	70
	Search-based optimization with simulation	50
p29	**Inadequate MH allocation policies**	
	Simulation	80
	Search-based optimization with simulation	90
p30	**Insufficient MH capacity**	
	Simulation	70
	Queueing	60
	Petri nets	40
	Queueing + Simulation	80
	Search-based optimization with simulation	90
p31	**Outmoded MH equipment (e.g., speed, capacity, and quality)**	
	Other (e.g., time series, layout modeling)	90
	Simulation	50
p32	**Poor layout design**	
	Other (e.g., time series, layout modeling)	90
	Simulation	80
p33	**Insufficient capacity of resources**	
	Simulation	90

Table continues on the next page...

Continued from the previous page...

	Queueing	70
	Queueing + Simulation	70
p34	**Poor resource allocation policies (strategies)**	
	Simulation	70
	Queueing	60
	Petri nets	40
	Search-based optimization with simulation	90
	Poor due date settings procedures	
	Other (e.g., time series, layout modeling)	80
	Search-based optimization with simulation	90
	Simulation	70
p35	**Poor production planning**	
	Other (e.g., time series, layout modeling)	80
	Simulation	90
	Search-based optimization with simulation	70
p36	**Inaccurate time standards**	
	Other (e.g., time series, layout modeling)	80
	Simulation	60
p37	**Inaccurate/missing process plans**	
	Non-modeling approach (e.g., observation)	90
p38	**Lack of backup plans**	
	Non-modeling approach (e.g., observation)	90
p39	**Lack of alternate plans**	
	Non-modeling approach (e.g., observation)	90
p40	**Poor scheduling (lack of alternate routings)**	
	Simulation	70
	Queueing	60
	Petri nets	40
	Queueing + Simulation	80
	Search-based optimization with simulation	90
p41	**Usage of inappropriate tools and fixtures**	
	Non-modeling approach (e.g., observation)	90
p42	**Excessive machine capacity**	
	Simulation	90
	Queueing	80
	Petri nets	70
	Search-based optimization with simulation	70
p43	**Excessive material handler capacity**	
	Simulation	90
	Queueing	80
	Petri nets	70
	Search-based optimization with simulation	70
p44	**Highly variable demand**	
	Other (e.g., time series, layout modeling)	90
	Simulation	60
	Search-based optimization with simulation	50
p45	**Lack of demand**	
	Other (e.g., time series, layout modeling)	90
p46	**Shrinking market (overall)**	
	Other (e.g., time series, layout modeling)	90
p47	**Outmoded philosophies**	
	Other (e.g., time series, layout modeling)	90

Table continues on the next page...

Continued from the previous page...

	Simulation	80
	Search-based optimization with simulation	80
p48	**Inaccurate blueprints**	
	Non-modeling approach (e.g., observation)	90
	Poor part design (unnecessarily complex products)	
	Non-modeling approach (e.g., observation)	90
p49	**Poor job instructions**	
	Non-modeling approach (e.g., observation)	90
p50	**Obsolete Technology**	
	Non-modeling approach (e.g., observation)	90
p51	**Slow decision making**	
	Non-modeling approach (e.g., observation)	90
p52	**Wrong marketing strategy**	
	Other (e.g., time series, layout modeling)	90
p53	**Poor record keeping**	
	Other (e.g., time series, layout modeling)	90
	Simulation	50
p54	**Poor benchmarking**	
	Non-modeling approach (e.g., observation)	90
p55	**Incorrect BOMs**	
	Other (e.g., time series, layout modeling)	90
	Simulation	70
p56	**Inaccurate routings**	
	Other (e.g., time series, layout modeling)	90
	Simulation	70
p57	**Improper performance measurement system**	
	Other (e.g., time series, layout modeling)	90
	Simulation	40
p58	**Poor information management system**	
	Other (e.g., time series, layout modeling)	90
	Simulation	40
p59	**Poor planing by customers**	
	Other (e.g., time series, layout modeling)	90
	Simulation	60

LIST OF MAPPINGS OF THE WHAT-IF QUESTIONS TO THE TOOLS

W.	**Matching What-if Questions to the Tools**	**Confidence**
#	*What-if questions are underlined, tools are indented*	**Level (0-100)**
	With regard to the Physical Objects	
w1	**Add/remove a work station**	
	Simulation	90
	Queueing	80
	Queueing + Simulation	90
w2	**Add/remove an assembly station**	
	Simulation	90
	Queueing	80
	Queueing + Simulation	60
w3	**Add/remove a material handler**	
	Simulation	90
	Queueing	80
	Queueing + Simulation	60
w4	**Add/remove a stock room**	
	Simulation	90
w5	**Add/remove a product from the product-mix**	
	Simulation	90
	Queueing	80
	Queueing + Simulation	80
w6	**Change plant layout (add/remove an aisle)**	
	Simulation	80
	Other (e.g., time series, layout modeling)	90
w7	**Add/remove a manufacturing line**	
	Simulation	80
	Queueing + Simulation	90
w8	**Add/remove tools, fixtures, pallets, etc.**	
	Simulation	90
	Petri nets	60
w9	**Change equipment reliability parameters (e.g., add/remove a maintenance crew)**	
	Simulation	90
	Queueing	60
	Queueing + Simulation	80
w10	**Add/remove operators**	
	Simulation	90
	Queueing + Simulation	80
w11	**Change operator capabilities (cross-training)**	
	Simulation	90
	Queueing	70
w12	**Change worker assignments**	
	Simulation	90
	Queueing	70

Table continues on the next page...

Continued from the previous page...

w13	**Change product design**	
	Simulation	90
	Queueing	60
	Queueing + Simulation	80
w14	**Change # of work shifts (for some stations)**	
	Simulation	80
	Queueing	60
w15	**Change plant location**	
	Simulation	70
	Other (e.g., time series, layout modeling)	80
w16	**Change buffer capacities**	
	Simulation	90
	Queueing	60
	Queueing + Simulation	80
	With regard to the Information Objects	
w17	**Change bill of materials (add/remove a component)**	
	Simulation	90
	Queueing	60
	Queueing + Simulation	90
w18	**Change routing (add/remove an alternate routing)**	
	Simulation	90
	Queueing	60
	Petri nets	40
	Queueing + Simulation	90
w19	**Combine operations**	
	Simulation	90
	Queueing	70
	Queueing + Simulation	60
w20	**Change demand for the product mix**	
	Simulation	90
	Queueing	70
	Petri nets	40
	Queueing + Simulation	60
	Other (e.g., time series, layout modeling)	80
w21	**Change lot sizes**	
	Simulation	90
	Queueing	70
	Queueing + Simulation	80
w22	**Change product versions**	
	Other (e.g., time series, layout modeling)	90
	Non-modeling techniques (e.g., observation)	60
w23	**Change sampling rates**	
	Simulation	80
	Non-modeling techniques (e.g., observation)	70
w24	**Change operation sequence**	
	Simulation	90
	Queueing	70
	Queueing + Simulation	90
w25	**Change setup/processing times**	
	Simulation	90
	Queueing	80
	Queueing + Simulation	90

Table continues on the next page...

Continued from the previous page...

w26	**Change defect/scrap rate**	
	Simulation	90
	Queueing	70
	Queueing + Simulation	80
	With regard to the Control/Decisional Objects	
w27	**Change production philosophy (MRP, JIT, Heuristics)**	
	Simulation	80
w28	**Change plant inventory policy (for RM, FG, etc.)**	
	Simulation	80
w29	**Change material handler allocation policy**	
	Simulation	80
	Petri nets	50
w30	**Change primary/secondary part selection policies from input queues of resources**	
	Simulation	80
w31	**Change control domain (span of control)**	
	Simulation	80
	Petri nets	50
w32	**Change order acceptance and/or order release policies**	
	Simulation	80
w33	**Change operator assignment policies**	
	Simulation	90
	Queueing + Simulation	70
	Petri nets	50
w34	**Change material flow control policies (push vs. pull)**	
	Simulation	90
w35	**Change Kanban size (if applicable)**	
	Simulation	80
w36	**Change overtime policy**	
	Simulation	90

APPENDIX E

RULES IN EFES

Introduction

This appendix contains listings of EFES rules beginning on the next page. The listings are separated by knowledge bases. Within each knowledge base, there is a header section followed by the entity types and parameter definitions. The header section contains the name of the knowledge base and the date the listing is generated by HUMBLE.

The rule names are highlighted (boldfaced). Following each rule name a definition of the rule is given. The actual code (IF .. THEN .. expression) follows the definition part. At the end of each knowledge base listing, rule metrics are presented. In the rule metrics the total number of rules, entity types and parameters are summarized.

ProblemDefinitionKB, a HUMBLE knowledge base
as of 16 November 1996 6:41:54 pm

```
--------------------------------------------
Entity Types
--------------------------------------------
ProblemScenario
    typeAbove: nil
    createPrompt: Are there any problem scenarios to consider?
    addPrompt: Are there any other problem scenarios  to consider?
    assumePrompt: I am creating a problem scenario
    defaultName: Problem_Scenario
    parameters: #(#StructuredProblem)
    mainParameters: #()

Symptoms
    typeAbove: ProblemScenario
    createPrompt: Are there any symptoms to consider?
    addPrompt: Are there any other symptoms to consider?
    assumePrompt: I am creating a symptom
    defaultName: Symptom
    parameters: #(#Name)
    mainParameters: #(#Name)

--------------------------------------------
Parameter Definitions
--------------------------------------------
Name
    describes: Symptoms
    type: #('Inaccurate inventory records' 'Stockout and/or high inventory level of some Raw Material
items' 'Stockout and/or high inventory level of some Finished Good items' 'High WIP inventory (overall)'
'High WIP inventory in selected work centers' 'Frequently running out of storage space' 'Perishable items
going bad' 'Late deliveries from vendors' 'Low quality Raw Material from vendors' 'Low customer
evaluation' 'Excessive number of customer returns (warranty redemption)' 'Frequent customer order
cancellations' 'Frequent changes on customer orders' 'Excessive customer complaints' 'Loss of demand'
'Excessive amount of backorders' 'Absenteeism' 'Worker tardiness' 'High workmen compensation' 'High
level of injuries and deaths' 'High level of frustration' 'Slow learning curve' 'High level of demotivation
towards the work and the company' 'Excessive waiting time for secondary resources (tools, pallets,
fixtures, etc.)' 'Long waiting times for material movement' 'Long waiting times for machine tools' 'Missing
the due dates (frequent late customer deliveries)' 'Low production rate' 'Long production flow times'
'Excessive overtime' 'Jobs with excessive tardiness' 'High reject/scrap rate' 'High rework rate' 'Damaged
parts' 'High variable work loads' 'High equipment idle times' 'Frequent breakdowns/malfunctions'
'Excessive setup times' 'Excessive material movement' 'Excessive sub-contracting' 'Excessive reliance on
expediting (too many hot jobs)' 'Incomplete production orders' 'Production time differs from Predicted
time' 'Excessive changes to MPS' 'Long customer order response time' 'Conflicting work schedule'
'Resource conflicts' 'Poor flow of materials' 'Missing production work orders' 'Too many jobs at top
priority' 'Too many unplanned activities (due to poor planning)')
    prompt: What is the name of & ?
    promptFlag: askFirst
    explanation: "Name is to be selected from a given set"
    remark: "Standard array parameter used to store the names of the symptoms"
    deducingRules: #()
    changeBlock: '[:parameter |]'
```

ProblemDefinitionKB - continued

```
StructuredProblem
    describes: ProblemScenario
    type: String
    prompt: What is the structured problem for & ?
    promptFlag: askLast
    explanation: "Structured problem is the cause for the existance of symptoms in the scenario"
    remark: "Standard string parameter used to store the final structured problem of the current
    scenario"
    deducingRules: #(#TooManyUnplannedActivities #TooManyJobsAtTopPriority
#MissingProductionWorkOrders #PoorFlowOfMaterials #ResourceConflicts #ConflictingWorkSchedule
#LongCustomerOrderResponseTime #ExcessiveChangesToMPS
#ProductionTimeDiffersFromPredictedTime #IncompleteProductionOrders
#ExcessiveRelianceOnExpediting #ExcessiveSubContracting #ExcessiveMaterialMovement
#ExcessiveSetupTimes #FrequentBreakdowns #HighEquipmentIdleTimes #HighVariableWorkLoads
#DamagedParts #HighReworkRate #HighRejectRate #JobsWithExcessiveTardiness #ExcessiveOvertime
#LongProductionFlowTimes #LowProductionRate #MissingTheDueDates
#LongWaitingTimesForMachineTools #LongWaitingTimesForMaterialMovement
#ExcessiveWaitingTimeForSecondaryResources #HighLevelOfDemotivationTowardsTheCompany
#SlowLearningCurve #HighLevelOfFrustration #HighLevelOfInjuriesAndDeaths
#HighWorkmenCompensation #WorkerTardiness #Absenteeism #ExcessiveAmountOfBackorders
#LossOfDemand #ExcessiveCustomerComplaints #FrequentChangesOnCustomerOrders
#FrequentCustomerOrderCancellations #ExcessiveNumberOfCustomerReturns #LowCustomerEvaluation
#LowQualityRMFromVendors #LateDeliveriesFromVendors #PerishableItemsGoingBad
#FrequentlyRunningOutOfStorageSpace #HighWIPInventoryOnSelectedWCs #HighWIPInventoryOverall
#StockoutAndOrHighInventoryOfSomeFGItems #StockoutAndOrHighInventoryOfSomeRMItems
#InaccurateInventoryRecords)
    changeBlock: '[:parameter |]'

--------------------------------------------
Rules
--------------------------------------------
Absenteeism
    "Ties the symptom 'Absenteeism' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Absenteeism'])
        then: [StructuredProblem is: 'Poor employee discipline' withCertainty: 0.5.
              StructuredProblem is: 'Inadequate reward/incentive system' withCertainty: 0.7.
              StructuredProblem is: 'Poor corporate culture' withCertainty: 0.6.
              StructuredProblem is: 'Improper performance measurement system' withCertainty: 0.5.
              StructuredProblem is: 'Poor job instructions' withCertainty: 0.3]

"Executes in the context of ProblemScenario entities"
-------
DamagedParts
    "Ties the symptom 'Damaged parts' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Damaged parts'])
        then: [StructuredProblem is: 'Outmoded MH equipment' withCertainty: 0.7.
              StructuredProblem is: 'Poor part design (complex products)' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"
```

```
-------
ConflictingWorkSchedule
    "Ties the symptom 'Conflicting work schedule' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Conflicting work schedule'])
        then: [StructuredProblem is: 'Poor resource allocation policies' withCertainty: 0.6.
              StructuredProblem is: 'Poor production planning' withCertainty: 0.8.
              StructuredProblem is: 'Inaccurate time standards' withCertainty: 0.7.
              StructuredProblem is: 'Inaccurate/missing process plans' withCertainty: 0.6.
              StructuredProblem is: 'Inaccurate blueprints' withCertainty: 0.6.
              StructuredProblem is: 'Poor job instructions' withCertainty: 0.4.
              StructuredProblem is: 'Poor information system' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"

-------
ExcessiveAmountOfBackorders
    "Ties the symptom 'Excessive amount of backorders' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Excessive amount of backorders'])
        then: [StructuredProblem is: 'Insufficient work force' withCertainty: 0.7.
              StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.8.
              StructuredProblem is: 'Long lead times' withCertainty: 0.7.
              StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.8.
              StructuredProblem is: 'Highly variable demand' withCertainty: 0.7.
              StructuredProblem is: 'Obsolete technology' withCertainty: 0.7]

"Executes in the context of ProblemScenario entities"

-------
ExcessiveChangesToMPS
    "Ties the symptom 'Excessive changes to MPS' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Excessive changes to MPS'])
        then: [StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.8.
              StructuredProblem is: 'Highly variable demand' withCertainty: 0.7.
              StructuredProblem is: 'Poor corporate culture' withCertainty: 0.4.
              StructuredProblem is: 'Poor inventory policies' withCertainty: 0.5.
              StructuredProblem is: 'Unreliable vendors' withCertainty: 0.4.
              StructuredProblem is: 'Lack of alternate plans' withCertainty: 0.5.
              StructuredProblem is: 'Poor production planning' withCertainty: 0.7.
              StructuredProblem is: 'Poor information system' withCertainty: 0.4]

"Executes in the context of ProblemScenario entities"
```

```
-------
ExcessiveCustomerComplaints
   "Ties the syptom 'Excessive customer complaints' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Excessive customer complaints'])
      then: [StructuredProblem is: 'Low quality' withCertainty: 0.8.
           StructuredProblem is: 'Inadequate value/price relation' withCertainty: 0.7.
           StructuredProblem is: 'Low customer service' withCertainty: 0.6.
           StructuredProblem is: 'Long lead times' withCertainty: 0.8.
           StructuredProblem is: 'Poor due date settings procedures' withCertainty: 0.5.
           StructuredProblem is: 'Wrong marketing strategy' withCertainty: 0.7]

"Executes in the context of ProblemScenario entities"

-------
ExcessiveMaterialMovement
   "Ties the symptom 'Excessive material movement' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Excessive material movement'])
      then: [StructuredProblem is: 'Poor layout design' withCertainty: 0.8.
           StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.5.
           StructuredProblem is: 'Poor production planning' withCertainty: 0.6.
           StructuredProblem is: 'Poor scheduling (lack of alternate routings)' withCertainty: 0.6.
           StructuredProblem is: 'Poor job instructions' withCertainty: 0.4.
           StructuredProblem is: 'Obsolete technology' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"

-------
ExcessiveNumberOfCustomerReturns
   "Ties the syptom 'Excessive number of customer returns (warranty redemption)'
   to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Excessive number of customer returns '])
      then: [StructuredProblem is: 'Low quality' withCertainty: 0.9.
           StructuredProblem is: 'Inadequate value/price relation' withCertainty: 0.7]

"Executes in the context of ProblemScenario entities"

-------
ExcessiveOvertime
   "Ties the symptom 'Excessive overtime' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Excessive overtime'])
      then: [StructuredProblem is: 'Insufficient work force' withCertainty: 0.7.
           StructuredProblem is: 'Inflexible work force' withCertainty: 0.4.
           StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.5.
           StructuredProblem is: 'Highly unreliable machines' withCertainty: 0.4.
           StructuredProblem is: 'Highly variable demand' withCertainty: 0.7.
           StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.6.
           StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.8]
```

```
-------
ExcessiveRelianceOnExpediting
   "Ties the symptom 'Excessive reliance on expediting (too many hot jobs)'
   to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Excessive reliance on expediting (too many hot jobs)'])
      then: [StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.7.
           StructuredProblem is: 'Unreliable vendors' withCertainty: 0.4.
           StructuredProblem is: 'Lack of backup vendors' withCertainty: 0.4.
           StructuredProblem is: 'Highly variable demand' withCertainty: 0.8.
           StructuredProblem is: 'Poor due date settings procedures' withCertainty: 0.7.
           StructuredProblem is: 'Poor production planning' withCertainty: 0.8]

"Executes in the context of ProblemScenario entities"

-------
ExcessiveSetupTimes
   "Ties the symptom 'Excessive setup times' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Excessive setup times'])
      then: [StructuredProblem is: 'Improper batch sizes (too small or too large)' withCertainty: 0.8.
           StructuredProblem is: 'Insufficient work force' withCertainty: 0.6.
           StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.7.
           StructuredProblem is: 'Usage of inappropriate tools and fixtures' withCertainty: 0.7]

"Executes in the context of ProblemScenario entities"

-------
ExcessiveSubContracting
   "Ties the symptom 'Excessive sub-contracting' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Excessive sub-contracting'])
      then: [StructuredProblem is: 'Insufficient work force' withCertainty: 0.7.
           StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.8.
           StructuredProblem is: 'Lack of required level of education' withCertainty: 0.6.
           StructuredProblem is: 'Inflexible work force' withCertainty: 0.4.
           StructuredProblem is: 'Highly variable demand' withCertainty: 0.6.
           StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"

-------
ExcessiveWaitingTimeForSecondaryResources
   "Ties the symptom 'Excessive waiting time for secondary resources (tools, pallets, fixtures, etc.)'
   to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Excessive waiting time for secondary resources '])
      then: [StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.9.
           StructuredProblem is: 'Poor resource allocation policies' withCertainty: 0.7.
           StructuredProblem is: 'Usage of inappropriate tools and fixtures' withCertainty: 0.6.
           StructuredProblem is: 'Slow decision making' withCertainty: 0.3]
```

ProblemDefinitionKB - continued

```
-------
FrequentBreakdowns
   "Ties the symptom 'Frequent breakdowns/malfunctions' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Frequent breakdowns/malfunctions'])
      then: [StructuredProblem is: 'Lack of required level of education' withCertainty: 0.5.
            StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.6.
            StructuredProblem is: 'Low RM quality' withCertainty: 0.7.
            StructuredProblem is: 'Highly unreliable machines' withCertainty: 0.8.
            StructuredProblem is: 'Inadequate maintenance' withCertainty: 0.8.
            StructuredProblem is: 'Tool wear' withCertainty: 0.6.
            StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.7]

"Executes in the context of ProblemScenario entities"

-------
FrequentChangesOnCustomerOrders
   "Ties the syptom 'Frequent changes on customer orders' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Frequent changes on customer orders'])
      then: [StructuredProblem is: 'Shrinking market (overall)' withCertainty: 0.7.
            StructuredProblem is: 'Poor benchmarking' withCertainty: 0.6.
            StructuredProblem is: 'Wrong marketing strategy' withCertainty: 0.7.
            StructuredProblem is: 'Poor plannig by customers' withCertainty: 0.8]

"Executes in the context of ProblemScenario entities"

-------
FrequentCustomerOrderCancellations
   "Ties the syptom 'Frequent customer order cancellations' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Frequent customer order cancellations'])
      then: [StructuredProblem is: 'Long lead times' withCertainty: 0.8.
            StructuredProblem is: 'Low customer service' withCertainty: 0.6.
            StructuredProblem is: 'Inadequate value/price relation' withCertainty: 0.8.
            StructuredProblem is: 'Wrong marketing strategy' withCertainty: 0.7.
            StructuredProblem is: 'Poor benchmarking' withCertainty: 0.6.
            StructuredProblem is: 'Poor plannig by customers' withCertainty: 0.7]

"Executes in the context of ProblemScenario entities"

-------
FrequentlyRunningOutOfStorageSpace
   "Ties the syptom 'Frequently running out of storage space' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Frequently running out of storage space'])
      then: [StructuredProblem is: 'Poor employee discipline' withCertainty: 0.5.
            StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.4.
            StructuredProblem is: 'Poor inventory policies' withCertainty: 0.7.
            StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.6.
            StructuredProblem is: 'Poor storage design' withCertainty: 0.9.
```

ProblemDefinitionKB - continued

```
            StructuredProblem is: 'Unreliable vendors' withCertainty: 0.5.
            StructuredProblem is: 'Highly variable demand' withCertainty: 0.6.
            StructuredProblem is: 'Poor information system' withCertainty: 0.4]

"Executes in the context of ProblemScenario entities"

-------
HighEquipmentIdleTimes
    "Ties the symptom 'High equipment idle times' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'High equipment idle times'])
        then: [StructuredProblem is: 'Excessive machine capacity' withCertainty: 0.8.
            StructuredProblem is: 'Excessive material handler capacity' withCertainty: 0.6.
            StructuredProblem is: 'Lack of demand' withCertainty: 0.7.
            StructuredProblem is: 'Shrinking market (overall)' withCertainty: 0.5.
            StructuredProblem is: 'Insufficient work force' withCertainty: 0.6.
            StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.7.
            StructuredProblem is: 'Poor scheduling (lack of alternate routings)' withCertainty: 0.6]

"Executes in the context of ProblemScenario entities"

-------
HighLevelOfDemotivationTowardsTheCompany
    "Ties the symptom 'High level of demotivation towards the work and the company'
    to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'High level of demotivation towards the work and the
    company'])
        then: [StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.5.
            StructuredProblem is: 'Inadequate reward/incentive system' withCertainty: 0.9.
            StructuredProblem is: 'Poor job instructions' withCertainty: 0.5.
            StructuredProblem is: 'Improper performance measurement system' withCertainty: 0.8.
            StructuredProblem is: 'Poor safety measures' withCertainty: 0.7.
            StructuredProblem is: 'Poor (bad) ergonomics' withCertainty: 0.6.
            StructuredProblem is: 'Poor corporate culture' withCertainty: 0.5.
            StructuredProblem is: 'Outmoded philosophies' withCertainty: 0.4]

"Executes in the context of ProblemScenario entities"

-------
HighLevelOfFrustration
    "Ties the symptom 'High level of frustration' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'High level of frustration'])
        then: [StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.9.
            StructuredProblem is: 'Inadequate reward/incentive system' withCertainty: 0.7.
            StructuredProblem is: 'Poor job instructions' withCertainty: 0.8.
            StructuredProblem is: 'Improper performance measurement system' withCertainty: 0.7.
            StructuredProblem is: 'Poor corporate culture' withCertainty: 0.5.
            StructuredProblem is: 'Poor safety measures' withCertainty: 0.5.
            StructuredProblem is: 'Insufficient work force' withCertainty: 0.6.
            StructuredProblem is: 'Outmoded philosophies' withCertainty: 0.4]

"Executes in the context of ProblemScenario entities"

-------
HighLevelOfInjuriesAndDeaths
    "Ties the symptom 'High level of injuries and deaths' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'High level of injuries and deaths'])
        then: [StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.8.
            StructuredProblem is: 'Lack of required level of education' withCertainty: 0.5.
            StructuredProblem is: 'Poor safety measures' withCertainty: 0.9.
            StructuredProblem is: 'Poor (bad) ergonomics' withCertainty: 0.7.
            StructuredProblem is: 'Poor employee discipline' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"

-------
HighRejectRate
    "Ties the syptom 'High reject/scrap rate' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'High reject/scrap rate'])
        then: [StructuredProblem is: 'Highly unreliable machines' withCertainty: 0.8.
            StructuredProblem is: 'Poor employee discipline' withCertainty: 0.6.
            StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.7.
            StructuredProblem is: 'Poor (bad) ergonomics' withCertainty: 0.4.
            StructuredProblem is: 'Inadequate maintenance' withCertainty: 0.7.
            StructuredProblem is: 'Tool wear' withCertainty: 0.7.
            StructuredProblem is: 'Usage of inappropriate tools and fixtures' withCertainty: 0.5.
            StructuredProblem is: 'Poor part design (complex products)' withCertainty: 0.5.
            StructuredProblem is: 'Poor job instructions' withCertainty: 0.4]

"Executes in the context of ProblemScenario entities"

-------
HighReworkRate
    "Ties the syptom 'High rework rate' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'High rework rate'])
        then: [StructuredProblem is: 'Highly unreliable machines' withCertainty: 0.8.
            StructuredProblem is: 'Poor employee discipline' withCertainty: 0.6.
            StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.7.
            StructuredProblem is: 'Poor (bad) ergonomics' withCertainty: 0.4.
            StructuredProblem is: 'Inadequate maintenance' withCertainty: 0.7.
            StructuredProblem is: 'Tool wear' withCertainty: 0.7.
            StructuredProblem is: 'Usage of inappropriate tools and fixtures' withCertainty: 0.5.
            StructuredProblem is: 'Poor part design (complex products)' withCertainty: 0.5.
            StructuredProblem is: 'Poor job instructions' withCertainty: 0.4]

"Executes in the context of ProblemScenario entities"
```

```
-------
HighVariableWorkLoads
    "Ties the symptom 'High variable work loads' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'High variable work loads'])
        then: [StructuredProblem is: 'Highly variable demand' withCertainty: 0.8.
               StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.8.
               StructuredProblem is: 'Poor production planning' withCertainty: 0.7.
               StructuredProblem is: 'Outmoded philosophies' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"

-------
HighWIPInventoryOnSelectedWCs
    "Ties the symptom 'High WIP inventory in selected work centers' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'High WIP inventory in selected work centers'])
        then: [StructuredProblem is: 'Inflexible work force' withCertainty: 0.5.
               StructuredProblem is: 'Low RM quality' withCertainty: 0.6.
               StructuredProblem is: 'Bottleneck' withCertainty: 0.9.
               StructuredProblem is: 'Deadlock' withCertainty: 0.6.
               StructuredProblem is: 'Highly unreliable machines' withCertainty: 0.8.
               StructuredProblem is: 'Tool wear' withCertainty: 0.5.
               StructuredProblem is: 'Poor layout design' withCertainty: 0.7.
               StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.5.
               StructuredProblem is: 'Poor scheduling (lack of alternate routings)' withCertainty: 0.4.
               StructuredProblem is: 'Incorrect BOMs' withCertainty: 0.3.
               StructuredProblem is: 'Inaccurate routings' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"

-------
HighWIPInventoryOverall
    "Ties the symptom 'High WIP inventory (overall)' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'High WIP inventory (overall)'])
        then: [StructuredProblem is: 'Inflexible work force' withCertainty: 0.3.
               StructuredProblem is: 'Insufficient work force' withCertainty: 0.5.
               StructuredProblem is: 'Bottleneck' withCertainty: 0.7.
               StructuredProblem is: 'Deadlock' withCertainty: 0.6.
               StructuredProblem is: 'Highly unreliable machines' withCertainty: 0.7.
               StructuredProblem is: 'Inadequate MH allocation policies' withCertainty: 0.4.
               StructuredProblem is: 'Insufficient MH capacity' withCertainty: 0.5.
               StructuredProblem is: 'Outmoded MH equipment' withCertainty: 0.5.
               StructuredProblem is: 'Poor layout design' withCertainty: 0.6.
               StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.7.
               StructuredProblem is: 'Poor resource allocation policies' withCertainty: 0.6.
               StructuredProblem is: 'Lack of alternate plans' withCertainty: 0.4.
               StructuredProblem is: 'Poor scheduling (lack of alternate routings)' withCertainty: 0.8.
               StructuredProblem is: 'Outmoded philosophies' withCertainty: 0.6.
               StructuredProblem is: 'Inaccurate blueprints' withCertainty: 0.4.
```

ProblemDefinitionKB - continued

```
            StructuredProblem is: 'Incorrect BOMs' withCertainty: 0.5.
            StructuredProblem is: 'Inaccurate routings' withCertainty: 0.5.
            StructuredProblem is: 'Highly variable demand' withCertainty: 0.6]

"Executes in the context of ProblemScenario entities"

-------
HighWorkmenCompensation
    "Ties the symptom 'High workmen compensation' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'High workmen compensation'])
        then: [StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.6.
            StructuredProblem is: 'Lack of required level of education' withCertainty: 0.6.
            StructuredProblem is: 'Failing to comply with OSHA, EPA regulations' withCertainty: 0.9.
            StructuredProblem is: 'Poor safety measures' withCertainty: 0.8.
            StructuredProblem is: 'Poor (bad) ergonomics' withCertainty: 0.7.
            StructuredProblem is: 'Poor job instructions' withCertainty: 0.4]

"Executes in the context of ProblemScenario entities"

-------
InaccurateInventoryRecords
    "Ties the syptom 'Inaccurate inventory records' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Inaccurate inventory records'])
        then: [StructuredProblem is: 'Insufficient work force' withCertainty: 0.5.
            StructuredProblem is: 'Theft' withCertainty: 0.7.
            StructuredProblem is: 'Poor storage design' withCertainty: 0.8.
            StructuredProblem is: 'Poor record keeping' withCertainty: 0.9.
            StructuredProblem is: 'Poor information system' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"

-------
IncompleteProductionOrders
    "Ties the symptom 'Incomplete production orders' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Incomplete production orders'])
        then: [StructuredProblem is: 'Theft' withCertainty: 0.6.
            StructuredProblem is: 'Poor inventory policies' withCertainty: 0.6.
            StructuredProblem is: 'Deadlock' withCertainty: 0.4.
            StructuredProblem is: 'Lack of alternate plans' withCertainty: 0.4.
            StructuredProblem is: 'Poor scheduling (lack of alternate routings)' withCertainty: 0.5.
            StructuredProblem is: 'Inaccurate blueprints' withCertainty: 0.7.
            StructuredProblem is: 'Poor job instructions' withCertainty: 0.5.
            StructuredProblem is: 'Slow decision making' withCertainty: 0.5.
            StructuredProblem is: 'Poor information system' withCertainty: 0.6.
            StructuredProblem is: 'Incorrect BOMs' withCertainty: 0.6.
            StructuredProblem is: 'Inaccurate routings' withCertainty: 0.6.
            StructuredProblem is: 'Poor production planning' withCertainty: 0.8]
```

```
"Executes in the context of ProblemScenario entities"

-------
JobsWithExcessiveTardiness
    "Ties the symptom 'Jobs with excessive tardiness' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Jobs with excessive tardiness'])
        then: [StructuredProblem is: 'Theft' withCertainty: 0.4.
               StructuredProblem is: 'Low RM quality' withCertainty: 0.5.
               StructuredProblem is: 'Deadlock' withCertainty: 0.7.
               StructuredProblem is: 'Poor due date settings procedures' withCertainty: 0.8.
               StructuredProblem is: 'Inaccurate time standards' withCertainty: 0.6.
               StructuredProblem is: 'Lack of alternate plans' withCertainty: 0.4.
               StructuredProblem is: 'Inaccurate blueprints' withCertainty: 0.7.
               StructuredProblem is: 'Poor job instructions' withCertainty: 0.5.
               StructuredProblem is: 'Slow decision making' withCertainty: 0.3.
               StructuredProblem is: 'Incorrect BOMs' withCertainty: 0.6.
               StructuredProblem is: 'Inaccurate routings' withCertainty: 0.6]

"Executes in the context of ProblemScenario entities"

-------
LateDeliveriesFromVendors
    "Ties the syptom 'Late deliveries from vendors' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Late deliveries from vendors'])
        then: [StructuredProblem is: 'Unreliable vendors' withCertainty: 0.8.
               StructuredProblem is: 'Lack of backup vendors' withCertainty: 0.7.
               StructuredProblem is: 'Highly variable demand' withCertainty: 0.6.
               StructuredProblem is: 'Poor information system' withCertainty: 0.4]

"Executes in the context of ProblemScenario entities"

-------
LongCustomerOrderResponseTime
    "Ties the symptom 'Long customer order response time' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Long customer order response time'])
        then: [StructuredProblem is: 'Long lead times' withCertainty: 0.8.
               StructuredProblem is: 'Poor due date settings procedures' withCertainty: 0.8.
               StructuredProblem is: 'Outmoded philosophies' withCertainty: 0.6]

"Executes in the context of ProblemScenario entities"

-------
LongProductionFlowTimes
    "Ties the symptom 'Long production flow times' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Long production flow times'])
        then: [StructuredProblem is: 'Bottleneck' withCertainty: 0.8.
               StructuredProblem is: 'Deadlock' withCertainty: 0.6.
```

```
            StructuredProblem is: 'Highly unreliable machines' withCertainty: 0.6.
            StructuredProblem is: 'Improper batch sizes (too small or too large)' withCertainty: 0.5.
            StructuredProblem is: 'Long lead times' withCertainty: 0.9.
            StructuredProblem is: 'Poor due date settings procedures' withCertainty: 0.9.
            StructuredProblem is: 'Inaccurate time standards' withCertainty: 0.6.
            StructuredProblem is: 'Poor production planning' withCertainty: 0.7]

"Executes in the context of ProblemScenario entities"

-------
LongWaitingTimesForMachineTools
    "Ties the symptom 'Long waiting times for machine tools' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Long waiting times for machine tools'])
        then: [StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.9.
            StructuredProblem is: 'Poor resource allocation policies' withCertainty: 0.8.
            StructuredProblem is: 'Inflexible work force' withCertainty: 0.5.
            StructuredProblem is: 'Insufficient work force' withCertainty: 0.7]

"Executes in the context of ProblemScenario entities"

-------
LongWaitingTimesForMaterialMovement
    "Ties the symptom 'Long waiting times for material movement' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Long waiting times for material movement'])
        then: [StructuredProblem is: 'Inadequate MH allocation policies' withCertainty: 0.8.
            StructuredProblem is: 'Insufficient MH capacity' withCertainty: 0.9.
            StructuredProblem is: 'Outmoded MH equipment' withCertainty: 0.7.
            StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.3.
            StructuredProblem is: 'Poor resource allocation policies' withCertainty: 0.3.
            StructuredProblem is: 'Inflexible work force' withCertainty: 0.2.
            StructuredProblem is: 'Insufficient work force' withCertainty: 0.4]

"Executes in the context of ProblemScenario entities"

-------
LossOfDemand
    "Ties the syptom 'Loss of demand' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Loss of demand'])
        then: [StructuredProblem is: 'Low quality' withCertainty: 0.8.
            StructuredProblem is: 'Low customer service' withCertainty: 0.7.
            StructuredProblem is: 'Inadequate value/price relation' withCertainty: 0.7.
            StructuredProblem is: 'Shrinking market (overall)' withCertainty: 0.7.
            StructuredProblem is: 'Long lead times' withCertainty: 0.8.
            StructuredProblem is: 'Wrong marketing strategy' withCertainty: 0.8]

"Executes in the context of ProblemScenario entities"

-------
```

```
LowCustomerEvaluation
   "Ties the syptom 'Low customer evaluation' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Low customer evaluation'])
      then: [StructuredProblem is: 'Low quality' withCertainty: 0.8.
             StructuredProblem is: 'Inadequate value/price relation' withCertainty: 0.7.
             StructuredProblem is: 'Low customer service' withCertainty: 0.6.
             StructuredProblem is: 'Long lead times' withCertainty: 0.8.
             StructuredProblem is: 'Poor due date settings procedures' withCertainty: 0.5.
             StructuredProblem is: 'Wrong marketing strategy' withCertainty: 0.7]

"Executes in the context of ProblemScenario entities"

-------
LowProductionRate
   "Ties the symptom 'Low production rate' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Low production rate'])
      then: [StructuredProblem is: 'Bottleneck' withCertainty: 0.8.
             StructuredProblem is: 'Deadlock' withCertainty: 0.6.
             StructuredProblem is: 'Highly unreliable machines' withCertainty: 0.6.
             StructuredProblem is: 'Improper batch sizes (too small or too large)' withCertainty: 0.5.
             StructuredProblem is: 'Long lead times' withCertainty: 0.9.
             StructuredProblem is: 'Poor due date settings procedures' withCertainty: 0.9.
             StructuredProblem is: 'Inaccurate time standards' withCertainty: 0.6.
             StructuredProblem is: 'Poor production planning' withCertainty: 0.7]

"Executes in the context of ProblemScenario entities"

-------
LowQualityRMFromVendors
   "Ties the syptom 'Low quality Raw Material from vendors' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Low quality Raw Material from vendors'])
      then: [StructuredProblem is: 'Unreliable vendors' withCertainty: 0.8.
             StructuredProblem is: 'Lack of backup vendors' withCertainty: 0.7.
             StructuredProblem is: 'Highly variable demand' withCertainty: 0.6]

"Executes in the context of ProblemScenario entities"

-------
MissingProductionWorkOrders
   "Ties the symptom 'Missing production work orders' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Missing production work orders'])
      then: [StructuredProblem is: 'Theft' withCertainty: 0.6.
             StructuredProblem is: 'Lack of required level of education' withCertainty: 0.6.
             StructuredProblem is: 'Poor job instructions' withCertainty: 0.7.
             StructuredProblem is: 'Inaccurate blueprints' withCertainty: 0.8.
             StructuredProblem is: 'Poor record keeping' withCertainty: 0.4.
             StructuredProblem is: 'Inaccurate routings' withCertainty: 0.6.
```

ProblemDefinitionKB - continued

```
            StructuredProblem is: 'Poor information system' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"


-------
MissingTheDueDates
    "Ties the symptom 'Missing the due dates (frequent late customer deliveries)'
    to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Missing the due dates (frequent late customer deliveries)'])
        then: [StructuredProblem is: 'Bottleneck' withCertainty: 0.8.
            StructuredProblem is: 'Deadlock' withCertainty: 0.6.
            StructuredProblem is: 'Highly unreliable machines' withCertainty: 0.6.
            StructuredProblem is: 'Improper batch sizes (too small or too large)' withCertainty: 0.5.
            StructuredProblem is: 'Poor due date settings procedures' withCertainty: 0.9.
            StructuredProblem is: 'Inaccurate time standards' withCertainty: 0.6.
            StructuredProblem is: 'Poor production planning' withCertainty: 0.7]

"Executes in the context of ProblemScenario entities"


-------
PerishableItemsGoingBad
    "Ties the syptom 'Perishable items going bad' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Perishable items going bad'])
        then: [StructuredProblem is: 'Poor inventory policies' withCertainty: 0.8.
            StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.8.
            StructuredProblem is: 'Poor storage design' withCertainty: 0.6.
            StructuredProblem is: 'Highly variable demand' withCertainty: 0.6.
            StructuredProblem is: 'Poor record keeping' withCertainty: 0.3.
            StructuredProblem is: 'Poor information system' withCertainty: 0.4]

"Executes in the context of ProblemScenario entities"


-------
PoorFlowOfMaterials
    "Ties the symptom 'Poor flow of materials' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Poor flow of materials'])
        then: [StructuredProblem is: 'Poor production planning' withCertainty: 0.8.
            StructuredProblem is: 'Inadequate MH allocation policies' withCertainty: 0.6.
            StructuredProblem is: 'Insufficient MH capacity' withCertainty: 0.6.
            StructuredProblem is: 'Outmoded MH equipment' withCertainty: 0.6.
            StructuredProblem is: 'Poor layout design' withCertainty: 0.7.
            StructuredProblem is: 'Outmoded philosophies' withCertainty: 0.4.
            StructuredProblem is: 'Poor production planning' withCertainty: 0.7.
            StructuredProblem is: 'Poor scheduling (lack of alternate routings)' withCertainty: 0.6]

"Executes in the context of ProblemScenario entities"


-------
```

ProblemDefinitionKB - continued

ProductionTimeDiffersFromPredictedTime

```
    "Ties the symptom 'Production time differs from Predicted time' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Production time differs from Predicted time'])
        then: [StructuredProblem is: 'Inaccurate time standards' withCertainty: 0.8.
              StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.6.
              StructuredProblem is: 'Inadequate reward/incentive system' withCertainty: 0.5.
              StructuredProblem is: 'Low RM quality' withCertainty: 0.4.
              StructuredProblem is: 'Highly unreliable machines' withCertainty: 0.6.
              StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.5.
              StructuredProblem is: 'Poor production planning' withCertainty: 0.8]

"Executes in the context of ProblemScenario entities"
```

ResourceConflicts

```
    "Ties the symptom 'Resource conflicts' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Resource conflicts'])
        then: [StructuredProblem is: 'Poor resource allocation policies' withCertainty: 0.8.
              StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.4.
              StructuredProblem is: 'Poor production planning' withCertainty: 0.7.
              StructuredProblem is: 'Inaccurate time standards' withCertainty: 0.6.
              StructuredProblem is: 'Inaccurate/missing process plans' withCertainty: 0.5.
              StructuredProblem is: 'Poor scheduling (lack of alternate routings)' withCertainty: 0.7.
              StructuredProblem is: 'Highly variable demand' withCertainty: 0.4.
              StructuredProblem is: 'Inaccurate blueprints' withCertainty: 0.5.
              StructuredProblem is: 'Poor job instructions' withCertainty: 0.5.
              StructuredProblem is: 'Poor information system' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"
```

SlowLearningCurve

```
    "Ties the symptom 'Slow learning curve' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Slow learning curve'])
        then: [StructuredProblem is: 'Lack of required level of education' withCertainty: 0.9.
              StructuredProblem is: 'Poor employee discipline' withCertainty: 0.5.
              StructuredProblem is: 'Poor job instructions' withCertainty: 0.6.
              StructuredProblem is: 'Poor (bad) ergonomics' withCertainty: 0.8.
              StructuredProblem is: 'Inadequate reward/incentive system' withCertainty: 0.8]

"Executes in the context of ProblemScenario entities"
```

StockoutAndOrHighInventoryOfSomeFGItems

```
    "Ties the symptom 'Stockout and/or high inventory level of some Finished Good items'
    to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Stockout and/or high inventory level of some Finished Good
        items'])
        then: [StructuredProblem is: 'Theft' withCertainty: 0.4.
            StructuredProblem is: 'Poor inventory policies' withCertainty: 0.6.
            StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.8.
            StructuredProblem is: 'Insufficient capacity of resources' withCertainty: 0.4.
            StructuredProblem is: 'Poor production planning' withCertainty: 0.7.
            StructuredProblem is: 'Highly variable demand' withCertainty: 0.6.
            StructuredProblem is: 'Outmoded philosophies' withCertainty: 0.5.
            StructuredProblem is: 'Poor record keeping' withCertainty: 0.6.
            StructuredProblem is: 'Poor information system' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"
```

StockoutAndOrHighInventoryOfSomeRMItems

```
    "Ties the symptom 'Stockout and/or high inventory level of some Raw Material items'
    to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Stockout and/or high inventory level of some Raw Material
        items'])
        then: [StructuredProblem is: 'Theft' withCertainty: 0.7.
            StructuredProblem is: 'Poor inventory policies' withCertainty: 0.9.
            StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.7.
            StructuredProblem is: 'Low RM quality' withCertainty: 0.4.
            StructuredProblem is: 'Incorrect BOMs' withCertainty: 0.4.
            StructuredProblem is: 'Highly variable demand' withCertainty: 0.6.
            StructuredProblem is: 'Poor record keeping' withCertainty: 0.7.
            StructuredProblem is: 'Poor information system' withCertainty: 0.5]

"Executes in the context of ProblemScenario entities"
```

TooManyJobsAtTopPriority

```
    "Ties the symptom 'Too many jobs at top priority' to the possible causes/problems"

    if: (anyOf: Symptoms have: [Name = 'Too many jobs at top priority'])
        then: [StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.7.
            StructuredProblem is: 'Poor production planning' withCertainty: 0.7.
            StructuredProblem is: 'Wrong marketing strategy' withCertainty: 0.5.
            StructuredProblem is: 'Highly variable demand' withCertainty: 0.8.
            StructuredProblem is: 'Poor information system' withCertainty: 0.6]

"Executes in the context of ProblemScenario entities"
```

```
TooManyUnplannedActivities
   "Ties the symptom 'Too many unplanned activities (due to poor planning)'
   to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Too many unplanned activities (due to poor planning)'])
      then: [StructuredProblem is: 'Poor demand forecasting' withCertainty: 0.7.
           StructuredProblem is: 'Poor production planning' withCertainty: 0.9.
           StructuredProblem is: 'Wrong marketing strategy' withCertainty: 0.5.
           StructuredProblem is: 'Highly variable demand' withCertainty: 0.8.
           StructuredProblem is: 'Poor information system' withCertainty: 0.6]

"Executes in the context of ProblemScenario entities"

-------
WorkerTardiness
   "Ties the symptom 'Worker tardiness' to the possible causes/problems"

   if: (anyOf: Symptoms have: [Name = 'Worker tardiness'])
      then: [StructuredProblem is: 'Lack of training, know-how' withCertainty: 0.9.
           StructuredProblem is: 'Lack of required level of education' withCertainty: 0.7.
           StructuredProblem is: 'Inadequate reward/incentive system' withCertainty: 0.7.
           StructuredProblem is: 'Poor corporate culture' withCertainty: 0.5.
           StructuredProblem is: 'Poor job instructions' withCertainty: 0.7.
           StructuredProblem is: 'Improper performance measurement system' withCertainty: 0.3]

"Executes in the context of ProblemScenario entities"

-------


--------------------------------------------
Rule Metrics
--------------------------------------------
Total Number of Entity Types: 2
Total Number of Rules: 51
An Average Rule Tests 1.0 Parameters
An Average Rules Makes Conclusions About 1.0 Parameters
Total Number of Parameter Definitions: 2
2 parameters with 51 associated rules, 100%
```

ProblemToToolKB, a HUMBLE knowledge base
as of 16 November 1996 6:41:55 pm

```
--------------------------------------------
Entity Types
--------------------------------------------
ToolSelectionScenarioForAProblem
    typeAbove: nil
    createPrompt: Are there any Problem scenarios to consider?
    addPrompt: Are there any other Problem scenarios  to consider?
    assumePrompt: I am creating a Problem scenario
    defaultName: ProblemToTool_Scenario
    parameters: #(#Tool #selectedProblem)
    mainParameters: #()

--------------------------------------------
Parameter Definitions
--------------------------------------------
selectedProblem
    describes: ToolSelectionScenarioForAProblem
    type: String
    prompt: What is the selected problem for & ?
    promptFlag: askFirst
    explanation: "Selected problem is the the scenario that needs to be addressed by a tool"
    remark: "Standard string parameter used to store the final selected problem of the current scenario"
    deducingRules: #()
    changeBlock: '[:parameter |]'

Tool
    describes: ToolSelectionScenarioForAProblem
    type: String
    prompt: What is the most appropriate set of tool that can address & ?
    promptFlag: askLast
    explanation: "Tool is the solver(s) that can address the given situation"
    remark: "Standard string parameter used to store the final tool(s) for the current scenario"
    deducingRules: #(#OutmodedMHEquipmentOrPoorRecordKeeping
#ImproperPerformanceMeasurementSystemOrPoorInformationSystem
#IncorrectBOMsOrInaccurateRoutings #ExcessiveMachineOrMHCapacity
#PoorDueDateSettingsProceduresOrPoorProductionPlanning #InsufficientMHCapacityOrPoorScheduling
#LowQualityOrToolWearOrPoorPlaningByCustomers
#PoorDemandForecastingOrLackOfBackupVendors #ExcessiveOrInsufficientWorkForce
#concludesToOtherAnalysisTechniques #concludesToNonModelingApproaches #PoorInformationSystem
#OutmodedPhilosophies #HighlyVariableDemand #InaccurateTimeStandards
#PoorResourceAllocationPolicies #InsufficientCapacityOfResources #PoorLayoutDesign
#InadequateMHAllocationPolicies #LongLeadTimes #LowCustomerService #ImproperBatchSizes
#InadequateMaintenance #HighlyUnreliableMachines #Deadlock #Bottleneck #UnreliableVendors
#PoorInventoryPolicies)
    changeBlock: '[:parameter |]'
```

```
------------------------------------------
Rules
------------------------------------------
Bottleneck
    "Ties the selected problem 'Bottleneck' to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Bottleneck')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing' withCertainty: 0.7.
              Tool is: 'Search-based optimization with simulation' withCertainty: 0.7]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
concludesToNonModelingApproaches
    "Ties the selected problems to non-modeling approaches.
    Here the available tools are not suitable for the specified problems"

    if: (selectedProblem = 'Poor employee discipline') |
        (selectedProblem = 'Lack of training, know-how') |
        (selectedProblem = 'Lack of required level of education') |
        (selectedProblem = 'Inflexible work force') |
        (selectedProblem = 'Inadequate reward/incentive system') |
        (selectedProblem = 'Failing to comply with OSHA, EPA regulations') |
        (selectedProblem = 'Poor safety measures') |
        (selectedProblem = 'Poor corporate culture') |
        (selectedProblem = 'Poor (bad) ergonomics') |
        (selectedProblem = 'Theft') |
        (selectedProblem = 'Low RM quality') |
        (selectedProblem = 'Inaccurate/missing process plans') |
        (selectedProblem = 'Lack of backup plans') |
        (selectedProblem = 'Lack of alternate plans') |
        (selectedProblem = 'Usage of inappropriate tools and fixtures') |
        (selectedProblem = 'Inaccurate blueprints') |
        (selectedProblem = 'Poor part design (complex products)') |
        (selectedProblem = 'Poor job instructions') |
        (selectedProblem = 'Obsolete technology') |
        (selectedProblem = 'Slow decision making') |
        (selectedProblem = 'Poor benchmarking')

        then: [Tool is: 'Non-Analytic techniques (e.g. Observation)' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
```

```
concludesToOtherAnalysisTechniques
    "Ties the selected problems to other modeling approaches (analytic techniques)
    such as statistical solutions, time series, layout modeling etc.
    These tools (techniques) are not available in the proposed system"

    if: (selectedProblem = 'Poor storage design') |
        (selectedProblem = 'Inadequate value/price relation') |
        (selectedProblem = 'Lack of demand') |
        (selectedProblem = 'Shrinking market (overall)') |
        (selectedProblem = 'Wrong marketing strategy')

        then: [Tool is: 'Other analysis techniques' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
Deadlock
    "Ties the selected problem 'Deadlock' to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Deadlock')

        then: [Tool is: 'Simulation' withCertainty: 0.6.
            Tool is: 'Petri nets' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
ExcessiveMachineOrMHCapacity
    "Ties the selected problem 'Excessive machine capacity'
    or 'Excessive material handler capacity' to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Excessive machine capacity') |
        (selectedProblem = 'Excessive material handler capacity')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
            Tool is: 'Queueing' withCertainty: 0.8.
            Tool is: 'Petri nets' withCertainty: 0.7.
            Tool is: 'Search-based optimization with simulation' withCertainty: 0.7]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
ExcessiveOrInsufficientWorkForce
    "Ties the selected problem 'Insufficient work force'
    or 'Excessive work force' to the possible tool(s)/solver(s)."

    if: (selectedProblem = 'Insufficient work force') |
        (selectedProblem = 'Excessive work force')
        then: [Tool is: 'Simulation' withCertainty: 0.8.
            Tool is: 'Queueing + Simulation' withCertainty: 0.4.
            Tool is: 'Other analysis techniques' withCertainty: 0.7]
```

ProblemToToolKB - continued

```
"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
HighlyUnreliableMachines
    "Ties the selected problem 'Highly unreliable machines' to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Highly unreliable machines')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
               Tool is: 'Queueing' withCertainty: 0.6.
               Tool is: 'Search-based optimization with simulation' withCertainty: 0.5]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
HighlyVariableDemand
    "Ties the selected problem 'Highly variable demand' to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Highly variable demand')

        then: [Tool is: 'Simulation' withCertainty: 0.6.
               Tool is: 'Search-based optimization with simulation' withCertainty: 0.5.
               Tool is: 'Other analysis techniques' withCertainty: 0.9]


"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
ImproperBatchSizes
    "Ties the selected problem 'Improper batch sizes (too small or too large)'
    to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Improper batch sizes (too small or too large)')

        then: [Tool is: 'Simulation' withCertainty: 0.7.
               Tool is: 'Queueing' withCertainty: 0.6.
               Tool is: 'Queueing + Simulation' withCertainty: 0.8.
               Tool is: 'Search-based optimization with simulation' withCertainty: 0.9]


"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
ImproperPerformanceMeasurementSystemOrPoorInformationSystem
    "Ties the selected problem 'Improper performance measurement system'
    or 'Poor information system' to the possible tool(s)/solver(s)"

    if:  (selectedProblem = 'Improper performance measurement system') |
         (selectedProblem = 'Poor information system')
         then: [Tool is: 'Simulation' withCertainty: 0.7.
                Tool is: 'Other analysis techniques' withCertainty: 0.9]
```

ProblemToToolKB - continued

```
"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
InaccurateTimeStandards
    "Ties the selected problem 'Inaccurate time standards' to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Inaccurate time standards')

        then: [Tool is: 'Simulation' withCertainty: 0.6.
              Tool is: 'Other analysis techniques' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
InadequateMaintenance
    "Ties the selected problem 'Inadequate maintenance' to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Inadequate maintenance')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing' withCertainty: 0.6]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
InadequateMHAllocationPolicies
    "Ties the selected problem 'Inadequate MH allocation policies'
    to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Inadequate MH allocation policies')

        then: [Tool is: 'Simulation' withCertainty: 0.8.
              Tool is: 'Search-based optimization with simulation' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
IncorrectBOMsOrInaccurateRoutings
    "Ties the selected problem 'Incorrect BOMs'
    or 'Inaccurate routings' to the possible tool(s)/solver(s)"

    if:  (selectedProblem = 'Incorrect BOMs') |
         (selectedProblem = 'Inaccurate routings')

        then: [Tool is: 'Simulation' withCertainty: 0.7.
              Tool is: 'Other analysis techniques' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
```

```
InsufficientCapacityOfResources
    "Ties the selected problem 'Insufficient capacity of resources'
    to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Insufficient capacity of resources')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing' withCertainty: 0.7.
              Tool is: 'Queueing + Simulation' withCertainty: 0.7]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
InsufficientMHCapacityOrPoorScheduling
    "Ties the selected problem 'Insufficient MH capacity'
    or 'Poor scheduling (lack of alternate routings)' to the possible tool(s)/solver(s)"

    if:  (selectedProblem = 'Insufficient MH capacity') |
        (selectedProblem = 'Poor scheduling (lack of alternate routings)')

        then: [Tool is: 'Simulation' withCertainty: 0.7.
              Tool is: 'Queueing' withCertainty: 0.6.
              Tool is: 'Petri nets' withCertainty: 0.4.
              Tool is: 'Queueing + Simulation' withCertainty: 0.8.
              Tool is: 'Search-based optimization with simulation' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
LongLeadTimes
    "Ties the selected problem 'Long lead times' to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Long lead times')

        then: [Tool is: 'Simulation' withCertainty: 0.8.
              Tool is: 'Queueing' withCertainty: 0.7.
              Tool is: 'Petri nets' withCertainty: 0.7.
              Tool is: 'Search-based optimization with simulation' withCertainty: 0.5]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
LowCustomerService
    "Ties the selected problem 'Low customer service' to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Low customer service')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Search-based optimization with simulation' withCertainty: 0.5.
              Tool is: 'Other analysis techniques' withCertainty: 0.9]
```

ProblemToToolKB - continued

```
"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
LowQualityOrToolWearOrPoorPlaningByCustomers
   "Ties the selected problem 'Low quality'
   or 'Tool wear' or 'Poor planning by customers to the possible tool(s)/solver(s)"

   if: (selectedProblem = 'Low quality') |
      (selectedProblem = 'Tool wear') |
      (selectedProblem = 'Poor plannig by customers')

      then: [Tool is: 'Simulation' withCertainty: 0.6.
           Tool is: 'Other analysis techniques' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
OutmodedMHEquipmentOrPoorRecordKeeping
   "Ties the selected problem 'Outmoded MH equipment'
   or 'Poor record keeping' to the possible tool(s)/solver(s)"

   if: (selectedProblem = 'Outmoded MH equipment') |
      (selectedProblem = 'Poor record keeping')

      then: [Tool is: 'Simulation' withCertainty: 0.5.
           Tool is: 'Other analysis techniques' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
OutmodedPhilosophies
   "Ties the selected problem 'Outmoded philosophies' to the possible tool(s)/solver(s)"

   if: (selectedProblem = 'Outmoded philosophies')

      then: [Tool is: 'Simulation' withCertainty: 0.8.
           Tool is: 'Search-based optimization with simulation' withCertainty: 0.5.
           Tool is: 'Other analysis techniques' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
PoorDemandForecastingOrLackOfBackupVendors
   "Ties the selected problem 'Poor demand forecasting'
   or 'Lack of backup vendors' to the possible tool(s)/solver(s)"

   if: (selectedProblem = 'Poor demand forecasting') |
      (selectedProblem = 'Lack of backup vendors')

      then: [Tool is: 'Simulation' withCertainty: 0.3.
           Tool is: 'Other analysis techniques' withCertainty: 0.9]
```

ProblemToToolKB - continued

```
"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
PoorDueDateSettingsProceduresOrPoorProductionPlanning
   "Ties the selected problem 'Poor due date settings procedures'
   to 'Poor production planning' to the possible tool(s)/solver(s)"

   if: (selectedProblem = 'Poor due date settings procedures') |
      (selectedProblem = 'Poor production planning')

      then: [Tool is: 'Simulation' withCertainty: 0.7.
            Tool is: 'Search-based optimization with simulation' withCertainty: 0.9.
            Tool is: 'Other analysis techniques' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
PoorInformationSystem
   "Ties the selected problem 'Poor information system' to the possible tool(s)/solver(s)"

   if: (selectedProblem = 'Poor information system')

      then: [Tool is: 'Simulation' withCertainty: 0.7.
            Tool is: 'Other analysis techniques' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
PoorInventoryPolicies
   "Ties the selected problem 'Poor inventory policies' to the possible tool(s)/solver(s)"

   if: (selectedProblem = 'Poor inventory policies')

      then: [Tool is: 'Simulation' withCertainty: 0.6.
            Tool is: 'Queueing' withCertainty: 0.4.
            Tool is: 'Queueing + Simulation' withCertainty: 0.7.
            Tool is: 'Search-based optimization with simulation' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
PoorLayoutDesign
   "Ties the selected problem 'Poor layout design' to the possible tool(s)/solver(s)"

   if: (selectedProblem = 'Poor layout design')

      then: [Tool is: 'Simulation' withCertainty: 0.8.
            Tool is: 'Other analysis techniques' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAProblem entities"
```

```
-------
PoorResourceAllocationPolicies
    "Ties the selected problem 'Poor resource allocation policies'
    to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Poor resource allocation policies')

        then: [Tool is: 'Simulation' withCertainty: 0.7.
              Tool is: 'Queueing' withCertainty: 0.6.
              Tool is: 'Petri nets' withCertainty: 0.4.
              Tool is: 'Search-based optimization with simulation' withCertainty: 0.9]


"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------
UnreliableVendors
    "Ties the selected problem 'Unreliable vendors' to the possible tool(s)/solver(s)"

    if: (selectedProblem = 'Unreliable vendors')

        then: [Tool is: 'Simulation' withCertainty: 0.5.
              Tool is: 'Other analysis techniques' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAProblem entities"

-------


-------------------------------------------
Rule Metrics
-------------------------------------------
Total Number of Entity Types: 1
Total Number of Rules: 28
An Average Rule Tests 1.0 Parameters
An Average Rules Makes Conclusions About 1.0 Parameters
Total Number of Parameter Definitions: 2
2 parameters with 28 associated rules, 100%
```

WhatIfToToolKB, a HUMBLE knowledge base
as of 16 November 1996 6:41:57 pm

```
------------------------------------------
Entity Types
------------------------------------------
ToolSelectionScenarioForAWhatIf
    typeAbove: nil
    createPrompt: Are there any What-If scenarios to consider?
    addPrompt: Are there any other What-If scenarios  to consider?
    assumePrompt: I am creating a What-If scenario
    defaultName: What-IF_Scenario
    parameters: #(#whatIfQuestion #Tool)
    mainParameters: #()

------------------------------------------
Parameter Definitions
------------------------------------------
Tool
    describes: ToolSelectionScenarioForAWhatIf
    type: String
    prompt: What is the most appropriate set of tools that can address & ?
    promptFlag: askLast
    explanation: "Tool is the solver(s) that can address the given situation"
    remark: "Standard string parameter used to store the final tool(s) for the current scenario"
    deducingRules: #(#ChangeLotSizesOrChangeScrapRate
#ChangeOperatorCapabilitiesOrChangeWorkerAssignments
#ChangeEquipmentReliabilityParametersOrChangeProductDesignOrChangeBufferCapacities
#ChangeMaterialHandlerAllocationPolicyOrChangeSpanOfControl #concludesInSimulationToolOnly01
#concludesInSimulationToolOnly02 #AddOrRemoveAWorkStationOrChangeSetupOrProcessingTimes
#AddOrRemoveAnAssemblyStationOrAMaterialHandler #ChangeOperatorAssignmentPolicies
#ChangeOperationSequence #ChangeSamplingRates #ChangeProductVersions
#ChangeDemandForProductMix #CombineOperations #ChangeRouting #ChangeBillOfMaterials
#ChangePlantLocation #ChangeNumberOfWorkShifts #AddOrRemoveOperators
#AddOrRemoveToolsFixturesPallets #AddOrRemoveAManufacturingLine #ChangePlantLayout
#AddOrRemoveAProduct)
    changeBlock: '[:parameter |]'

whatIfQuestion
    describes: ToolSelectionScenarioForAWhatIf
    type: String
    prompt: What is the what-if question for & ?
    promptFlag: askFirst
    explanation: "What-if question is the the scenario that needs to be addressed by a tool"
    remark: "Standard string parameter used to store the final what-if question of the current scenario"
    deducingRules: #()
    changeBlock: '[:parameter |]'
```

```
------------------------------------------
Rules
------------------------------------------
AddOrRemoveAManufacturingLine
    "Ties the what-if question 'Add/remove a manufacturing line'
    to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Add/remove a manufacturing line')

        then: [Tool is: 'Simulation' withCertainty: 0.8.
              Tool is: 'Queueing' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
AddOrRemoveAnAssemblyStationOrAMaterialHandler
    "Ties the what-if question 'Add/remove an assembly station'
    or 'Add/remove a material handler' to the possible tool(s)/solver(s)"

    if:  (whatIfQuestion = 'Add/remove an assembly station') |
         (whatIfQuestion = 'Add/remove a material handler')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing' withCertainty: 0.8.
              Tool is: 'Queueing + Simulation' withCertainty: 0.6]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
AddOrRemoveAProduct
    "Ties the what-if question 'Add/remove a product from the product-mix'
    to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Add/remove a product from the product-mix')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing' withCertainty: 0.8.
              Tool is: 'Queueing + Simulation' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
AddOrRemoveAWorkStationOrChangeSetupOrProcessingTimes
    "Ties the what-if question 'Add/remove a work station'
    or 'Change setup/processing times' to the possible tool(s)/solver(s)"

    if:  (whatIfQuestion = 'Add/remove a work station') |
         (whatIfQuestion = 'Change setup/processing times')
        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing' withCertainty: 0.8.
              Tool is: 'Queueing + Simulation' withCertainty: 0.9]
```

WhatIfToToolKB - continued

```
"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
AddOrRemoveOperators
    "Ties the what-if question 'Add/remove operators' to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Add/remove operators')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing + Simulation' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
AddOrRemoveToolsFixturesPallets
    "Ties the what-if question 'Add/remove tools, fixtures, pallets, etc.'
    to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Add/remove tools, fixtures, pallets, etc.')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing + Simulation' withCertainty: 0.6]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeBillOfMaterials
    "Ties the what-if question 'Change bill of materials (add/remove a component)'
    to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Change bill of materials (add/remove a component)')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing' withCertainty: 0.6.
              Tool is: 'Queueing + Simulation' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeDemandForProductMix
    "Ties the what-if question 'Change demand for the product mix'
    to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Change demand for the product mix')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing' withCertainty: 0.7.
              Tool is: 'Petri nets' withCertainty: 0.4.
              Tool is: 'Queueing + Simulation' withCertainty: 0.6.
              Tool is: 'Other analysis techniques' withCertainty: 0.8]
```

WhatIfToToolKB - continued

```
"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeEquipmentReliabilityParametersOrChangeProductDesignOrChangeBufferCapacities
   "Ties the what-if question 'Change equipment reliability parameters (add/remove a maintenance crew)'
   or 'Change product design' or 'Change buffer capacities' to the possible tool(s)/solver(s)"

   if:  (whatIfQuestion = 'Change equipment reliability parameters (add/remove a maintenance crew)') |
        (whatIfQuestion = 'Change product design') |
        (whatIfQuestion = 'Change buffer capacities')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing' withCertainty: 0.6.
              Tool is: 'Queueing + Simulation' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeLotSizesOrChangeScrapRate
   "Ties the what-if question 'Change lot sizes'
   or 'Change defect/scrap rate' to the possible tool(s)/solver(s)"

   if:  (whatIfQuestion = 'Change lot sizes') |
        (whatIfQuestion = 'Change defect/scrap rate')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Queueing' withCertainty: 0.7.
              Tool is: 'Queueing + Simulation' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeMaterialHandlerAllocationPolicyOrChangeSpanOfControl
   "Ties the what-if question 'Change material handler allocation policy'
   or 'Change control domain (span of control)' to the possible tool(s)/solver(s)"

   if:  (whatIfQuestion = 'Change material handler allocation policy') |
        (whatIfQuestion = 'Change control domain (span of control)')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
              Tool is: 'Petri nets' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeNumberOfWorkShifts
   "Ties the what-if question 'Change # of work shifts (for some stations)'
   to the possible tool(s)/solver(s)"

   if: (whatIfQuestion = 'Change # of work shifts (for some stations)')
```

WhatIfToToolKB - continued

```
        then: [Tool is: 'Simulation' withCertainty: 0.8.
               Tool is: 'Queueing' withCertainty: 0.6]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeOperationSequence
    "Ties the what-if question 'Change operation sequence'
    to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Change operation sequence')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
               Tool is: 'Queueing' withCertainty: 0.7.
               Tool is: 'Queueing + Simulation' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeOperatorAssignmentPolicies
    "Ties the what-if question 'Change operator assignment policies'
    to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Change operator assignment policies')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
               Tool is: 'Petri nets' withCertainty: 0.5.
               Tool is: 'Queueing + Simulation' withCertainty: 0.7]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeOperatorCapabilitiesOrChangeWorkerAssignments
    "Ties the what-if question 'Change operator capabilities (cross-training)'
    or 'Change worker assignments' to the possible tool(s)/solver(s)"

    if:  (whatIfQuestion = 'Change operator capabilities (cross-training)') |
         (whatIfQuestion = 'Change worker assignments')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
               Tool is: 'Queueing' withCertainty: 0.7]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangePlantLayout
    "Ties the what-if question 'Change plant layout (add/remove an aisle)'
    to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Change plant layout (add/remove an aisle)')
```

WhatIfToToolKB - continued

```
        then: [Tool is: 'Simulation' withCertainty: 0.8.
               Tool is: 'Other analysis techniques' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangePlantLocation
    "Ties the what-if question 'Change plant location' to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Change plant location')

        then: [Tool is: 'Simulation' withCertainty: 0.7.
               Tool is: 'Other analysis techniques' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeProductVersions
    "Ties the what-if question 'Change product versions' to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Change product versions')

        then: [Tool is: 'Non-Analytic techniques (e.g. Observation)' withCertainty: 0.6.
               Tool is: 'Other analysis techniques' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeRouting
    "Ties the what-if question 'Change routing (add/remove an alternate routing)'
    to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Change routing (add/remove an alternate routing)')

        then: [Tool is: 'Simulation' withCertainty: 0.9.
               Tool is: 'Queueing' withCertainty: 0.6.
               Tool is: 'Petri nets' withCertainty: 0.4.
               Tool is: 'Queueing + Simulation' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
ChangeSamplingRates
    "Ties the what-if question 'Change sampling rates' to the possible tool(s)/solver(s)"

    if: (whatIfQuestion = 'Change sampling rates')

        then: [Tool is: 'Non-Analytic techniques (e.g. Observation)' withCertainty: 0.7.
               Tool is: 'Simulation' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"
```

WhatIfToToolKB - continued

```
-------
CombineOperations
   "Ties the what-if question 'Combine operations' to the possible tool(s)/solver(s)"

   if: (whatIfQuestion = 'Combine operations')

      then: [Tool is: 'Simulation' withCertainty: 0.9.
           Tool is: 'Queueing' withCertainty: 0.7.
           Tool is: 'Queueing + Simulation' withCertainty: 0.6]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
concludesInSimulationToolOnly01
   "Ties a set of what-if question to the 'Simulation' tool"

   if:  (whatIfQuestion = 'Change production philosophy (MRP, JIT, Heuristics)') |
      (whatIfQuestion = 'Change plant inventory policy (for RM, FG, etc.)') |
      (whatIfQuestion = 'Change primary/secondary part selection policies from input queues of
                          resources') |
      (whatIfQuestion = 'Change order acceptance and/or order release policies') |
      (whatIfQuestion = 'Change Kanban size (if applicable)')

      then: [Tool is: 'Simulation' withCertainty: 0.8]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------
concludesInSimulationToolOnly02
   "Ties a set of what-if question to the 'Simulation' tool"

   if:  (whatIfQuestion = 'Add/remove a stock room') |
      (whatIfQuestion = 'Change material flow control policies (push vs. pull)') |
      (whatIfQuestion = 'Change overtime policy')

      then: [Tool is: 'Simulation' withCertainty: 0.9]

"Executes in the context of ToolSelectionScenarioForAWhatIf entities"

-------


-------------------------------------------
Rule Metrics
-------------------------------------------
Total Number of Entity Types: 1
Total Number of Rules: 23
An Average Rule Tests 1.0 Parameters
An Average Rules Makes Conclusions About 1.0 Parameters
Total Number of Parameter Definitions: 2
2 parameters with 23 associated rules, 100%
```

APPENDIX F

SMALLTALK-80 CODE

Introduction

This appendix contains listings of Smalltalk-80 code beginning on the next page. The listings are relevant portions of new and/or modified classes and methods that were used for the development of EFES.

The listings are separated by class. Within each class, there is a header section followed by the listings of related methods. The header section contains the class hierarchy specifications as well as the names of all instance and class variables. A comment segment concludes the header section.

The methods are divided into groups (protocols) of related methods. This grouping is arbitrary but usually provides some insight as to the general intend of the method in the group. The group headers are designated by the character string "!*className* methodsFor: *groupName*". The last grouping under a method (if listed) is the group for class methods. These methods are used by the class rather than instances of the class. A good example of their use is the creation of a new instance.

Methods listings always start with the method name including any incoming parameters. The names are free form except that a colon is used to separate the parameters from the name of the method. The code itself follows Smalltalk-80 convention. Any text within the method enclosed by quotation marks is a comment. All methods terminate with an exclamation point (!).

Related SmallTalk-80 Code For Class: BaseModel

```
Model subclass: #BaseModel

    instanceVariableNames: 'name fileName plant itemMaster bomMaster routingMaster
    plantLayoutModel '
    classVariableNames: 'CurrentBaseModel '
    poolDictionaries: ''
    category: 'Base Model'!

BaseModel comment:

        'This is the model of the whole organization.  Current implementation is limited only to the
        manufacturing aspects of the organization.  Serves as a substrate from which tool and problem
        dependant models can be generated.

        instanceVariableNames
        name            <String>  name of the organization
        plant           <Plant>       manufacturing system
        ItemMaster      <Dictionary> key: product name value: <Item>
        bomMaster       <Dictionary> key: product name value: <BomPart>
        routingMaster       <Dictionary> key: product name value: <Routing>

        classVariableNames
        CurrentBaseModel <BaseModel>reference to the active instance of BaseModel'!

!BaseModel methodsFor: 'efes extension'!

numberOfProducts
    ^self finishedGoods size!

productBredths
    | aColl |
    aColl := SortedCollection new.
    self finishedGoods do: [:fg | aColl add: (self bomMaster at: fg) bredth].
    ^aColl!

productDepths
    | aColl |
    aColl := SortedCollection new.
    self finishedGoods do: [:fg | aColl add: (self bomMaster at: fg) depth].
    ^aColl!

productRoutings
    | aColl |
    aColl := SortedCollection new.
    self finishedGoods do: [:fg | aColl add: (self routingMaster at: fg) routingList size].
    ^aColl! !
```

Related SmallTalk-80 Code For Class: EnvironmentInterface

```
GPApplicationModelWithHelp subclass: #EnvironmentInterface

    instanceVariableNames: 'baseModelInterface baseModelName solverName solverDict solverList
        outputInterface parentInterface '
    classVariableNames: 'ActiveInterface '
    poolDictionaries: ''
    category: 'Environment Interface'!

EnvironmentInterface comment:

        'The EnvironmentInterface class is the main menu or launcher for the
        entire manufacturing system analysis environment.  From this interface
        the user can access the base model, or specify a problem for analysis and
        execute the solver.

        There are several windows associated with this class.  The windows are accessed
        via the buttons of the main interface for this class.  When new solvers are added,
        the intialize class will have to be altered.  The current text messages are listed in
        a so-named protocol.

        instanceVariableNames
        baseModelName    <ValueHolder on a String> holds name of current BaseModel
        solverName       <ValueHolder on a String>  holds name of selected solver
        solverDict   <Dictionary> keys are execution tools (solvers) and the entries are the
          appropriate translator
        solverList    <SelectionInList> keys of solverDict'!

!EnvironmentInterface methodsFor: 'efes extention'!

startAdvisory
    "Switch the control to the Experimental Frame Expert System class"

    | efes |
    efes := ExperimentalFrameExpertSystem new.
    efes parentInterface: self.
    efes startAdvisory!

!EnvironmentInterface methodsFor: 'base model interfaces'

newBaseModel
    ^self baseModelInterface newBaseModel!

reviewTools
    ^self baseModelInterface reviewTools! !
```

Related SmallTalk-80 Code For Class: ExperimentalFrameExpertSystem

```
EnvironmentInterface subclass: #ExperimentalFrameExpertSystem

    instanceVariableNames: 'structuredProblemsList whatIfAnalysisList mgfSysComplexity
        whatIfQuestion selectedProblem expertInputMatrix adjustingFactors
        intermediateRecommendations '
    classVariableNames: "
    poolDictionaries: "
    category: 'Experimental Frame Expert System'!

!ExperimentalFrameExpertSystem methodsFor: 'accessing'!

adjustingFactors
    ^adjustingFactors!

intermediateRecommendations
    ^intermediateRecommendations!

mgfSysComplexity
    ^mgfSysComplexity!

selectedProblem
    ^selectedProblem!

whatIfQuestion
    ^whatIfQuestion! !

!ExperimentalFrameExpertSystem methodsFor: 'initialize'!

initialize
    "Here, we initialize the instance variables related to the EFES"

    super initialize.

    mgfSysComplexity := 'high'.

    adjustingFactors := Dictionary new.
    adjustingFactors at:'Simulation' put: 0.15; at:'Queueing' put: -0.15; at:'Petri nets' put:-0.10;
        at:'Queueing + Simulation' put: 0.17; at:'Search-based optimization with simulation' put: 0.25.

    expertInputMatrix := (Matrix new:3 by:5) collection:#(50 -25 17 -50 50 -50 -25 25 -25 -17 17 25 25 -
        50 50).

    (structuredProblemsList := OrderedCollection new) addAll: #('Poor employee discipline' 'Lack of
        training, know-how' 'Lack of required level of education' 'Inflexible work force' 'Inadequate
        reward/incentive system' 'Failing to comply with OSHA, EPA regulations' 'Poor safety measures'
        'Poor corporate culture' 'Poor (bad) ergonomics' 'Insufficient work force' 'Excessive work force'
        'Theft' 'Poor inventory policies' 'Poor demand forecasting' 'Poor storage design' 'Unreliable vendors'
        'Lack of backup vendors' 'Low RM quality' 'Bottleneck' 'Deadlock' 'Highly unreliable machines'
        'Inadequate maintenance' 'Improper batch sizes (too small or too large)' 'Low quality' 'Tool wear'
        'Low customer service' 'Inadequate value/price relation' 'Long lead times' 'Inadequate MH
        allocation policies' 'Insufficient MH capacity' 'Outmoded MH equipment' 'Poor layout design'
        'Insufficient capacity of resources' 'Poor resource allocation policies' 'Poor due date settings
    procedures' 'Poor production planning' 'Inaccurate time standards' 'Inaccurate/missing process
    plans' 'Lack of backup plans' 'Lack of alternate plans' 'Poor scheduling (lack of alternate routings)'
    'Usage of inappropriate tools and fixtures' 'Excessive machine capacity' 'Excessive material handler
    capacity' 'Highly variable demand' 'Lack of demand' 'Shrinking market (overall)' 'Outmoded
    philosophies' 'Inaccurate blueprints' 'Poor part design (complex products)' 'Poor job instructions'
    'Obsolete technology' 'Slow decision making' 'Wrong marketing strategy' 'Poor record keeping'
    'Poor benchmarking' 'Incorrect BOMs' 'Inaccurate routings' 'Improper performance measurement
    system' 'Poor information system' 'Poor plannig by customers').

  (whatIfAnalysisList := OrderedCollection new) addAll: #('&& Physical &&' 'Add/remove a work
    station' 'Add/remove an assembly station' 'Add/remove a material handler' 'Add/remove a stock
    room' 'Add/remove a product from the product-mix' 'Change plant layout (add/remove an aisle)'
    'Add/remove a manufacturing line' 'Add/remove tools, fixtures, pallets, etc.' 'Change equipment
    reliability parameters (add/remove a maintenance crew)' 'Add/remove operators' 'Change operator
    capabilities (cross-training)' 'Change worker assignments' 'Change product design' 'Change # of
    work shifts (for some stations)' 'Change plant location' 'Change buffer capacities' '&&
    Informational &&' 'Change bill of materials (add/remove a component)' 'Change routing
    (add/remove an alternate routing)' 'Combine operations' 'Change demand for the product mix'
    'Change lot sizes' 'Change product versions' 'Change sampling rates' 'Change operation sequence'
    'Change setup/processing times' 'Change defect/scrap rate' '&& Control/Decisional &&' 'Change
    production philosophy (MRP, JIT, Heuristics)' 'Change plant inventory policy (for RM, FG, etc.)'
    'Change material handler allocation policy' 'Change primary/secondary part selection policies from
    input queues of resources' 'Change control domain (span of control)' 'Change order acceptance
    and/or order release policies' 'Change operator assignment policies' 'Change material flow control
    policies (push vs. pull)' 'Change Kanban size (if applicable)' 'Change overtime policy').! !

!ExperimentalFrameExpertSystem methodsFor: 'efes question types'!

binaryChoice: title
  | theAnswer |
  theAnswer := Hypothesis new: (GPDialog confirm: title).
  ^theAnswer!

fillInTitled: title
  | theAnswer |
  "get the answer from among a set of choices"

  theAnswer := Hypothesis new: (GPDialog request: title initialAnswer: '').
  theAnswer = (Hypothesis new: String new) | (Hypothesis new: (self
    does: theAnswer value asUppercase
    match: 'UNKNOWN'
    outTo: (theAnswer value size min: 3)))
  ifIsTrue:[^Hypothesis unknown]
  ifIsFalse:[^theAnswer]!

multipleChoice: aCollection titled: title
  | theAnswer |
  "get the answer from among a set of choices"

  theAnswer := Hypothesis new: (GPDialog select: title fromList: aCollection lines: 20).
  theAnswer value == nil
```

```
        ifTrue: [^Hypothesis unknown]
        ifFalse: [^theAnswer]! !

!ExperimentalFrameExpertSystem methodsFor: 'efes private'!

createRequestFrom: aString in: aParameter
    "create a string title for a request by substituting appropriate entity
    names or parameter values into the title"

    | inStream outStream param questionString entity |
    entity := aParameter entity.
    inStream := ReadStream on: aString.
    outStream := WriteStream on: (String new: aString size).
    [inStream atEnd]
        whileFalse:
            [outStream nextPutAll: (inStream upTo: $&).
            inStream atEnd
                ifFalse:
                    [questionString := inStream upTo: Character space.
                    questionString = ''
                        ifTrue:
                            [outStream nextPutAll: entity name.
                            inStream peek = $? ifFalse: [outStream space]]
                        ifFalse:
                            [param := entity parameterAt: questionString asSymbol.
                            outStream nextPutAll: param value printString.
                            outStream space]]].
    ^outStream contents!

does: aString match: anotherString outTo: anInteger
    | max |
    max := (aString size min: anotherString size) min: anInteger.
    1 to: max do: [:each |
        (aString at: each) = (anotherString at: each) ifFalse:[^false]].
    ^true! !

!ExperimentalFrameExpertSystem methodsFor: 'efes parameter questioning'!

request: aParameter
    "If the parameter definition name is 'mgfSysComplexity' then get it from the interrogator"

    | definition reqString theAnswer tp |
    (tp := aParameter definition parameterName) = 'mgfSysComplexity'
        | (tp = 'whatIfQuestion')
        | (tp = 'selectedProblem')
            ifTrue: [^Hypothesis new: (self perform: aParameter definition parameterName asSymbol)].
    definition := aParameter definition.
    reqString := self createRequestFrom: definition prompt in: aParameter.
    definition type class == Array
        ifTrue:
            [theAnswer := self multipleChoice: definition type titled: reqString.
```

Related SmallTalk-80 Code For Class: ExperimentalFrameExpertSystem -continued

```
            aParameter valueIsKnown: true.
            ^theAnswer].
    definition type == #YN
        ifTrue:
            [theAnswer := self binaryChoice: reqString.
            aParameter valueIsKnown: true.
            ^theAnswer].
    theAnswer := self fillInTitled: reqString.
    aParameter valueIsKnown: true.
    definition type = Number ifTrue: [theAnswer value: theAnswer value asNumber].
    ^theAnswer! !

!ExperimentalFrameExpertSystem methodsFor: 'efes extention'!
```

displayResults

```
    | intRec sc anOrdColl in |
    intRec := intermediateRecommendations copy.
    sc := OrderedCollection new.
    sc := intRec values asSortedCollection reverse.
    anOrdColl := OrderedCollection new.
    sc do: [:i | intRec keysAndValuesDo: [:k :v | i = v
                ifTrue:
                    [anOrdColl add: k; add: v; add: v.
                    intRec removeKey: k]]].
    in := EFES_TableInterface new.
    in initializeWith: anOrdColl.
    in efes: self.
    self closeRequest.
    in open!
```

getUserPreferences

```
    | u |
    u := UserPreferencesInterface new.
    u efes: self.
    u open!
```

startAdvisory

```
    "This part is to ask a few questions to the user in order to identify the scenario details"

    | choiceOne choiceTwo |
    ScheduledControllers scheduledControllers do: [ :each | each collapse].
    GPDialog warn:
'You are starting a consultation session with EFES

        © Dursun Delen & Center For CIM
'.
    (GPDialog
        choose: 'Choose one of the following study types ...'
        labels: #('Problematic Scenario' 'What-if Analysis')
        values: #(1 2)
    default: #(false false)) = 2
    ifTrue:
      [choiceOne := GPDialog
            select: 'Choose one what-if question from the following list ... '
            fromList: whatIfAnalysisList
            lines: 15.
      choiceOne isNil
        ifFalse:
          [whatIfQuestion := choiceOne.
          self startWhatIfToTool]]
    ifFalse: [(GPDialog
        choose: 'Choose one of the following ...'
        labels: #('Problem is Known' 'Problem is Unknown')
        values: #(3 4)
        default: #(false false)) = 3
        ifTrue:
          [choiceTwo := GPDialog
                select: 'Choose one problem from the following list ...'
                fromList: structuredProblemsList
                lines: 20.
          choiceTwo isNil
            ifFalse:
              [selectedProblem := choiceTwo.
              self startProblemToTool]]
        ifFalse: [self startDerivationOfStrProbFromSymptoms]]!

startDerivationOfStrProbFromSymptoms
  "By utilizing problemDefinition knowledge base, this method provides user
  with a list of most possible problems in a prioritized order"

  | efes parameter hp newDict sc aColl choice |
  newDict := Dictionary new.
  aColl := OrderedCollection new.
  efes := KnowledgeBase allBases at: 'ProblemDefinitionKB'.
  efes initEntities.
  efes initEntityTypes.
  efes interrogator: self.
  parameter := efes findOut: #StructuredProblem.
  parameter hypotheses values do: [:conc |
    (hp := conc at: #combined) certainty > 0 ifTrue: [newDict at: hp value put: hp certainty]].
  sc := newDict values asSortedCollection reverse.
  sc do: [:i | newDict keysAndValuesDo: [:k :v | i = v
        ifTrue:
          [aColl add: k , ' -- with cwa of ' , (v * 100) truncated printString , '%'.
          newDict removeKey: k]]].

  choice := GPDialog
        select: 'Choose one problem from the following list and continue ...'
        fromList: aColl
        lines: 10.
    choice isNil ifFalse: [structuredProblemsList do: [:each | (each , '*' match: choice)
                ifTrue: [selectedProblem := each]].
                self startProblemToTool]!

startProblemToTool
    "By utilizing problemToTool knowledge base, this method provides user
    with a list of most appealing tools in a prioritized order"

    | efes parameter hp newDict |
    newDict := Dictionary new.
    efes := KnowledgeBase allBases at: 'ProblemToToolKB'.
    efes initEntities.
    efes initEntityTypes.
    efes interrogator: self.
    parameter := efes findOut: #Tool.
    parameter hypotheses values do: [:conc | (hp := conc at: #combined) certainty > 0
            ifTrue: [newDict at: hp value put: hp certainty]].
    intermediateRecommendations := newDict.
    intermediateRecommendations keys size > 1 ifTrue:[self getUserPreferences]
        ifFalse:[self displayResults]!

startWhatIfToTool
    "By utilizing whatIfToTool knowledge base, this method provides user
    with a list of most appealing tools in a prioritized order"

    | efes parameter hp newDict |
    newDict := Dictionary new.
    efes := KnowledgeBase allBases at: 'WhatIfToToolKB'.
    efes initEntities.
    efes initEntityTypes.
    efes interrogator: self.
    parameter := efes findOut: #Tool.
    parameter hypotheses values do: [:conc | (hp := conc at: #combined) certainty > 0
            ifTrue: [newDict at: hp value put: hp certainty]].
    intermediateRecommendations := newDict.
    intermediateRecommendations keys size > 1 ifTrue:[self getUserPreferences]
        ifFalse:[self displayResults]! !

!ExperimentalFrameExpertSystem methodsFor: 'efes entity questioning'!

moreOf: anEntityType
    "This is the message which HUMBLE uses to ask whether more of some sort
    of entity exist, after establishing that at least one exists using the oneOf:
    message. The programmer interested in creating his own interrogator
    class may want to intercept this message and check for any special cases.  It
    is likely that this message and oneOf: will need to check the same cases."

    ^GPDialog confirm: anEntityType addPrompt!
```

```
oneOf: anEntityType
    "This is the message which HUMBLE uses to ask whether one of some sort
    of entity exists. The programmer interested in creating his own interrogator
    class may want to intercept this message and check for any special cases."

    ^GPDialog confirm: anEntityType createPrompt! !

!ExperimentalFrameExpertSystem methodsFor: 'aspects'!

adjustingFactors: aDict
    adjustingFactors := aDict!

intermediateRecommendations: aDict
    intermediateRecommendations := aDict!

mgfSysComplexity: aString
    mgfSysComplexity := aString! !
```

Related SmallTalk-80 Code For Class: UserPreferencesInterface

```
ApplicationModel subclass: #UserPreferencesInterface
   instanceVariableNames: 'efes timeToSolution levelOfDetail mSC1 mSCW2 mSCW4 mSC3 mSC5
      mSCW5 mSC4 mSC2 mSCW1 mSCW3 '
   classVariableNames: "
   poolDictionaries: "
   category: 'Experimental Frame Expert System'!

!UserPreferencesInterface methodsFor: 'aspects'!

efes
   ^efes!

efes: anExpErame
   efes := anExpErame!

levelOfDetail
   "This method was generated by UIDefiner.  Any edits made here
   may be lost whenever methods are automatically defined.  The
   initialization provided below may have been preempted by an
   initialize method."

   ^levelOfDetail isNil
      ifTrue:
         [levelOfDetail := 0.00 asValue]
      ifFalse:
         [levelOfDetail]!

mSC1
   "This method was generated by UIDefiner.  Any edits made here
   may be lost whenever methods are automatically defined.  The
   initialization provided below may have been preempted by an
   initialize method."

   ^mSC1 isNil
      ifTrue:
         [mSC1 := 0 asValue]
      ifFalse:
         [mSC1]!

mSC2
   "This method was generated by UIDefiner.  Any edits made here
   may be lost whenever methods are automatically defined.  The
   initialization provided below may have been preempted by an
   initialize method."

   ^mSC2 isNil
      ifTrue:
         [mSC2 := 0 asValue]
      ifFalse:
         [mSC2]!
```

mSC3

```
    "This method was generated by UIDefiner.  Any edits made here
    may be lost whenever methods are automatically defined.  The
    initialization provided below may have been preempted by an
    initialize method."

    ^mSC3 isNil
        ifTrue:
            [mSC3 := 0 asValue]
        ifFalse:
            [mSC3]!
```

mSC4

```
    "This method was generated by UIDefiner.  Any edits made here
    may be lost whenever methods are automatically defined.  The
    initialization provided below may have been preempted by an
    initialize method."

    ^mSC4 isNil
        ifTrue:
            [mSC4 := 0 asValue]
        ifFalse:
            [mSC4]!
```

mSC5

```
    "This method was generated by UIDefiner.  Any edits made here
    may be lost whenever methods are automatically defined.  The
    initialization provided below may have been preempted by an
    initialize method."

    ^mSC5 isNil
        ifTrue:
            [mSC5 := 0 asValue]
        ifFalse:
            [mSC5]!
```

mSCW1

```
    "This method was generated by UIDefiner.  Any edits made here
    may be lost whenever methods are automatically defined.  The
    initialization provided below may have been preempted by an
    initialize method."

    ^mSCW1 isNil
        ifTrue:
            [mSCW1 := 0 asValue]
        ifFalse:
            [mSCW1]!
```

mSCW2

```
"This method was generated by UIDefiner.  Any edits made here
may be lost whenever methods are automatically defined.  The
initialization provided below may have been preempted by an
initialize method."

^mSCW2 isNil
    ifTrue:
        [mSCW2 := 0 asValue]
    ifFalse:
        [mSCW2]!
```

mSCW3

```
"This method was generated by UIDefiner.  Any edits made here
may be lost whenever methods are automatically defined.  The
initialization provided below may have been preempted by an
initialize method."

^mSCW3 isNil
    ifTrue:
        [mSCW3 := 0 asValue]
    ifFalse:
        [mSCW3]!
```

mSCW4

```
"This method was generated by UIDefiner.  Any edits made here
may be lost whenever methods are automatically defined.  The
initialization provided below may have been preempted by an
initialize method."

^mSCW4 isNil
    ifTrue:
        [mSCW4 := 0 asValue]
    ifFalse:
        [mSCW4]!
```

mSCW5

```
"This method was generated by UIDefiner.  Any edits made here
may be lost whenever methods are automatically defined.  The
initialization provided below may have been preempted by an
initialize method."

^mSCW5 isNil
    ifTrue:
        [mSCW5 := 0 asValue]
    ifFalse:
        [mSCW5]!
```

```
timeToSolution
    "This method was generated by UIDefiner.  Any edits made here
    may be lost whenever methods are automatically defined.  The
    initialization provided below may have been preempted by an
    initialize method."

    ^timeToSolution isNil
        ifTrue:
            [timeToSolution := 0.00 asValue]
        ifFalse:
            [timeToSolution]! !

!UserPreferencesInterface methodsFor: 'actions'!

cancel
    "This stub method was generated by UIDefiner"

    self closeRequest!

continue
    "This stub method was generated by UIDefiner"

    | upi |
    upi := UserPreferencesInterface2 new.
    upi timeToSolution: self timeToSolution.
    upi efes: self efes.
    upi levelOfDetail: self levelOfDetail.
    upi mSC: (mSC1 value * mSCW1 value + (mSC2 value * mSCW2 value) + (mSC3 value * mSCW3
value) + (mSC4 value * mSCW4 value) + (mSC5 value * mSCW5 value) / (mSCW1 value + mSCW2
value + mSCW3 value + mSCW4 value + mSCW5 value) / 10) asValue.
    self closeRequest.
    upi open!

explainLOD

    GPDialog warn:
'You can change the preference level of output by adjusting the slider.
User should know that requiring more datailed output would result in a longer execution time.
In some cases, requesting detailed output will work in favor of simulation-related analysis tools ...

Pressing :
        "Continue" will resume the process
        "Cancel" will terminate the process and close the interface.'!

explainMSC

    GPDialog warn:
'This box displays all 5 sub-factors that are calculated based on the current Base Model
and used to calculate the complexity of the manufacturing system under consideration.
Even though user cannot change these sub-factor values,  the weights for these factors
are adjustable.
```

```
Pressing :
        "Continue" will resume the process
        "Cancel" will terminate the process and close the interface.'!

explainTTS

    GPDialog warn:
'You can change the time-to-solution level by adjusting the slider.
User should know that requiring quick answer will work in favor of analitical tools.
This factor should be left 0 if a strong preference is not exist ...

Pressing :
        "Continue" will resume the process
        "Cancel" will terminate the process and close the interface.'! !

!UserPreferencesInterface methodsFor: 'initialize'!

initialize
    "These values are to be determined from the BaseModel"

    | aBM aPBD1 firstValue aPBD2 secondValue aPBD3 thirdValue |

    aBM := BaseModel currentBaseModel.

    mSC1 := aBM plant numberOfStations asValue.
    mSC2 := aBM plant numberOfMHs asValue.
    mSC3 := aBM numberOfProducts asValue.

    aPBD1 := PointBasedData newWith: aBM productDepths.
    firstValue := ((aPBD1 maximum) + (aPBD1 minimum) + (4 * (aPBD1 meanEstimate))) / 6.
    firstValue > 10 ifTrue: [firstValue := 10].

    aPBD2 := PointBasedData newWith: aBM productBredths.
    secondValue := ((aPBD2 maximum) + (aPBD2 minimum) + (4 * (aPBD2 meanEstimate))) / 6.
    secondValue > 10 ifTrue: [secondValue := 10].

    mSC4 := ((firstValue + secondValue) / 2) asValue.

    aPBD3 := PointBasedData newWith: aBM productRoutings.
    thirdValue := ((aPBD3 maximum) + (aPBD3 minimum) + (4 * (aPBD3 meanEstimate))) / 6.
    thirdValue > 10 ifTrue: [thirdValue := 10].
    mSC5 := thirdValue asValue.

    "These are the default weights (10)"
    mSCW1 := 10 asValue.
    mSCW2 := 10 asValue.
    mSCW3 := 10 asValue.
    mSCW4 := 10 asValue.
    mSCW5 := 10 asValue! !
"-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- "!
```

Related SmallTalk-80 Code For Class: UserPreferencesInterface - continued

```
UserPreferencesInterface class
    instanceVariableNames: ''!

!UserPreferencesInterface class methodsFor: 'interface specs'!

windowSpec
    "UIPainter new openOnClass: self andSelector: #windowSpec"

    <resource: #canvas>
    ^#(#FullSpec ...)

windowSpecOld
    "UIPainter new openOnClass: self andSelector: #windowSpecOld"

    <resource: #canvas>
    ^#(#FullSpec ...)

!UserPreferencesInterface class methodsFor: 'resources'!

helpViewerImage
" Created by GPImageEditor v1.2"

    <resource: #image>
    ^OpaqueImage
        figure:
        (Image extent: 32@32 depth: 4 bitsPerPixel: 4 palette: ...)
```

Related SmallTalk-80 Code For Class: UserPreferencesInterface2

```
UserPreferencesInterface subclass: #UserPreferencesInterface2
    instanceVariableNames: 'mSCW mSC lODW tTSW '
    classVariableNames: "
    poolDictionaries: "
    category: 'Experimental Frame Expert System'!

!UserPreferencesInterface2 methodsFor: 'aspects'!

lODW
    "This method was generated by UIDefiner.  Any edits made here
    may be lost whenever methods are automatically defined.  The
    initialization provided below may have been preempted by an
    initialize method."

    ^lODW isNil
        ifTrue:
            [lODW := 0 asValue]
        ifFalse:
            [lODW]!

mSC
    "This method was generated by UIDefiner.  Any edits made here
    may be lost whenever methods are automatically defined.  The
    initialization provided below may have been preempted by an
    initialize method."

    ^mSC isNil
        ifTrue:
            [mSC := 0 asValue]
        ifFalse:
            [mSC]!

mSCW
    "This method was generated by UIDefiner.  Any edits made here
    may be lost whenever methods are automatically defined.  The
    initialization provided below may have been preempted by an
    initialize method."

    ^mSCW isNil
        ifTrue:
            [mSCW := 0 asValue]
        ifFalse:
            [mSCW]!

tTSW
    "This method was generated by UIDefiner.  Any edits made here
    may be lost whenever methods are automatically defined.  The
    initialization provided below may have been preempted by an
    initialize method."
    ^tTSW isNil
        ifTrue:
            [tTSW := 0 asValue]
        ifFalse:
            [tTSW]! !

!UserPreferencesInterface2 methodsFor: 'initialize'!

initialize
    "These are also the default weights (10)"

    mSCW := 10 asValue.
    tTSW := 10 asValue.
    lODW := 10 asValue! !

!UserPreferencesInterface2 methodsFor: 'actions'!

continue
    | m1 m2 aColl totW m3 aDict intRec sc anOrdColl m in |
    m1 := (Matrix new: 3 by: 5)
                collection: #(0.5 -0.25 0.17 -0.5 0.5 -0.5 -0.25 0.25 -0.25 -0.17 0.17 0.25 0.25 -0.5 0.5).
    aColl := OrderedCollection new.
    totW := lODW value + tTSW value + mSCW value.
    aColl add: mSC value * (mSCW value / totW); add: timeToSolution value * (tTSW value / totW); add:
levelOfDetail value * (lODW value / totW).
    m2 := (Matrix new: 1 by: 3)
                collection: aColl.
    m3 := m2 * m1.
    aDict := Dictionary new.
    aDict at: 'Simulation' put: (m3 atPoint: 1 @ 1); at: 'Queueing' put: (m3 atPoint: 2 @ 1); at: 'Petri nets'
put: (m3 atPoint: 3 @ 1); at: 'Queueing + Simulation' put: (m3 atPoint: 4 @ 1); at: 'Search-based
optimization with simulation' put: (m3 atPoint: 5 @ 1).
    efes adjustingFactors: aDict.
    intRec := efes intermediateRecommendations copy.
    sc := OrderedCollection new.
    sc := intRec values asSortedCollection reverse.
    anOrdColl := OrderedCollection new.
    sc do: [:i | intRec keysAndValuesDo: [:k :v | i = v
            ifTrue:
                [anOrdColl add: k; add: (v asFixedPoint: 3).
                (efes adjustingFactors includesKey:k) ifTrue:[m := (efes adjustingFactors at: k) * v + v]
                    ifFalse: [m := v].
                m > 1
                    ifTrue: [anOrdColl add: 1.000]
                    ifFalse: [anOrdColl add: (m asFixedPoint: 3)].
                intRec removeKey: k]]].
    in := EFES_TableInterface new.
    in initializeWith: anOrdColl.
    in efes: self efes.
    self closeRequest.
    in open
```

Related SmallTalk-80 Code For Class: UserPreferencesInterface2

```
efes
    ^efes!

efes: anExpErame
    efes := anExpErame!

explainAll

    GPDialog warn:
'This box displays all 3 factor (Manufacturing System Complexity, Time-To-Solution, Level-Of-Detail)
values.
Even though these factor values are only display purposes,  the weights for the factors are adjustable.

Pressing :
        "Continue" will resume the process
        "Redo" will take the process back to user preferences interface
        "Cancel" will terminate the process and close the interface.'!

levelOfDetail: aNumber
    levelOfDetail := aNumber!

mSC: aNumber
    mSC := aNumber!

redo
    self closeRequest.
    self efes getUserPreferences!

timeToSolution: aNumber
    timeToSolution := aNumber! !
"-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- "!

UserPreferencesInterface2 class
    instanceVariableNames: ''!

!UserPreferencesInterface2 class methodsFor: 'interface specs'!

windowSpec
    "UIPainter new openOnClass: self andSelector: #windowSpec"

    <resource: #canvas>
    ^#(#FullSpec ...)
```

Related SmallTalk-80 Code For Class: EFES_TableInterface

```
ApplicationModel subclass: #EFES_TableInterface
    instanceVariableNames: 'cellContents sightingsTable tableInterface efes firstOrderChanged '
    classVariableNames: "
    poolDictionaries: "
    category: 'Experimental Frame Expert System'!


!EFES_TableInterface methodsFor: 'initialize-release'!

initialize
    | list |
    super initialize.

    "Create a collection of sightings data."
    list := TwoDList
                on: #('Simulation' 0.95 0.80 'Queuing' 0.56 0.75 'Petri Nets' 0.22 0.15 'Simulation +
                        Queuing' 0.46 0.75 'Search Based Optimization with Simulation' 0.15 0.45) copy
                columns: 3
                rows: 5.
    sightingsTable := SelectionInTable with: list.

    "Create a table interface and load it with the sightings."
    tableInterface := TableInterface new selectionInTable: sightingsTable.
    tableInterface columnWidths: #(250 80 120);
                columnLabelsArray: #(" " ");
                columnFormats: #(#centered #centered #centered);
                columnLabelsFormats: #(#centered #centered #centered).

    firstOrderChanged := false.!

initializeWith: aColl
    | list |
    super initialize.

    "Create a collection of sightings data."
    list := TwoDList
                on: (aColl value) copy
                columns: 3
                rows: ((aColl size) / 3).
    sightingsTable := SelectionInTable with: list.

    "Create a table interface and load it with the sightings."
    tableInterface := TableInterface new selectionInTable: sightingsTable.
    tableInterface columnWidths: #(250 80 121);
                columnLabelsArray: #(" " ");
                  columnFormats: #(#left #centered #centered);
                  columnLabelsFormats: #(#centered #centered #centered).

    (aColl size > 5) ifTrue:[(aColl at:3) < (aColl at:6) ifTrue:[firstOrderChanged := true]
            ifFalse:[firstOrderChanged := false]]! !
```

Related SmallTalk-80 Code For Class: EFES_TableInterface - continued

```
!EFES_TableInterface methodsFor: 'accessing'!

efes
    ^efes!

efes: anEFES
    efes := anEFES!

firstOrderChanged
    ^firstOrderChanged!

sightingsTable
    ^sightingsTable!

tableInterface
    ^tableInterface! !

!EFES_TableInterface methodsFor: 'actions'!

changeView

    "This stub method was generated by UIDefiner"

    ^self!

close
    self closeRequest.
    ScheduledControllers scheduledControllers do: [:each |
            each model class = EnvironmentInterface ifTrue: [each model builder window expand]]!

continue
    | choice solver |
    tableInterface selectionInTable selectionIndexHolder value x: 1.
    tableInterface selectionInTable selectionIndexHolder value y = 0
        ifTrue:
            [GPDialog warn: 'Sorry ... You need to select a tool to continue ...'.
            ^self]
        ifFalse: [choice := tableInterface selectionInTable selection].

    choice isNil ifFalse: [('Simulation*' match: choice)
            ifTrue: [solver := 'Simulation']
            ifFalse: [('Queueing +*' match: choice)
                    ifTrue: [solver := nil]
                    ifFalse: [('Queueing*' match: choice)
                            ifTrue: [solver := 'QNASolver']
                            ifFalse: [('Petri nets*' match: choice)
                                    ifTrue: [solver := 'Petri nets']]]]].

    solver isNil
        ifTrue: [GPDialog warn: 'Sorry ...
    Selected tool is not supported by the current envoronmment.
    Please select another solver and continue ...']
        ifFalse:
            [efes parentInterface solverName value: solver.
            self closeRequest.
            ScheduledControllers scheduledControllers do: [:each | each model class =
EnvironmentInterface ifTrue: [each model builder window expand]]]!

explainResults

    GPDialog warn:
'User can choose any solver from the given list.
List is in a prioritized order.  Weights on the second column are based on the structured problem.
The third column displays the weights which are adjusted based on the user preferences.

Pressing :
        "Continue" will resume the process
        "Redo" will take the process back to user preferences interface
        "Cancel" will terminate the process and close the interface.'!

redo
    self closeRequest.
    self efes getUserPreferences! !

!EFES_TableInterface methodsFor: 'interface opening'!

postBuildWith: aBuilder
    firstOrderChanged ifTrue: [(aBuilder componentAt: #changeView) beVisible]!

postOpenWith: aBuilder
"   firstOrderChanged ifTrue: [GPDialog warn: 'Because of the user prefrences, the FIRST PRIORITIZED
solver changed ..!!'].
    self halt."! !
"-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- "!

EFES_TableInterface class
    instanceVariableNames: ''!


!EFES_TableInterface class methodsFor: 'interface specs'!

windowSpec
    "UIPainter new openOnClass: self andSelector: #windowSpec"

    <resource: #canvas>
    ^#(#FullSpec ...

!EFES_TableInterface class methodsFor: 'initialize-release'!
```

Related SmallTalk-80 Code For Class: EFES_TableInterface - continued

```
newWith: aColl
    | newList |
    newList := self new initializeWith: aColl.
    newList openInterface: #windowSpec!

newWith: aColl labeled: aString
    | newList |
    newList := self new initializeWith: aColl.
    newList label: aString.
    newList openInterface: #windowSpec! !

!EFES_TableInterface class methodsFor: 'resources'!

changeView
    "UIMaskEditor new openOnClass: self andSelector: #changeView"

    <resource: #image>
    ^CachedImage on: (Image extent: 112@34 depth: 2 bitsPerPixel: 2 palette: ...)

helpViewerImage
" Created by GPImageEditor v1.2"

    <resource: #image>
    ^OpaqueImage
        figure:
        (Image extent: 32@32 depth: 4 bitsPerPixel: 4 palette: ...)

resultsImage
" Created by GPImageEditor v1.2"

    <resource: #image>
    ^  ( CachedImage on:
            (Image extent: 127@38 depth: 8 bitsPerPixel: 8 palette: ...)
```

APPENDIX G

INSTITUTIONAL REVIEW BOARD FORM

OKLAHOMA STATE UNIVERSITY
INSTITUTIONAL REVIEW BOARD
HUMAN SUBJECTS REVIEW

Date: 12-12-95 **IRB#:** EG-96-004

Proposal Title: ROLE OF EXPERT SYSTEMS IN AN ADVANCED MODELING ENVIRONMENT FOR MANUFACTURING SYSTEMS

Principal Investigator(s): David B. Pratt, Dursun Delen

Reviewed and Processed as: Exempt

Approval Status Recommended by Reviewer(s): Approved

ALL APPROVALS MAY BE SUBJECT TO REVIEW BY FULL INSTITUTIONAL REVIEW BOARD AT NEXT MEETING.
APPROVAL STATUS PERIOD VALID FOR ONE CALENDAR YEAR AFTER WHICH A CONTINUATION OR RENEWAL REQUEST IS REQUIRED TO BE SUBMITTED FOR BOARD APPROVAL.
ANY MODIFICATIONS TO APPROVED PROJECT MUST ALSO BE SUBMITTED FOR APPROVAL.

Comments, Modifications/Conditions for Approval or Reasons for Deferral or Disapproval are as follows:

Provisions received and approved.

Signature: John H. Wyckoff Date: December 20, 1995

Chair of Institutional Review Board

2

VITA

Dursun Delen

Candidate for the Degree of

Doctor of Philosophy

Thesis: DEVELOPMENT OF AN EXPERT SYSTEM BASED EXPERIMENTAL FRAME FOR MODELING OF MANUFACTURING SYSTEMS

Major Field: Industrial Engineering and Management

Biographical:

Personal Date: Born in Of, Trabzon, Turkey, February 5, 1964, the son of Süleyman and Ayse Delen. Married Handan Elibol on May 27, 1996.

Education: Graduated from Fatih Vatan Lisesi (high school), Istanbul, Turkey in May, 1982; received Bachelor of Science Degree in Industrial Engineering from Istanbul Technical University, Istanbul, Turkey, May, 1986; received Master of Engineering Degree in Industrial Engineering from Yildiz University, Istanbul, Turkey, May 1988; completed the requirements for the Doctor of Philosophy degree at Oklahoma State University in May, 1997.

Professional Experience: Computer System Controller, Aksay Worldwide Shipping Company, Istanbul, Turkey, July, 1987 to August 1988; System Analyst, Turkish Navy, Taskizak Dockyard Computer Information Center, Istanbul, Turkey, December, 1988 to November 1989; System Analyst and Computer Programmer, Ars-Com Computer Consulting, Inc., Istanbul, Turkey, December, 1989 to May, 1990; Research Associate, School of Industrial Engineering and Management at Oklahoma State University, January, 1994 to present; Lecturer, School of Industrial Engineering and Management at Oklahoma State University, August, 1996 to present.