

PARAMSOTHY JEYAKUMAR

A DESIGN APPROACH
TO INTEGRATED INTELLIGENT
SYSTEMS IN A MANUFACTURING
ENVIRONMENT



NTH
UNIVERSITETET I TRONDHEIM
NORGES TEKNISKE HØGSKOLE

DOKTOR INGENIØRAVHANDLING 1995:112
INSTITUTT FOR PRODUKSJONS- OG
KVALITETSTEKNIKK
TRONDHEIM

**A DESIGN APPROACH TO INTEGRATED
INTELLIGENT SYSTEMS IN A
MANUFACTURING ENVIRONMENT**

Paramsothy Jeyakumar

DR. ING. THESIS / Ph.D. DISSERTATION

**University of Trondheim
The Norwegian Institute of Technology - NTH
Department of Production and Quality Engineering**

Trondheim, Norway

PREFACE

This work has been carried out in the period from January 1992 to November 1995 in the Department of Production and Quality Engineering at the University of Trondheim, the Norwegian Institute of Technology (NTH).

First of all I would like to thank Professor Øyvind Bjørke for giving me the opportunity to conduct my Ph.D. studies.

I would like to take this opportunity to thank my supervisor professor Kesheng Wang for his help and guidance through the three years of hard work and his criticism and patience in reading this dissertation.

I also want to thank Associate Professor Knut Egelie; Nina Moxnes and Stewart Clark for giving me the opportunity to finish my Ph.D. research at the University of Manchester Institute for Science and Technology (UMIST) for a period of six months.

I am grateful to Dr. Roger G. Hannam at the Department of Mechanical Engineering, UMIST, for his inspiring discussions and immense assistance during my stay in Manchester.

Special thanks are also due to my wife Sivanutha for her patience and encouragement throughout my study of the Ph.D. course in spite of the difficult time she had to providing me with necessary financial assistance.

Also, I wish, sincerely to, thank my friend Bjørnar Iversen with whom I shared very infighting ideas before writing this thesis.

Finally, I want to thank the staff at the Department of Production and Quality Engineering at the Norwegian Institute of Technology and SINTEF (The Foundation for Science and Industrial Research at the Norwegian Institute of Technology) for their assistance and collaboration.

IV

TO MY FATHER

SUMMARY

The purpose of this work is to provide a framework and principles for designing Integrated Intelligent Manufacturing Systems (IIMS) in a manufacturing environment. Integrated Intelligent Manufacturing systems are based on the integration of expert systems, multimedia, graphic systems, database systems and multimedia techniques. They offer a rich environment for creating computer applications that can increase productivity and flexibility enormously, and act as intelligent assistants and decision makings. Emphasis of this thesis is placed on implementation aspects, particular interest being centred on manufacturing environments. Practical applications of Artificial Intelligent techniques are highlighted and a guide to available advanced tools is given. Case studies are incorporated both in order to introduce basic concepts and to explain more detailed applications.

The kernel of the modern manufacturing technology is to implement the integration of domains of specific knowledge, the integration of industrial manufacturing, decision-making automation, and management and marketing activities. The development and application of Integrated Intelligent System techniques will quicken this process and improve companies' production efficiency and product quality.

This thesis provide the concept, methodology, and implementation techniques of an Integrated Intelligent System (IIS), as well as IIS applications in real-world manufacturing industries. IIS is a knowledge integration environment, which consists of six modules: a meta-system, a graphic package, a data base system, a spreadsheet system, numerical computations and an expert system. The integrated software environments allows the running of programs written in different languages, and communicate among the programs as well as exchange of data between programs and database. These isolated intelligent systems, numerical packages and other modules are under the control of a supervising intelligent system, which is called a meta-system. The meta-system manages the selection, coordination, operation and communication of these programs.

This thesis consists of seven chapters. The first three chapters introduce fundamental framework concepts, methodology and implementation of IIS. IIS applications are described in the following four chapters which discuss, respectively, Integrated Intelligent Production Management system (IIPMS), an Integrated Intelligent Process Planning Systems (IIPPS) for machining processes, an Integrated Intelligent Process Support System (IIPSS) and an expert system for diagnosing faults in a chocolate factory. The tutorial nature of some of the Chapters serves as an introduction to the application techniques which may be employed, particularly for industrial problem solving.

This thesis also has identified some of the more important guidelines and has outlined how a low-cost knowledge-processing system (called KnowledgePro) can be used to facilitate their realisation. The KnowledgePro system offers a 'rich' and 'intelligent' environment to build an IIS.

The future of IIS technology looks exciting. The success already experienced with the technology have generated great interest. We can expect to see a number of IIS developed and the intelligence and capability of these systems increase dramatically. As the knowledge engineering discipline moves on and improves, IIS will be developed with the ability to deal with more and more complex problems. IIS will move from the role of assistant to the role of adviser to the role of monitor to the role of controller. The IIS technology is integrated with other advanced technologies and like the vision and speech technologies it will generate more interest in the future.

SAMMENDRAG

Hensikten med dette doktorarbeidet er å skaffe til veie et rammeverk og prinsipper for design av Integreerte Intelligente Produksjonstekniske Systemer (IIMS) innenfor et produksjonsmiljø. IIMS er basert på en integrasjon av ekspertsystemer, grafiske systemer, databasesystemer og multimediateknikker. Systemene gir gode muligheter for utvikling av datamaskinanvendelser, og for økt produktivitet og fleksibilitet, og virker som intelligente assistenter og beslutningstakere. I doktorarbeidets implementeringsaspekter er det lagt vekt på dette, spesielt med tanke på produksjonsomgivelser. Det legges vekt på produktanvendelser av kunstig intelligens-teknikker og gi en oversikt over tilgjengelige avanserte verktøy. Casestudier integreres både for å introdusere grunnleggende konsepter og for å forklare mer detaljerte anvendelser.

Kjernen i moderne produksjonsteknologi er å implementere integrasjon av spesifikke kunnskapsområder, integrasjon av industriell produksjon, automasjon av beslutningstaking, ledelse og markedsføringsaktiviteter. Utvikling og implementering av Integreerte Intelligente System (IIS) teknikker vil akselerere denne prosessen og forbedre bedrifters produksjonseffektivitet og produktkvalitet.

Dette doktorarbeidet skaffer til veie konseptet, metoder og implementeringsteknikker for et IIS, og viser noen IIS anvendelser i produksjonsindustrier. IIS er en kunnskapsintegreert omgivelse som består av seks moduler: et meta-system, en grafisk modul, et database system, et regneark system, numeriske beregninger og et ekspert system. Den integrerte softwareomgivelsen tillater kjøring av programmer skrevet i ulike programmeringsspråk og utveksling av data mellom programmer og databaser. Disse isolerte intelligente systemer, numeriske moduler og andre moduler kontrolleres av et styrende intelligent system, som kalles et meta-system. Meta-systemet håndterer utvelgelse, koordinasjon, operasjon og kommunikasjon mellom de ulike programmene.

VIII

Dette arbeidet består av syv kapitler. De første tre kapitlene introduserer fundamentale konsepter for rammeverk, metoder og implementering av IIS. IIS-applikasjonene beskrevet i de følgende fire kapitler, diskuterer henholdsvis integrerte intelligente produksjonshåndteringssystemer, integrerte intelligente prosessplanleggingssystemer for maskineringsprosesser, et integrert intelligent prosesstøttesystem og et ekspertsystem for diagnostisering av en sjokoladefabrikk. Sammensetninger av kapitler kan ses på som en introduksjon til de applikasjonsteknikker som kan innføres, spesielt for industriell problemløsning.

Dette doktorarbeidet har også identifisert noen av de viktigere retningslinjer og har skissert hvordan et lavkostnads kunnskapsystem (systemet kalt KnowledgePro) kan brukes for lettere virkeliggjøring. KnowledgePro systemet gir en god og intelligent omgivelse for å bygge et IIS.

Fremtiden ser lovende ut for IIS - teknologien. Det er allerede oppnådd suksess med teknologien, og den har vekket stor interesse. Det kan ventes at antallet utviklede IIS og disse systemenes intelligens og evner vil øke dramatisk. Etter hvert som den kunnskapsbaserte ingeniørdisiplinen går videre og forbedres, vil IIS bli utviklet med evne til å takle mer og mer komplekse problemer. IIS vil endres fra rollen som assistent, til rådgiver og videre til overvåker, og tilslutt til rollen som styrer. Når IIS teknologien kan integreres med andre avanserte teknologier, som for eksempel visjons-og taleteknologier, vil dette generere mer interesse i fremtiden.

TABLE OF CONTENTS

PREFACE	III
SUMMARY	V
SUMMENDRAG	VII
TABLE OF CONTENTS	IX
LIST OF FIGURES	XIII
LIST OF TABLES	XV

1. INTRODUCTION	1
1.1 Artificial Intelligence in Manufacturing Engineering	1
1.1.1 Symbolic reasoning systems in manufacturing	4
1.1.2 Coupling intelligent systems in manufacturing	5
1.2 Integrated Intelligent Systems Approach	6
1.3 General view of this thesis	8
1.4 Reference	10
2. A TOOL FOR BUILDING INTELLIGENT SYSTEMS	11
2.1 Introduction	11
2.1 Why KnowledgePro	12
2.3 Application environment	13
2.3.1 Procedural knowledge	13
2.3.2 Relational knowledge	14
2.4 The development environment of intelligent systems	14
2.4.1 Intelligent system building	14
2.4.2 Hypertext implementation	17
2.4.3 Knowledge visualisation and tracing	18
2.4.4 Integration with other program tools	19
2.5 The user environment	19
2.5.1 Screen design	19
2.5.2 Custom report generation	20
2.6 Conclusions	20
2.7 Reference	20

3. META-SYSTEM STRUCTURE	21
3.1 Introduction	21
3.2 The concept of meta-system	21
3.3 Why we need a meta-system in manufacturing environment?	23
3.3.1 Fundamental concept of intelligent manufacturing.....	25
3.4 Function of the meta-system in integrated intelligent systems.....	26
3.5 Configuration of meta-system	27
3.5.1 Interface to external environment.....	28
3.5.2 Interface to internal subsystem.....	28
3.5.3 Meta-knowledge base.....	28
3.5.4 Inference mechanism.....	29
3.6 Implementation of meta-system in KnowledgePro Environment	20
3.7 Conclusions	31
3.8 Reference	32
4. INTEGRATED INTELLIGENT PRODUCTION MANAGEMENT SYSTEM (IIPMS)	33
4.1 Introduction	33
4.2 Problem definition	34
4.2.1 Procedure for break-even analysis.....	35
4.3 Solution by integrated intelligent systems.....	38
4.3.1 Integration environment	40
4.3.2 Problem solving method.....	43
4.4 Conclusions	49
4.5 References	50
5. INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEMS (IIPPS) FOR MACHINING PROCESSES	51
5.1 Introduction	51
5.2 Integrated intelligent system approach to process planning	54
5.2.1 The process planning function	54
5.2.2 The integrated intelligent system approach	54
5.3 Structure of IIPPS	55
5.4 Geometric feature extraction.....	57
5.5 Profile description	61
5.6 The modelling of process	63
5.7 Decision trees	64
5.8 Problem solving strategy	66

5.9	An expert system module for manufacturing logic and operation	69
5.9.1	Knowledge acquisition	69
5.9.2	Knowledge representation	70
5.9.3	Inference strategies	71
5.10	The function of meta-system	72
5.11	Verification and validation	72
5.12	Conclusions	79
5.13	References	79
6.	COUPLING QUANTITATIVE AND COMPUTING QUALITATIVE REASONING IN THE OPERATION OF PROCESS INDUSTRIES	81
6.1	Introduction	81
6.1.1	Quantitative and qualitative model in Integrated Intelligent system	83
6.2	Sulfite pulping process	84
6.3	Problems in the pulping process	86
6.4	Problem solving strategies	88
6.5	The quantitative model	89
6.5.1	The calculation process	92
6.6	Qualitative knowledge acquisition and representation	93
6.6.1	Coupling system	95
6.7	System Construction	98
6.7.1	Operating procedure of IIPSS	99
6.8	Implementation and demonstration	101
6.9	Conclusions	105
6.10	References	106
7.	AN EXPERT SYSTEM FOR DIAGNOSING FAULTS IN A CHOCOLATE FACTORY	109
7.1	Introduction	109
7.2	An overview of expert system diagnosis	110
7.3	Expert system model	111
7.4	Knowledge representation	112
7.4.1	Production rule representation	112
7.4.2	Knowledge elicitation	114
7.4.3	Decision trees	119
7.4.4	Structuring the knowledge base	120
7.5	Explanation facility	122
7.6	Knowledge acquisition	123
7.7	Backward chaining inference strategy	124

XII

7.8 Implementation.....	125
7.9 Conclusions	127
7.10 References	128

APPENDIX

APPENDIX 1 Listing of Integrated Intelligent Process Planning system [IIPPS].	129
APPENDIX 2 Program of Integrated Intelligent Process support system [IIPSS]	159
APPENDIX 3 Listing of case study program for chocolate factory	185
APPENDIX 4 Listing of IIPMS Program	191
GLOSSARY	199

LIST OF FIGURES

Figure 1.1 Intelligent manufacturing system	2
Figure 1.2 The core of symbolic reasoning systems	4
Figure 1.3 Structure of qualitative and quantitative analyses	6
Figure 1.4 Integration of intelligent systems.....	8
Figure 2.1 Edit window for using Rules	15
Figure 2.2 Edit window for using Topics	16
Figure 2.3 Layout window for using Rules or Topics	17
Figure 2.4 Hierarchical structure of a knowledge base.....	18
Figure 3.1 Integration of manufacturing software systems.....	24
Figure 3.2 Requirements for manufacturing systems and the effects of intelligent manufacturing.....	26
Figure 3.3 The core of the meta-system.....	27
Figure 3.4 Integrated Intelligent System.....	30
Figure 3.5 PC-based Intelligent System in a software environment.....	31
Figure 4.1a Example of liner Break-even chart	36
Figure 4.1b Example of non-liner Break-even chart	36
Figure 4.2 Break-even points for multiple alternatives.....	37
Figure 4.3a Make-or-buy comparison.....	39
Figure 4.3b Plot of profit functions for competing alternatives	39
Figure 4.4 The integrating intelligent system structure	40
Figure 4.5a Decision tree for break-even analysis.....	41
Figure 4.5b Layout of Alternatives I and II	42
Figure 4.6 Consultation introduction screen	44
Figure 4.7 Consultation introduction screen.....	45
Figure 4.8 Excel spreadsheet screen for cost functions	45
Figure 4.9 Input screen for IIPMS	45
Figure 4.10 Result of break-even analysis with 150 units	46
Figure 4.11 Result of break-even analysis with 250 units	46
Figure 4.12 Excel spreadsheet screen for Profit functions	47
Figure 4.13 Input screen for IIPMS	47
Figure 4.14 Result of break-even analysis with 150 units	48
Figure 4.15 Result of break-even analysis with 250 units	48
Figure 4.15 Result of break-even analysis with 250 units	48

Figure 5.1	The Integration of CAD, CAM and CAPP	53
Figure 5.1	The Integration of CAD, CAM and CAPP	53
Figure 5.2	IIPPS structure	56
Figure 5.3	IIPPS screen menu	58
Figure 5.4	Profile description scheme for rotational parts	62
Figure 5.5	Real world operator	63
Figure 5.6	The decision tree for external profile	65
Figure 5.7	Path logic in IIPPS	66
Figure 5.8a	Symbolic representation of a rotational part	73
Figure 5.8b	Example of input part drawing	74
Figure 5.9	IIPPS consultation screen	77
Figure 5.10	IIPPS consultation screen	78
Figure 5.11	IIPPS consultation screen	78
Figure 6.1	Material flow diagram of sulfite pulping and paper process operation..	84
Figure 6.2	Temperature and Pressure curves of sulfite cooking	86
Figure 6.3	Calculation of the quantitative system	92
Figure 6.4	Coupling input variables	95
Figure 6.5	Integrated architecture in IIPSS	98
Figure 6.6	The main window of Integrated Intelligent Process Support System	99
Figure 6.7	Input data	101
Figure 6.8	Qualitative reasoning results for operation chip load	102
Figure 6.9	Estimated cooking time for individual input variables [Level I]	102
Figure 6.10	Estimated cooking time based on selected input variables [Level II] ..	103
Figure 6.11	Estimated cooking time based on all input variables [Level III]	103
Figure 6.12	Input data for quantitative computation	104
Figure 6.13	Quantitative output values	105
Figure 7.1	Expert system architecture	111
Figure 7.1	Expert system architecture	111
Figure 7.2	Rule-based knowledge representation	113
Figure 7.3	Decision tree	119
Figure 7.4	Knowledge base search procedure	120
Figure 7.5	Knowledge base search procedure A	120
Figure 7.6	Knowledge base search procedure B	121
Figure 7.7	Knowledge base search procedure C	121
Figure 7.8	Knowledge base search procedure D	121
Figure 7.10	Knowledge base search procedure F	122
Figure 7.11	Example with backward chaining	124
Figure 7.12 - 15	Consultation window layout screen in KnowledgePro.	126 - 127

LIST OF TABLES

Table 6.1	Desired conditions for pulping quality variables	93
Table 6.2	Description of each situation	94
Table 6.3	Description of each situation at level I	96
Table 6.4	Cooking time ranges for coupled weight values.....	97
Table 7.1	The trigger-hypothesis rules, [ITS].....	115
Table 7.2	The hypothesis-fault rules, [CAU].....	116
Table 7.3	The fault-test rules, [CON].....	117
Table 7.4	The fault-action rules, [ITD]	118

1. INTRODUCTION

New information techniques will provide the foundation for manufacturing systems of the 1990's. As these systems evolve, both the basic control systems and company operations will change. In response to competitive thrusts increasingly based on flexibility, the integration of manufacturing islands will evolve and necessitate successful applications of knowledge system technology. The effective integration of the Artificial Intelligence technology will face challenges that are organisational logistical and technical in nature.

J. Davis and M. D. Oliff (1988), Requirements for manufacturing planning island using knowledge based technology

1.1 ARTIFICIAL INTELLIGENCE IN MANUFACTURING ENGINEERING

For a long time, computers have been used for design, manufacturing, control, diagnosing and monitoring in manufacturing industries. With the appearance of computer technology, manufacturing methodologies have developed rapidly. The systematic knowledge of industrial manufacturing has created an environment facilitating the introduction of Artificial Intelligence (AI). Existing computer hardware and programming technologies have been developed to implement more complicated and flexible manufacturing systems [1]. On the other hand, the increasing demand for more effective information processing methods and manufacturing strategies in industrial applications cannot be met by current computing techniques.

The advanced technologies in manufacturing have addressed the research interests of Integrated Intelligent Manufacturing Systems (IIMS), which surround the theory and applications of artificial intelligence technology, computer technology and manufacturing technology (Figure 1.1). An Integrated Intelligent Manufacturing System is developed for improving product quality, increasing productivity and ensuring operational safety. The major objectives of intelligent manufacturing technology is to improve manufacturing efficiency through artificial intelligence.

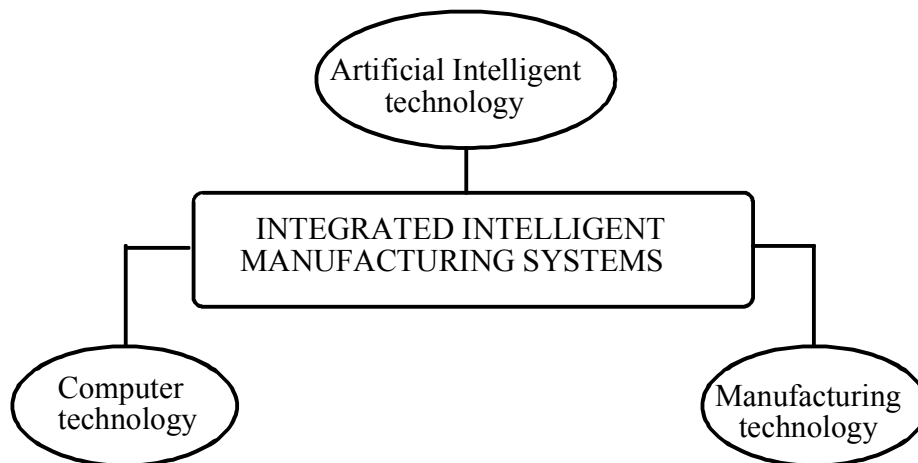


Figure 1.1 Intelligent manufacturing system.

Many engineering problems are not yielding to purely algorithmic computation. They are usually ill-structured problems that deal with non-numerical or non-algorithmic information and are suitable for the use of Artificial Intelligence. Artificial Intelligence has widely applied various disciplines, including manufacturing. It aims at processing non-numerical information, using heuristics and simulating a human being's capability in problem-solving. Expert System (or knowledge-based system) technology is one of the most active branches in Artificial Intelligence research. An Expert System is also a computer program that acquires the knowledge of human experts and applies it to make inferences for users with less training or experience in solving various problems. Expert System provides a programming methodology for solving ill-structured problems that are difficult to handle by purely algorithmic methods. An Expert System is constructed in a way that it allows us to capture the way that people reason and think. The experience from developing Expert System to solve engineering problems has shown that their power is most obvious when the considered problem is adequately complex. By use of Artificial Intelligence techniques, the search could be minimised.

Design is a special process for generating, analysing, evaluating models in manufacturing. The generation of models is a synthesising process and evaluation is an analysing process. As a result, design activities include two kinds of tasks:

1. numerical computation (design analysis) and,
2. symbolic reasoning (design synthesis).

The current Computer-Aided Design (CAD) technology is very efficient at carrying out the numerical computation, but inefficient for symbolic reasoning. Symbolic reasoning make a more notable impact on the manufacturing cost. Therefore, the implementation of design synthesis automation will improve the efficiency of engineering design. Introducing Artificial intelligence technology into the existing

CAD systems is a very effective means of implementing the intelligent design environment that has the capacity for creative design [2], [3].

An important content in Computer Integrated Manufacturing Systems (CIMS) is an intelligent robot. An intelligent robot should be able to sense (seeing and touching), to think (decision-making), and to act (moving and manipulating). AI technology can help us in dealing with the following four types of robotics problems [4].

1. kinematics and design,
2. robot selection,
3. workspace layout and
4. planning and maintenance.

Computer Aided Process Planning (CAPP), proposes to build a bridge for implementing the integration of Computer Aided Product Design (CAD) and Manufacturing (CAM). A number of problems on intelligent CAPP, including representation of non-geometric and geometric features, identification of parameters and specifications, problem-solving strategies, have been discussed [5].

Most of the AI applications in industrial processes have been studied for process control and design, [6] and most of the existing intelligent systems for industrial control are implemented for design purposes [7]. Control system design is a very complex and not a completely understood process. Such design is largely a trial-and-error process, in which human design experts often employ heuristics or the rules of thumb. An intelligent system can provide a user-computer interface in such a process to the efficiency of design [8]. The primary function of the controller can be improved by introducing AI technology into the control system [9].

The production planning process in a manufacturing environment is a complex task. Intelligent planning is engaged in understanding the nature of problem solving by a group of intelligent agents and studying problem solving and planning strategies. Intelligent planning for a Flexible Manufacturing System (FMS) aims at implementing the automation of production scheduling in the CIM environment through linking an information management system with a material manufacturing environment.

The intelligent maintenance system is one of the most successful AI applications in manufacturing. Maintenance of equipment involves a diagnostic procedure incorporating many rules as well as judgement decisions. Own experience and experts' knowledge are very important factors when an engineer locates a failure problem and implements an appropriate correction. Intelligent maintenance systems are utilised to assist maintenance personnel in performing complex repairs by presenting menu-driven instruction guides for diagnosing, predicting, interpreting and monitoring equipment faults [10], [11].

1.1.1 SYMBOLIC REASONING SYSTEMS IN MANUFACTURING ENGINEERING

Symbolic reasoning systems are developed to solve ill-structured problems that are difficult to handle by purely algorithmic methods. The knowledge base of a symbolic reasoning system contains two kinds of knowledge:

1. public knowledge and,
2. private knowledge (expertise).

Symbolic reasoning is a specific computer program, implemented by a specific programming technique, and used to solve a specific problem in a specific domain. The separation of database, knowledge base and inference engine in the expert system allows us to organise the different models and domain expertise efficiently because these components can be designed and modified separately. These systems are widely applied in the research of intelligent manufacturing systems [4].

In our research KnowledgePro was used as a programming tool. This language is used for developing symbolic reasoning systems based on production rules. The tool consists of three components (Figure 1.3):

- i. knowledge base (production memory),
- ii. inference engine (executor) and
- iii. database (working memory)

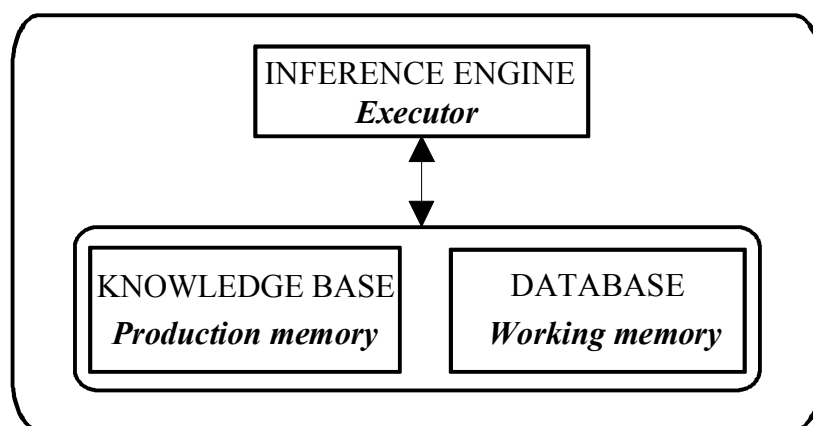


Figure 1.2 The core of symbolic reasoning systems

The production memory contains the general knowledge about a problem. The expert knowledge of the problem is described by a set of production rules stored in

production memory. The typical production rule is described as **IF** (condition), **THEN** (action). Each production rule contains of a condition and action.

The working memory is a special buffer-like data structure and holds the knowledge that is accessible to the whole system. Each unit of working memory is an attribute value element. Any attribute that is not assigned a value for particular instance, is given the default value designated as 'nil'.

The inference engine is an executor. It must determine which rules are relevant to a given data memory configuration and select one to apply. This control strategy is also called conflict resolution. The inference engine can be described as a finite state machine with a cycle consisting of three steps:

1. matching rules,
2. selecting rules and,
3. executing rules.

1.1.2 COUPLING INTELLIGENT SYSTEMS IN MANUFACTURING ENGINEERING

Several existing symbolic reasoning systems can only process symbolic information and make heuristic inferences. The deficiency of numerical computation limits their capability to solve real engineering problems. Computer-aided manufacturing techniques followed the development of mathematical computing theory and techniques. Mathematical modelling is not the only means to describe real manufacturing problems [5]. Many manufacturing problems are ill-structured ones that deal with non-numerical information and non-algorithmic procedures, and are suitable for the application of Artificial Intelligence technology [6].

In Chapter 6, the qualitative and the quantitative analyses together have been applied to help the human operator in the sulfite pulping process to produce good quality pulp. Figure 1.4 shows the structure of qualitative and quantitative analyses. Generally, qualitative decisions are mainly based on symbolic and graphical information, while quantitative analysis are obtained from numerical information. Both methods often complement each other.

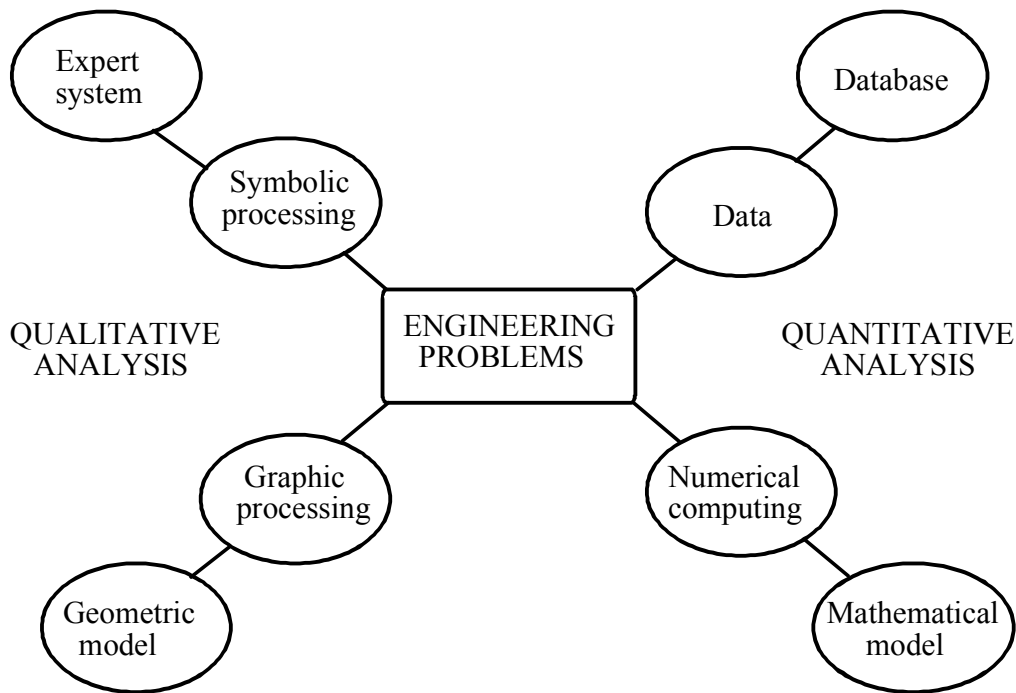


Figure 1.3 Structure of qualitative and quantitative analyses.

In a manufacturing environment, not only a qualitative description of system behaviour is required, but also a quantitative analysis is needed. A main disadvantage of existing symbolic reasoning systems is their incompetence to handle numerical (quantitative) computation. Artificial intelligence should emphasise symbolic processing and non-algorithmic inference [6], but it should be noted that utilisation of numerical computation packages will make intelligent systems more powerful in dealing with engineering problems. Artificial intelligence cannot be used as a substitute for numerical computation.

1.2 THE INTEGRATED INTELLIGENT SYSTEMS APPROACH

Today it is clear that many of the most valuable applications of intelligent system technology are those in which intelligent systems access databases used for other purposes or link to other types of software or systems. Many important applications in the future will combine conventional data processing with intelligent system technology. Knowledge engineering will be combined with information engineering. A lot of software will have an artificial-intelligence component as a small part of its overall code. Because of this it has become important that intelligent systems are run

on the machines and use the software interfaces that are employed for conventional computing. They must work with the operating systems, database systems, user interfaces, and programming languages that constitute the world of data processing. Intelligent systems will provide databases with new, innovative techniques and tools for data design and retrieval.

So far, many intelligent systems have been developed. However, their capability of dealing with complicated manufacturing problems is very limited. The most urgent problems to be solved are how to coordinate symbolic reasoning, numerical computation, as well as computer graphics, and how to integrate heterogeneous intelligent systems.

With the accumulated experience of building expert systems, integrating different intelligent systems into a large-scale knowledge environment is often necessary but difficult. In such an environment, a supervisory system that controls and manages the heterogeneous intelligent subsystems is required. The supervisory system has to provide integration functions in the following phases:

1. Integration of knowledge of different disciplinary domains.
2. Integration of empirical expertise and analytical knowledge.
3. Integration of various objectives, such as research and development, system design and implementation, process operation and control.
4. Integration of different symbolic processing systems (expert systems).
5. Integration of different numerical computation packages.
6. Integration of symbolic processing systems, numerical computation systems, and computer graphics packages.

Phases 1 and 2 are at the knowledge level. Phase 1 also indicates the characteristics of modern engineering techniques. Phase 3 functions at both the knowledge and the functional levels. Phases 4 through 6 perform their integration functions at the functional level, through the problem-solving level to the program level.

Applications of intelligent subsystems in manufacturing can be classified as follows (Figure 1.2):

- intelligent design
- intelligent process planning
- intelligent operation
- intelligent production planning
- intelligent testing and assembling
- intelligent maintenance

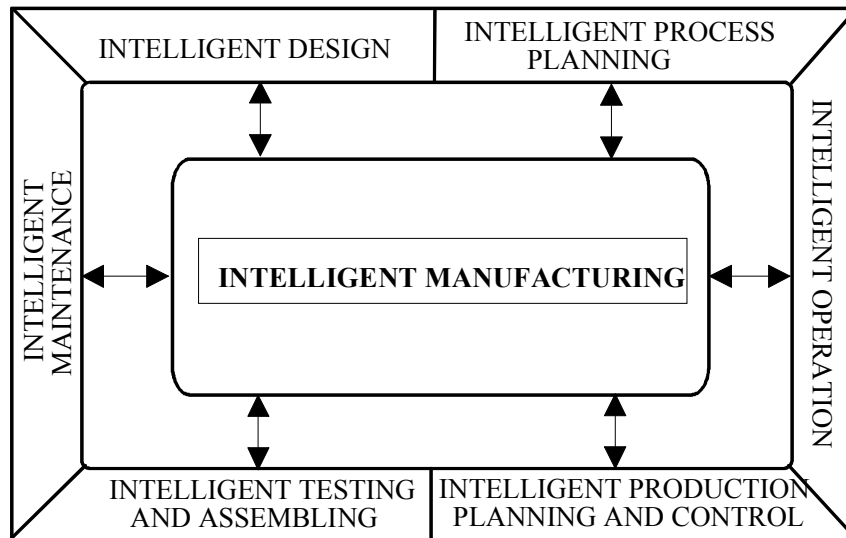


Figure 1.4 Integration of intelligent systems

Integrated Intelligent System (IIS) is a large-scale knowledge integrated environment. IIS consists of symbolic reasoning systems, numerical computation packages, database systems, computer graphics packages and a meta-system to solve complex engineering problems. The integrated software environment allows the running of programs written in different languages and communication among the programs as well as the exchange of data between programs and database. All the subsystems are under the control of a supervising intelligent system, namely, the meta-system (Chapter 3). The meta-system manages the selection, coordination, operation and communication of these subsystems.

1.3 GENERAL VIEW OF THIS THESIS

International computation has intensified the requirement for high quality products that can compete in the global marketplace. As a result of this increased competition, the pace of product or system development has been quickened, thus forcing manufacturers into an era in which continuous quality improvement is a matter of survival, not simply competitive advantage. To overcome problems in manufacturing we need a new technology which can solve the problems easier and quicker. In this thesis we provide a concept of Integrated Intelligent Systems (IIS). To build a model of the system a programming tool is needed. We have found that KnowledgePro can

help to build an Integrated Intelligent System in an easy way. This software is low cost and can be run on a PC. The fundamental principles are described in Chapter 2.

With the experience of building an Intelligent System we see that integrating several intelligent systems into a large scale knowledge environment is often important and necessary but difficult. In such an environment, we need a supervisory system that can control and manage the different intelligent systems. This supervisory system provides many integration functions in different phases. Such a kind of supervisory system is defined as a **meta-system** in our terminology. In other words, a meta-system is a kernel to manage and control the selection, communication, coordination and cooperation of the subsystems which are used in integrated intelligent systems. Chapter 3 gives more details about this system and its role in Integrated Intelligent Systems.

In Chapter 4 an Integrated Intelligent Production Management System is developed (IIPMS). The example presented in this chapter involves the selection of an alternative for the production of a part in a manufacturing operation. First we present the problem definition, then we present some concepts of break-even points for the comparison of alternatives on the basis of cost or profit. Finally, we present an intelligent system model to solve the problem presented.

In Chapter 5 an Integrated Intelligent Process Planning System (IIPPS) is developed for solving a real world problem in the domain of machining process planning, which is a small scale prototype of a large system that would be implemented in a manufacturing organisation. This system drives part geometry and other related information from a computer aided design (CAD) database in the form of a descriptive language, and produces a detailed machining process plan to manufacture parts.

in Chapter 6 presents the construction of the Integrated Intelligent Process Support System (IIPSS) for a batch sulfite pulping process. This chapter first presents some background information about sulfite pulping at a real industrial plant in order to introduce the application of the Integrated Intelligent Process Support System (IIPSS). Then problems in the pulping process, solving strategies, the quantitative model and the qualitative model are discussed. The system construction based on the concept of integrated intelligent systems and the construction of the IIPSS are discussed. Finally, implementation and demonstration are presented, and conclusions drawn.

The last Chapter presents the development of an expert advisor, that can be used to find the faults in the chocolate factory.

1.4 REFERENCE

- [1] Bitran, G. and Papageorge, T. Integration on manufacturing policy and corporate strategy with the aid of expert systems, in Proc. of the First In. Conf. on Expert Systems and Intelligent Manufacturing (ed. Oliff, M.D.), Menolo Park, CA, pp. 13-44 (1987).
- [2] Mann, R.W. Computer-Aided Design, Mcgraw-Hill Tear Book Science and Technology (1965).
- [3] Brown, D.c. and Chandrasekaran, B. An approach to expert system for mechanical design. Proc. of Trends and Applications, National Bureau of Standards, Gaithersberg, MD, (1983).
- [4] Soroka, B. I. 'Expert systems and robotics.' Proc. of the Robots 8th Conf., pp. 113-40 (1984).
- [5] Cheung, y.p. and Dowd, A-L. 'Artificial intelligence in process planning.' Computer-Aided Engineering, 5, 153-56 (1988).
- [6] Stephanopoulos, G. and Davis, J. 'Artificial intelligence in process systems engineering. CACHÉ Monograph Series, 1(1990).
- [7] Pang, G.K.H and MacFarlane, A.G.J. 'An Expert System Approach to Computer-Aided Design of Multivaivable Systems, Springer-Verlag, New York. (1987).
- [8] Lamont, G. and Schiller, M. W. 'The role of artificial intelligence in computer aided design of control system'. Proc. Ieee 26th Conf. on Decision and Control, Los Angles, CA, pp. 1960-65 (1987).
- [9] Astrom, k.j. 'Processes control-past, present, and future.' IEEE Control Magazine, August, 3-10. (1985).
- [10] Miller, R. K. and Walker, T.C. 'Artificial Intelligence Applications in Engineering', SEAI Tecnical Publications, Georgia. (1988).
- [11] Rao, M., Wang, Q., Coward, J. and Lamb, D.K. 'Maintenance support system for truck condition monitoring.' Proc of CSchE, Toronto. pp.242-43 (1992).
- [12] Talukdar, S.N., Cardozo, E. and Leao, L.V. Toast: power system operator's assistant. IEEE Computer, July, 53-60 (1986).
- [13] Rao, M., Jiang, T.S. and Tsai, J.P. An intelligent direction selector for the controller's action in multiloop control systems. Int. J. Intelligent Systems, 3, 361-79 (1988).
- [14] Buchanan, B.G. Expert systems in an overview of automated reasoning and related fields. Journal of Automation Reasoning, 1, 28-34 (1985).

2. A TOOL FOR BUILDING INTELLIGENT SYSTEMS

2.1 INTRODUCTION

For a variety of reasons there is an increasing interest in the use of Artificial Intelligent (AI) Technology. To build an intelligent system we need a software which is easy to use. In the past, the tasks to design and develop intelligent production planning, intelligent design systems, intelligent operation systems, intelligent control systems, intelligent maintenance systems and intelligent process planning systems have always been difficult and time consuming [1]. Fortunately, as the latest improvement and refinement of the available development tools has increased, so the realisation of these tasks has become easier than before.

There are available a substantial number of intelligent software tools. Unfortunately, many of the intelligent development tools that are in current use suffer from a number of severe limitations. Some of the more important of these are:

1. Difficulty of use.
2. Limited power and flexibility.
3. No ability to interface with other software.

A suitable development tool that can overcome some or all of these limitations is therefore required. The tool must be able to allow user to implement our tasks easy and when designing an intelligent system for use in real world situations many guidelines need to be taken into account.

One of the important facts is media utilisation. Media utilisation paradigms describe how the knowledge storage facilities be organised and controlled. The other important utilisation paradigms is hyper-media. Hyper-media (for example hyper-text) refers to the ability of a designer to inter link units of knowledge in a non-linear fashion [3],[4]. This concept is described later in this chapter. Most systems fail to realise the need for facilities which will allow intelligent system designers to structure multi-media knowledge in an acceptable way. Through the use of suitable knowledge engineering environments and expert system shells, this will be overcome. Therefore this chapter describes:

1. Our motivation for using KnowledgePro as a building tool.
2. Some "Evaluation Scenarios" that we have used in order to assess the utility of the KnowledgePro system.

2.2 WHY KNOWLEDGEPRO?

Most of us are interested in the use of low-cost commercially available knowledge processing systems for the production of intelligent systems. The system that we have been exploring is called KnowledgePro[5].

KnowledgePro was one of the first tools to combine both intelligent systems and multimedia technologies in a single development tool to be run on a PC. Knowledge Garden regards this package as a new kind of communications tool that allows the personal computer to become a medium for communicating ideas and information. KnowledgePro (Windows) incorporates all the features of its DOS-based package for developing Microsoft Windows applications. The review that follows will indicate those features specific to the MS- Window 3.x version of the product.

Major reasons for wanting to use the KnowledgePro system as a tool for building the intelligent systems are as follows:

1. Ease of use.
2. Its build-in-hyper-text facility.
3. Its powerful input and output facilities.
4. Consistent and friendly end-user interface.
5. The facilities it offers for dynamic programming.
6. The fact that it can act as an expert-system shell.
7. Its powerful range of primitive commands and functions.
8. Its ability to interface with other software items and packages.

KnowledgePro allows rapid building complex window applications. The development environment is made up of:

- debugging tools
- a multi-document editor
- sample code and applications
- design tools for point and click design
- a flexible, customisation help system
- a librarian and library of topics to extend the language
- a rich Object Oriented Programming (OOP), list processing language.

Since it is a high level language which is learned quickly by inexperienced programmers, it provides low level access to the window Application Program Interface (API) and messages.

There are six basic commands in KnowledgePro to learn by a potential users for developing a window.

- **say**
- **ask**
- **if.....then**
- **#m.....#m**
- **window**
- **close-window**

All these commands are used in figure 2.1 and 2.2.

The KnowledgePro system can act as an expert system shell. It can also be used to create domain-specific expert systems. It therefore offers a mechanism for producing an intelligent system which ebbed rules that can be used in different areas.

2.3 APPLICATIONS ENVIRONMENT

We can classify the application environment in two categories.

1. Procedural knowledge
2. Relational knowledge

2.3.1 PROCEDURAL KNOWLEDGE

KnowledgePro is actually a high-level computer language that facilitates intelligent system development using a verity of strategies. One can represent procedural knowledge as a collection of rules or else as a set of "topics", the basic structural unit of the language. A topic can contain commands (like a procedure), store values (like a variable), return values (like a function), be assigned properties (like a frame), be arranged hierarchically and thereby demonstrate inheritance behavior like system commands, or even thread to hypertext. Knowledge Maker, a companion product, is

capable of inducing a set of rules from a group of examples which developers then can import directly into KnowledgePro or a number of other development tools. Although KnowledgePro's inference mechanism is primarily chaining, developers can program this tool to chain in a forward direction as well.

The producer of KnowledgePro has carefully avoided including uncertainty handling features in KnowledgePro, preferring to force the developer to ask the user more specific questions then to hedge the system's advice by associating confidence factors with its recommendations.

2.3.2 RELATIONAL KNOWLEDGE

KnowledgePro has a finely developed hypertext component that allows developers to include the information associated with a node either externally (as in a hyper stack) or internally (as part of the program source code). As with procedural knowledge, the "topic" structure is the primary vehicle for manipulating hypertext in KnowledgePro.

2.4 THE DEVELOPMENT ENVIRONMENT OF INTELLIGENT SYSTEMS

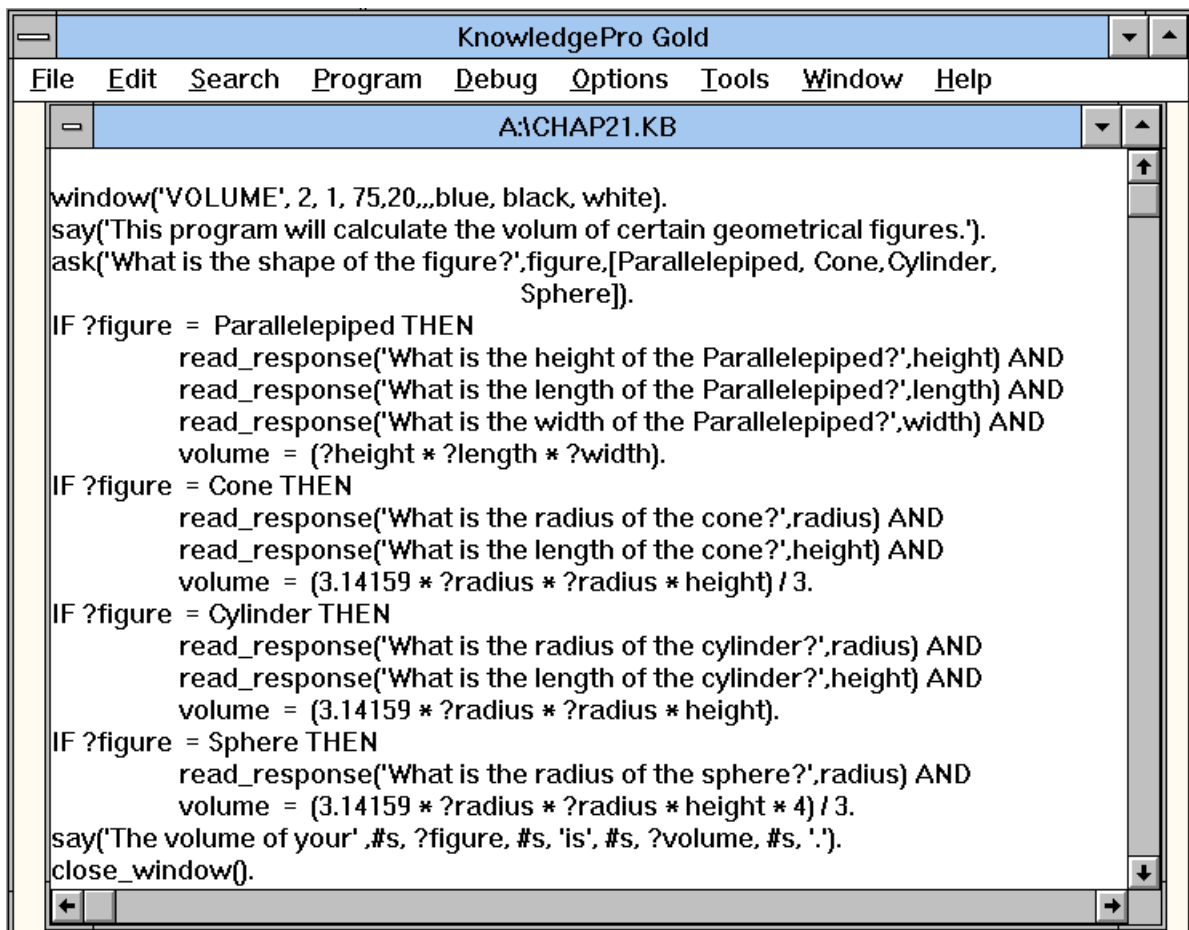
2.4.1 INTELLIGENT SYSTEM BUILDING

KnowledgePro is not a menu-driven tool. As a high-level compiled language, it contains over 100 keywords (though only a handful are required to develop straightforward systems) are strict syntax rules. It contains a wealth of string manipulation functions (much like LISP), which are convenient for more sophisticated development projects. KnowledgePro also contains its own editor and takes full advantage of Windows features such as the Clipboard.

To illustrate the difference between developing a system using rules and one using topics, consider the task of writing an application to calculate the volume of parallelepiped, cone, cylinder and sphere. Figure 2.1 is a script written in KnowledgePro that uses rules to solve this problem.

The "window" command opens a window titled "VOLUME". A blue border will surround a black background, and text appearing in this window will be white. The window's upper left-hand corner is in column 2, row 1, and the window is 75 columns wide and 20 rows long.

The "say" statement simply displays the indicated text, while the "ask" statement not only displays a question, but also provides a menu of possible answers (contained in square brackets). The user's response to the question is stored in the variable "figure".



```

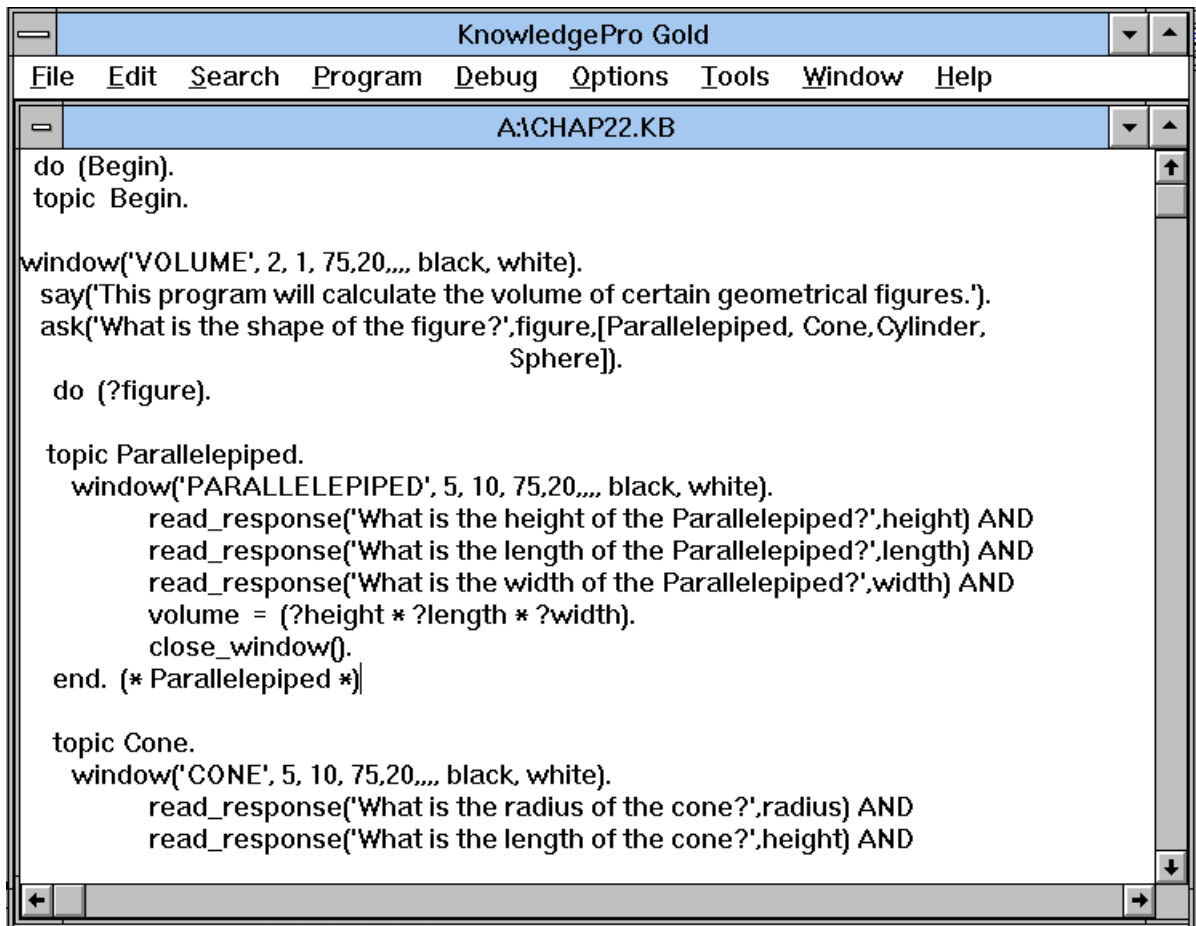
window('VOLUME', 2, 1, 75, 20, blue, black, white).
say('This program will calculate the volum of certain geometrical figures.').
ask('What is the shape of the figure?', figure, [Parallelepiped, Cone, Cylinder,
        Sphere]).
IF ?figure = Parallelepiped THEN
    read_response('What is the height of the Parallelepiped?', height) AND
    read_response('What is the length of the Parallelepiped?', length) AND
    read_response('What is the width of the Parallelepiped?', width) AND
    volume = (?height * ?length * ?width).
IF ?figure = Cone THEN
    read_response('What is the radius of the cone?', radius) AND
    read_response('What is the length of the cone?', height) AND
    volume = (3.14159 * ?radius * ?radius * height) / 3.
IF ?figure = Cylinder THEN
    read_response('What is the radius of the cylinder?', radius) AND
    read_response('What is the length of the cylinder?', height) AND
    volume = (3.14159 * ?radius * ?radius * height).
IF ?figure = Sphere THEN
    read_response('What is the radius of the sphere?', radius) AND
    volume = (3.14159 * ?radius * ?radius * height * 4) / 3.
say('The volume of your' ,#s, ?figure, #s, 'is', #s, ?volume, #s, '.').
close_window().
  
```

Figure 2.1 Edit window for using **Rules**

Depending on the value of the variable "figure", (indicated by "?figure"), one of the two rules will fire. In either case the system asks the user another question or two (this time no menu of possible answers appears), the answers to which will be stored in the variables "height", "length", "width", or "radius", as appropriate. The system then applies the area formula or the particular figure to the value(s) of the variable(s).

Finally, the answer is output by another "say" statement and the window closes. (The "#s" symbol in the final "say" statement is formatting command).

Now consider the same problem, this time solved by topics rather than rules (Figure 2.2). This script contains four topics, "Parallelepiped", "Cone", "Cylinder" and "Sphere", nested within the main topic, "Begin". When users respond to the "ask" statement in "topic Begin" their answer is stored in the variable "figure". The "do(?figure)" command then accordingly executes either "topic Rectangle" or "topic Circle". Each of these topics opens a smaller window, overlaying the original full-screen window, prompts the user for data, computes the area of the figure, and closes its window. Finally, "topic Begin" outputs the answer and closes the main window.



```

do (Begin).
topic Begin.

window('VOLUME', 2, 1, 75,20,,,, black, white).
say('This program will calculate the volume of certain geometrical figures.').
ask('What is the shape of the figure?',figure,[Parallelepiped, Cone,Cylinder,
      Sphere]).

do (?figure).

topic Parallelepiped.
  window('PARALLELEPIPED', 5, 10, 75,20,,,, black, white).
  read_response('What is the height of the Parallelepiped?',height) AND
  read_response('What is the length of the Parallelepiped?',length) AND
  read_response('What is the width of the Parallelepiped?',width) AND
  volume = (?height * ?length * ?width).
  close_window().
end. (* Parallelepiped *)

topic Cone.
  window('CONE', 5, 10, 75,20,,,, black, white).
  read_response('What is the radius of the cone?',radius) AND
  read_response('What is the length of the cone?',height) AND

```

Figure 2.2 Edit window for using **Topics**

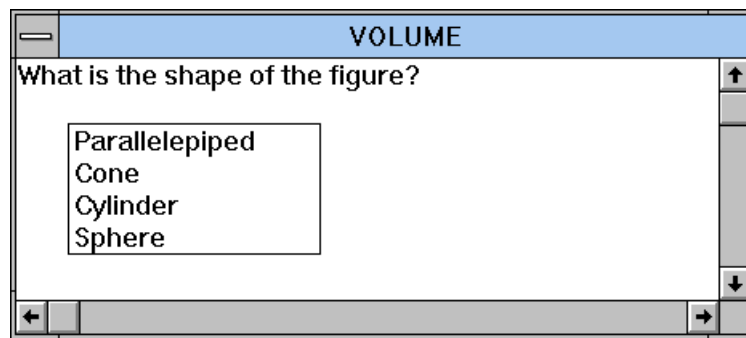


Figure 2.3 Layout window for using Rules or Topics

Though the topics Parallelepiped, Cone, Cylinder and Sphere contain mathematical calculations and windowing commands, the script of Figure 2.2 only begins to demonstrate the versatility of the topic structure in KnowledgePro. Figure 2.3 shows the layout window in both cases.

2.4.2 HYPERTEXT IMPLEMENTATION

To convert a phrase in KnowledgePro, *every* word and phrase is a button, but the "#m#" notation still serves to group a collection of words as a single hypertext concept. Corresponding to the button should be a topic of the same name that KnowledgePro will execute upon a user's clicking on the buttoned phrase (using mouse function key). For example, suppose the first "say" statement in topic "Begin" of Figure 2.2 were modified to read:

```
"say (This program will calculate the volume of certain  
#mgeometrical figures#m.)."
```

and the following topics were added to the script:

```
topic 'geometrical figures'.  
window('Geometrical Figures', white, green, white, 2, 1, 75, 20).  
say('This program handles only parallelepiped, cone, cylinder, and  
sphere.').  
close_window().  
end.
```

During run time, the phrase "geometrical figure" would be highlighted and, if the user elected to click on this phrase, topic "geometrical figures" executes, resulting in a sub window opening and displaying the sentence in its "say" statement. If the message to be displayed were lengthy, an option would be to store it in an external file, say "geometry" and rewrite topic "geometrical figures" as follows:

```
topic 'geometrical figures'.  
window('Geometrical Figures', white, green, white, 2, 1, 75, 20).  
text is read (geometry).  
close_window().  
end.
```

In this case the content of the file named "geometry" is stored in the variable named "text" which the "say" statement then displays.

2.4.3 KNOWLEDGE VISUALISATION AND TRACING

By turning on the DEBUG feature, a developer can readily access a visual that displays the hierarchy of the knowledge base (as depicted in Figure 2.4).

When the TRACE option is invoked, each step of the execution of a knowledge base displays either on the screen or to a specified file, thus facilitating both debugging and report generation. The EVALUATE feature opens a window and allow execution of any KnowledgePro command (for example, checking the current value of a variable), which is a helpful attribute when debugging a knowledge base [6].

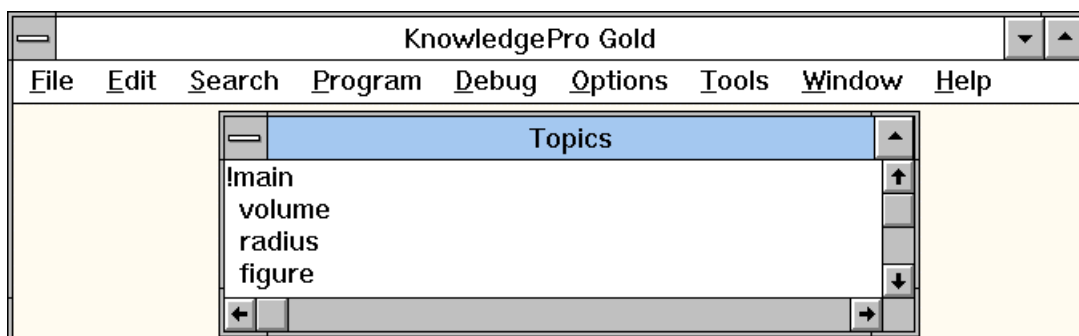


Figure 2.4 Hierarchical structure of a knowledge base.

2.4.4 INTEGRATION WITH OTHER PROGRAM TOOLS

Because of its ability to make calls to the operating system, KnowledgePro can run any external program without leaving the KnowledgePro environment. A special external library of utility programs achieves integration with other popular applications programs, graphic package, data bases, spreadsheet and standard programming languages. The current KnowledgePro external library supports dBASE, Lotus 1-2-3, PC Paint, Turbo Pascal, and C functions.

2.5 THE USER ENVIRONMENT

2.5.1 SCREEN DESIGN

As illustrated in Figures 2.1 and 2.2, KnowledgePro, consistent with its image as a communications tool, offers a variety of methods for user interaction. In addition to the "ask" and "read_response" statement, other input commands include "read_character" (which reads one character), "menu" (which generates a menu of acceptable responses), and "edit_file" (which allows the user to input as many lines as necessary). When menus appear on the screen, users can select their responses either by mouse or by the arrow keys and the RETURN key. Developers can also include the full array of Windows graphical-input devices including buttons, radio buttons, icons, and check boxes.

As suggested by Figures 2.1 and 2.2, developers using KnowledgePro have full control of window sizing, placement, and colours (border, background, and text) and can therefore achieve any desired effect. One can also create a list of windows having different characteristics and then invoke them by number throughout the script. The KnowledgePro also includes a special knowledge base to help developers automate the creation of different types of windows.

2.5.2 CUSTOM REPORT GENERATION

Except for the TRACE command described earlier, KnowledgePro does not automatically generate any reports of a session. The WRITE command, however, allows customised file creation that may include text and values of variables. Developers can write to disk, to a printer, to the current window, or directly to a serial port.

2.6 CONCLUSIONS

This chapter has so far described the KnowledgePro system and the way in which it can be used to develop an intelligent system. Since KnowledgePro can integrate other applications such as data base, graphic packages, etc..., we can use it to develop a successful meta-system (chapter 4). This chapter also identified some of the more important guidelines and has outlined how a low-cost knowledge processing system can be used to facilitate their realisation.

We have used the facilities that it offers (in combination with other program tools) in order to produce a successful Integrated Intelligent System (IIS) for use in a number of different subject areas (Chapter 4 to 7).

2.7 REFERENCE

- [1] Braker.P.G.Authour languages for CAL.MACmillan. Basingstoke, UK (1987)
- [2] Braker. P.G. Authoring electronic books. Paper submitted to IFIP 5th worldconference on computers in education,sydney, Australia.(1989)
- [3] Megary. J.Hyper-text and compact disks: the challange of multi-media learning. British jurnal of educational technology 19 ,(3),172-183. (1988).
- [4] Mc Aleese R.hyper-text: Theory into practice. Blackwell scientific, Oxford (1989)
- [5] Thompson, B.and Thompson, B.Knowledgepro user manual knowledge garden Inc, Ny (1991).
- [6] Kesheng Wang, 'An expert system for fault dignosis in car engines' (June 1993).

3 META-SYSTEM STRUCTURE

3.1 INTRODUCTION

The study presented in this chapter is a broader study of the structure and structuring of meta-system. The meta-system approach to Integrated Intelligent Systems presented in this chapter is set in a control systems framework. The meta-system has its own database, knowledge base and inference engine, but the meta-system gives its activities into the separated, strictly ordered phases of information. The meta-system makes it possible for us to integrating and utilising new knowledges. Any communication between programs must really on translation by the meta-system. This configuration makes it possible for us to add or delete programs easily. The most significant fact is that successfully developed software packages can be applied wherever needed. The time and money required in doing repeat work can be saved, making commercial application of intelligent systems feasible.

3.2 THE CONCEPT OF A META-SYSTEM

'Meta' stems from Greek where its meaning is 'after'. Metaphysics start where physics end, that is, it comes after physics. 'Meta' nowadays is mostly used in the meaning of 'above'. Methodology is the science of scientific method. Methodology stands above science and is therefore considered as a metascience. In order to describe the metascientific properties of methodology one usually distinguishes between object-language and metalanguage. A sentence like ' $10 + 10 = 100$ ' is an object-language sentence. A sentence like " $10 + 10 = 200$ " is an arithmetical equation but is, however, a metalanguage sentence. The sentence no longer deals with the objects but with a higher-level property. It stands above the first sentence and it is in this sense that the concept of a meta-system is used. A system is defined as a set of elements and relationships between the elements. As long as one deals with these elements and relationships - the objects of the system - the considerations on the system are at object level. When this level is exceeded, the considerations are at meta-systemic level.

Beer [1,2] explicitly refers to the meta-system when comparing the neuro-physiology of the brain to the organisation of the firm. Later, the idea of a meta-system is elaborated into a cybernetic meta-system model. The concept of a meta-system has implicitly or explicitly been incorporated in the design of computer hardware and software.

Churchman's [3] incursion into the design of inquiring systems is premised on the need for designing the meta-system for human inquiry. Churchman explicitly states the need not only for management science, but also for a science of management, a decision-making science from which we can learn to decide on how to decide, that is, to make metadecisions.

Klir [4,5] conceptualised a hierarchy of epistemological levels of systems which are differentiated by the level of knowledge regarding the set of variables and of potential states which is contained at each level. Mathematics is considered the metalanguage of science. According to Rapport [6] mathematics serve to bring out the isomorphism's existing across particular sciences and thus acts as the metalanguage of General Systems Theory. Within mathematics one can refer to more specific metalanguages. Thus the probability theory has been called the metalanguage of uncertainty and fuzzy set theory the metalanguage of ambiguity [7,9]. Van Gigch [10] discussed the need for a meta-system in the context of a comparison of methodologies for systems design and problem-solving. The concept of a meta-system also appears in decision-making sciences, particularly the related fields of policy making and planning science. Faludi [11] distinguishes between three levels in planning theory, namely metaplanning, procedural theories and substantive theories on planning. The first deals with the design of the planning system, the second with the way plans operate and the last with the area of concern. Faludi explicitly defines metaplanning as the design of planning agencies and their procedures. Some initial hypotheses about a theory of metaplanning are developed by Emshoff [12]. Mitroff and Betz [13] explicitly refer to metadecision-making.

As can clearly be seen from this brief and surely incomplete review of the concept of meta-system in some of the literature, the concept is used in many other different ways. The interpretation of the concept adopted here is related to the first mentioned methodological approach which stems from the difference between object-level language and meta-level language. The study presented in this chapter is a broader study of the meta-system structure and structuring.

3.3 WHY WE NEED A META-SYSTEM IN A MANUFACTURING ENVIRONMENT?

Many intelligent systems have been developed but so far, their ability to dealing with complicated manufacturing problems is still limited. There are some disadvantages existing in intelligent systems [15]. Some of the disadvantages are shown below:

1. Lack of integration of different intelligent systems, software packages and commercial AI tools;
2. Lack of efficient management in intelligent systems;
3. Lack of coordination of symbolic reasoning, neural networks, numerical computation and graphics representation;
4. Difficulty in modifying knowledge bases by end-user rather than the original producers; and
5. Lack of parallel configuration to deal with a multiplicity of knowledge representation and problem solving strategies.

The main problems to be solved in a modern manufacturing environment are how to coordinate data base systems, numerical computations, neural networks, fuzzy logic programs and software systems.

With the experience of building Intelligent Systems, we see that integrating several intelligent systems into a large scale knowledge environment is often important and necessary but difficult. In such an environment, we need a supervisory system that can control and manage the different intelligent systems. This supervisory system provides many integration functions in different phases. Such a supervisory system is defined as a meta-system in our terminology. In other words, a meta-system is a kernel to manage and control the selection, communication, coordination and cooperation of the sub-systems which are used in integrated intelligent systems.

A meta-system is referred to as the control mechanism of the meta-level knowledge, to acquire, maintain, integrate, coordinate, and utilize knowledge from different domains. Knowledge can be classified into domain knowledge and meta knowledge. The domain knowledge is defined as facts, laws, formulae and rules in specific domains of knowledge. The meta knowledge is defined as knowledge about the domain knowledge, and can be used to manage, control and utilize the domain knowledge. A

meta-system can be viewed as an expert system to control, supervise, and coordinate the subsystem and solve the problems among them. A meta-system has its own inference engine, data base and knowledge base, but it does not give the solution to any particular problem within the specific knowledge domain. It only serves as a function which manages the whole software system. For example, the integration of a manufacturing software system is shown in Figure 3.1. The terminology meta-system is widely used in the software engineering field. This new conceptual framework can serve as a universal configuration to develop high-performance intelligent systems for many complicated applications.

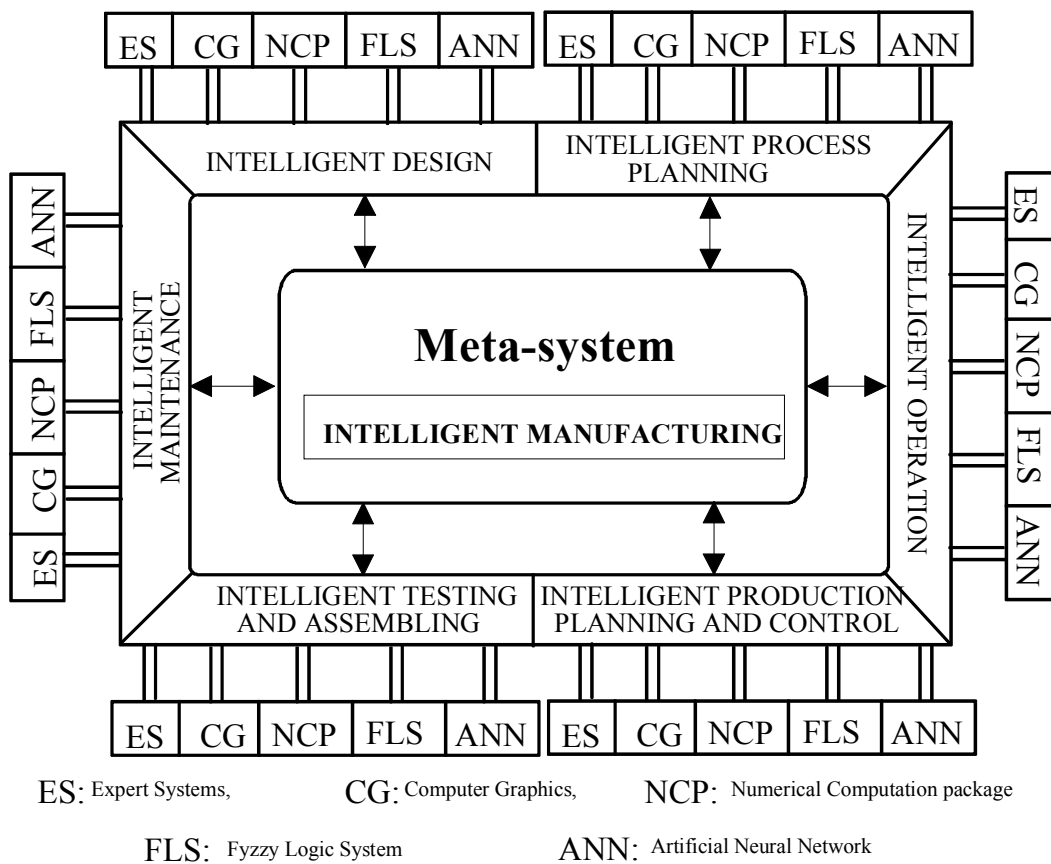


Figure 3.1 Integration of manufacturing software systems.

The meta-system is developed to support the production of information processing systems throughout their life cycle. They greatly help programmers to reduce the time and cost of software development, and to maintain and improve the existing software environment. These meta-systems have provided significant benefits to database systems and software environments.

We have used KnowledgePro as a tool to build a meta-system. The reason why we choose KnowledgePro as a building tool was discussed in chapter 2.

3.3.1 FUNDAMENTAL CONCEPT OF INTELLIGENT MANUFACTURING

In order to realize a high-precision, flexible manufacturing system with easy operation and 24-hour unmanned capability, a completely new concept for "intelligent manufacturing" has been introduced. In manufacturing technology, high precision and high reliability are usually required. Today, completely unmanned operation, flexible operation, and easy operation are also required. Traditional techniques for improving machine tool behavior are not sufficient to satisfy these requirements. Therefore, a fundamentally new concept is necessary. The authors' approach for "intelligent manufacturing" is a possible solution [17].

The relationship between requirements for manufacturing systems and the effects of intelligent manufacturing on these requirements are shown in Figure 3.2. Intelligent manufacturing not only realizes the requirements of ordinary manufacturing systems, but also makes manufacturing in alien worlds, such as in the atomic-scale world or in space, as easy as manufacturing in the normal world. In addition, intelligent manufacturing provides for self evolving manufacturing systems that use automatic data collection and self-learning, and supports the development of creativity by allowing unskilled designers to manufacture complex parts easily.

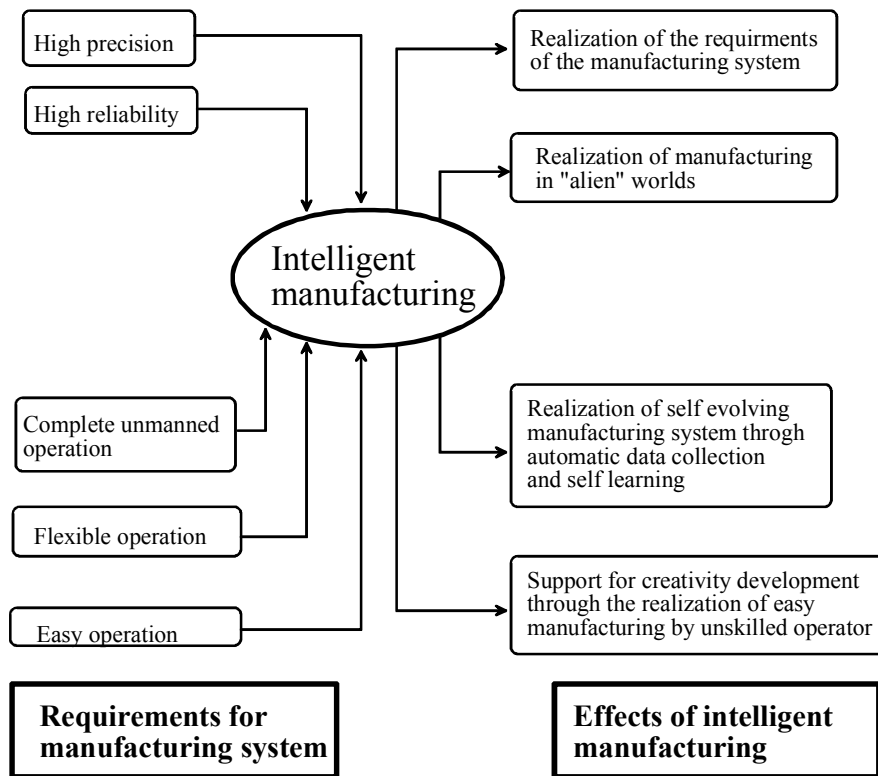


Figure 3.2. Requirements for manufacturing systems and the effects of intelligent manufacturing [17]

Through the use of intelligent manufacturing systems, the manufacturing technology may enter a new stage of development. Moreover, such a system may open new possibilities for manufacturing in alien worlds, such as in space or on the atomic scale.

3.4 FUNCTION OF THE META-SYSTEM IN INTEGRATED INTELLIGENT SYSTEMS

The main function of a meta-system in Integrated Intelligent Systems are described as follows:

- It is responsible for selecting and operating all subsystems. The meta-system monitors the operating process within a subsystem, records the data such as execution time and processes data files such as saving.

- It coordinates the execution of different tasks. In integrated intelligent system the meta-system is working as a coordinator to manage all symbolic reasoning systems, numeric computation routines and database system.
- The meta-system can set an order to invoke subsystems or routines which are needed for a specific task.
- The meta-system is responsible for the integration function. That is why we can acquire new knowledge. It gives us integrating and utilising new knowledges. Any communication between programs must rely on translation by the meta-system. This configuration makes it possible for us to add or delete programs easily. The most significant fact is that successfully developed software packages can be applied wherever needed. The time and money required in doing repeat work can be saved, making commercial application of intelligent system feasible.

For expert users, the meta-system permit them to manipulate, modify and retrieve the objects of knowledge base in subsystems. In this way the system function can be improved by codifying more creative knowledge and experiences from expertise.

3.5 CONFIGURATION OF A META-SYSTEM.

Like other expert systems the meta-system has its own database, knowledge base, and inference engine (Figure 3.3). But it gives its activities into the separated, strictly ordered phases of information collecting and processing. The meta-system configuration has the following components:

1. An interface to the external environment,
2. An interface to internal subsystems,
3. A meta-knowledge base and
4. Inference mechanism.

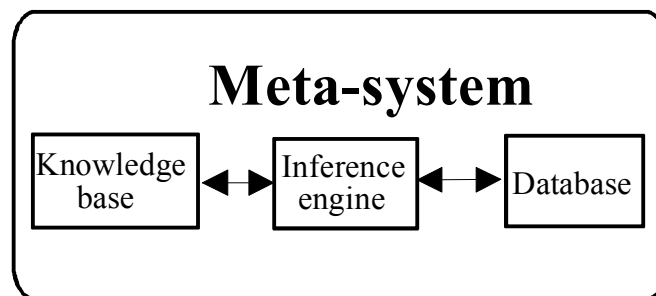


Figure 3.3 The core of the meta-system

3.5.1 INTERFACE TO EXTERNAL ENVIRONMENT

The interface to the external environment builds the communication between the users and internal software systems as well as among the external software systems (subsystems). The interface plays a key role in an open structured software system in two ways:

- i. Codify human expertise into the computer system such that it can adopt the most creative intelligence and knowledge in decision making.
- ii. Communicate with other intelligent software systems to extend the system into a much larger scale for more complicated tasks.

3.5.2 INTERFACE TO INTERNAL SUBSYSTEM

This component of the meta-system is established based on each specific application. The internal interface connects any individual subsystems which are used in problem solving and under the control and management by the meta-system. Each module of the interface converts a nonstandard data from a specific subsystem into standard form in the integrated intelligence environment. Conversion between the standard forms of different languages is carried out by the meta-system.

3.5.3 META-KNOWLEDGE BASE

The meta knowledge base is the intelligence resource of the meta-system. It provides a foundation for the meta-system to carry out the managerial tasks. The meta-knowledge base consist of a compiler and a structured frame knowledge representation facility. The compiler converts the external knowledge representation, which is obtained through the editor and is easy to understand by users, into internal representation forms available in the inference mechanism. The structure of knowledge representation can be production rule or frame or a combination of the two. Characterised by diversity and variety in nature, the meta-knowledge may be better represented in object-oriented frame structures.

There are several modules in the frame to represent different components of meta-knowledge. These components are functioned for specific purposes. For example, the communication standardisation module for heterogeneous subsystems and the conflicting resolution module are formed for a general management purpose at supervisory level. The module for knowledge about subsystems and the task assignment module,, have to be built according to each specific problem. The meta-knowledge base employs an open organisation structure. It allows new intelligent functionality to enter the meta-knowledge base to engage more duties in decision-making.

3.5.4 INFERENCE MECHANISM

The inference mechanism in a meta-system adopts various inference methods, such as forward chaining, backward chaining, certain reasoning, uncertain reasoning, conflict reasoning. The inference mechanism performs operations and processing on the meta-knowledge. It also carries out various actions based on the reasoning results that are passing data between any two subsystems, or sorting new data in the data base. Therefore, there are some functional modules in the mechanism, which further extends the functionality of the inference mechanism.

3.6 IMPLEMENTATION OF META-SYSTEM IN KNOWLEDGEPRO ENVIRONMENT

The meta-system implementation with KnowledgePro has several advantages. The four main reasons to use KnowledgePro as implementation language as follows:

- i. The KnowledgePro language is versatile in both symbolic reasoning and numerical computation.
- ii. The KnowledgePro language is a high level language, which means convenient for program coding and control on hardware, as well as it is easy to use.
- iii. The KnowledgePro language can easily access other language environments by interface written in a KnowledgePro language.

In Integrated Intelligent System, various computer languages, such as dBASE III, Auto CAD, Excel, and C++ are used to build different subsystems. The standardisation of communication among them is very important for the integration of these subsystems. A detail discussion, implementation and conclusion about meta-system implementations is given in Chapter 4 to 7.

The meta-system built by the KnowledgePro environment provides efficient and comfortable to increase problem solving strategy in Integrated Intelligent Systems (IIS). The four different IIS in Chapter 4 to 7 are constructed to solve different tasks. Figure 3.4 shows an Integrated Intelligent System and Figure 3.5 shows the hole structure of a meta-system.

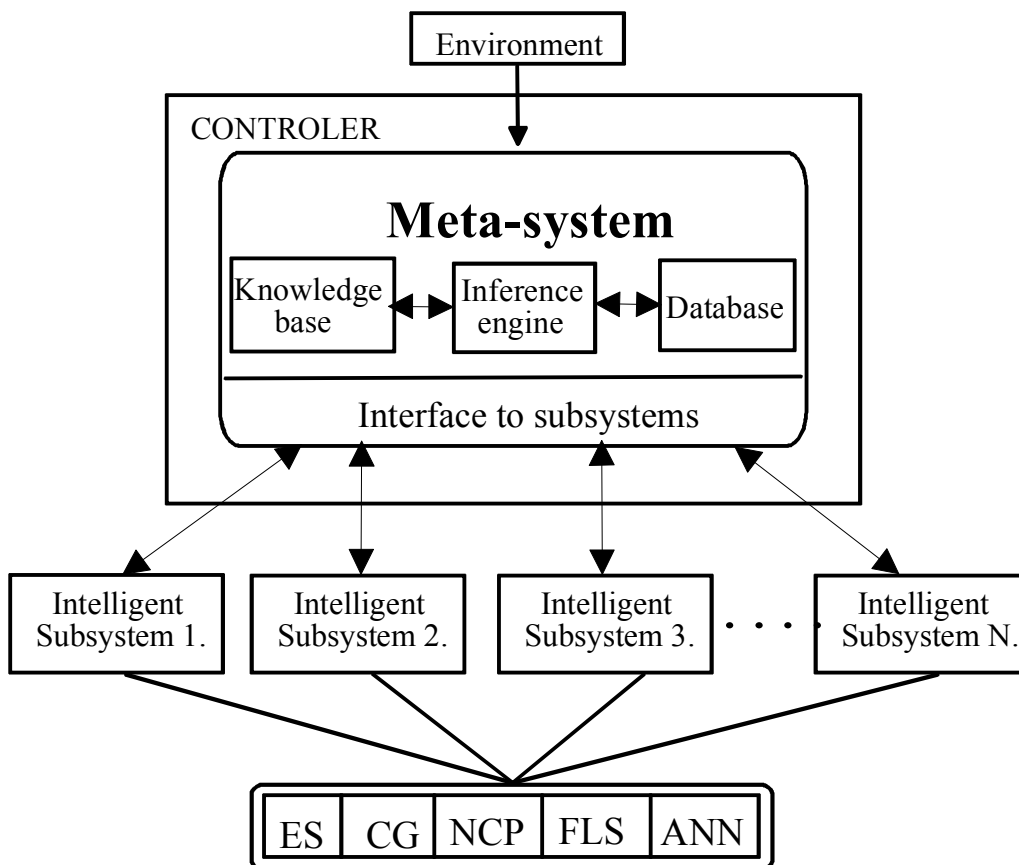


Figure 3.4 Integrated Intelligent System

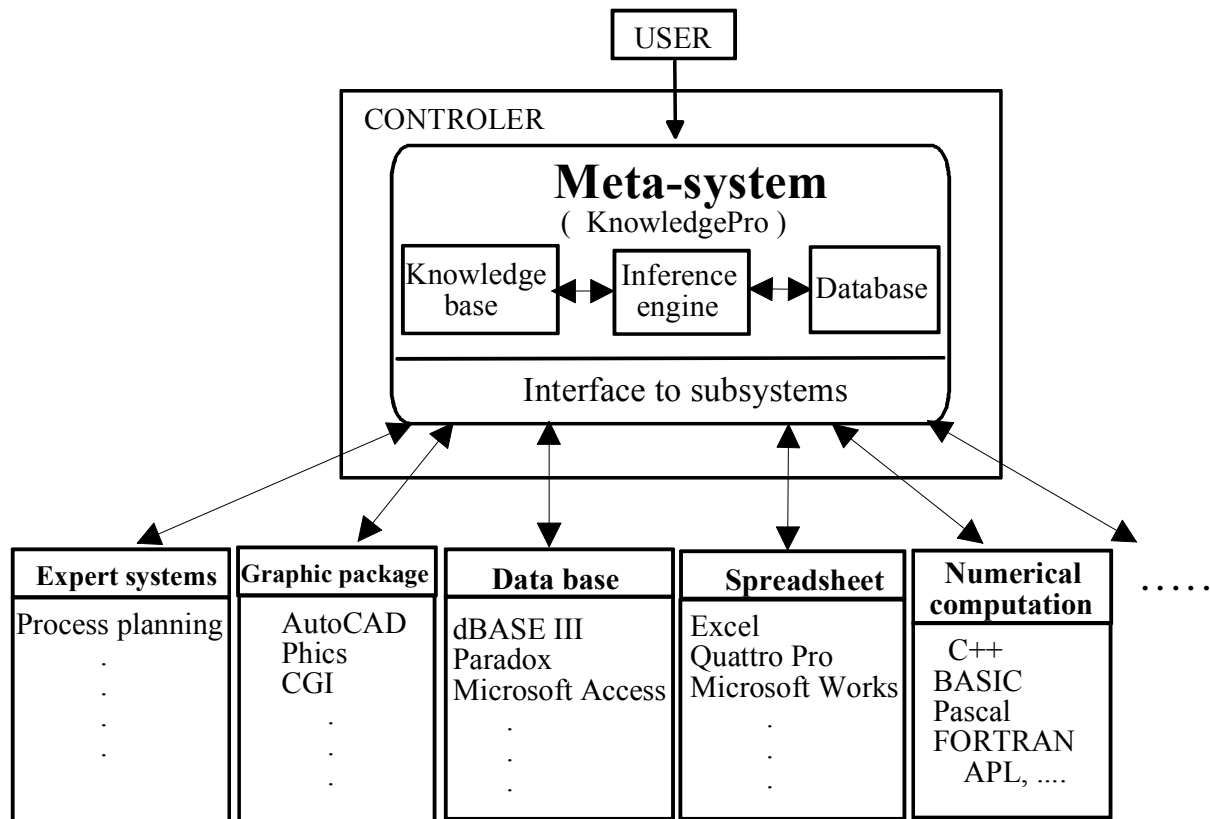


Figure 3.5 PC-based Intelligent System in a software environment

3.7 CONCLUSIONS

The purpose of this chapter was to draw a general concept of a meta-system and its functions. Explicitly incorporating the meta-system into the Integrated Intelligent Systems has the clear advantage of openly setting out the roles of the meta-systems. It then becomes clear that the implementation of this concept in Chapter 4 to 7 exhibits the deferent IIS with separate functions. A meta-system gives an opportunity to modify knowledge bases by end-user rather than the original producers. This new conceptual design framework can serve as a universal configuration to develop high-performance intelligent systems for many complicated applications.

3.6 REFERENCES

- [1] S. Beer. "Decision and Control (Wiley, New York, 1966) and Platform for Change (Wiley, New York , 1975).
- [2] S. Beer, Brain of the Firm, 2nd Ed. (Wiley, New York, 1981).
- [3] C. W. Churchman, "The Design of Inquiring Systems (Basic Books, NewYork, 1971).
- [4] G. J. Klir, "Identification of Generative Structure in Empirical Data", Int. J. of General Systems 3, 89-104 (1976).
- [5] Social Issue of "Reconstruct ability Analysis", Int. J. of General Systems 7, 1 (1981).
- [6] A Rapoport, "Mathematical Aspects of General System Theory", General System 11, 3-12 (1977), and General System Theory, (Abacus, Kent, Eng., 1986).
- [7] J. P. van Gigch, "A Methodological Comparison of the Science, Systems and Meta-system Paradigms", Int. J. Man-Machine Studies **11**. 651-663 (1979).
- [8] J. P. van Gigch and L.L.Pipino, "From Absolute, to Probable, and Fuzzy in Decision Making", Kybernetes **9**. 47-55 (1980).
- [9] W. J. M. Kickert. "Fuzzy Theories in Decision Making" (Nijhoff. Boston and Leiden, 1978).
- [10] J. P. van Gigch. "A Methodological Comparison of the Science, Systems and Meta-system Paradigm", Op. Cit.
- [11] A. Faludi. Planning Theory (Pergamon, Oxford, 1973).
- [12] J. R. Emshoff, "Planning the Process o Improving the Planning Process: A Case Study in Meta-Planing", Management Science **24**. 1095-1108 (1978).
- [13] I. I. Mitroff and F. Betz, "Dialectical Decision Theory: A meta-Theory of Decision Making", Management Science **19**. 11-24 (1972).
- [14] Browston, L., Farrell, R., Kant, E and Martin, N. (1985) 'Programming Expert Systems In OPS5', Addison-Westly, Reading MA
- [15] Cha, J, Zhao, Z. and Rao, M. (1991) Applications of intelligence engineering in Manufacturing process. Proc. of the First Int. Conf. on Manufacturing Technology, HK, pp.179-82.
- [17] Hatamura, Y. "Intelligent manufacturing and its fundamental structure", in Proc Jpn Soc Precision Eng Sept 1989, 297-298.

4. INTEGRATED INTELLIGENT PRODUCTION MANAGEMENT SYSTEM (IIPMS)

4.1 INTRODUCTION

Modern industrial and manufacturing systems have become highly complex, advanced and with possibly incorrect or incomplete information. This characteristic arises out of the need to improve their efficiency. Artificial Intelligence (AI) techniques, and intelligent systems in particular, play an important role in tackling complexity in manufacturing systems. Interest in AI techniques stems from the way that humans cope with the complexities of the physical world. Intelligent systems allow a computer to use heuristic, or rule-of-thumb, methods to solve complex or poorly defined problems that would be impractical to solve using conventional programming.

In the markets, there are a lot of intelligent systems which can be used for different purposes. The fact is that it is difficult to find a suitable system for a special task. Flexible properties may exist but such a system is too expensive or too complex. The best way is to develop your own system which can be easily build up and can solve special problems in your own company by the domain engineers, rather than knowledge engineers [4]. The company also has some software it has used before and it is worthwhile to use easy-programming environments to integrate this software into the intelligent system [1][3].

KnowledgePro is chosen as the programming environment because it combines both intelligent systems and hypertext technologies in one single development tool to run on a PC [2]. The program is easy to use and relatively inexpensive, and has good interface capabilities to external programs. All of this makes it possible for the domain engineers to develop systems to solve special problems in their own company [6][7][8]. The development principle of the system can be used for education training, researching and industrial applications building.

The comparison of alternatives is an important task in manufacturing decision making analysis. Alternatives may need to be selected on the basis of cost, profit, efficiency, productivity, quality, or any other criterion of interest [5][9]. The example presented in

this chapter involves the selection of an alternative for the production of a part in a manufacturing operation. First we present the problem definition, then we present some concepts of break-even points for the comparison of alternatives on the basis of cost or profit. Finally, we present an intelligent system model to solve the problem presented.

In this chapter, we will apply the concepts and framework of Integrated Intelligent System which integrates an intelligent system with a spreadsheet and a computational program written in conventional program languages. A break-even analysis is taken as an example to demonstrate how to build an integrated intelligent system. The integrated intelligent system is developed by use of KnowledgePro, a programming environment. This approach is suitable for small and medium sized enterprises to develop an intelligent system in order to help the managers make a decision in production the process.

4.2 PROBLEM DEFINITION

The relationship of revenues, production costs and machine capacities is a critical factor in production planning. The level of output at which a course of action begins to yield a profit, is its break-even point, or equivalent point. A break-even analysis considers the economic implications of production plans to determine preferred pricing, servicing, manufacturing and scheduling policies.

A break-even model relates fixed costs, variable costs, and revenue to the quantity of units produced. Relationships are conveniently displayed on graphs to assist communication among decision makers. Mathematical expressions of the relationships support the graphs and are used to test the sensitivity of break-even decisions. Any variable in the model can be altered to observe its effect on the other variables. In this way, alternative courses of action may be simulated and evaluated in the planning stage.

A break-even model can be used for analysing the following cases:

- (1) Revenue, cost, and capacity relationships
- (2) Multiple comparisons.

4.2.1 PROCEDURE FOR BREAK-EVEN ANALYSIS

Once all the cost aspects of the alternatives are documented and properly organised, several types of economic analyse can be performed. An example of the procedure for a break-even analysis is presented here. The procedure is modeled later as a component of an intelligent system solution to the problem stated above. The total cost of an operation may be expressed as the sum of the fixed and variable costs with respect to output quantity. That is:

$$TC(x) = FC + VC(x)$$

Where

x = number of units produced,

$TC(x)$ = total cost of producing x units,

FC = total fixed cost,

$VC(x)$ = the total variable costs associated with producing x units.

The total revenue resulting from the sale of x units is defined as:

$$TR(x) = px,$$

Where p is the price per unit. The profit due to the production and sale of x units of the product is calculated as:

$$P(x) = TR(x) - TC(x).$$

The break-even point of an operation is defined as the value of a given parameter that will result in neither a profit nor a loss. The parameter of interest may be the number of units produced, the number of hours of operation, the number of units of a resource type allocated, or any other measure of interest. At the break-even point we have the following relationship:

$$TR(x) = TC(x)$$

or

$$P(x) = 0.$$

In some cases, the relationship between cost and a parameter of interest can be expressed in a mathematical formula. For example, there may be a linear cost relationship between the total cost of a manufacturing alternative and the number of units produced. The cost expression facilitates straightforward analysis for decision-making purposes. In some cases, the relationship between cost or profit and a parameter

of interest can be expressed in a linear function, which is shown in Figure 4.1a. In other cases, break-even chart shows as a non-linear function of output, which is shown in Figure 4.1b. The cost or profit expressions facilitate straightforward analysis for decision-making purposes.

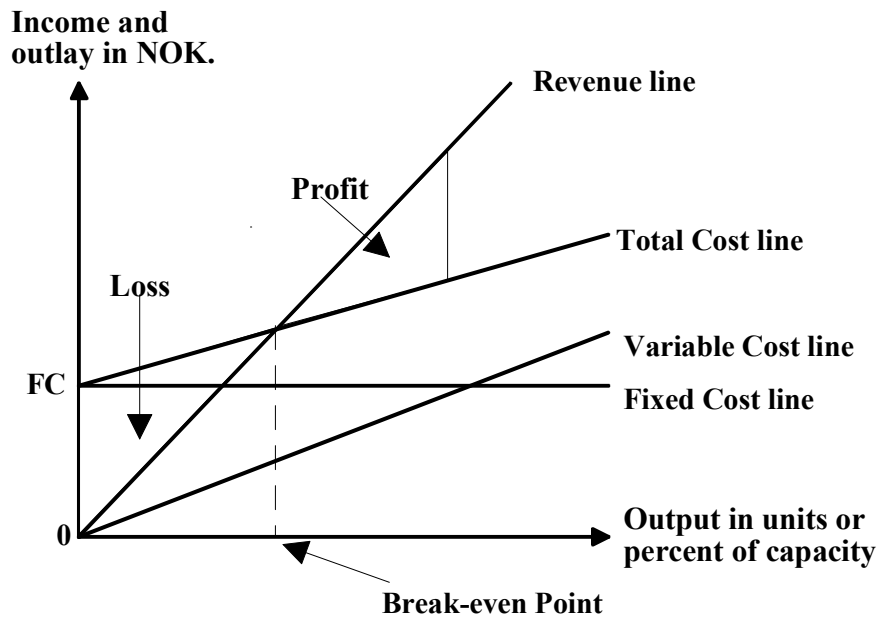


Figure 4.1a Example of liner Break-even chart

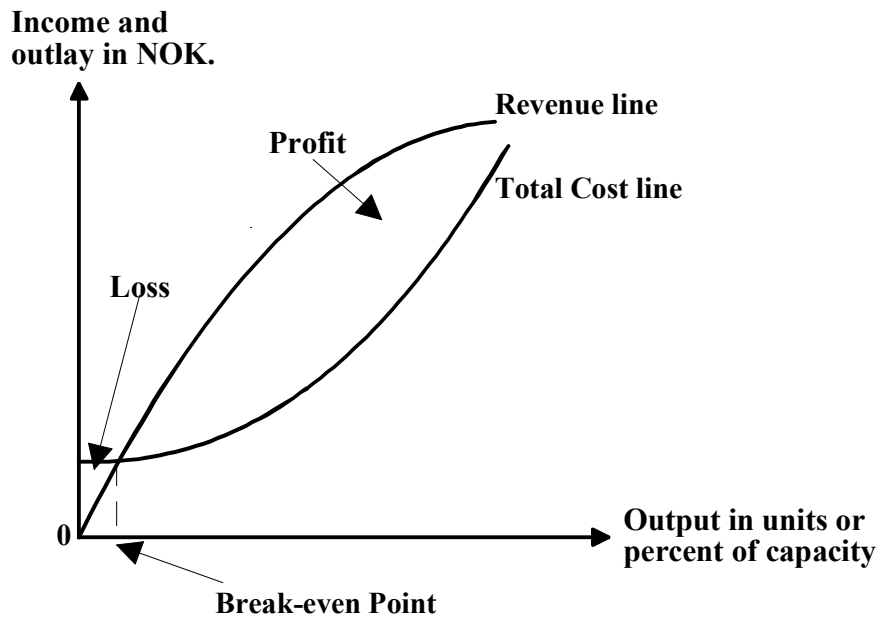


Figure 4.1b Example of non-liner Break-even chart

Figure 4.2 shows examples of multiple break-even points when multiple alternatives are compared. When two alternatives are compared, the break-even points refer to the point of indifference between the two alternatives. In Figure 4.2, x_1^* represents the point where both alternatives A and B are equally desirable, x_2^* represents where A and C are equally desirable, and x_3^* represents where B and C are equally desirable. The figure shows that if we are operating below a production level of x_2 units, then alternative C is the preferred alternative out of the three. If we are operating at a level of more than x_2 units, then alternative A is the best choice.

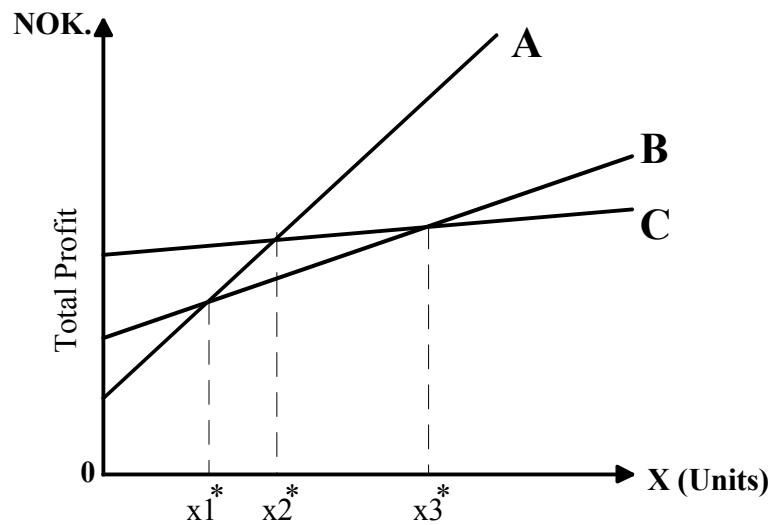


Figure 4.2 Break-even points for multiple alternatives

4.3 SOLUTION BY INTEGRATED INTELLIGENT SYSTEMS

Two alternatives are being considered for production of a new product. The required analysis involves determining which alternative should be selected on the basis of how many units of the product are produced per year. On the basis of past records, there is a known relationship between the number of units produced per year, x , and the net annual profit, $p(x)$, from each alternative. The level of production is expected to be between zero and 250 units per year. The net annual profits (in NOK thousands) are given below for each alternative:

$$\text{Alternative 1 (A): } P(x) = (2.8)x - 205$$

$$\text{Alternative 2 (B): } P(x) = (0.018)x^2 - 310$$

This problem can be solved analytically by finding the intersection points of the profit function and evaluating the respective profits over the given range of product units. The analytic solution is the approach used in the intelligent system solution discussed later in this chapter. Figure 4.3b presents a break-even chart for a graphical solution of the problem.

The Figure 4.3b shows the simultaneous plot of the profit functions for the competing alternatives. The plot shows that Alternative B should be selected if between zero and 186.8 (186 actual units) are to be produced. Alternative A should be selected if between 186 and 250 units are to be produced. Alternative A or B if product units equal to 186.8. The break-even chart can be described by the use of the following production rules:

Rule 1

IF units is greater than or equal to 0
AND units is less than 186.8
THEN selection is Alternative A

Rule 2

IF units is greater than 186.8
AND units is less than 250
THEN selection is Alternative B

Rule 3

IF units is equal to 186.8
THEN selection is Alternative A or Alternative B.

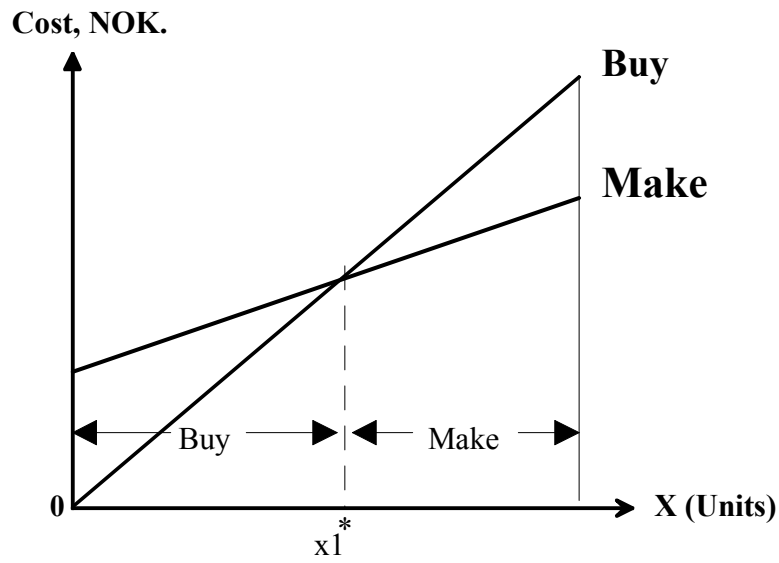


Figure 4.3a. Make-or-buy comparison

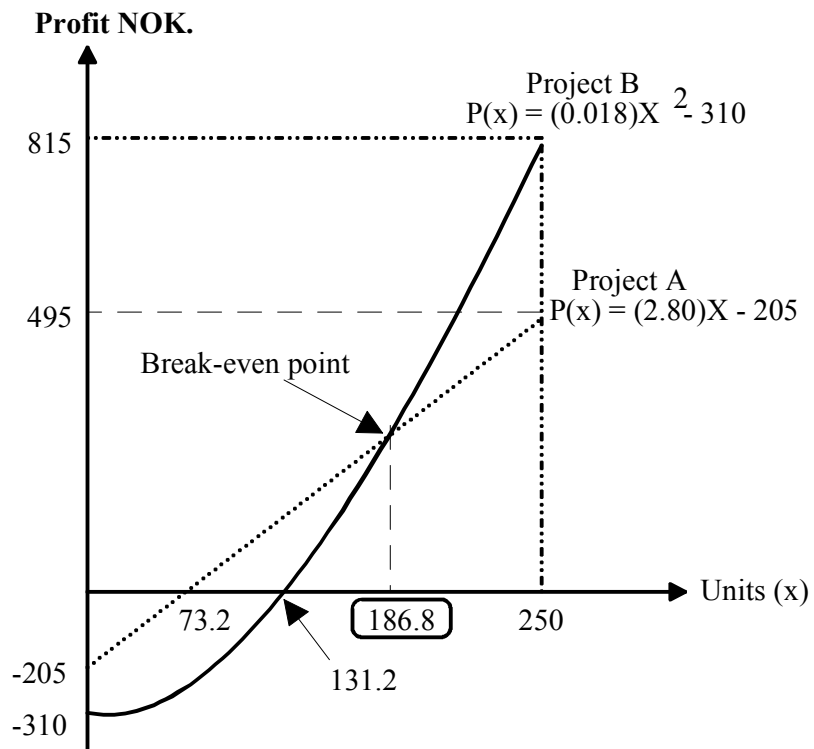


Figure 4.3b. Plot of profit functions for competing alternatives

4.3.1 INTEGRATION ENVIRONMENT

In the beginning of this chapter, the problem involves the selection of a manufacturing alternative based on break-even analysis. Figure 4.4 presents the intelligent system structure for comparing the two manufacturing alternatives. KnowledgePro in the interface engine is used to develop the intelligent system. Detailed information about the alternatives to be compared is stored in the external data base file. The external spreadsheet file stores the data on the cost or profit functions associated with each alternative. The external program (written in C, C++, FORTRAN, PASCAL, or basic) performs the computational analysis needed to obtain the break-even points.

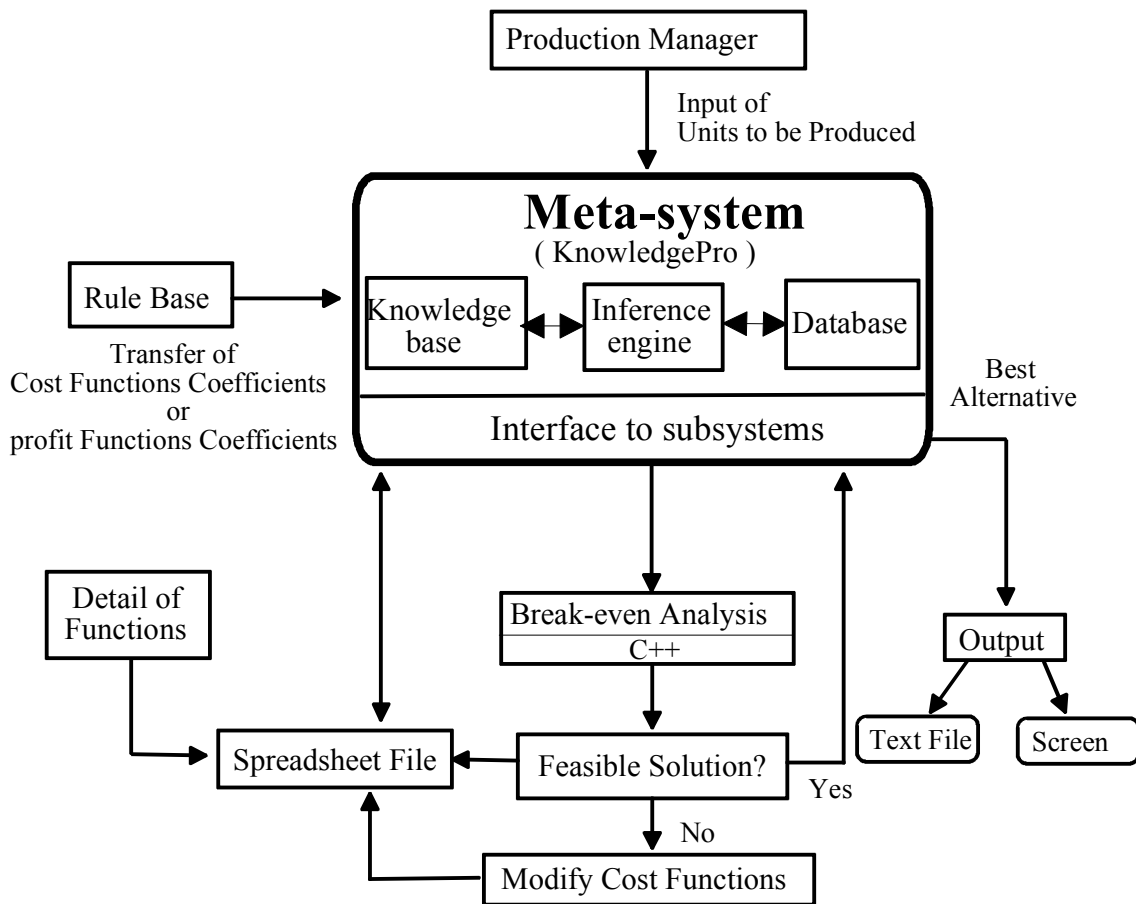


Figure 4.4 The integrating intelligent system structure

If the quadratic equation used to compute the break-even points does not have real roots, the system informs the user to select new function coefficients. The EXCEL spreadsheet file is then updated with the new function coefficients and the consultation is repeated. The user inputs the number of units to be produced. The number of units is used to instantiate parameters in the rule base, where a conclusion is reached on which alternative to select. The consultation result is presented to the user on the

screen. The user has an option of requesting that the result be saved to a text file in order to be printed out later. Figure 4.5a presents the decision tree for the break-even procedure.

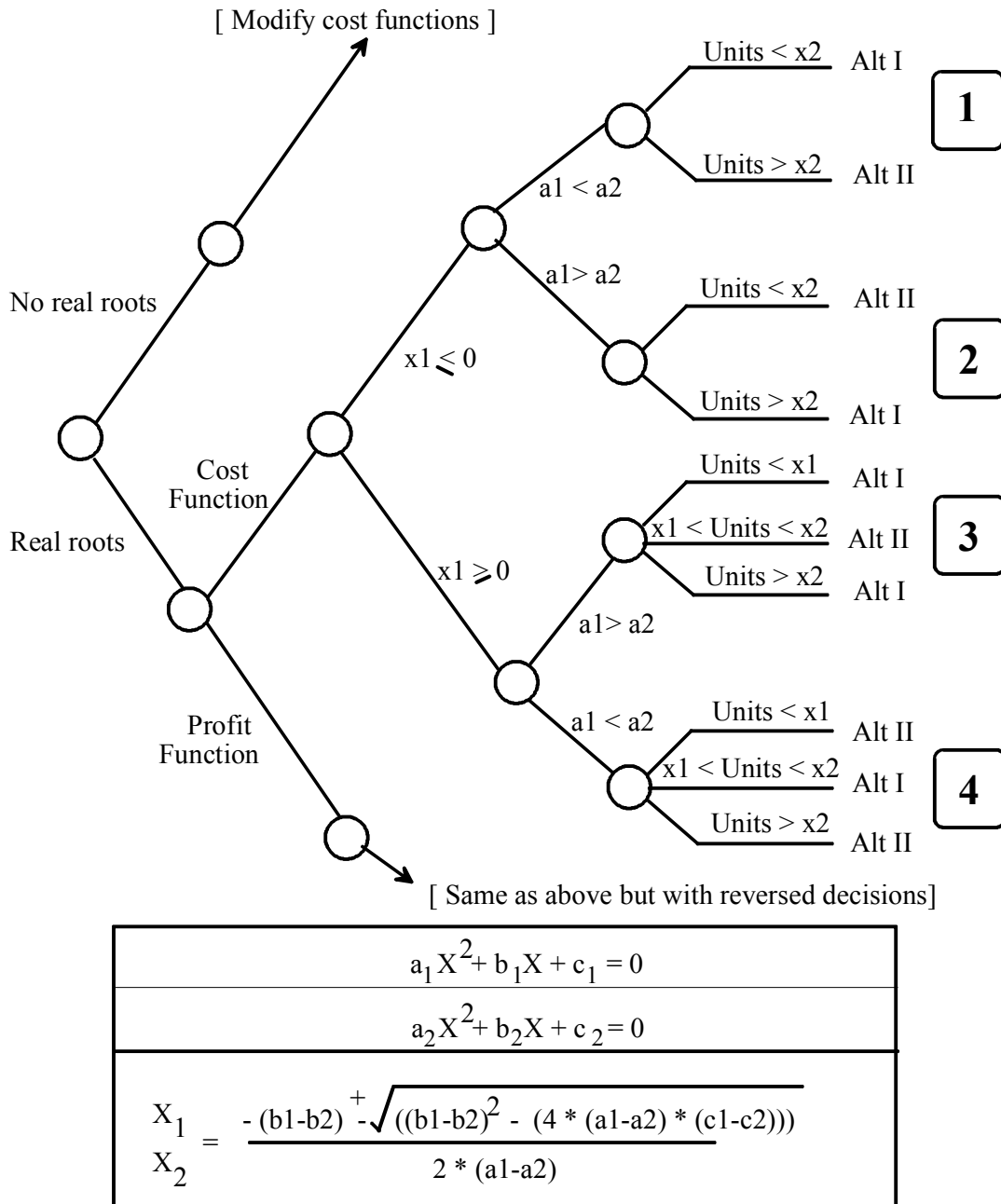


Figure 4.5a Decision tree for break-even analysis

More details of the numbers inside the rounded rectangle are given in Figure 4.5b. The user interface is an important component of the intelligent system. An attractive design of the user interface can be pleasing and appealing to user. A well designed interface encourages the user to explore the capabilities of the intelligent system. The intelligent system above, named Integrated Intelligent Production Management System

IIPMS, makes good use of the multiple window capabilities of KnowledgePro to enhance the user interface.

Examples of screen displays during a IIPMS consultation are presented in Figures 4.6 to 4.11. A list of the complete KnowledgePro program for IIPMS and a C++ program for calculating the break-even points are presented in Appendix 4.

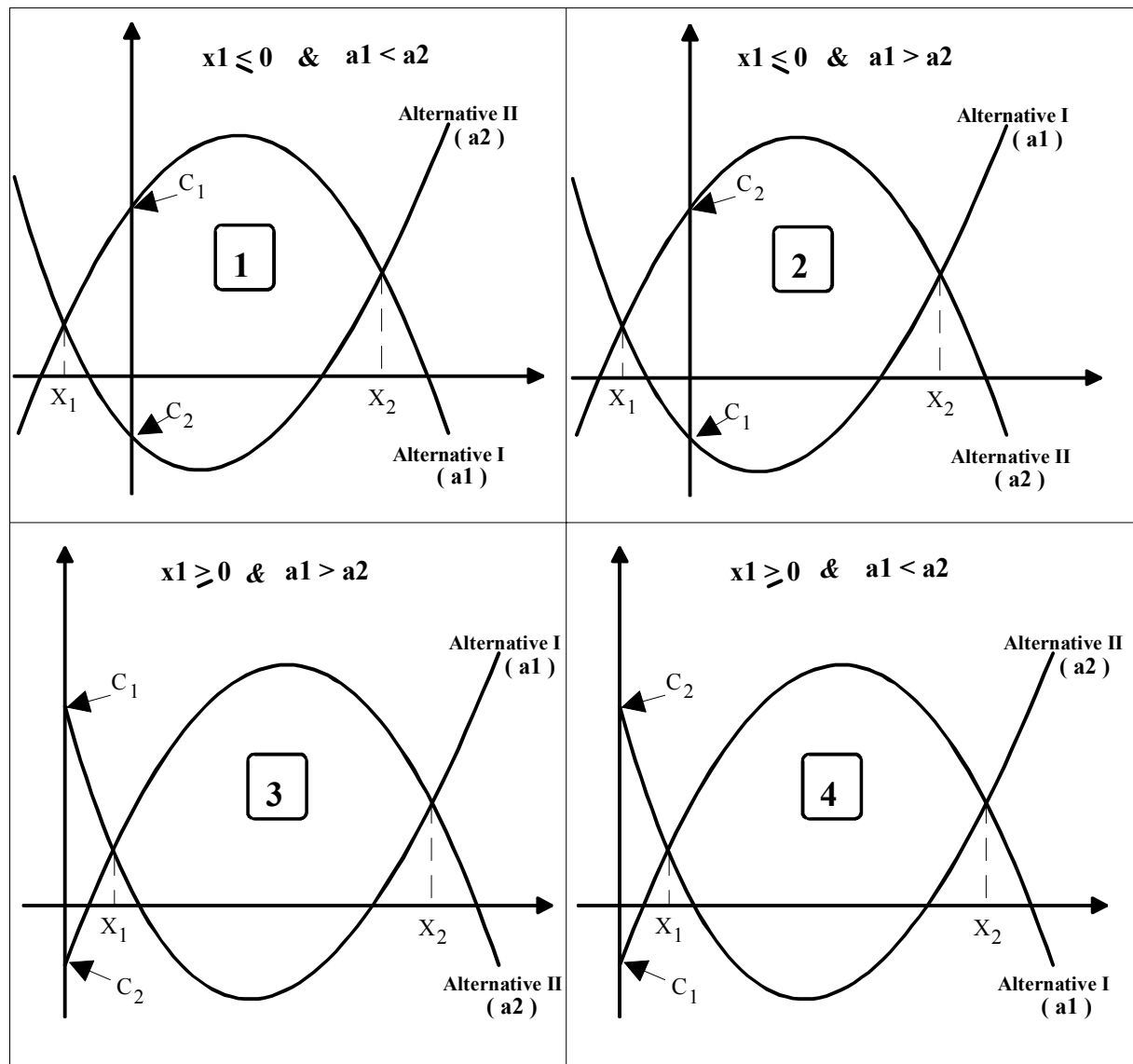


Figure 4.5b Layout of Alternatives I and II.

4.3.2 PROBLEM SOLVING METHOD

In a production operation it is desired to determine the point(s) of equivalence (break-even points) for two competing alternatives, **1** and **2**. The net cost or net profit for each alternative is expressed in terms of quadratic functions of the formula shown below:

$$F_i = aX^2 + bX + c$$

where

i = alternative number

X = number of units to be produced

Rules have to be developed to determine at what values of X each alternative should be selected, depending on the given cost or profit functions. The specifications for the intelligent system design are presented below:

1. Create a spreadsheet file (using Microsoft Excel or any other package) to store the coefficients of the two cost or profit functions as shown in the layout below:

Alternative	a	b	c	Type
1	0	2.80	-205	Cost
2	0.018	0	-310	Cost

The example in the spreadsheet layout above is for the case where the functions given are *profit functions* expressed as:

$$\text{Alternative 1: } C_1 = (2.8)x - 205$$

$$\text{Alternative 2: } C_2 = (0.018)x^2 - 310$$

2. Write a computer program (using BASIC, PASCAL, C , FORTRAN, or any other language) to solve a quadratic function given the appropriate coefficients. The quadratic function to be solved will be the one needed to obtain the break-even point(s) of the competing alternatives.
3. Develop a data base file that contains the description and other details about the two alternatives being compared.

4. Create a KnowledgePro knowledge base that will contain the rules for recommending which alternative to select when a value of X is put in by the user.

The intelligent system will read the function coefficients from the spreadsheet file and determine whether they are cost or profit functions. It will then call the computer program to solve the break-even points. In the break-even solution, the intelligent system will use the knowledge base rules to determine which alternative to select. Before presenting the recommendation to the user, the intelligent system will read all the information about the recommended alternative from the data base file. The recommendation and the associated information are then printed on the screen for the user. At this point, the user will be given the option of storing the consultation result in an external text file. If the equation to solve for the break-even points does not have real roots, the intelligent system will notify the user to modify the coefficients in the spreadsheet file.

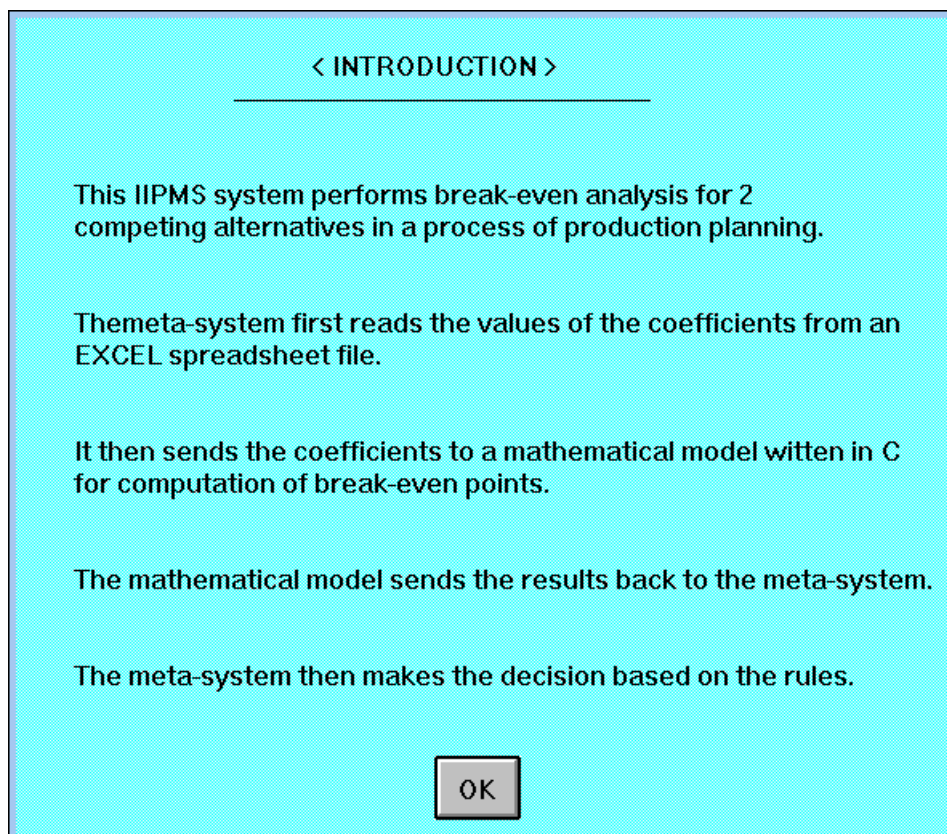


Figure 4.6 Consultation introduction screen

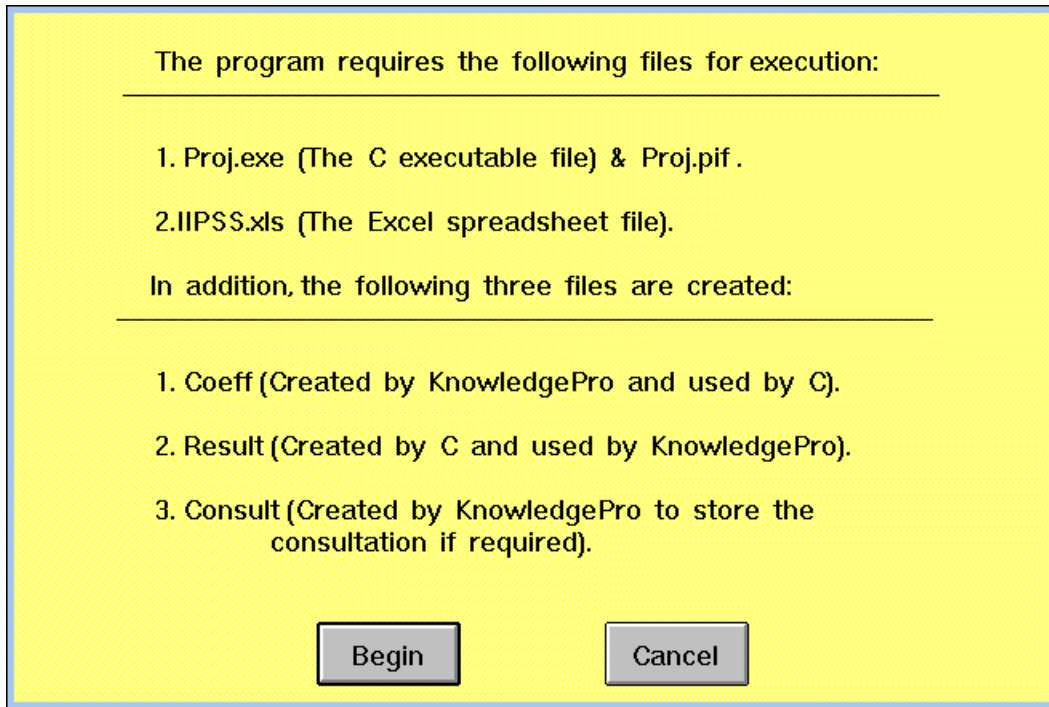


Figure 4.7 Consultation introduction screen

IIPMS.XLS						
	A	B	C	D	E	F
1	A	B	C	FUNCTION_TYPE		
2	0	2.800	-205	cost		
3	0.018	0	-310	cost		
4						

Figure 4.8 Excel spreadsheet screen for cost functions

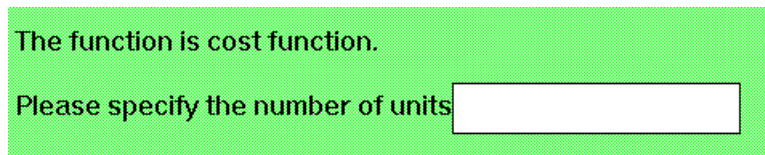


Figure 4.9 Input screen for IIPMS

The function is cost function.

Please specify the number of units

Computing the break-even points, please wait... Done.

The type of the functions is cost functions.

The break-even points are 186.785660 and -31.230095

What we suggest to choose is alternative: 2

The alternation function that you choose is:
 $0.018 * X * X + 0 * X - 310$

Do you want to store the consultation? yes no

Figure 4.10 Result of break-even analysis with 150 units

The function is cost function.

Please specify the number of units

Computing the break-even points, please wait... Done.

The type of the functions is cost functions.

The break-even points are 186.785660 and -31.230095

What we suggest to choose is alternative: 1

The alternation function that you choose is:
 $0 * X * X + 2.800 * X - 205$

Do you want to store the consultation? yes no

Storing your consultation... Done.

Figure 4.11 Result of break-even analysis with 250 units

Figures 4.8 to 4.11 are drawn out from cost function calculation. Figures 4.10 and 4.11 show that the cost function gives alternative 2 for input of 150 units and alternative 1 for input of 250 units.

If the function type is profit, then the answer will be reversed. Figures 4.12 to 4.15 show that the profit function gives alternative 1 for input of 150 units and alternative 2 for input of 250 units. Comparing the answer with the graph in Figure 4.3b proves that IIPMS gives the correct decision. The decision tree in Figure 4.5a helps to make the correct decision rules written in KnowledgePro. A few of rules from the IIPMS program are listed below:

```

topic answer.
  if ?function_type is cost
  then answer is lower.
  if ?function_type is profit
  then answer is higher.
  end. (* end of answer *)

topic the_altern.
  if ?answer is lower and ?poly1>?poly2
  then the_altern is 2.
  if ?answer is lower and ?poly1<=?poly2
  then the_altern is 1.
  if ?answer is higher and ?poly1>=?poly2
  then the_altern is 1.
  if ?answer is higher and ?poly1<?poly2
  then the_altern is 2.
  end. (* end of the_altern *)
    
```

	A	B	C	D	E	F
1	A	B	C	FUNCTION_TYPE		
2		0 2.800	-205	profit		
3	0.018		0	-310	profit	
4						

Figure 4.12 Excel spreadsheet screen for Profit functions

The function is profit function.

Please specify the number of units

Figure 4.13 Input screen for IIPMS

The function is profit function.

Please specify the number of units

Computing the break-even points, please wait... Done.

The type of the functions is profit functions.

The break-even points are 186.785660 and -31.230095

What we suggest to choose is alternative: 1

The alternation function that you choose is:
 $0 *X*X + 2.800 *X - 205$

Do you want to store the consultation? yes no

Figure 4.14 Result of break-even analysis with 150 units

The function is profit function.

Please specify the number of units

Computing the break-even points, please wait... Done.

The type of the functions is profit functions.

The break-even points are 186.785660 and -31.230095

What we suggest to choose is alternative: 2

The alternation function that you choose is:
 $0.018 *X*X + 0 *X - 310$

Do you want to store the consultation? yes no

Storing your consultation... Done.

Figure 4.15 Result of break-even analysis with 250 units

4.4 CONCLUSIONS

Intelligent systems are being used in a number of industrial applications. As the modern manufacturing system, become more complex the use of intelligent systems will continue to grow. In this chapter, the integrated intelligent production selecting system (IIPMS) has been developed for break-even analysis, using a low-cost PC-based integrated intelligent development environment (KnowledgePro) in order to help a manager to make a decision in production processes. The approach focuses on the integration of intelligent systems and several external softwares. The advantages of the integration system are listed below.

- It can cope with a number of complex and ill-structured problems in production and manufacturing engineering;
- It can use the software package and tools which have been well employed before;
- The domain engineers can develop their own system in a faster, cheaper and easier way;
- It is suitable for the small and medium sized companies to employ AI techniques a on PC platform.

Although only a break-even analysis is used as an example in this chapter. The integrated intelligent systems seem to be applicable to a diversity of areas, which is discussed in other chapters, including production and manufacturing engineering. More work is needed to apply the technique to a wider range of applications to see the full potential of the approach. Eventually, research and development in integrated intelligent systems may reveal insight into human decision making and lead to new types of intelligent computing systems.

4.5. REFERENCES

- [1] Bielawski, L. and Lewand, R., Intelligent system design - integrating intelligent systems, hypermedia, and database technologies, John Wiley & Sons, 1991.
- [2] Knowledge garden, KnowledgePro Windows version 2.0 - User manual, Knowledge Garden, Inc. 1991.
- [3] Medsker, L., and Turban, F., Integrating intelligent system and neural computing for decision support, Intelligent system with applications, Vol. 7, No. 4, pp. 553-562, 94.
- [4] Rao, M., Cha, J. and Zhou, J., Integrated software system for intelligent manufacturing, Artificial intelligence applications in manufacturing, Ed. Famili, A., Nau, D. S. and Kim, S. H., AAAI Press/The MIT Press, pp. 385-400, 1992.
- [5] Riggs, J. L., Production system: planning, analysis, and control, Fourth edition, John Wiley & Sons, Inc., Singapore, 1987.
- [6] Wang, K., Jeyakumar, P. and Kjenstad, D. An intelligent system for diagnosing faults in a chocolate factory, Applications and impacts information Processing '94 - Proceedings of the IFIP 13th world computer congress, Hamburg, Vol. II, pp. 86-91, 1994.
- [7] Wang, K., Development of an intelligent diagnosis system using KnowledgePro, International Symposium on Manufacturing Science & Technology for 21st Century, Beijing, China, 1994.
- [8] Wang, K., The selection of the inverse kinematics solutions for robot manipulators using intelligent systems approach, Scandinavian Symposium on robotics, Helsinki-Stockholm, 1994.
- [9] Sigvard C. Lepsøe og John Sjøtrø 'Bedriftsøkonomi', Universitetsforlaget 1993.

5 INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEMS (IIPPS) FOR MACHINING PROCESSES

5.1 INTRODUCTION

This chapter discusses the integration of Computer Aided Design (CAD) and Computer Aided Process Planning (CAPP), which is an important area in the manufacturing environment using the Integrated Intelligent System (IIS). The framework, principle and implementation of IIS for machining process planning are presented. An Integrated Intelligent Process Planning System (IIPPS) has a parallel hierarchical structure in general, and a meta-system as its kernel to manage and control the selection, communication, coordination and operation of the subsystems. IIPPS is implemented to demonstrate the feasibility and capability of Integrated Intelligent Systems.

Process planning is generally defined as the subsystem responsible for converting design data into work instructions. It is a function within a manufacturing facility that establishes the machining process, identifies the machines capable of performing these processes and sets the machining parameters in order to convert raw material to a desired form as specified in an engineering drawing.

Traditional process planning relies heavily on human decisions and manual calculations. Manual planning, depending on the experience of the planner, may be highly labour intensive and error susceptible. By utilising a computer and intelligent programming languages, human errors and working time can be reduced to a great extent.

Integrated Intelligent Process Planning System (IIPPS) is developed for solving a real world problem in the domain of machining process planning, which is a small scale prototype of a large system that would be implemented in a manufacturing organisation. This system drives part geometry and other related information from a computer aided design (CAD) database in the form of a descriptive language, and produces a detailed machining process plan to manufacture parts. The part drawing is first created using a CAD package which stores the geometry in a database as line and arc segments. Then Integrated Intelligent System incorporates manufacturing logic to generate plans of machining processes.

There are two systematic approaches to computer-aided process planning (CAPP) systems:

1. the variant approach
2. the generative approach

The variant approach to process planning is based on the concept of group technology (GT). The parts are first grouped into families and standard plans are stored for each part family. Planning for new parts involves retrieval of existing plans for the part family, and modification of the plan for the part. Compared to manual planning, it increases the information management capabilities moderately. The main disadvantage of the variant process, however, is that it still requires an expert planner to edit and modify the plans for specific components. The computer is just a tool to assist in manual process planning activities. Some examples using this approach are CAPP [1], MIPLAN [2], AUTOCAP [3], and TOJICAP [4].

In the generative approach, the planning logic is built into the system. All process plans for a part are created from scratch. Ideally, all planning functions such as process selection and sequencing, tool and parameter selection, fixturing design, etc. should be done by the system with little or no human assistance. Several generative systems have been developed: APPAS [5], CPPP [6], AUTOPLAN [7], GENPLAN [8], TIPPS [9], PROPLAN [10], and XPLAN [11]. Input for generative systems can come either as a text input where the user answers a number of questions or graphic input where the machinable features are extracted from the geometric data from a CAD database. To have a fully integrated system, the latter is preferred. Different methods have been developed for feature extraction: syntactic pattern identification [12, 13], geometry decomposition [14], expert system and logic [15], and a graph-based approach [16, 17]. Although these methods have had some success in extracting machinable features from a 3D geometry, they can only handle a limited set of geometric variations and are computationally expensive in some cases.

Most of the current generative process planning systems are not truly automated planning systems. They still require some form of human input and decision making in interfacing with CAD. The interacting between CAPP and CAM is also underdeveloped and rarely reported. The need is thus to develop a generative process planning that can interface with a CAD system through automated feature recognition and provide its output to a CAM module for automated part programming.

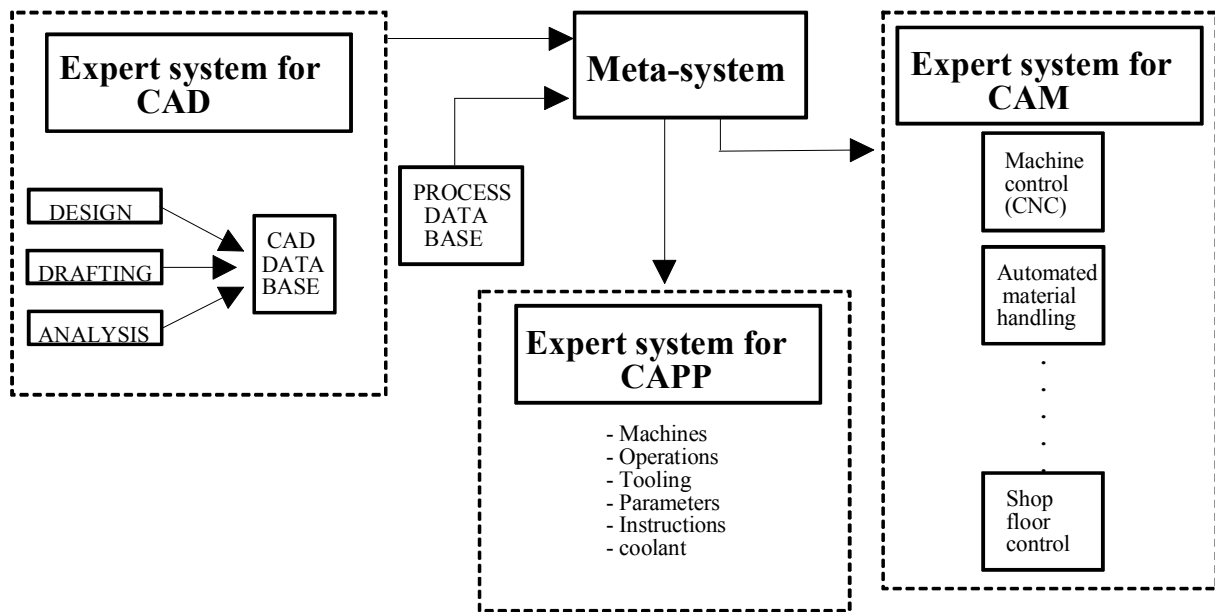


Figure 5.1 The Integration of CAD, CAM and CAPP

This chapter discusses an approach that has attempted to overcome some of these shortcomings. This approach has tried to develop a system for integrating a CAD system with manufacturing decision logic, so that process plans can be generated with little or no manual intervention. A significant departure from earlier approaches has been:

1. The use of CAD database as the primary input source for part features
2. The use of a database system to translate the geometrical representation to symbolic representation
3. The use of a meta-system to control the information flows among systems
4. The application Artificial Intelligent (AI) in developing an expert system program incorporating manufacturing logic to generate process plans (Figure 5.1).

The following sections of the chapter will present the overall research approach and describe how to devel IIPPS. The integrated intelligent system was developed for process planning to rotational machine parts.

5.2 INTEGRATED INTELLIGENT SYSTEM APPROACH TO PROCESS PLANNING

5.2.1 THE PROCESS PLANNING FUNCTION

A typical process planner is a well experienced machinist or production engineer and who is familiar with the production process and the capabilities of various machines on the factory floor. He has a vast resource of accumulated knowledge which allows him to solve the task of process planning by semi-logical methods such as recognising a familiar pattern and recalling the appropriate machining process.

Process planning is a very complex task requiring a good deal of professional expertise. Manual process planning is a timeconsuming task and there is no guarantee of efficiency in the planning of machining sequences. The initial plan may be revised several times in an attempt to achieve feasibility. Digital computers have been used to automate some subtasks of process planning. In developing a process plan, the planner can retrieve process plans for a similar part and modify them to suit the new part. But, this entails searching a large number of files to account for every feature of the part. Moreover, if the new part has distinctly different features, manual process planning is necessitated.

5.2.2 THE INTEGRATED INTELLIGENT SYSTEM APPROACH

The goal of this research is to design and implement an Integrated Intelligent System (IIS) that can:

- (1) Analyse the geometry of a part based on a symbolic representation,
- (2) Produce a logical sequence of operations to manufacture the part,
- (3) Recommend appropriate machines and machining parameters.

The Integrated Intelligent Process Planning System called IIPPS has been developed to realise the above goal. The system has been tested on a spectrum of symmetric rotational components.

Process planning is a very complex task. The main difficulty in process planning is the number of specialised information that must be processed within a limited time frame. The process planner must sort out numerous possibilities and develop a feasible plan. Apart from that, the process planner has to handle a large amount of information on the subject of process planning.

The most efficient process planners are those engineers who have spent several years working for a particular organisation and have seen the pattern of parts that can be produced in that environment. A novice to this environment learns the art of process planning by observing these experts at work. When these experts are ready to leave the organisation, the novices retain a few capsules of their predecessor's knowledge. Integrated Intelligent Systems have found a way to preserve the knowledge of human experts and in recent years have surpassed human performance in many real-world applications.

Integrated Intelligence Systems have significant advantages over conventional structured computer programs in applications such as process planning. One major advantage is that Integrated Intelligent Systems offer a modular architecture for building large programs. Knowledge in the form production rules may be added, deleted or modified in the knowledge base, without any change in the control structure. Integrated Intelligent Systems can acquire knowledge interactively through dialogue with a user. Integrated Intelligent Systems have the ability to explain the line of reasoning used in any particular situation. The sequence of rule invocation when expressed in natural language constitutes an explanation for program behaviour. Finally, an Integrated Intelligent System's ability to perform symbol manipulation permits users to choose a variety of representation schemes for components, and also permits an easier translation of CAD data into the selected representation.

5.3 STRUCTURE OF IPPS

An Integrated Intelligent System for solving a real world problem in any domain would take at least a few years to develop. IPPS, which was developed for this research, is thus a small scale prototype of a larger system that would be implemented in a manufacturing organisation. A well-bounded problem was chosen and solved within a limited time frame. The current working domain of IPPS is the category of symmetric rotational parts that are produced primarily on turning machines such as lathe and screw machines. However, a comprehensive list of operations that are performed on these machines was considered in the development of IPPS.

The whole structure of IPPS is shown in Figure 5.2. The major components are as follows:

1. Graphics subsystem, which uses AutoCAD to input data and extract geometric features.
2. Data subsystem, which uses dBASE III to organise data and extraction.
3. Meta-system, which uses KnowledgePro to control the information flows among the subsystems.
4. Expert subsystem, which uses KnowledgePro to make machining decisions.

The knowledge base consists of production rules on various aspects such as selection of machine tools, selection of machining parameters, etc. These rules are of the form:

IF (set of conditions)
THEN (set of actions to be taken)

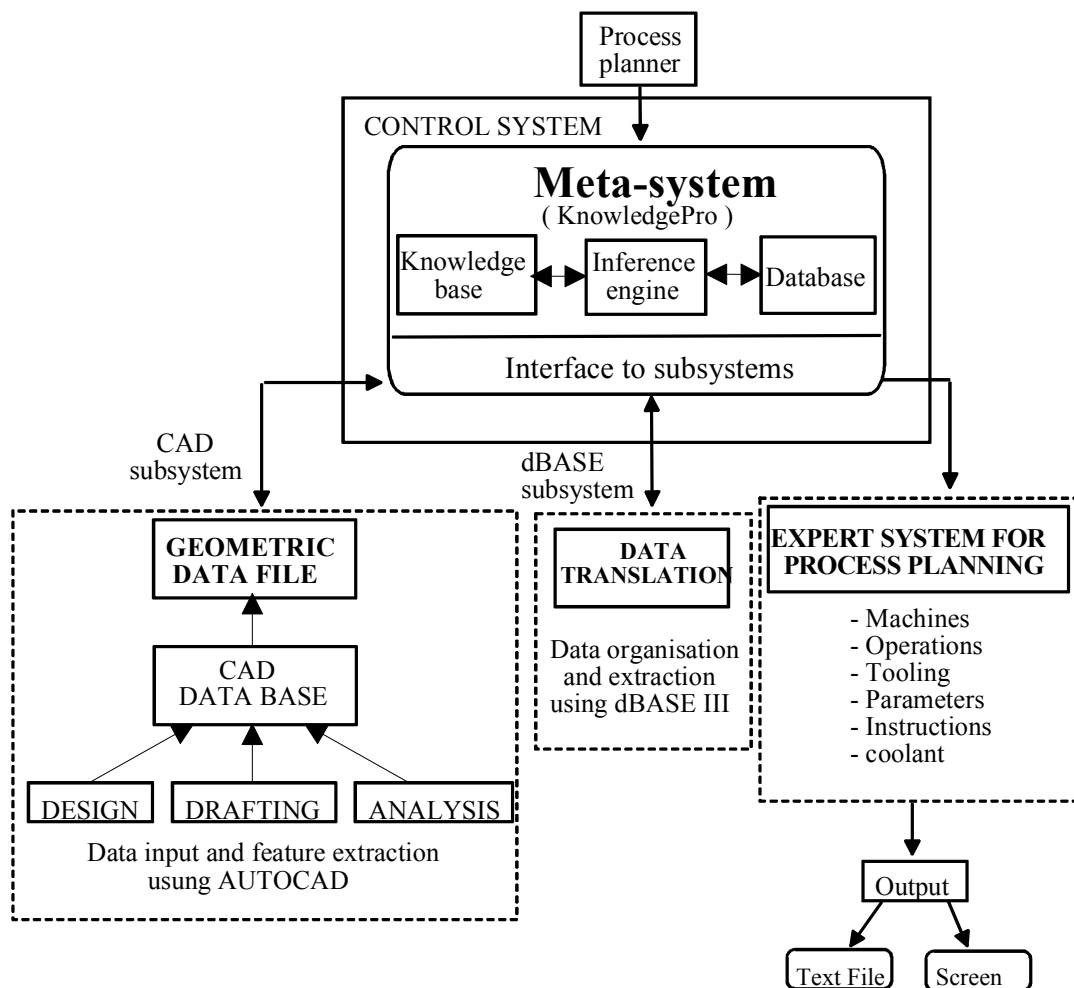


Figure 5.2 IPPS structure

The rules are stored internally in the language of choice and the equivalent English version is generated to make it readable by the user. The rule may be considered as a piece of domain specific information. The actions recommended by the rule are justified if the set of preconditions is fulfilled.

The control structure interacts with all other parts of the Integrated Intelligent System in deriving a process plan. Major tasks accomplished by the control structure are:

1. Using embedded problem solving knowledge to process data.
2. Establishing an interactive environment and a user-friendly interface.
3. Offer explanation to user regarding the system's line of reasoning.
4. Acquisition of knowledge in the form of production rules.

Representation schemes have been worked out for workpieces, machines, tooling etc. in order to provide the Integrated Intelligent System with all the details necessary to drive a process plan. These schemes will be described in detail in the next section.

The workspace stores partial results obtained in planning and this information will be used to decide on optimal or near-optimal machining sequences. The raw material undergoes several state transitions as it is machined to obtain a final form.

5.4 GEOMETRIC FEATURE EXTRACTION

The main advantage of the generative approach to process planning is that little or no manual intervention is required for preparation of a process plan. Many generative process planning systems have been developed, but very few of them can interface with CAD systems. In the usual approach, the user has to enter the part coordinate data and the other necessary data by answering many questions, which takes more time. To overcome this dilemma, IPPS was designed with the aim of developing a CAD interface with the meta-system. The first step was to find a CAD software that is used extensively in industry. Since the expert system shell to be used was PC-based, it was necessary that the CAD software also be PC-based. Even though there are many microcomputer CAD systems available, AutoCAD is by far the most prominent. One of the major reasons for AutoCAD's position as the standard for PC-based CAD software is its adaptability. It also offers a high degree of flexibility and customisation. It allows a user to tailor the software to his or her programming needs, and it can be modified to meet the needs of drafters and designers in many design situations.

Customisation of AutoCAD in IPPS was necessary because the input module for producing part drawings are designed for inexperienced users of AutoCAD.

Customised AutoCAD benefits the user by presenting a more compact and familiar interface that reduces learning time. Most of the time spent producing and editing drawings in AutoCAD is spent in selecting and/or typing commands in a precise order and then selecting the correct options within these commands. Customisation allows a user to combine a long series of keystrokes into just a few. With customisation, questions are posed to the user while the part drawing is made through use of the custom menu provided. A sample of the menu is shown in Figure 5.3. The descriptions of the items on the menu are presented below:

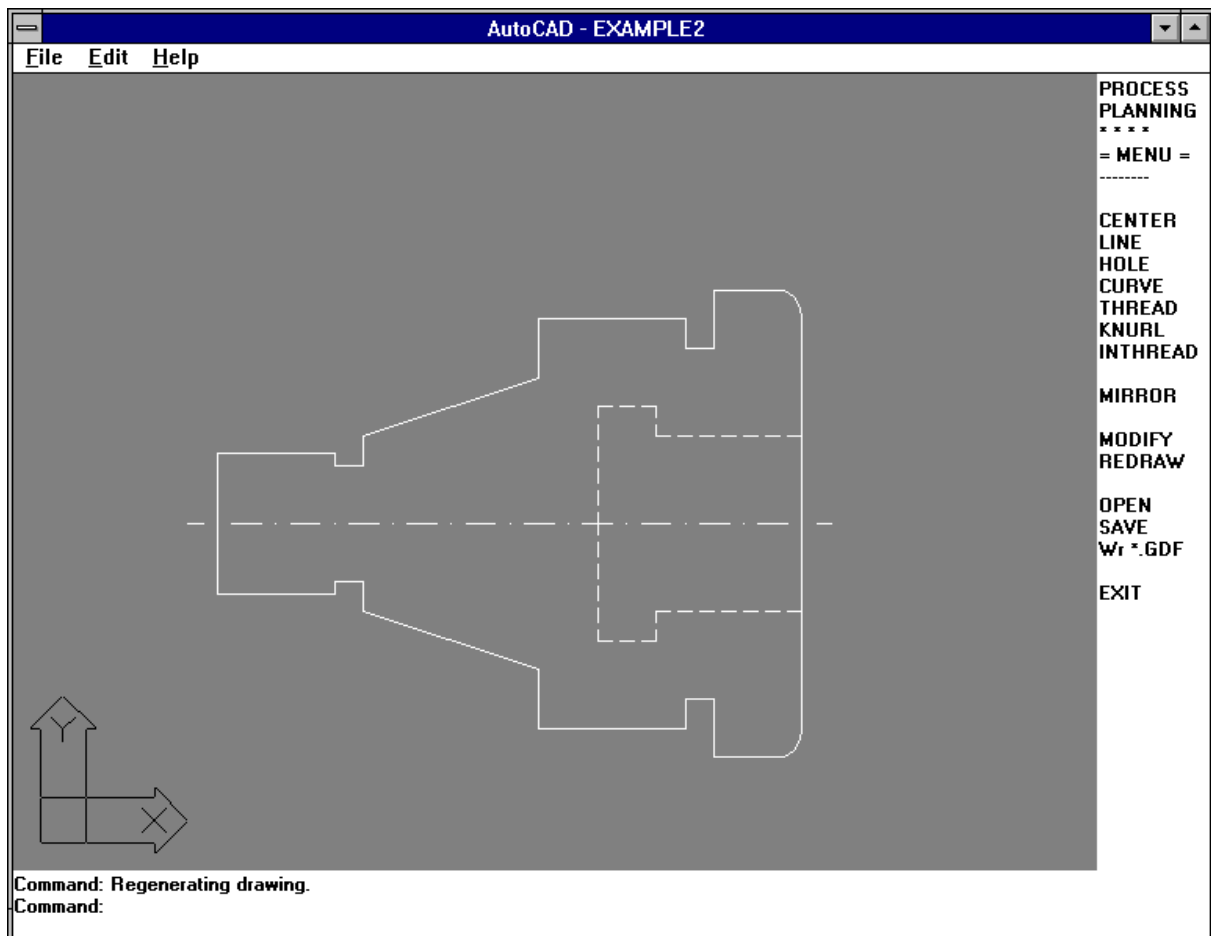


Figure 5.3 IIPPS screen menu

- CENTER:** This is used to draw center line. It must be the first step in producing new part drawings.
- LINE:** This is used to draw external straight surfaces on a part.
- CURVE:** This is used to draw round fillets.
- HOLE:** This is used to draw internal machining surfaces.
- THREAD:** This is used to draw external threaded surfaces.
- KNURL:** This is used to draw knurling surfaces.
- INTTHREAD:** This is used to draw internal threaded surfaces.
- MIRROR:** This is used to generate mirror images around center line.

- MODIFY:** This is used to modify an object. It erases the objects first, user then must enter a new object to replace the erased object immediately.
- REDRAW:** This is used to redraw the part drawings on the screen.
- OPEN:** This is used to open and read a part drawing.
- SAVE:** This is used to save a part drawing.
- WR *.GDF :** This is used to write the geometric data of a part drawing. The file[*.GDF] is the output of AutoCAD module.
- EXIT:** This terminates the AutoCAD module.

The code for the customised menu is stored in a file with the extension MNU. This file extension identifies an AutoCAD screen menu file, which contains screen-menu prompts as well as the commands that are executed when options are selected for the printing device. Each line in a custom menu contains a specific command sequence. Each sequence begins at the leftmost position and can continue to the right indefinitely using a wraparound technique. A drawing database is maintained simultaneously for each drawing. The only way to access this data is by using AutoLISP. AutoLISP is an implementation of the LISP programming language embedded within AutoCAD. AutoLISP allows AutoCAD developers to write programs and functions in a powerful high level language that is well suited to graphics applications. It adheres closely to the syntax and conventions of common LISP. Even though it is a small subset of common LISP, it has many additional functions specific to AutoCAD. AutoCAD instructions that have been written in AutoCAD are called LISP routines. LISP routines are contained in ASCII files that are called LISP files. LISP files can have any valid DOS file name and they always have the file extension LSP.

In AutoCAD, each drawing element is considered as an entity. For example, a line is an entity. For each entity, there is an association list which contains dotted pairs of information about that entity. For example, an association list for a line drawn from (2.00, 4.00) to (6.00, 9.5) is as shown below:

```
((-1. <Entity name: 60000014>)
(0. "LINE")           ; Entity type
(8. "0")              ; Layer
(10. 2.00 4.00)       ; Start point
(11. 6.00 9.5)        ; End point
)
```

The first number in the dotted pair is the code that represents a particular item. For example, -1 represents the entity name, 10 represents the starting point of the line. A comprehensive set of AutoLISP functions provides access to AutoCAD entities and to the graphics screen and input devices. A user can select entities, retrieve their values, and modify them. But no functions are provided for creating entities. An entity name is like a pointer into a file maintained by AutoCAD's drawing editor, from which

AutoLISP can find the entity's data base record and its vectors (if on screen). This feature, by which the user can modify the entity data base, is used by IPPS for extracting the additional parameters of an entity such as tolerance, surface finish, threads per inch, and so on. When the user is drawing an entity on the screen using the menu provided, interactive questions are asked concerning relevant parameter values. There are some values in an association list which are not useful for the feature extraction. These values are searched in the entity data base and replaced by the parameter values entered by the user. A sample of this procedure is presented below.

In the association list for a threading surface, a dotted pair (38. 9) represents the layer name of an entity which is not useful since IPPS makes use of only one layer. So, in this case, the value 9 is replaced by a threads-per-inch value entered by the user. After this procedure is completed, the list associated with the threading surface contains all the information needed to select the machining parameters for that surface. After the part drawing is complete, the next job is to extract the required information from the entity data base. There are functions in AutoLISP that can be used to access a particular piece of data. Through use of these functions, the required data is extracted. For example, the individual dotted pairs that represent the values can be retrieved by using the **assoc** command. To retrieve the values, the **cdr** command is used. To explain **cdr** command the following piece of program is taken from IPPS.

```
(cdr(assoc 10 1))
```

where,

10 is the code which is a pointer for the parameter value
1 is the name of the list in which the entity data is stored
assoc locates the dotted pair which contains the value 10
cdr retrieves the second value in the selected dotted pair

All these commands are embedded in the AutoCAD program which acts as a feature extractor of the input part drawing. The program is written in such a way that the output is in a format that can be interfaced with dBASE III. The output of the LSP file is an ASCII file.

5.5 PROFILE DESCRIPTION

After the data about the part drawing is extracted from AutoCAD, the data obtained should be represented in a way that is suitable for processing by use of the available problem-solving knowledge. This is accomplished by a control structure written in dBASE III. The output ASCII file from AutoCAD is retrieved in dBASE III.

There are many reasons why dBASE III was chosen for developing the control structure. As discussed, the output of the AutoCAD is an ASCII file which contains the numbers and characters associated with the part drawing. The control structure should be a tool which can take this data as input, convert it into a format that can be used by the knowledge base and be interfaced with the chosen intelligent system. Also, it should provide the necessary programming environment. All of this can be accomplished with dBASE III.

Schemes of data representations are worked out for the part drawing entities. The method used in this research is particularly suited to represent symmetric and rotational parts. These parts are assumed to have two profiles, one external and one internal. Each profile is completely described using primitive segments 'LINE' and 'ARC'. A 'LINE' is a straight edge on a part and is specified using the coordinates of the endpoints of the line. Additional parameters are included in the description of a 'LINE' to account for surface features, such as a threaded surface. An 'ARC' is a curved surface on a part, such as a groove or a fillet. It is specified by the coordinates of the endpoints of the arc, the coordinates of the center of the arc and the angle swept by the arc. Additional parameters may be added depending on the surface feature of the arc. See Figure 5.4.

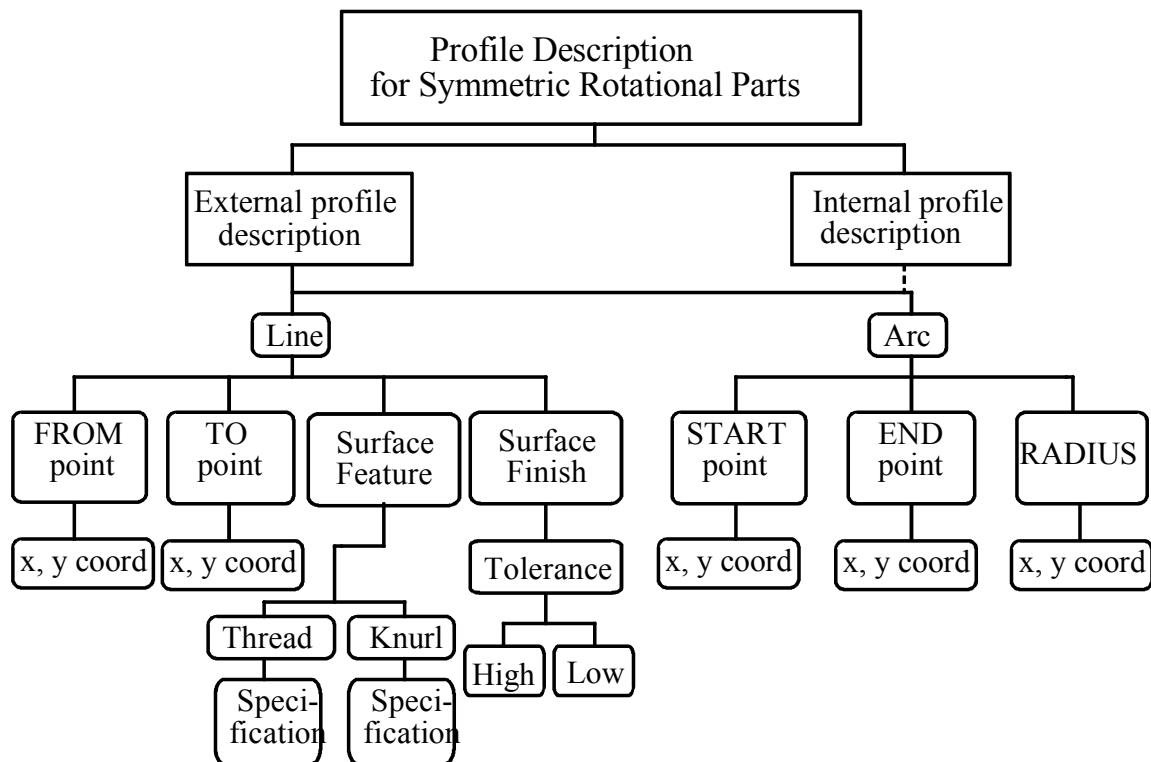


Figure 5.4 Profile description scheme for rotational parts

The entity data is stored in the output ASCII file in an order which the user has drawn on the part drawing. Suppose a user draws an entity A followed by entities B, C, and D. If he later needs to make a change in entity A, he/she would need to replace A with a modified entity A'. In this case, the order in which the output data is stored should be A'BCD, but it is actually stored as BCDA'. To find the machining operation sequence the data should be rearranged in the order A'BCD, which is the first step performed by the dBASE III program.

Experienced operators play an important role in finding the operation sequence for parts. Operations associated with process planning are the various machining operations that change the shape of an object from a raw material form to a final desired form. Every machining operation has a unique way of changing the state of material and each one is called for, depending on the situation. Several factors may be evaluated as preconditions, such as the surface to be modified, the existence of certain features in raw materials, and so on. Not all operations are capable of generating desired shapes. For example, a turning operation is applicable only if the raw material is cylindrical and its dimensions are within the specified limits of the machine. A boring operation can be performed only if the part is cylindrical. For these reasons, it is very important to know the precedence of the required operations. The precedence logic used in IIPPS is based on the developers' own experience as well as published guides.

5.6 THE MODELING OF PROCESS

Operators in a production system change the state of the system. The operators associated with process planning are the various machining operations that change the shape of material from raw form to a final desired shape. Every machining operation has a unique way of changing the state of material and each one is called for depending on the situation. Several factors may be evaluated as pre-conditions such as, the surface to be modified, the existence of certain features in raw material etc.

Not all operators apply at all times. For example, a turning operation is applicable only if the raw material is cylindrical and its dimensions are within specified limits of a machine. A boring operation can be performed only if there exists a pre-drilled hole. Threading on a surface can be done only if the part is cylindrical and all burrs have been removed by turning. It is worthwhile to note that precedence of operators is one of the main considerations in planning. Nonetheless, at some states more than one operator may be applicable. Figure 5.5 shows a subset of operators in the real world and how they are applied to change surfaces.

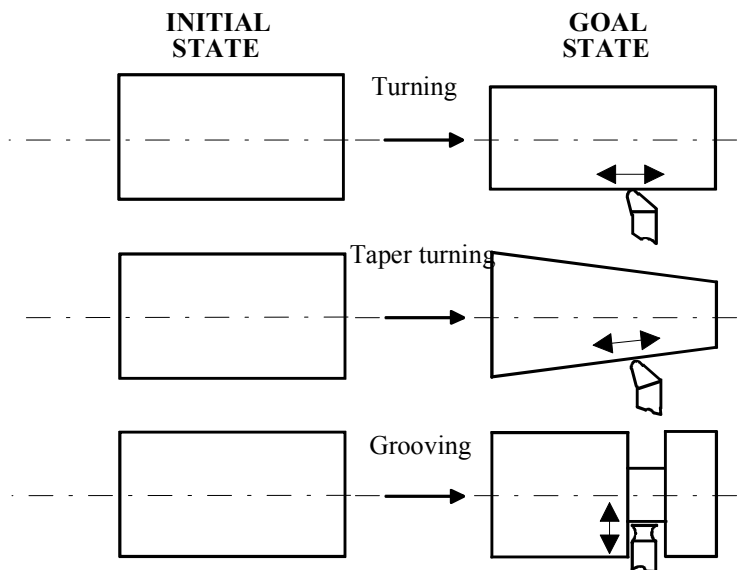


Figure 5.5 Real world operator

The machining operations are described as follows:

Drilling	:-	create hole
Boring	:-	enlarge hole
Turning	:-	decrease diameter
Filleting	:-	create a curved surface
Taper turning	:-	chop an angle

5.7 DECISION TREES

An interface tree for process planning symmetric rotational parts is shown in Figure 5.6. The external and internal profiles of a part are composed of lines and arcs. If the segment is LINE, three possible situations occur. The Line may be vertical, Horizontal or slant. A vertical line is designated as vert+ if it is traced away from the center line of the workpiece and vert- otherwise. Similarly we have horiz+, horiz-, slant+, and slant-. This classification helps to identify hidden edges in the outer or inner profile. For example, if the system encounters the sequence horiz+, vert-, and horiz- it expects a groove at that junction. Above each of these segments is stored a distinct shape. Above a horiz+ is a rectangle whose height is the difference between the raw material diameter and the y coordinate of the part profile under it. Nothing is stored above vertical segments, and above slant segments we have triangles. An ARC may be classified as convex+ or convex- and the corresponding shape is a sector. This tree is used to select the machining operation required and is updated after each operation is executed. The other parameters such as machines, tools, tool materials, coolant, speed, feed and depth-of-cut are chosen based on rules from the knowledge base.

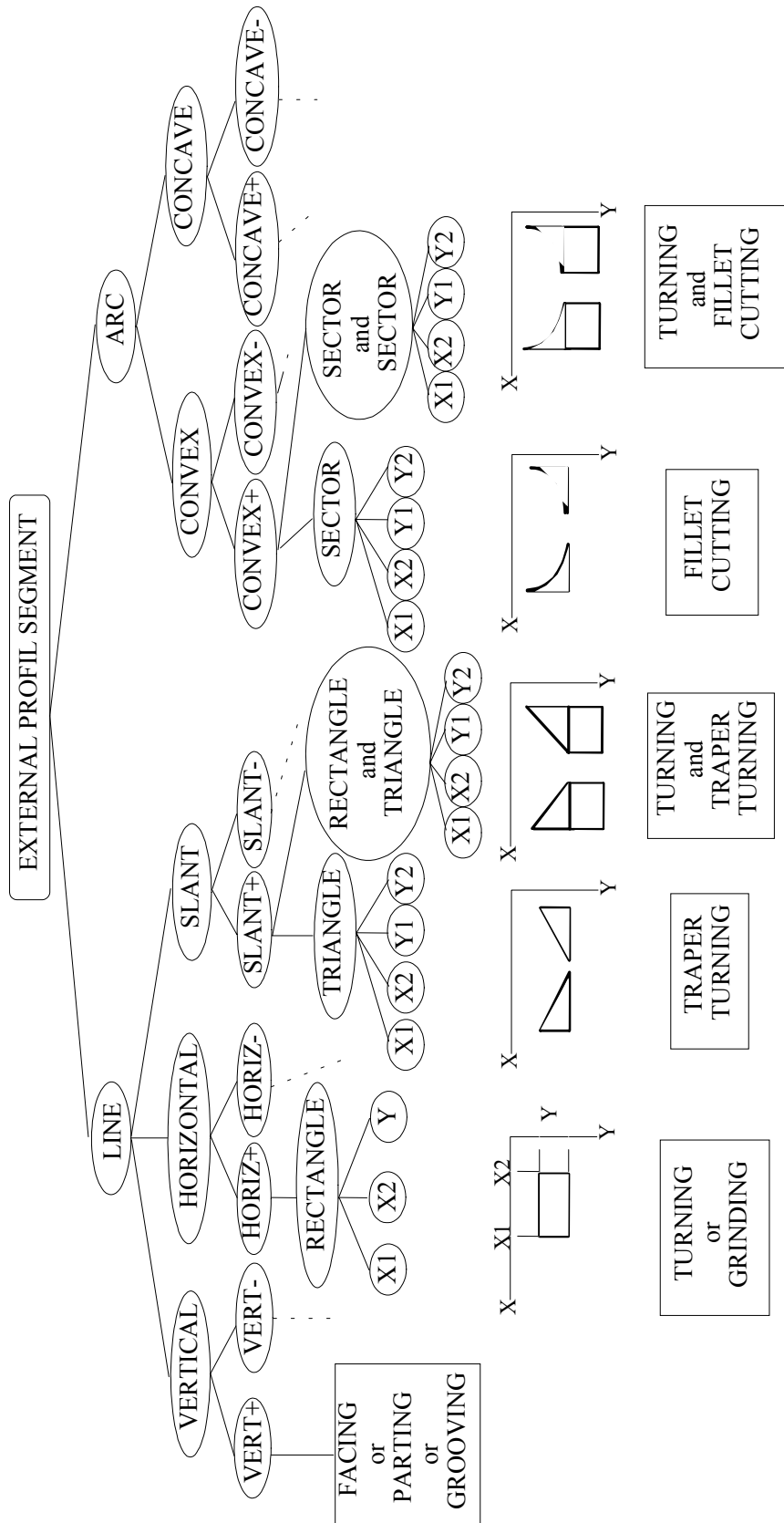


Figure 5.6 The decision tree for external profile

5.8 PROBLEM SOLVING STRATEGY

The process planner's job involves interpreting engineering drawing, reasoning about a component's geometry and quality, and finally making decisions about how the part can be best produced. This can be done by identifying the cutting precedences and by selecting processes, parameters, and tools to be used. If a computer-based system is to perform the task of process planning at a level comparable to humans, we need to supplement its numerical capabilities by providing logic that translates the piece-by-piece description into higher order relationships. This can be done by defining the interrelationships between the states of the production system.

In the approach used in IPPS, the part drawing is considered as a two-dimensional graph, which shows the area to be removed to get the final form of the part from the raw material bar stock. The area above a horizontal line is considered as a rectangle and the operation which removes the rectangular area is turning. Similarly, taper-turning is considered to remove a triangular area, which is the area above a slanted line. For finding out the number of passes in which the required material can be removed, it is necessary to find the depth of the area to be removed. Also for turning, drilling, and boring, the length of the cut is maximised. It is necessary to keep a record of the current (x,y) coordinates and the final coordinates. To achieve this, a special procedure was developed. For each edge of a surface, records of x and y coordinates, (x_{final}, y_{final}) and $(x_{current}, y_{current})$ are maintained. After each operation or cut, the $(x_{current}, y_{current})$ are updated. An example is shown in Figure 5.7

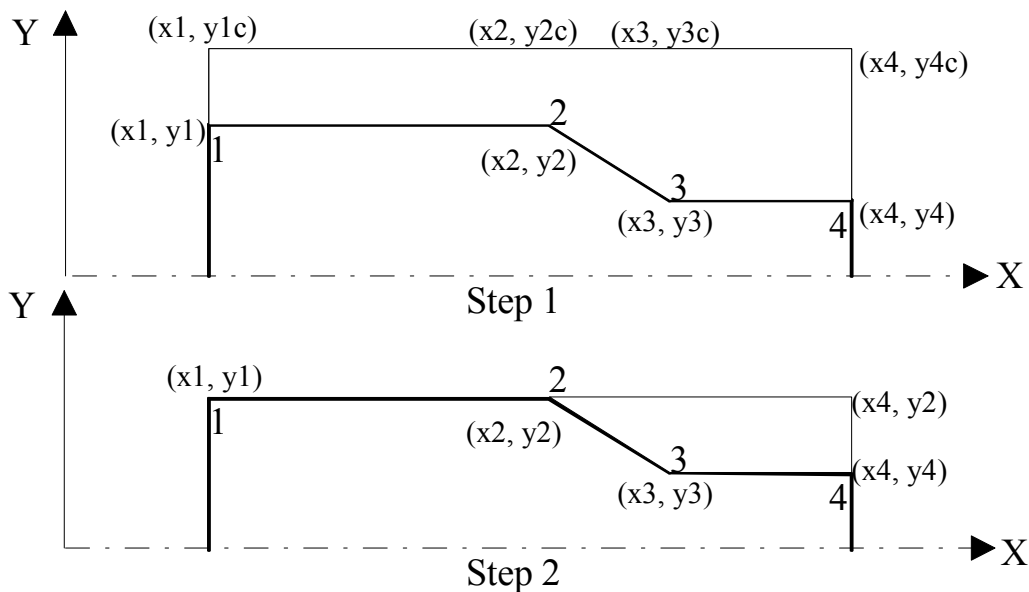


Figure 5.7 Path logic in IPPS

The figure shows two horizontal surfaces but at different levels. Before the machining is started (step1), the final coordinates of edges 1,2,3,4 are $(x1, y1)$, $(x2, y2)$, $(x3, y3)$ and $(x4, y4)$, respectively. Similarly, the current coordinates for the edges 1,2,3,4 are $(x1, y1c)$, $(x2, y2c)$, $(x3, y3c)$ and $(x4, y4c)$, respectively. Starting from the left at $(x1, y1c)$, in order to maximise the cut, the first cut is made to $x4$ instead of $x2$ or $x3$. The depth of the cut is $y1c-y1$. Because of this, the current coordinates of edges 1,2,3,4 change to $(x1, y1)$, $(x2, y2)$, $(x3, y1)$ and $(x4, y1)$, respectively. It is seen that the current coordinates for edges 1 and 2 are equal to the final coordinates (step2). When this condition is satisfied, the machining of that portion of the surface is considered to be complete. To start the next machining operation, the current coordinates of the edges are updated and the process logic is repeated. Note that numerically, $y1c=y2c=y3c=y4c$, $y1=y2$ and $y3=y4$. The logic used for determining the operation sequence is given below.

1. Find the maximum diameter of the part and determine the raw material diameter. The raw material diameter Y_{max} stored in temp&.dbf file.
2. Find the length of the part (X_{max}) and suggest the required raw material length so that the part can held in chuck while machining without affecting the actual part surface.

The temp&.dbf is listed bellow.

X1	Y1
11	8.5

This file can be read by KnowledgePro and then linked to the procedure of process planning.

3. Locate the center line and find the intersection point of the center line and the first external surface. That will be the starting point or reference point of machining.
4. If the surface is horizontal and the operation is turning, then try to maximise the length of the material that can be removed in one pass.
5. Threading, knurling, and so on, are differentiated by the entity colours in IIPPS. The colours are used for determining the operations.

In IIPPS the following colours are used to determine the operations:

COLOUR	OPERATION	ENTITY NUMBER
Red	Thread	1
Green	Knurl	3
Magenta	Inthread	6
Cyan	Hole	4

6. Before threading or knurling the part must be turned to a specific diameter.
7. If the line is vertical, then there is a step involved.
8. If the surface sequence is vertical downwards, horizontal and vertical upward, then the operation can be turning, forming, or grooving depending on the width of the step.
9. After completing the external operations, start internal operations.
10. If the line is horizontal and dotted, then use drilling and try to maximise the length of the drilling cut.
11. At the end of the process, perform any required parting operation at the required length.

Operation order is determined by dBASE III and stored in the file named temp%.dbf. The file temp&.dbf is listed below:

OPER1	OPER2	FROMX1	FROMY1	FROMX2	FROMY2	DEPTH	TOMIN	TOLPLUS
3	0	1	1,2	1	0	4,25	0	0
1	0	1	4	11	4	0,25	0	0
8	0	3	1	3,5	1	3	0,01	0
1	7	1	1,2	3	1,2	2,8	0,04	0
7	0	1	1,2	3	1,2	0	0,04	40
2	0	3,5	1,5	6,5	2,5	2,5	0,02	0
8	0	9	3	9,5	3	1	0,02	0
1	5	6,5	3,5	9	3,5	0,5	0,02	0
5	0	6,5	3,5	9	3,5	0	0,02	25
4	0	11	3,5	11	0	4	0,005	25
12	0	11	3,5	10,5	4	0,5	0,005	0
10	0	7,5	0,4	0	0,4	-7,5	0,02	0
11	0	7,5	2	8,5	2	1,6	0,02	0
11	6	8,5	1,5	11	1,5	1,1	0,004	0
6	0	8,5	1,5	11	1,5	0	0,004	30

After the part information is arranged in the required format and the sequence of operations is determined, there should be a way to store this information so that it can be interfaced with the knowledge base.

5.9 AN EXPERT SYSTEM MODULE FOR MANUFACTURING LOGIC AND OPTIMISATION

Apart from sequencing machining operations, process planners are also concerned with selecting optimum cutting conditions. Selecting of parameters such as feed, speed and depth of cut may be based on factors, such as, reliability of part produced, failure of tool, surface texture,..., etc. IIPPS uses an expert system module to solve these problems in process planning.

5.9.1 KNOWLEDGE ACQUISITION

Experienced operators play an important role in knowledge acquisition. Published guides give relevant information and parameters to build up an intelligent system. In machining process planning, the knowledge acquisition is quite difficult to deal with. A lot of parameters are involved. Each parameter plays an important role in a machining process. The following parameters vary from each raw material under a machining operation:

- selection of machining process;
- selection of machines;
- selection of tool material;
- selection of coolant;
- selection of cutting tools;
- selection of fixtures:
- selection of speed

Acquisition of expert knowledge has been a costly and lengthy process. The acquisition of full knowledge for building of IIPPS has taken many weeks.

Knowledge acquisition sources are:

1. Books and other published materials,
2. Interview with experienced operators and production planners,
3. Own experience.

5.9.2 KNOWLEDGE REPRESENTATION

The knowledge base consists of production rules of the form of IF <premise> THEN <conclusion>. Heuristics, data, and control information may be stored in the form of production rules. At present, most of the rules in IIPPS's knowledge base pertain to data. These rules suggest tools to be used, tool material to be selected, the type of coolant that would be most effective in reducing the heat generated during machining, the speed at which the workpiece should rotate, the rate at which the cutting tool should remove the metal (feed), and the amount of material that should be removed in one pass of the cutting tool (depth of cut). An example of some production rules from IIPPS is presented below:

(ON CHOICE OF CUTTING SEED)

```
topic cut_speed.
if ?material=low_alloy_steel
    and ?tool=single_point
    or ?tool1=single_point
    then name='yes' and
    sp1='150_to_300' and
    sp2='300_to_400' and
    sp3='400_to_550' and
    sp4='550_to_700' and
    sp5='700_to_1200'.
```

```
topic depth_roughpass.
if ?tomin<=0.005
    then depth2=?depth-0.005 and
    finishcut=0.005 and
    feed1='0.004_to_0.005' and
    speed1=?sp5 and
    finishpass=1.
```

(ON CHOICE OF COOLANT)

```
topic coolant.
if ?material=low_alloy_steel
    or ?material=medium_alloy_steel
    then coolant='water_miscible_dry'.
```

(ON CHOICE OF TOOL AND TOOL-MATERIAL)

```
if ?oper1=threading
    then tool='v_tool' and
    toolmat='carbide' and
    done='yes' and
    thread().
```

IF-THEN rules convey the “knowledge” or “expertise” contained in the knowledge base. Each indicates that if a certain condition occurs, then a certain action should be taken. IF can also represent the linkage of evidence and hypothesis. The use of IF-THEN rules has several advantages.

1. The easyness of acquiring knowledge from the expert because the IF-THEN format is similar to the way humans think.
2. The easyness of linking one rule interface to other rule interfaces through the use of metarules.
3. The easyness of implementing backward reasoning.
4. The modularity of knowledge representation.
5. The easyness to maintenance and modification.
6. The understandability of the knowledge represented.

5.9.3 INFERENCE STRATEGIES

Many expert system control strategies are in use today. One or more of these strategies are incorporated in the inference engine of each expert system [19].

- Backward chaining
- Forward chaining
- Breadth-first search
- Depth-first search
- Heuristic search
- Problem reduction
- Pattern matching
- Hierarchical control
- Unification
- Event -divert control

A Process plan contains all detailed operations that change the initial state of a part or product to its goal state. For the machining process the four planning components are as follows:

1. Initial state - raw blank material
2. Goal state - design spesification of the part
3. Operations - drilling, turning, grinding, threading, etc.
4. Resources - machining, tooling, labor etc.

A process planning system generates all machining operations for changing the initial state of a part to its goal state. As mentioned above, the engineering drawing of a part is the goal state. The raw blank material is the initial state. A process planning engineer will start from the goal state, through selecting and determining the necessary operations, resources to decide to choose the raw material, that is, to initial state. So the Integrated Intelligent Process Planning System (IIPPS) use the so called backward chaining as the inference engine.

5.10 THE FUNCTION OF META-SYSTEM

The meta-system can be referred to as the control mechanism of the Meta-level knowledge in IIPPS. The Meta-system can be considered as an expert system to control, coordinate the subsystem, and solve the problems among them via exchanging information between subsystems (Chapter 4). Like common expert systems the meta-system has its own database, knowledgebase, inference engine but it does not provide the solution to any particular problems within the specific domain. It supervises and manages the whole software system. The meta-system drives the inference engine for reasoning and decision-making, based on knowledge and data in the meta-knowledge base and data base. KnowledgePro is used as a development environment to develop the IIPPS.

5.11 VERIFICATION AND VALIDATION

Verification involves the determination of whether or not a system is functioning as intended. This may involve program debugging, error analysis, input acceptance, output generation, reasonableness of operation, run time control, and problem scooping. Verification, in effect, ensures that an intelligent system does not contain technical errors. Validation concerns a diagnosis of how closely a system's solution matches a human expert's solution. Validation ensures that the system does not contain logic errors. If an intelligent system is valid, then the conclusions or recommendations it offers can substitute for those of a human expert. The cost savings associated with implementing the system may also be directly evaluated. The validation of IIPPS was done by using different problem scenarios to simulate consultation sessions. The four modules of IIPPS (Meta-system, AutoCAD module, dBASE III module, and the expert system module) were validated both independently during development as well as collectively after completing the development.

Different part drawings were given and the answers provided by IPPS were checked with the answers of the domain expert. Iterative modification of the system were undertaken until the final answers of the system matched the experts' answers to an acceptable level of confidence.

The great majority of symmetric-rotational parts can be described in a single view. Symbolic representation of a rotational part is shown in Figure 5.8a.

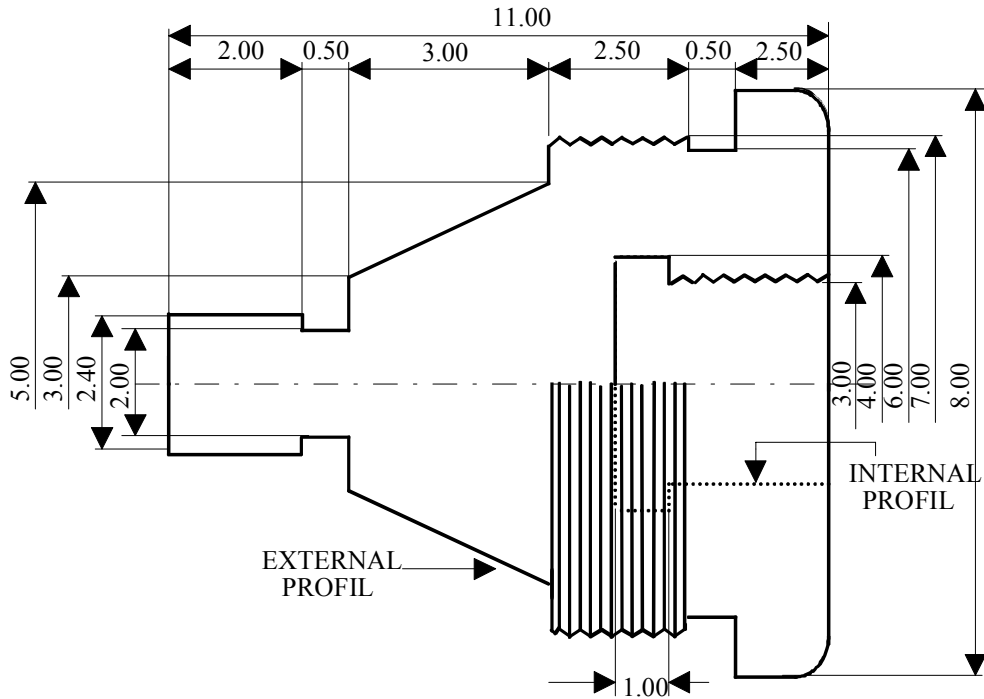


Figure 5.8a Symbolic representation of a rotational part

Figure 5.8b shows one of the sample parts and two segments (Line and Arc) of the corresponding process planning that was produced by the IPPS is given below.

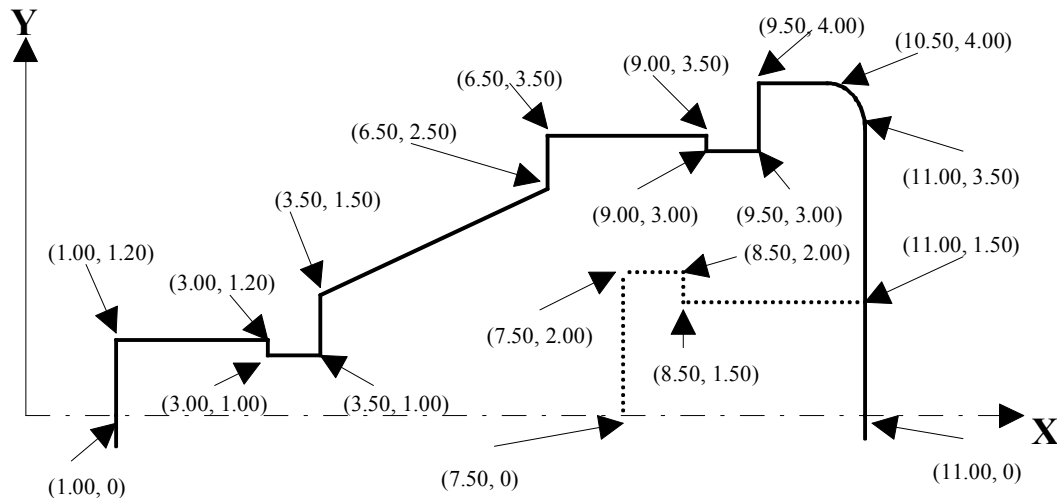


Figure 5.8b Example of input part drawing

A complete process plan generated by IIPPS is presented below:

INSTRUCTIONS

- 1) The raw material of the part is free_cutting_steel
- 2) Take the raw material with
 - Diameter = 8.500 inches
 - Length = 11.0000 inches
- 3) Hold the job in the lathe head stock chuck
- 4) Hold the second part of the job in the lathe tail stock for all operations except the internal operations
- 5) For holding the job in the tail stock use countersunk drill

STEP 1

The operation is : facing
 The tool is : parting_tool
 The tool material is : carbide
 The coolant to be used is : water_miscible_dry
 facing from : x= 1.0000 y= 1.2000
 to : x= 1.0000 y= 0.0000

STEP 2

The operation is : turning
 The tool is : single_point
 The tool material is : carbide
 The total depth of cut is : 0.2500
 The rough cut is : 0.245
 The number of roughpasses are : 1
 The speed is : 350_to_450
 The feed is : 0.060_to_0.18
 The coolant to be used is : water_miscible_dry
 turning from : x= 1.0000 y= 4.0000
 to : x= 11.0000 y=4.0000

STEP 3

The operation is : grooving
 The tool is : parting_tool
 The tool material is : carbide
 The coolant to be used is : water_miscible_dry
 grooving from : x= 3.0000 y= 1.0000
 to : x= 3.5000 y= 1.0000

STEP 4

The operation is : turning
 The tool is : single_point
 The tool material is : carbide
 The total depth of cut is : 2.8000
 The rough cut is : 0.4666666666666666
 The number of roughpasses are : 6
 The speed is : 175_to_350
 The feed is : 0.060_to_0.18
 The coolant to be used is : water_miscible_dry
 turning from : x= 1.0000 y= 1.2000
 to : x= 3.0000 y= 1.2000

STEP 5

The operation is : knurling
 The tool is : knurling_tool
 The tool material is : carbide
 The turns per inch are : 40
 The speed in rpm is : 700
 The surface speed is : 50
 The coolant to be used is : water_miscible_dry
 knurling from : x= 1.0000 y= 1.2000
 to : x= 3.0000 y= 1.2000

STEP 6

The operation is : tapeturning
 The tool is : single_point
 The tool material is : carbide
 The total depth of cut is : 2.5000
 The rough cut is : 0.5
 The number of roughpasses are : 5
 The speed is : 175_to_350
 The feed is : 0.060_to_0.18
 The coolant to be used is : water_miscible_dry
 tapeturning from : x= 3.5000 y= 1.5000
 to : x= 6.5000 y= 2.5000

STEP 7

The operation is : grooving
 The tool is : parting_tool
 The tool material is : carbide
 The coolant to be used is : water_miscible_dry
 grooving from : x= 9.0000 y= 3.0000
 to : x= 9.5000 y= 3.0000

STEP 8

The operation is : turning
 The tool is : single_point

The tool material is : carbide
 The total depth of cut is : 0.5000
 The rough cut is : 0.25
 The number of roughpasses are : 2
 The speed is : 350_to_450
 The feed is : 0.060_to_0.18
 The coolant to be used is : water_miscible_dry
 turning from : x= 6.5000 y= 3.5000
 to : x= 9.0000 y= 3.5000

STEP 9

The operation is : threading
 The tool is : v_tool
 The tool material is : carbide
 The threads per inch are : 25
 The speed in rpm is : 700
 The surface speed is : 50
 The coolant to be used is : water_miscible_dry
 threading from : x= 6.5000 y= 3.5000
 to : x= 9.0000 y= 3.5000

STEP 10

The operation is : parting
 The tool is : parting_tool
 The tool material is : carbide
 The coolant to be used is : water_miscible_dry
 parting from : x= 11.0000 y= 3.5000
 to : x= 11.0000 y= 0.0000

STEP 11

The operation is : filleting
 The tool is : form_tool
 The tool material is : carbide
 The coolant to be used is : water_miscible_dry
 filleting from : x= 11.0000 y= 3.5000
 to : x= 10.5000 y= 4.0000

STEP 12

The operation is : drilling
 The tool is : twist_drill
 The tool material is : HSS
 The tool geometry :
 Point angle : 125
 Lip clearance angle : 10_12
 Chisel point angle : 125_135
 Helix angle : 24_32
 The lubricant to be used : lard_or_soluble_oil
 The speed(sfpm) is : 50
 The feed(ipm) is : 0.01
 drilling from : x= 11.0000 y= 0.0000
 to : x= 7.5000 y= 0.0000

STEP 13

The operation is : boring

The tool is :	boring_tool
The tool material is :	carbide_bit
The total depth of cut is :	1.1000
The rough cut is :	0.365
The number of roughpasses are :	3
The speed is :	275_to_475
The feed is :	0.010_to_0.020
The coolant to be used is :	water_miscible_dry
boring from :	x= 8.5000 y= 1.5000
to :	x= 11.0000 y= 1.5000

STEP 14

The operation is :	boring
The tool is :	boring_tool
The tool material is :	carbide_bit
The total depth of cut is :	1.6000
The rough cut is :	0.4
The number of roughpasses are :	4
The speed is :	275_to_475
The feed is :	0.010_to_0.020
The coolant to be used is :	water_miscible_dry
boring from :	x= 7.5000 y= 2.0000
to :	x= 8.5000 y= 2.0000

STEP 15

The operation is :	inthreading
The tool is :	boring_v_tool
The tool material is :	carbide_bit
The threads per inch are :	30
The speed in rpm is :	700
The surface speed is :	50
The coolant to be used is :	water_miscible_dry
inthreading from :	x= 8.5000 y= 1.5000
to :	x= 11.0000 y= 1.5000

Selected screen displays during IIPPS consultation are presented in Figures 5.9 to 5.11.

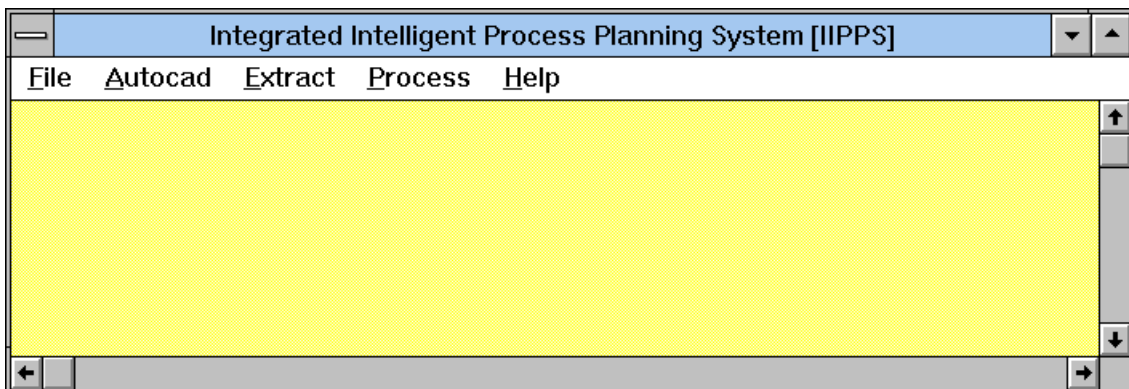


Figure 5.9 IIPPS consultation screen

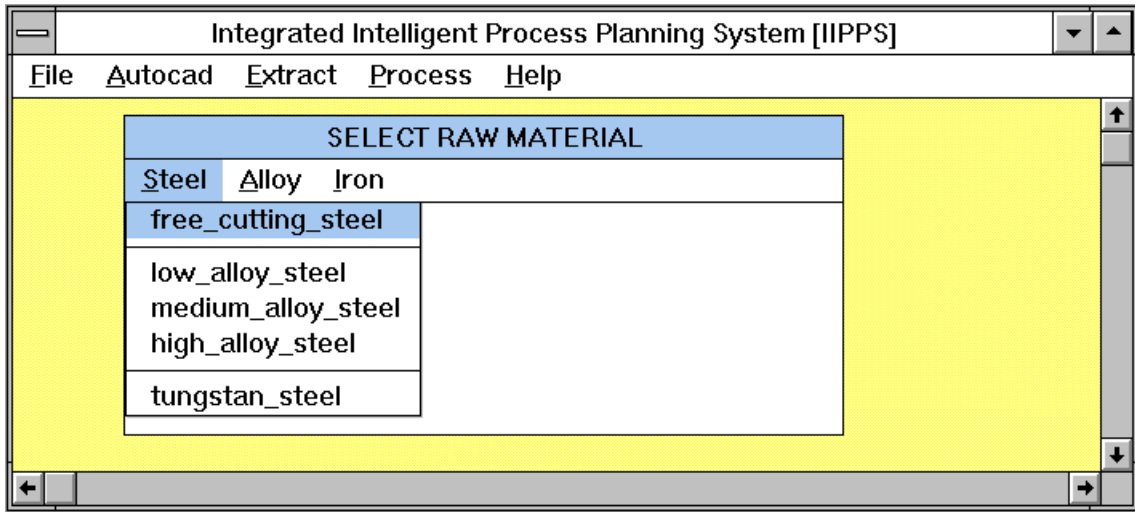


Figure 5.10 IIPPS consultation screen

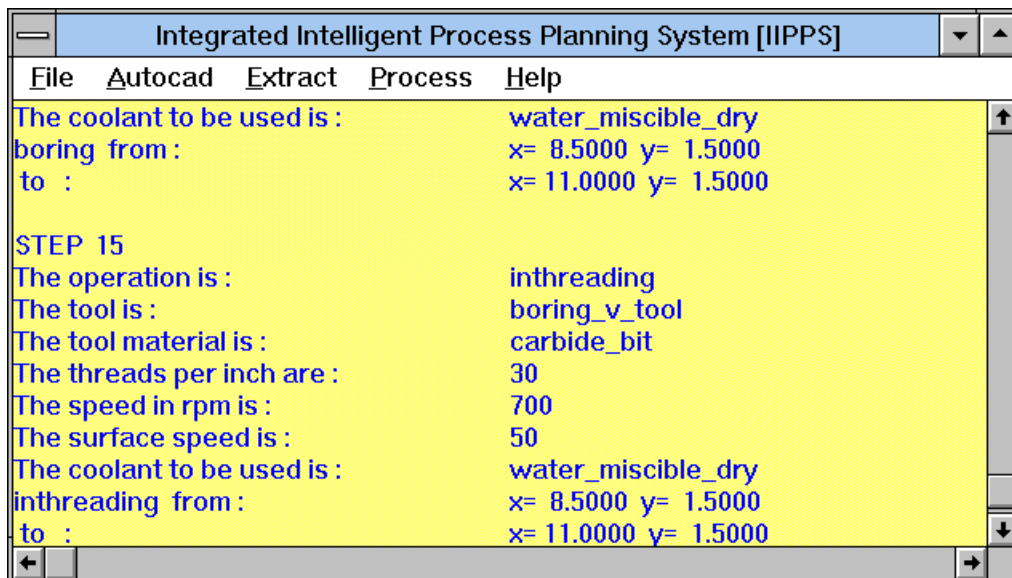


Figure 5.11 IIPPS consultation screen

5.12 CONCLUSIONS

The commercial PC-based software package AutoCAD and dBASE III have been interfaced using a program written in KnowledgePro. The program serves as a process planner utilising information from the AutoCAD database to interpret part geometry, and produce a process plan for the machining of the symmetrical rotational parts.

The PC-based process planning system developed has been instrumental in the reduction of process planning for symmetrical rotational parts, as compared to traditional manual systems. Most importantly, the system may be operated by a person with little machining or planning skills. Following the design of the part in AutoCAD, the only intervention required by the user is to follow the IPPS instructions to generate process planning. Reduction of planning time and elimination of the need for prior machining knowledge leads to significantly increased productivity.

The results from IPPS did show that the system was efficient and had the potential for practical application. It was further determined that the system could be improved considerably if it was applied to a particular manufacturing facility, so that the knowledge base/rule set could be tailored to a specific environment. The future scope of this work involves for example extension, and it is hoped that the results will make a significant contribution to the future development of the Integrated Intelligent Process Planning System for machining processes.

5.13 REFERENCES

- [1] C. H. link, "CAPP-CAM-1 automated process planning system", Proceedings of the 13th Numerical Control Society Annual Meeting and Technical Conference, Cincinnati, OH, March 1976.
- [2] TNO, Introduction to MIPLAN, Organization for Industrial Research, Inc., Waltham, MA, 1981.
- [3] T. T. El-Midany and B. J. Davies, "AutoCAD - a dialogue system for planning the sequence of operations for turning parts", International Journal of Machine Tool Design, pp. 175-191, 1981.
- [4] S. Zhang and W. D. Gao, "TOJICAP - a system for computer aided process planning for rotational parts", CIRP Annuals, 34(1), pp. 299-301, 1984.
- [5] R. A. Wysk, "An automated process planning and selection program: APPAS", PhD thesis, Purdue University West Lafayette, Indiana, 1977.

- [6] M. S. Dumm and M. S. Mann, "Computerized production process planning", Proc. of 15th Numerical Control Society Annual Meeting and Technical Conference, Chicago, IL, 1978.
- [7] K. H. Tempelhof, "A system of computer aided process planning for machine parts", SME Technical Paper, Series MS79-154, 1979.
- [8] K. Matsushima, N. Okada and T. Sata, "The integration of CAC and CAM by application of artificial intelligence techniques", *Annals of CIRP*, 31(1), pp. 329-332, 1982.
- [9] T. C. Chang, "TIPPS - a total integrated process planning system", PhD thesis, Virginia Polytechnic Institute, Blacksburg, Virginia, 1982.
- [10] C. Mouleeswaran, "PROPLAN - a knowledge based expert system for manufacturing process planning", MS thesis, University of Chicago, IL, 1984.
- [11] T. Lenau and L. Alting, "Xplan- an expert process planning system", Second International Expert Systems Conference, London, UK, 30 September-2 October 1986.
- [12] B. K. Choi and M. M. Barash, "STOPP: an approach to CAD/CAM integration", *Computer-Aided Design*, 17(4), pp. 162-168, May 1985.
- [13] X. Dong and M. Wozny, "Feature extraction for computer aided process planning", *ASME Computer in Engineering*, 31 July-4 August, 1988.
- [14] T. C. Woo, "Interfacing solid modeling to CAD and CAM: data structures and algorithms for decomposing a solid", Technical Report 83-6, Department of Industrial and Operations Engineering, University of Michigan.
- [15] H. Kung, "An investigation into development of process plans from solid geometric modeling representation", PhD thesis, Oklahoma State university, 1984.
- [16] M. R. Henderson and G. J. Chang, "FRAPP: automated feature recognition and process planning from solid model data", *ASME Conference of Computers in Engineering*, San Francisco, California, USA, 31 July-4 August, pp. 529-536, 1988.
- [17] H. Sakurai and D. C. Grossard, "Shape feature recognition from 3D solid models", *ASME Conference of Computers in Engineering*, San Francisco, California, USA 31 July-4 August, 1988.
- [18] Joshi, Ajay P., *PROCESS-PLUS: A Prototype Expert System for Generative Process Planning*, MS Thesis, school of Industrial Engineering, University of Oklahoma Norman OK, 1989.
- [19] James M., Steven O., 'Building expert systems' New Jersey 1989.

6. COUPLING QUANTITATIVE COMPUTING AND QUALITATIVE REASONING IN THE OPERATION OF PROCESS INDUSTRIES

6.1 INTRODUCTION

There is presently a very strong interest in the methods of quantitative and qualitative computing in science, engineering and business. As evidence by the applications in process industries, combining qualitative (symbolic) methods derived from artificial intelligence research with more traditional quantitative (numeric) methods. Traditional quantitative methods are obtained from numerical analysis, operations research, and simulation. This can field more powerful and more useful problem solving capabilities. This chapter focuses on how the methods and technique of coupling quantitative and qualitative computing can be used to help the human operator in the sulfite pulping process to produce good quality pulp.

Experienced operators who have spent many years at the plant have often accumulated much private knowledge. This very valuable knowledge could be lost when these experts retire. By use of intelligent systems we can codify their experience and expertise for future use, i.e. to store these most valuable assets for the company. In some cases replacement is unnecessary.

The main purpose of most chemical processes is to transform raw materials into high-quality products at the lowest cost, with minimum impact on the environment. Therefore, a chemical plant that is well designed, well controlled and well operated, should meet and satisfy the above requirements.

This chapter presents the construction of the Integrated Intelligent Process Support System (IIPSS) for a batch sulfite pulping process. All the relevant information on how to create the IIPSS system is taken from the Handbook for pulp & paper Technologists [1] and Ming Rao and Jean Corbin [2]. This system contains a qualitative reasoning system and a quantitative computing system, and is used to help human operators to produce satisfactory quality pulp. The safe operation of a chemical process depends on the human operator and the real-time process control system, both of which must perform well together. Many chemical industrial processes are very complex, in nature, and the development of a mathematical model for process control is so difficult that the performance of the real-time control system becomes poor.

In such a case, only the most experienced and skilled operators can handle the operation effectively.

High-quality pulp is often produced by a chemical process under batch operation. This chemical process utilises cooking liquor (cooking acid) to dissolve and remove the lignin from wood chips for the production of sulfite pulp. *Its primary objective is to cook wood chips to a desirable degree of delignification (i.e. desired kappa number).* When a batch digester is used, the quality of pulp is only sampled and tested at the end of the batch cycle. As a batch digester operates at high temperatures and high pressures, it is not practical to remove a pulp sample and analyse its quality (kappa number) on-line. For these processes, optimum operation and quality control depend heavily on the operators, who need to use their experience and skills to estimate the pulp quality and to decide the time to terminate the cooking process in order to produce good-quality pulp.

Artificial Intelligence (AI) techniques are being used more and more by researchers and engineers. It is suggested that AI techniques could be used in the operation and control of chemical processes at the human operation level, to solve ill-structured problems that are very difficult to handle by purely numerical algorithm methods. This is one way to assist human operators in evaluating the data and to advise them how to react to changing conditions. Since expert systems incorporate the knowledge of experienced operators and designers, the operators have the benefit of expert advice at short notice at all times.

The framework of the integrated intelligent system, which integrates both quantitative reasoning and quantitative computation with a multimedia technology, is used to develop the IIPSS. IIPSS can provide the estimated 'cooking time' and 'kappa numbers', as well as other type of process information. Its use has greatly increased interaction between human operations and the controlled process, and thus provides more information interpretation for process operation. IIPSS can also be used to train engineers and operators. This methodology paves the way for process operators to upgrade and improve the old industrial production environment using advanced manufacturing technology.

This chapter first presents some background information about sulfite pulping at a real industrial plant in order to introduce the application of Integrated Intelligent Process Support System (IIPSS). Then problems in the pulping process, solving strategies, the quantitative model and the qualitative model are discussed. The system construction based on the concept of integrated intelligent systems and the construction of the IIPSS are discussed. Finally, implementation and demonstration are presented, and conclusions drawn.

6.1.1 QUANTITATIVE AND QUALITATIVE MODEL IN INTEGRATED INTELLIGENT SYSTEM

The progress in Integrated Intelligent Systems (IIS) and particularly the success of Artificial Intelligence (AI) has led to the application of IIS to a wide variety of fields. The interest in applying IIS to Science and engineering is increasing rapidly[17]. Problem solving in the engineering domain requires common sense, heuristic knowledge and knowledge about mathematical models. The goal of Integrated Intelligent Systems in the operation of process industries is to reproduce the intelligent reasoning of experienced engineers and operators.

The fact that qualitative and quantitative mathematical models play a central role in Science and Engineering generates a number of questions.

- How can qualitative and quantitative models be integrated together?
- How can quantitative models be represented with standard AI tools?
- What is the relationship between the use of qualitative and quantitative models in traditional artificial Intelligence?

Some of these issues have been addressed at the first computation[18]. A relatively large literature exists on qualitative reasoning about physical problems[19, 20] but the main focus has been on reproducing human intuitive reasoning in everyday situations rather than the reasoning on an engineer or scientist, which necessarily involves the simultaneous use of both qualitative and quantitative methods.

There are several factors that are involved in the choice between a qualitative and a quantitative model. Some reasons to choose a qualitative model are:

- All qualitative models are computationally intractable.
- No satisfactory quantitative model exists or is known.
- No qualitative model can be used because of insufficient input data.

A main reason to choose a qualitative model is:

- A qualitative model is known that is fast enough and applicable in all relevant cases.

Therefore it is clear that symbolic reasoning (qualitative) can not replace numeric reasoning (quantitative) or vice versa, but rather they complement each other. The aim of this chapter is to show how different methods can be used interchangeably and how symbolic reasoning can control the selection methods.

6.2 SULFITE PULPING PROCESS

The cooking liquor is usually prepared by burning sulfur to produce SO_2 gas and then absorbing the SO_2 in an alkaline base solution. In the older calcium sulfite mills, limestone was used exclusively in the gas absorption tower, serving both as a packing and a chemical source of calcium to produce calcium bisulfite. In the newer mills, one of the soluble bases, in the form of NH_4OH , $\text{Mg}(\text{OH})_2$ or Na_2CO_3 , is used to absorb the SO_2 . A simplified flow diagram for the sulfite pulping system is shown in Figure 6.1.

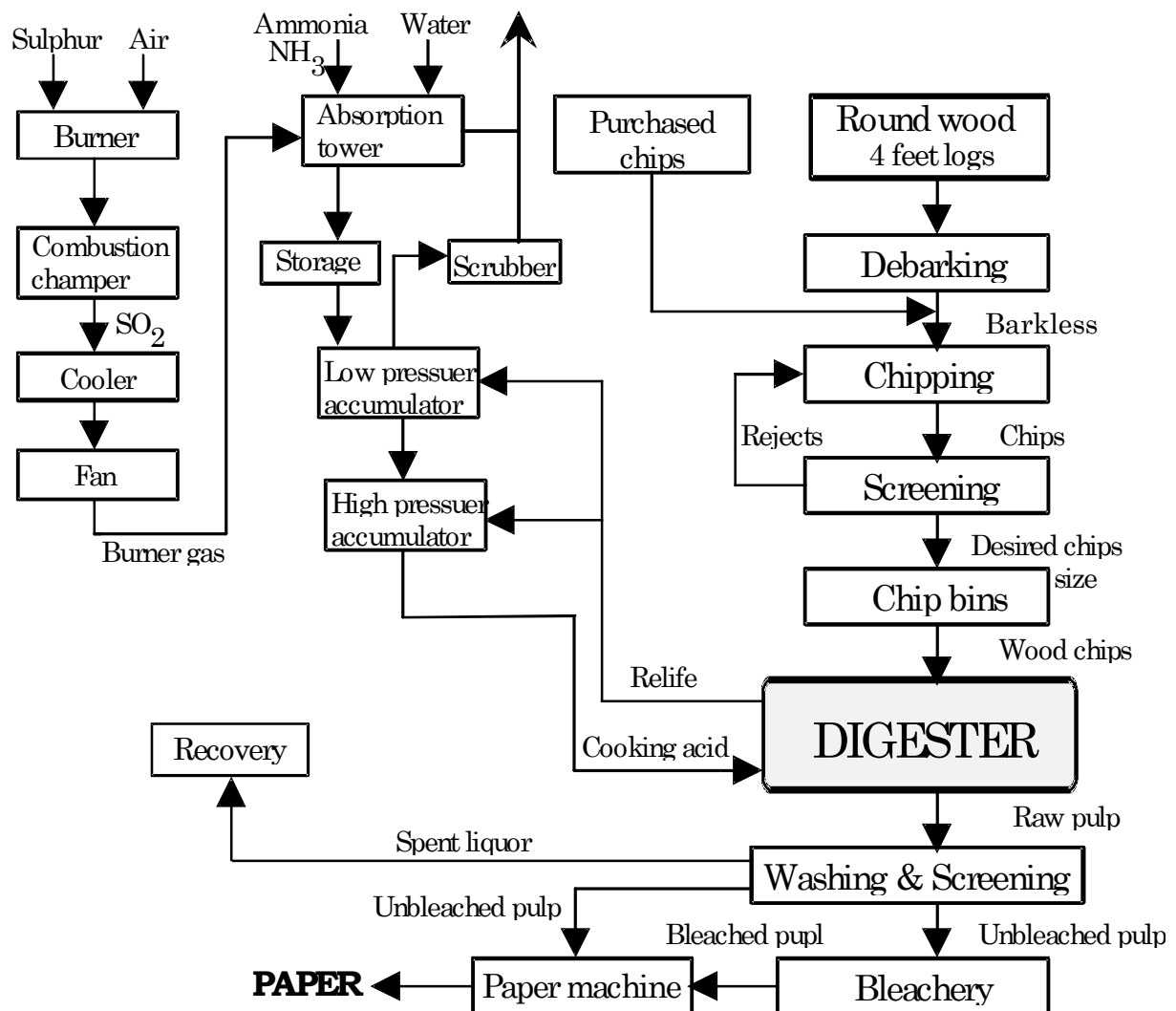


Figure 6.1 Material flow diagram of sulfite pulping and paper process operation.

The raw cooking acid after SO₂ absorption is a mixture of free SO₂ and combined SO₂ in the desired proportions. Before the raw acid is used in pulping, it is fortified with SO₂ relief gas from the digester (see Figure 6.1). This fortification usually takes place in low and high pressure accumulators.

Wood chips from a wood room and/or from a chip pile are transported to the chip bins located above the digester house, then sent to the digester through a chute. During chip filling, steam at 345 kPa is injected tangentially near the top of the digester: this creates a rotating motion of the chips to produce a dense packing per batch.

The digester is first filled with chips and capped, then sufficient hot acid from the high pressure accumulator is added to almost fill the vessel. After a certain amount of acid has entered the vessel, the circulation pump starts automatically. Meanwhile, a cooking liquor sample is taken, and its quality (total %SO₂, free %SO₂, combined %SO₂, and pH) is analysed. The digester is equipped with a steam-fed heat exchanger where the circulating cooking liquor is heated. Its temperature is controlled by manipulating the steam going to the heater.

As the temperature and pressure are increased, the hot acid is rapidly absorbed by the chips. The chemical reaction does not gain momentum until the temperature has exceeded 110°C, but it is important at this stage that the fibers are well “impregnated” with chemicals. Typical Sulfite cooking curves of temperature and pressure are shown in Figure 6.2.

The progress of the cook is followed by observing the colour of the liquor and by running periodic tests of residual SO₂. After the operator controls that the desired pulp quality has been reached, the cooking liquor is drained from the digester. This operation is called drain down. It not only reduces the total pressure in the vessel, but also makes room for the quenching liquor. Heat and pressure is reduced by relieving gas and steam to the accumulator. *A well judged drain down operation is most important because it heavily affects pulp quality.*

During the quenching phase, a cold weak liquor is introduced into the digester to further reduce pressure and temperature in the vessel. This weak liquor has practically no chemical strength and comes from the washing and screening processes of the plant. The quenching operation stops the cooking process to prevent dissolving of the cellulose portion of the wood chips, thus preventing loss of yield, and loss in pulp brightness and strength. After the quenching operation, the dump valve is opened and the vessel is emptied. The pulp passes through several processes for further refinement including rock trap, defiberizer, washing and screening, bleaching, and paper machines. If the digester is not emptied immediately after the quenching phase, pulp brightness, pulp yield, and pulp quality will be affected.

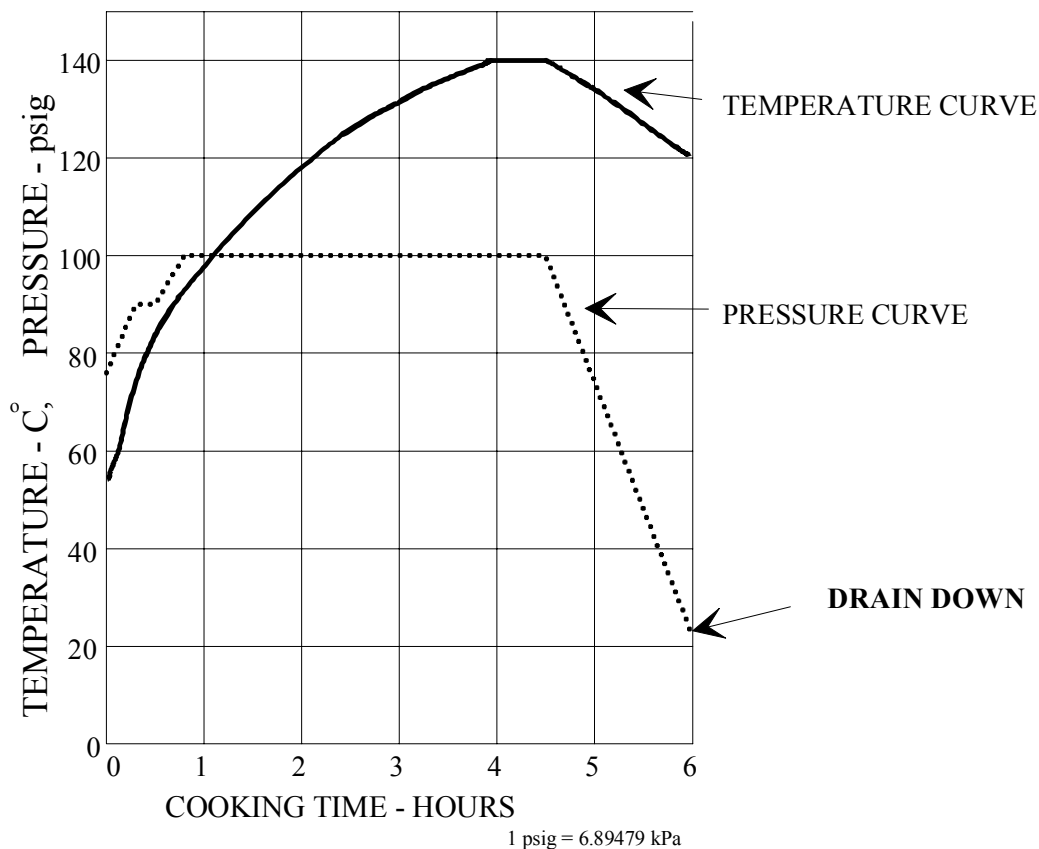


Figure 6.2 Temperature and Pressure curves of sulfite cooking.

6.3 PROBLEMS IN THE PULPING PROCESS

Digesters cook wood chips to a specified degree of delignification, measured by the kappa number (see Section 6.5). The kappa number is determined by a laboratory test of a pulp sample because it cannot be measured on-line in the mill [2]. The content of lignin in the wood (pulp) cannot be measured during the batch cooking cycle. The pulp sample is taken from the dump line after the cooking cycle is completed. The kappa number test is then performed on the sample, and the result is recorded on the digester log sheet. This value is used as a guide by the operations at the bleachery plant for the bleaching operation.

The target kappa number is 28. A value between 26 and 30 is considered satisfactory. In the case the laboratory test shows a kappa number higher than 30, indicating a high lignin content still remaining in the pulp, the bleachery operator needs to increase the rate of chlorine/chlorine dioxide for pulp bleaching. A low kappa number for the collected sample (i.e. below 26) indicates loss of pulp strength, pulp brightness, and

pulp yield in that particular batch cook. Occasionally, the quality is so bad that the pulp cannot be used for making good quality paper.

In a sulfite mill, under normal conditions, a cooking time is between 3:45 and 3:55 hours to obtain a satisfactory pulp quality (i.e. a kappa number between 26 and 30). Another difficulty is that the qualities of the raw materials (cooking acid and wood chips) that are changed into the digester are not always as expected. The cooking acid is produced in an upstream recovery process, and its quality changes significantly, which affects the cooking time. The chip quality, which varies in terms of moisture content, species, and chip age, has a similar effect on cooking time. For example, a certain quality of raw material will require a cooking time of 3:35 hours for the digester to achieve its objective, whereas under different circumstances, the batch cooking cycle might require four hours to obtain satisfactory pulp quality.

The operator's objective is to control the digester to produce the best pulp based on the given raw materials by choosing the appropriate operating conditions. The operator must decide when to terminate the process by pressing the down button so that good quality pulp can be obtained in the digester at the end of the process. Since no general mathematical model is available for this process, successful operation relies mainly on the operator's private knowledge such as, personal experience, expertise and heuristics, etc. However, such experienced operators are few and expensive to employ.

As mentioned above, pulp quality will be affected by several factors. Some factors are listed below:

1. Wrong interpretation of the process data by a human operator who is responsible for it. Even operators who have many years of operationing experience and know the process very well, are inefficient at quickly recalling the effects of particular data on one digester and different data on another digester. Thus, they are confused in operation.
2. The main criteria which has to be satisfied to produce pulp of the desired puerility, is to choose the right cooking time to maintain optimal operating conditions. Cooking time is a function of the raw materials quality (wood chip and cooking liquor) and the operating conditions (temperature and pressure).

The difficulties in the operation and control of the chemical batch process may be described as the following:

- Important private knowledge about the pulping operation cannot be accumulated, then transferred for general use.
- Many ill-structured problems that are difficult to solve by mathematical modelling pulp numerical control methods.
- In some mills, the kappa number can be estimated only by a few experienced operators, which greatly affects the pulp product quality.
- There exists little interaction between the process operators and the controlled process.
- Digesters are not always at the optimal operating conditions.

6.4 PROBLEM SOLVING STRATEGIES

The Integrated Intelligent Process Support System (IIPSS) is developed to solve the problems mentioned above.

- A mathematical cooking model (quantitative model) developed based on S_m -factor (general knowledge) can also be implemented in such a computer program, and used for supervisory control.
- IIPSS suggests the proper cooking time based on the available information concerning the process (qualitative model), and monitors the cooking process to estimate kappa numbers. By investigating the effect of all the available process quality input on the cooking time (private knowledge), the IIPSS can quickly process the knowledge and provide operational support. The private knowledge is obtained from the experts (operators), and coded into a computer system.

These two models use the framework of integrated intelligent system.

This methodology will increase the interaction between the process and human operators, provide more information interpretation during the pulping operation, thus

optimise the production environment. It transfers and accumulates the expert knowledge into computer programs so that those who do not have the operational experience can also control digesters at the level of expert operators. Obviously, using the newest developments in advanced technology to upgrade and improve the old industrial manufacturing facility, is another advantage.

6.5 THE QUANTITATIVE MODEL

Both qualitative reasoning and quantitative computation are needed for the operation and control of the cooking process. Qualitative reasoning is usually efficiently based on symbolic and graphic information (see Section 6.6), while quantitative computing is more conveniently performed using numerical information. Each method often complements the other [8].

The IIPSS has a quantitative computation system to calculate cooking time and kappa number. The mathematical model implemented in the IIPSS is based on the S_m -factor and it is programmed in the C⁺⁺ language. Its module is called 'pulp.ppc'. *This numerical calculating system can be used only when the chip size and quality are under normal conditions.* The qualitative reasoning system using heuristics and private knowledge can not guarantee perfect solutions. Therefore, it solves ill-formulated problems and generates suggestions about possible changes in the operating environment.

The succeeding section summarises the computation process to calculate the S_m -factor, and the equations to calculate the cooking time and kappa number.

The S_{factor} method is often used to calculate the degree of delignification [9]. This method includes both temperature and pressure.

$$S_{m1\text{-factor}}(\text{target}) = \int_0^t P_{\text{tot}} * k * dt. \quad (1)$$

and

$$\text{kappa number} = a_1 + a_2 * \int_0^t P_{\text{tot}} * k * dt + a_3 * [\text{TS}(0)] + a_4 * [\text{TS}(10)]. \quad (2)$$

where

a_1, a_2, a_3, a_4 = regression coefficients,

k = relative reaction rate,

P_{tot} = total pressure inside the digester (atm),

$TS(0)$ = percentage of total SO_2 of the cooking liquor during acid filling (before cooking). This value can be obtained during acid filling.

$TS(10)$ = percentage of total SO_2 of the circulating cooking liquor 10 minutes after the operating temperature has reached a maximum. This value can be obtained from the circulating liquor during the cooking cycle (e.g. approximately 2:10 hours after the cooking cycle has started).

The relative reaction rate (k), can be calculated by use of the following equation

$$k = e^{[F - E/R * T]} \quad (3)$$

where

E = activation energy (cal/mol),

F = frequency factor (constant),

R = gas constant (cal/mol-K), and

T = absolute cooking temperature (K).

For sulfite pulping, the above parameters can be chosen to have the following numerical values:

$E = 20200$ cal/mol,

$F = 27.25$, and

$R = 1.987$ cal/mol-K

The numerical values of the regression coefficients are given as follows:

$a_1 = 108.2$

$a_2 = 0.02170$

$a_3 = 8.336$, and

$a_4 = 7.425$.

By setting the kappa number to 28 (i.e. target kappa number) in equation (2) it can be rearranged in the following manner to calculate target S_m -factor.

$$S_{m\text{-factor}} = (28 - a_1 - a_3 * [TS(0)] + a_4 * [TS(10)]) / a_2. \quad (4)$$

S_{m1} -factor is calculated during the rising time operation stage, when the temperature and total pressure are increased to their operation conditions. When the temperature and total pressure are at their operating conditions the S_{m2} -factor can be calculated by a simplified equation as:

$$S_{m2}\text{-factor} = P_{tot} * k * \Delta t. \quad (5)$$

The total S_m -factor can be calculate by

$$S_m\text{-factor} = S_{m1}\text{-factor} + S_{m2}\text{-factor}. \quad (6)$$

To calculate the cooking time, the system requires the S_{m1} -factor and the time at which the cooking liquor sample test for the total $SO_2(10)$ is taken. Once the pressure and temperature of the digester are held constant at their operating conditions, the S_{m1} -factor can be calculated.

$$\text{Cooking time} = (S_{m1}\text{-factor} (\text{target}) - S_{m1}\text{-factor} / P_{tot} * k) + t_1. \quad (7)$$

where

t_1 = the time of which the total $SO_2(10)$ is taken (hour),

P_{tot} = operating pressure (atm), and k = relative reaction rate.

Once the process is at its operating temperature and pressure, the user can obtain an estimated current pulp kappa number. Kappa number calculated by equation (2) based on the value of the S_{mt} -factor at the requested time. S_{mt} -factor calculated by the following equation:

$$S_{mt}\text{-factor} = S_{m1}\text{-factor} + P_{tot} * k * [t - t_1]. \quad (8)$$

where t = any particular time after the target S_{mt} -factor is calculated.

Remaining cooking time can be calculated from the following equation:

$$\text{Remaining cooking time} = \text{cooking time} - t. \quad (9)$$

Equations (1) to (9) are used in the IIPSS to predict the **cooking time**, **kappa number**, and **S_m -factor**.

6.5.1 THE CALCULATION PROCESS

Required data from Microsoft excel are passed to the numerical computation model by the inference engine to calculate the **target S_m -factor**. The **target S_m -factor** is calculated based on the target kappa number (i.e. 28). The **target S_m -factor** can only be calculated approximately 2:10 hours after the cooking cycle has been started. This is because total SO_2 is required for the calculation of **target S_m -factor** , which is obtained 10 minutes after the operation temperature has reached the maximum value. Normally it takes 2 hours for the temperature to reach a maximum [2]. The result is displayed on the computer screen through the user interface. The quantitative system calculation process is shown in Figure 6.3.

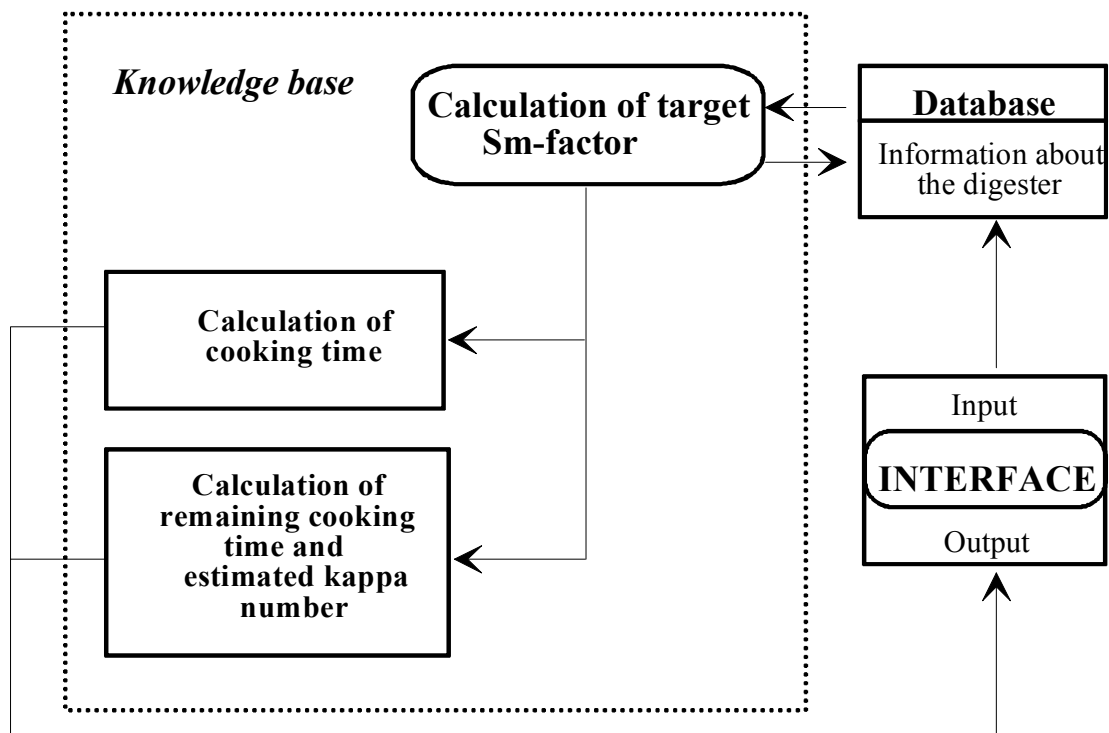


Figure 6.3 Calculation of the quantitative system.

6.6 QUALITATIVE KNOWLEDGE ACQUISITION AND REPRESENTATION

Knowledge acquisition is important to understand in detail the digester's operation and the related problems before starting the construction of IIPSS. The main purpose for the developers is to acquire the valuable knowledge that has been used in the digester's operation to produce sufficient products. Then, the acquired knowledge can be built into the IIPSS so that the knowledge can be easily transferred to the users. Another objective of the IIPSS is to increase the interaction between operators and the controlled process and give operators assistance to produce good quality pulp. The better organised and the more accurate the knowledge is, the more efficient the IIPSS is.

Once the process operation and related problem were studied, all the available quality inputs and outputs were investigated [16]. These process variables can be classified as follows:

Table 6.1 shows the desired conditions for all process variables. If a process input is outside these conditions, the desired pulp quality will not be achieved.

Table 6.1 Desired conditions for pulping quality variables

Quality inputs	Desired conditions
Load of wood chips	35% - 40%
Size of wood chips	1" - 3" x 1" - 3" x 1/4" - 1/2"
Quality of wood chips	Good quality (< 6 weeks old) 90% softwood - 10% hardwood
Total %SO ₂ of initial cooking acid	5.45% - 6.15%
Combined %SO ₂ of initial cooking acid	2.65% - 3.00%
Free % SO ₂ of initial cooking acid	2.80% - 3.15%
pH of initial cooking acid	3.4 - 3.6
Operating temperature	162 - 165 °C
Operating pressure	675 - 685 kPa
Cooking time of batch cycle	3:45 - 3:55 (hour)

Quality outputs	Desired conditions
Kappa number	26 - 30
Pulp brightness	62 - 68

Table 6.2 Description of each situation

Individual inputs	Description	Average cooking time
Wood chips load	22% - 29% gauge 30% - 34% gauge 35% - 40% gauge 41% - 46% gauge 47% - 55% gauge	3:20 - 3:30 hour 3:30 - 3:40 hour 3:45 - 3:55 hour 4:00 - 4:10 hour 4:10 - 4:20 hour
Wood chips quality	Good quality Old chips More than 10% hard wood Old chips & < 10% hardwood	3:45 - 3:55 hour 3:25 - 3:35 hour 3:25 - 3:35 hour 3:15 - 3:35 hour
Wood chips size	Standard size Over standard size Under standard size	3:45 - 3:55 hour 4:05 - 4:15 hour 3:25 - 3:55 hour
Total %SO ₂ of initial cooking liquor	4.00 - 4.80% 4.81 - 5.44% 5.45 - 6.17% 6.16 - 6.50% 6.51 - 7.40%	4:05 - 4:15 hour 3:55 - 4:05 hour 3:45 - 3:55 hour 3:35 - 3:45 hour 3:25 - 3:35 hour
pH of initial cooking liquor	2.5 - 2.9 3.0 - 3.3 3.4 - 3.6 3.7 - 4.0 4.1 - 4.6	3:25 - 3:35 hour 3:35 - 3:45 hour 3:45 - 3:55 hour 3:55 - 4:05 hour 4:05 - 4:15 hour
Operating temperature	154 - 157 °C 158 - 161 °C 162 - 165 °C 166 - 169 °C 170 - 175 °C	4:10 - 4:20 hour 4:00 - 4:10 hour 3:34 - 3:55 hour 3:30 - 3:40 hour 3:20 - 3:30 hour 3:10 - 3:20 hour
Operating pressure	575 - 624 kPa 625 - 674 kPa 675 - 685 kPa 686 - 735 kPa 736 - 785 kPa	4:05 - 4:15 hour 3:55 - 4:05 hour 3:45 - 3:55 hour 3:35 - 3:45 hour 3:25 - 3:35 hour

Table 6.2 shows the required cooking time for different process quality input conditions. All information was acquired from the operators. Because normal cooking time differs by less than 10 minutes (i.e. between 3:45 hours and 3:55 hours), a minimum of 10 minutes range for counting the cooking time is used in order to classify

the different input conditions, these ranges can be compiled together to derive a coupled cooking time (i.e. a near optimum cooking time range).

6.6.1 COUPLING SYSTEM

If some input variables are not within their values, the system needs to couple the effect on the cooking behaviour and to derive a coupled required cooking time. Figure 6.4 shows how the input variables can be coupled together. Thus, different cooking times can be obtained at three different levels.

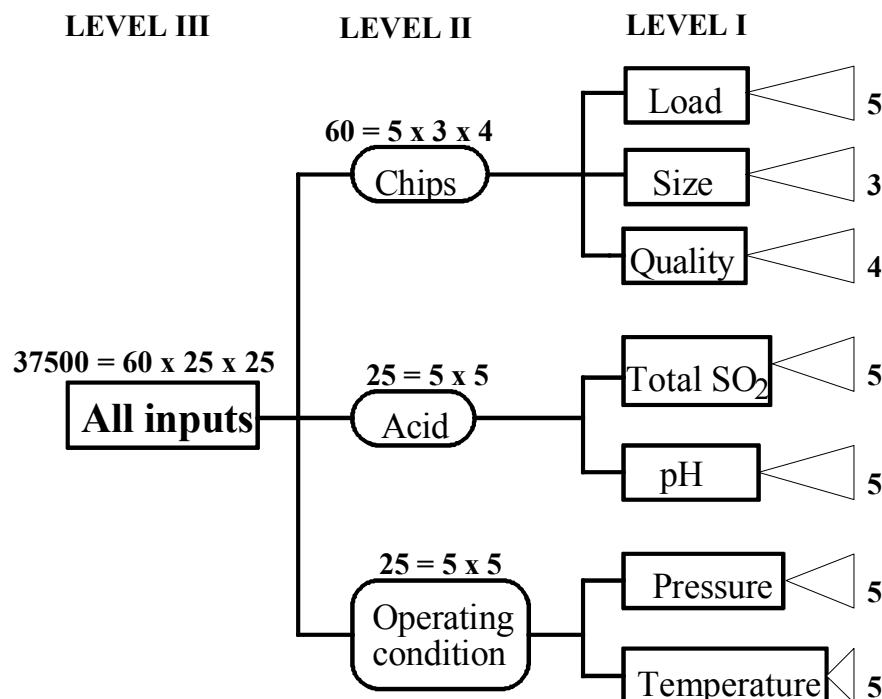


Figure 6.4 Coupling input variables.

Level **I** derives cooking time for each individual input variable. Level **II** gives the coupled cooking time for the wood chips (the system couples the effects of chip size, chip load, and chip quality together), the cooking acid (the system couples the effects of Total SO₂ and pH together), and finally the operating condition (the system couples the effects of operating temperature and operating pressure together).

Level **III** gives a coupled cooking time for all input variables together. The coupled cooking time is provided to users through the bar graph display by selection of the Result module. The compilation of the effect of these input variables on the cooking time is done by adding the weight values in the weight array of each input variable.

Table 6.3 provides the weight values for the different possible values of all input variables.

Table 6.3 Description of each situation at level I

Individual inputs	Description	Weights
Wood chips load	22% - 29% gauge	-4
	30% - 34% gauge	-2
	35% - 40% gauge	0
	41% - 46% gauge	+2
	47% - 55% gauge	+4
Wood chips quality	Good quality	0
	Old chips	-3
	More than 10% hard wood	-3
	Old chips & < 10% hardwood	-3
Wood chips size	Standard size	0
	Over standard size	+3
	Under standard size	3
Total %SO ₂ of initial cooking liquor	4.00 - 4.80%	+3
	4.81 - 5.44%	+1
	5.45 - 6.17%	0
	6.16 - 6.50%	-1
	6.51 - 7.40%	-3
pH of initial cooking liquor	2.5 - 2.9	+3
	3.0 - 3.3	+1
	3.4 - 3.6	0
	3.7 - 4.0	-1
	4.1 - 4.6	-3
Operating temperature	154 - 157 °C	+4
	158 - 161 °C	+2
	162 - 165 °C	0
	166 - 169 °C	-2
	170 - 175 °C	-4
Operating pressure	575 - 624 kPa	+3
	625 - 674 kPa	+1
	675 - 685 kPa	0
	686 - 735 kPa	-1
	736 - 785 kPa	-3

Table 6.4 gives the required cooking time corresponding to different weight values. If a cumulative weight value is higher or lower than those presented in the table, the system is unable to provide the coupled cooking time. Figure 6.4 shows also the number of possibilities that IIPSS can handle at each level. A total of 37500 different possible combinations exist at level III of the coupled system.

Table 6.4 Cooking time ranges for coupled weight values

Weight values	Required cooking time	Impact on cooking time
+9	4:35 - 4:45 hours	40 - 50 minutes longer
+8	4:30 - 4:40 hours	35 - 45 minutes longer
+7	4:25 - 4:35 hours	30 - 40 minutes longer
+6	4:20 - 4:30 hours	25 - 35 minutes longer
+5	4:15 - 4:25 hours	20 - 30 minutes longer
+4	4:10 - 4:20 hours	15 - 25 minutes longer
+3	4:05 - 4:15 hours	10 - 20 minutes longer
+2	4:00 - 4:10 hours	5 - 15 minutes longer
+1	3:55 - 4:05 hours	0 - 10 minutes longer
0	3:45 - 3:55 hours	normal cooking time
-1	3:35 - 3:45 hours	0 - 10 minutes longer
-2	3:30 - 3:40 hours	5 - 15 minutes longer
-3	3:25 - 3:35 hours	10 - 20 minutes longer
-4	3:20 - 3:30 hours	15 - 25 minutes longer
-5	3:15 - 3:25 hours	20 - 30 minutes longer
-6	3:10 - 3:20 hours	25 - 35 minutes longer
-7	3:05 - 3:15 hours	30 - 40 minutes longer
-8	3:00 - 3:10 hours	35 - 45 minutes longer
-9	2:55 - 3:05 hours	40 - 50 minutes longer

Quantitative and qualitative knowledge are integrated in the IIPSS and coordinated by a meta-system [8]. Numerical calculation and symbolic reasoning, as well as graphical representation, are integrated to facilitate the function of the IIPSS. The integrated architecture in IIPSS is shown in Figure 6.4. Both quantitative and qualitative systems can be used by the operators for decision-making. The mathematical model is only a function of the temperature, pressure, total SO₂, and time. *The quantitative computation system cannot be used when other input variables (i.e. chip size, chip quality, chip load, and pH of acid) are outside their normal condition.* Therefore, the quantitative computation system does not provide qualitative information about the effect of input variables on the final pulp product (such as effect on pulp strength, pulp brightness, etc.).

The user can obtain quality information about the process behaviour by the qualitative reasoning system. The qualitative system does not guarantee a perfect solution, but it can give the directions concerning the process behaviour changes. This type of system increases the interaction between operators and the actual process.

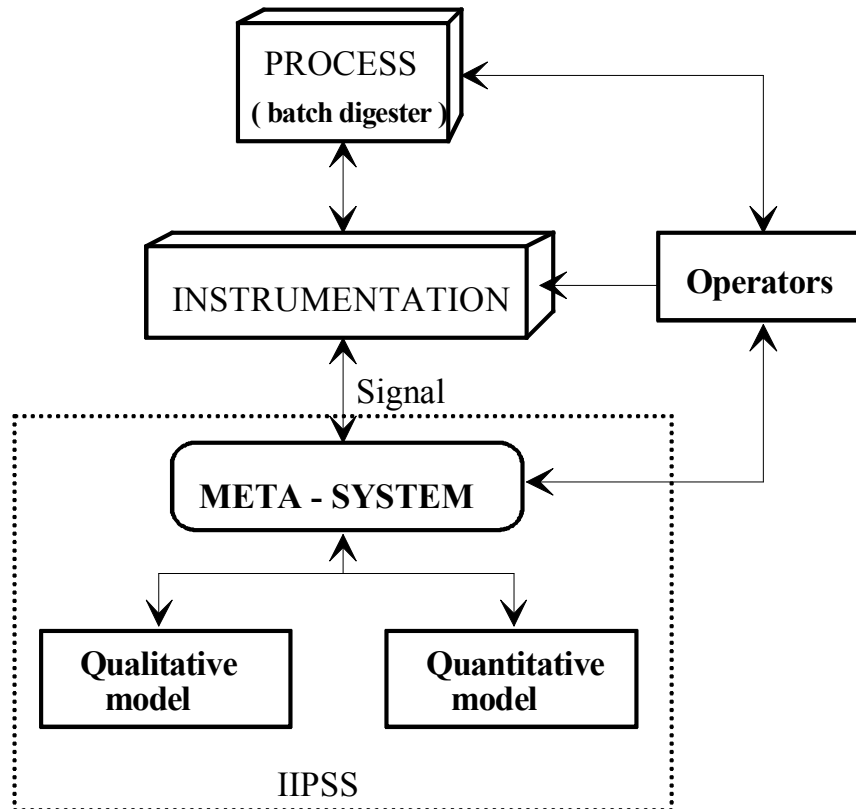


Figure 6.5 Integrated architecture in IIPSS.

6.7 SYSTEM CONSTRUCTION

IIPSS is written in Knowledge Pro language and developed using the Borland TURBU C++ compiler. The numerical computation program is written using C++ language. The IIPSS is designed to support the digester operation and to meet the operators' demands. By using a conventional programming language, the developer can customise the program, and modify it for the users' specific requirements. The interface of the IIPSS main window consists of an upper-menu, lower-menus, and different windows, which make the input/output of information much easier to interchange between the user and the IIPSS (Figure 6.6).

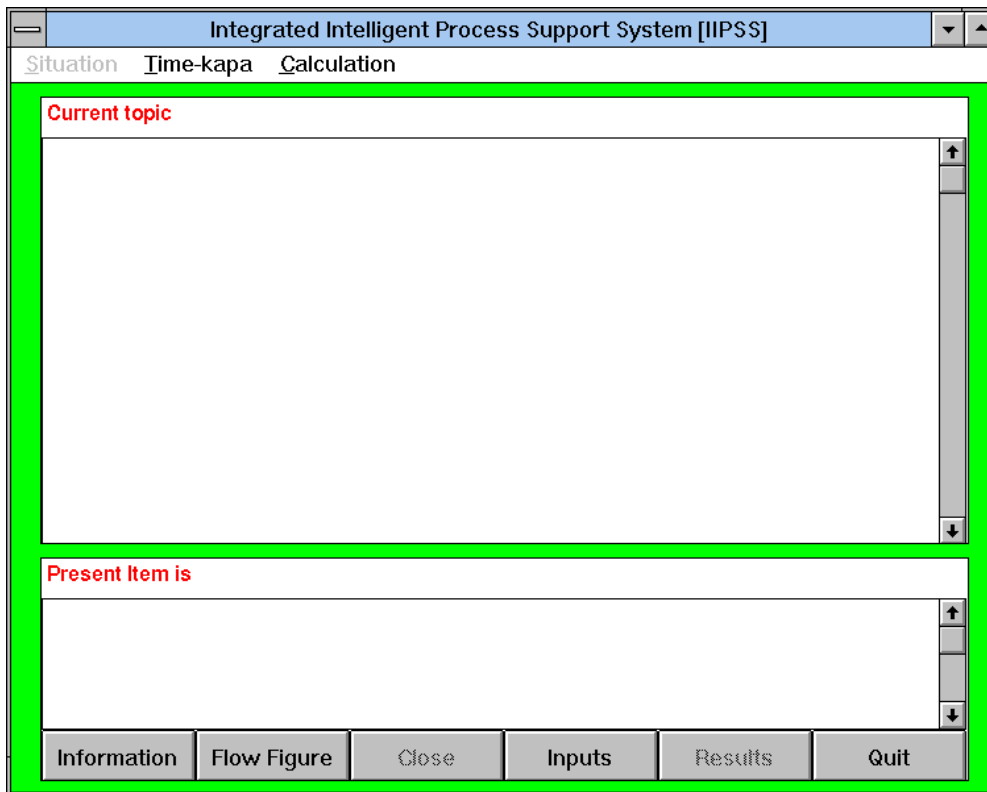


Figure 6.6 The main window of Integrated Intelligent Process Support System.

The main advantage of using a conventional programming language is that once the system (IIPSS) is completed and compiled, it can be installed on any PC-based computer without the purchase of any software or shell.

6.7.1 OPERATING PROCEDURE OF IIPSS

The IIPSS is a menu-driven system, which can be easily installed on any PC/MSDOS stage. The main function modules contain the following choices that can be selected by the user on the computer screen. There are two main menus.

Upper-menu

[Situation] [Time-Kappa] [Calculation]

Lower-menu

[Information] [Flow Figure] [Close] [Input] [Results] [Quit]

χ χ χ

[View Inform] [<= | =>] [Colse1]

χ

[Inform]

When the IIPSS is running, the main window as shown in Figure 6.6 appears. The upper-menu appears on the top of the display screen and the lower-menu appears on the bottom of the display screen. Each selection from the main menu provides a sub-menu and/or a window system to display information to the user, or to process the information better. The main functions of each selection are described as follows:

[Information]: The Information selection gives a brief introduction about the sulfite pulping process. It describes the purpose of digesting and the concept of delignification. It also gives the new user a quick understanding of each process stage.

[Flow Figure]: This selection provides a graphical display of the batch digester system which includes the main pulping, circulating pump, dump valve, and liquor heater. It also provides the material flow diagram of pulp and paper process operation.

[Input]: The input module takes the data entered by the user into the IIPSS. These inputs are presented in Table 6.2. The data required by the S_m -factor model are entered from the Time-Kappa menu.

[Results]: This selection provides qualitative knowledge based on the status of the data (i.e. information entered through the Input selection). The qualitative knowledge includes the estimated cooking time. All output values are described by bargraphs.

[Quit]: The Quit selection terminates the IIPSS application.

[Situation]: The Situation module gives quantitative knowledge based on the status of the data. It provides possible reasons for when a certain quality input is outside its desired operation conditions.

[Time-Kappa]: This module takes the qualitative data required by the S_m -factor model entered by the user into the IIPSS.

[Calculation]: This selection provides quantitative values. Which is the results from the S_m -factor model.

The other modules will be of help to the above mentioned modules.

6.8 IMPLEMENTATION AND DEMONSTRATION

Figure 6.7 shows the input variables assigned for a particular batch cycle. These data are entered into the IIPSS by selecting Input lower-menu. All qualitative input variables are entered on the same window. As an example, Figure 6.7 explains how the value for the deferent input variables are entered into the IIPSS. Once all the required data are entered, the user may first select Situation to obtain qualitative information about the process operation behaviour. Figure 6.8 displays the qualitative reasoning results for the operation chip load.

Integrated Intelligent Process Support System [IIPSS]

Situation Time-kapa Calculation

Please give input values here

Wood chips load	35 - 40% gauge	↓	pH of initial cooking liquor	3.4 - 3.6%	↓
Wood chips quality	Good quality	↓	Operation temperature	162 - 165 C	↓
Wood chips size	Standard size	↓	Operation pressure	675 - 685 kPa	↓
Total SO2 of initial cooking liquor	5.45 - 6.15%	↓	Combined %SO2	2.80	
Free SO2	2.95		Total SO2(10)	0.00	
S - factor model	0.00		Are you sure all the input values are correct?	Yes	

Figure 6.7 Input data.

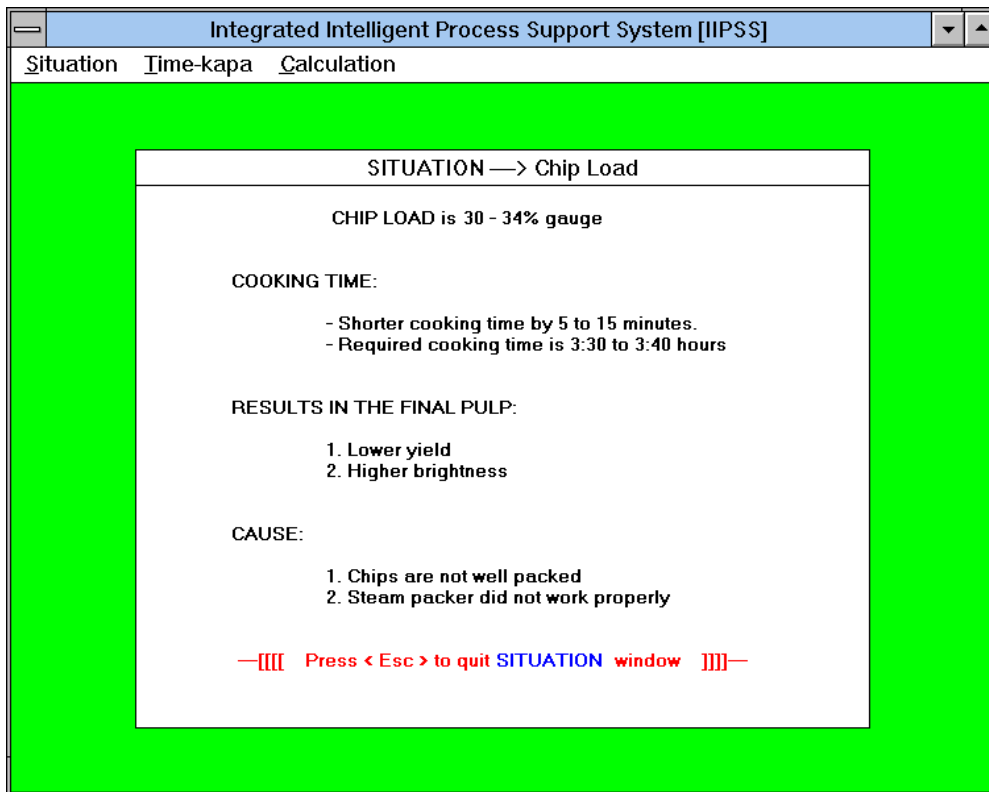


Figure 6.8 Qualitative reasoning results for operation chip load.

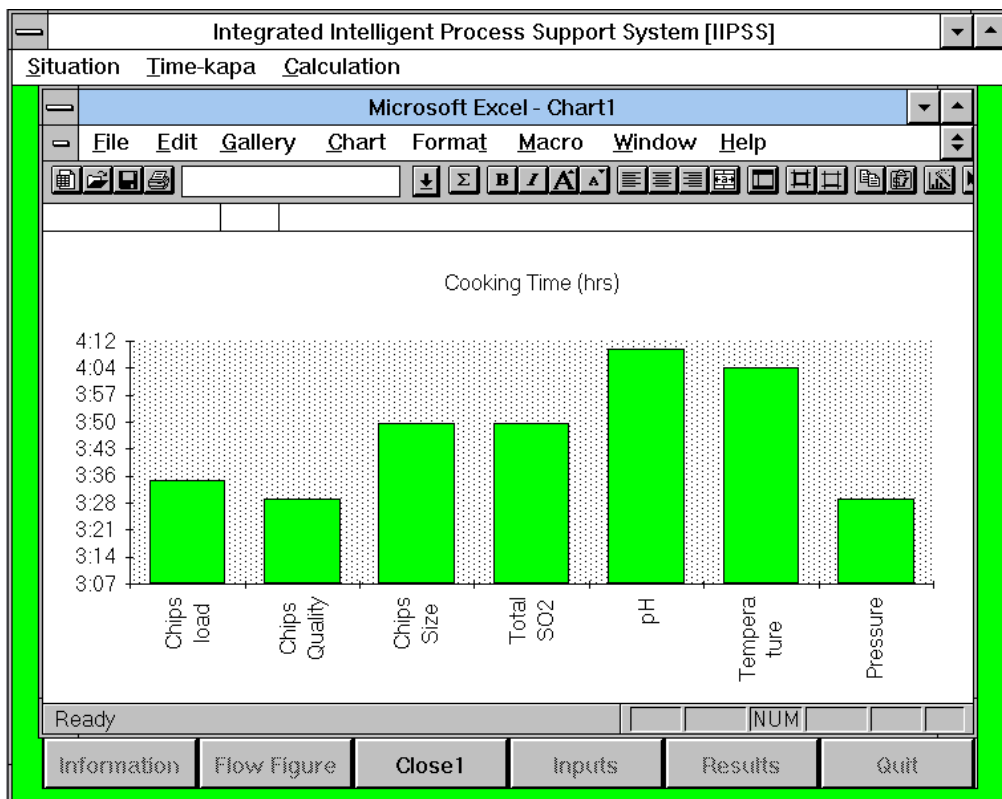


Figure 6.9 Estimated cooking time for individual input variables [Level I].

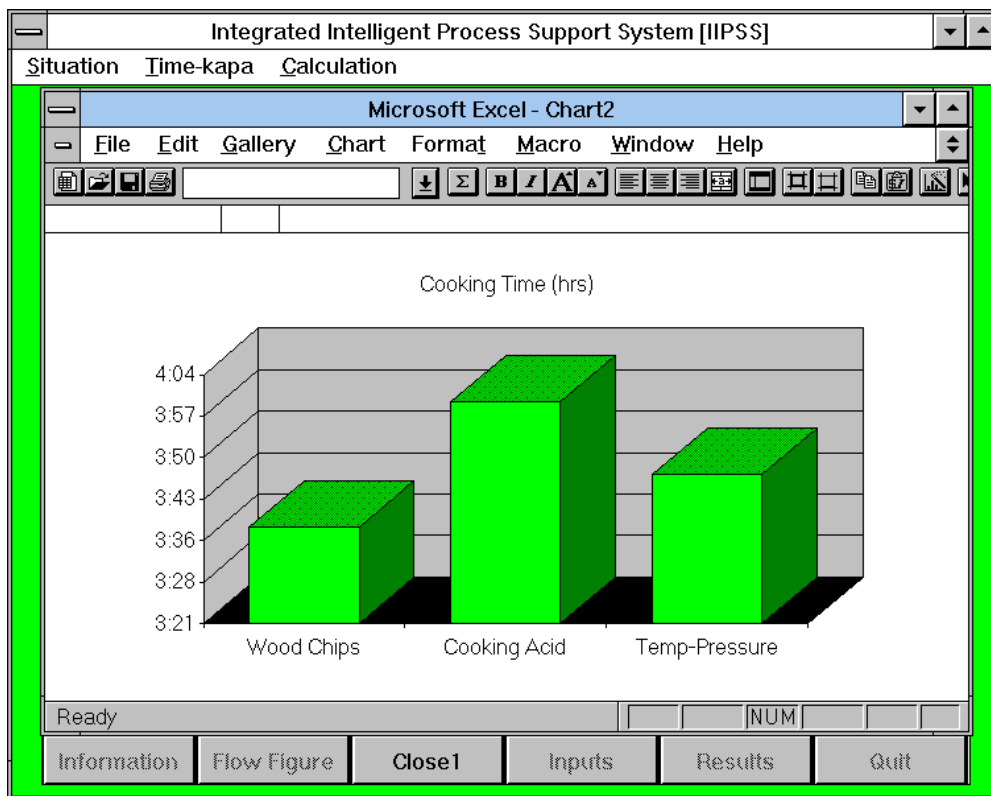


Figure 6.10 Estimated cooking time based on selected input variables [Level II].

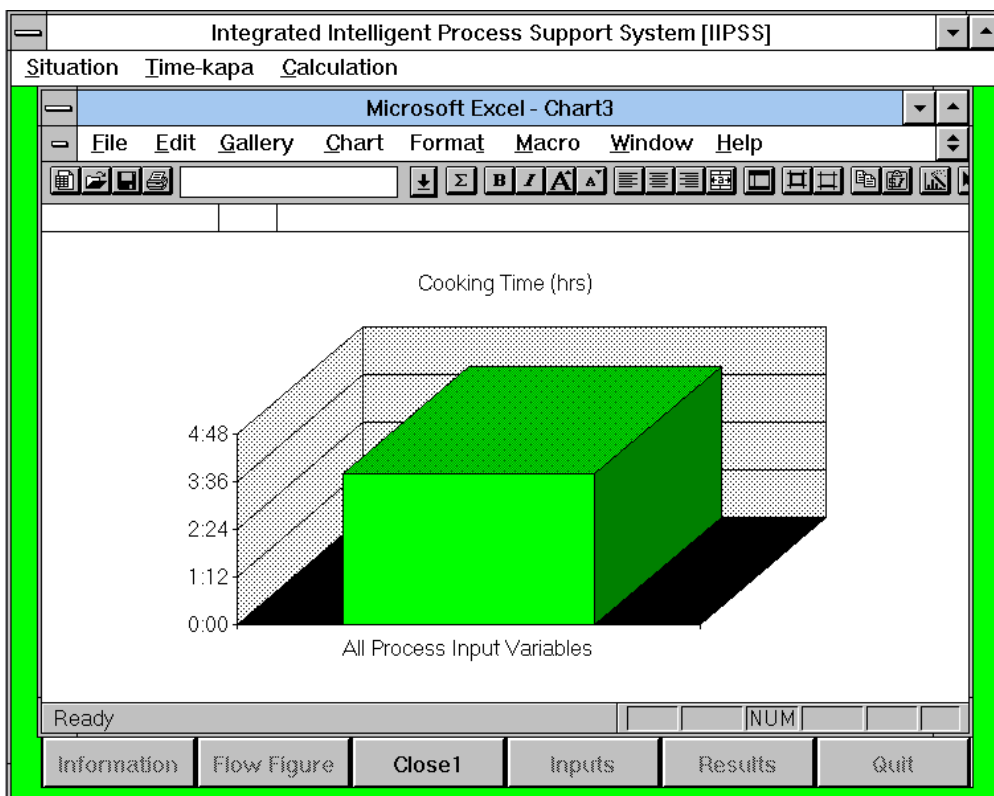


Figure 6.11 Estimated cooking time based on all input variables [Level III].

The qualitative illustration is demonstrated above. Quantitative illustration also demonstrated below. By selecting the Time-kappa model the following data are entered:

Chip quality:	old quality	Operation temperature:	165 °C
Chip load:	32%	Operating pressure:	680 kPa
Chip size:	standard size	Total %SO ₂ (10):	1.89%
Free SO ₂ :	3.05%	Time of sampling (t ₁):	2:10 hours
Combined SO ₂ :	2.90%	S _{m1} -factor:	125 atm/h
Total SO ₂ :	5.95%	pH:	3.5

Figure 6.12 shows the input data. Figure 6.13 provides the required target S_m-factor, estimated remaining cooking time and kappa number.

Integrated Intelligent Process Support System [IIPSS]

Situation Time-kapa Calculation

Please input values for QUANTITATIVE computation

What is the total SO₂ of cooking acid in percent during acid filling.....[x.xx] or [x.xx] 5,38

Enter the %total SO₂ of cooking acid sample collected 10 minutes after the reached a MAXIMUM.[x.xx] or [x.xx] 1,89

Enter the TIME the sample was collected (i.e. time from the start of the cycle.[x:xx]) 2:10 (h:min)

Enter the S - factor at that time the sample was collected.....[xx.xx] or [xxx.x] 124

Operation Temperature (*C) 163 Operation Pressure (kPa) 680

Enter TIME into cooking cycle..... [x:xx] 3:23 (h:min)

Are you sure all the input values are correct? Yes

Figure 6.12 Input data for quantitative computation.

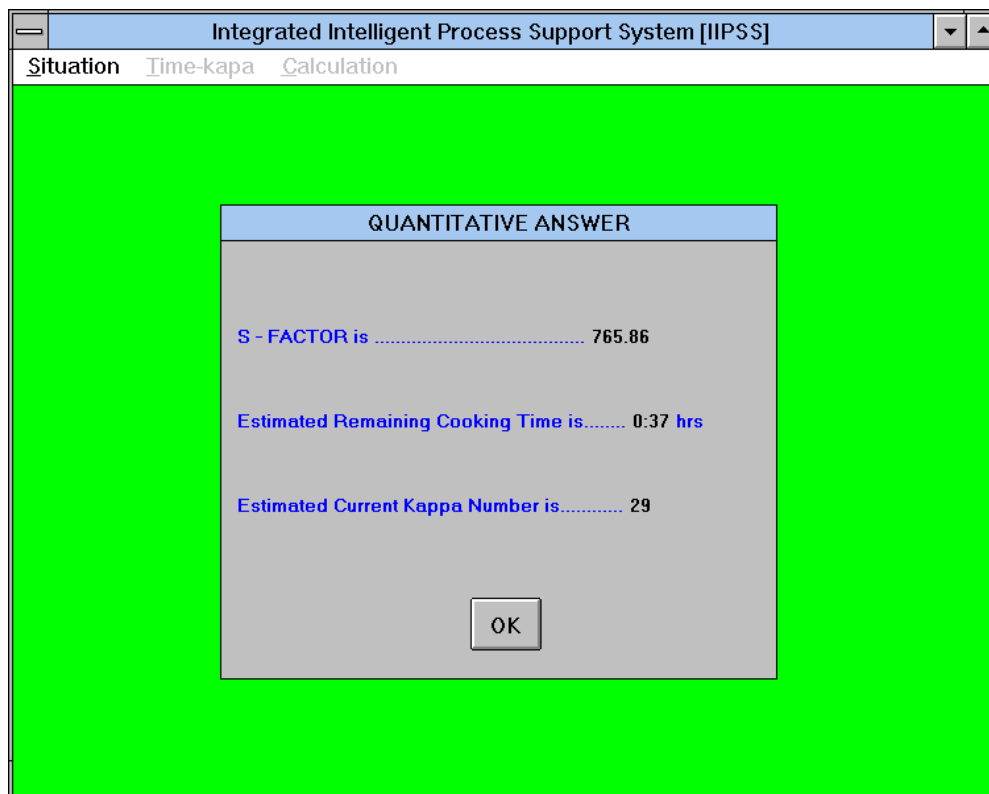


Figure 6.13 Quantitative output values

6.9 CONCLUSIONS

The Integrated Intelligent Process Support System (IIPSS) is implemented as an off-line system whose input data are entered by the user using a keyboard, and is not physically connected to the instrumentation of the digester. However, IIPSS can provide on-line process assistance which helps the operators control the digester's operation. The objective of this development is to provide assistance to human operators to achieve better operation of the batch chemical pulping process. IIPSS can be used by different users. For example, the system can be used to give operational support for operators, to help engineers better understand the process, as well as to train new employees. This system can easily be installed on any PC/MSDOS environment. The IIPSS intends to increase interaction between human operators and the pulping process by using both qualitative reasoning and quantitative computation.

The Qualitative reasoning function helps operators to select near optimal operation conditions based on the initial process variables and operation conditions. It gives cooking time for the production of good quality pulp, supervises the process operation, and diagnoses faults.

The quantitative calculation, the IIPSS can give on-line process assistance under the special process operation situation. *It can alter the operator about the remaining cooking time before the drain down operation for a cooking cycle.* The development of IIPSS for a real industrial application requires knowledge from different disciplines. This research work utilises knowledge from chemical engineering, computer science, artificial intelligence.

The IIPSS utilises variable rules to organise its knowledge base. Such a technique greatly reduces both memory space and running time. This also has a common-sense reasoning mechanism to prevent the system from processing impossible physical data. It is implemented using Knowledge Pro language, running on PC/MSDOS that offers the user the lowest development cost, and the higher flexibility to modify the system.

6.10 REFERENCES

- [1] G. A. Smook (1982), "Handbook for pulp & paper technologists".
- [2] Rao, m. and Corbin, J. "Intelligent Operation Support System for Batch Chemical Pulping Process", Engng Applic. Intell. Volu. 6. No. 4, pp. 357-380, 1993.
- [3] Corea, F.R., Peel, D. and Morris, A.J., (1989) "Control Structure Determination and Enhanced Self-Tuning using the MUSE AI Toolkit", proc. IEE Colloquium on "Knowledge Based Environments for Industrial Applications including Cooperating Expert Systems i Control", 9th June 1989.
- [4] Waye D and Terroux A. SIMSMART: synthetic intelligence (SI): the integration of AI and simulation. Pulp & Paper Can. 92. 83-87 (1991).
- [5] Nussbaum M. and Molina O. Intelligent manual: an aid for process engineering. Engng Applic. AI5, 43-49 (1992).
- [6] Surko P. Tips for knowledge acquisition. PC AI, pp. 14-18 (1989); Pulp & paper Can. 90, 112-114 (1989).
- [7] Haywood S. T. An empirical cooking model for magnesium bisulphite pulp.
- [8] Rao M. Integrated System for Intelligent Control, Lecture Note Series in Control and Information Science. Springer, Berlin (1991).
- [9] Bylund L. and Thorsell L. On-line cooking liquor analyzer a means for effective control of sulfite digester. Tappi Proc. Int. Conf. Sulphite Pulping, pp. 285-292 (1982).
- [10] Yorston F. H. and Liebergott N. Correlation of the rate of sulphite pulping with temperature and pressure. Pulp & paper Can. 93, 272-278 (1965).
- [11] Andrews M. and Barnes R. Better-quality pulp gained by using statistics to analyse chip variation. Pulp & Paper 86, 103-107 (1985).

- [12] Rao M. Frontiers and challenges of intelligent process control. Engng Applic. AI 5, 475481 (1992).
- [13] Rao M. and Qiu H. Process Control Engineering. Gordon and Breach, New York (1993).
- [14] Moore, R.L., (1989) "The G2 Real-Time Expert System for Process Control", Proc. International Symposium on Advanced Process Supervision and Real-Time Knowledge Based Control, Dep. Chemical and Process Engineering, University of Newcastle upon Tyne, Newcastle, UK.
- [15] Aynsley, M., Peel, D. and Morris, A.J., (1989) "A real-time Knowledge Based System for Fermentation control", vol. 3, pp. 2239-2244, proc. ACC, Pittsburgh.
- [16] Simith, D. and Hubert, B. (1983) "Digesters' Operation Manual", Fraser Inc., New Brunswick, Canada.
- [17] Knowledge-Based Expert Systems for Engineering Computational Mechanics Publication, Ashurst Lodges Southampton, S04 2AA, UK, 1988.
- [18] Kowalik, J.S., editor, Coupling Symbolic and Numerical computing in Expert System, North-Holland, Amsterdam, The Netherlands, 1986.
- [19] Sack, E., "Qualitative Mathematical Reasoning " Proc. Ninth International Joint Conference on Artificial Intelligence, American Association for Artificial Intelligence, 1985, pp.137.
- [20] Special Volume, "Qualitative Reasoning about Physical Systems", Artificial Intelligence, Vol.24, 1984.

7 AN EXPERT SYSTEM FOR DIAGNOSING FAULTS IN A CHOCOLATE FACTORY

7.1 INTRODUCTION

An expert system is a computer program that attempts to capture the knowledge and experience of one or more human experts in a small domain in order to make such expertise available on demand to the user of the program. One of the most widespread uses of expert systems is for diagnosis of malfunctioning machinery. In this case the human expert can range all the way from the office fix-it person to the engineer who designed the machine. On the basis of this expert's knowledge, the system will "know" what to consider first, depending on the nature of the breakdown. It begins to ask questions of the users about the nature of the problem. The choice of each new question depends on the users previous answer, so that in a few minutes it is possible to pinpoint a potential source of the problem. The user then follows instructions in testing that component. If fixing or replacing that part makes the machine run, then all is well. If, however, the machine in question still does not function properly, the user lets the system know that, and the expert system asks other questions about the next most likely thing to investigate. It will continue in this manner until either the machinery is up and running or it is time to consult a human expert. A well-written system will let the user know when it's time to call in a human expert and it knows its limits.

There is a lot of literature that describes expert systems for faults diagnosis.[1][11] However, few have described the development of an expert system in all details. This chapter presents the development of an expert advisor that can be used to find the faults in the chocolate factory by using the programming tool, KnowledgePro. KnowledgePro is chosen as the development tool because it combines both expert systems and hypertext technologies in a single development tool to run on a PC. The program is easy to use and relatively inexpensive, and has good interface capabilities to external programs. All this makes it possible to use the program for educational training, researching and industrial applications building.[6]

7.2 AN OVERVIEW OF EXPERT SYSTEM DIAGNOSIS

Most expert system diagnostic systems can be classified as operating from either a structural (or functional) basis or a symptom recognition basis.

Symptom recognition reasoning is typified by a rule-based system that attempts to match observed symptoms to fault causes by using a short series of inferences with symptoms as inputs.[2] This is the traditional diagnostic expert system approach, which is sometimes referred to as "shallow reasoning". Such a system is straightforward to implement but it generally incorporates little, or no, knowledge of the structure of the domain, or of potential interactions with other domains.

By contrast, the structural reasoning expert system attempts to construct a frame of inference that takes advantage of known functional relationships in the problem domain in analysing the defective behaviour and arriving at a diagnosis.[2,3,5,7-10]

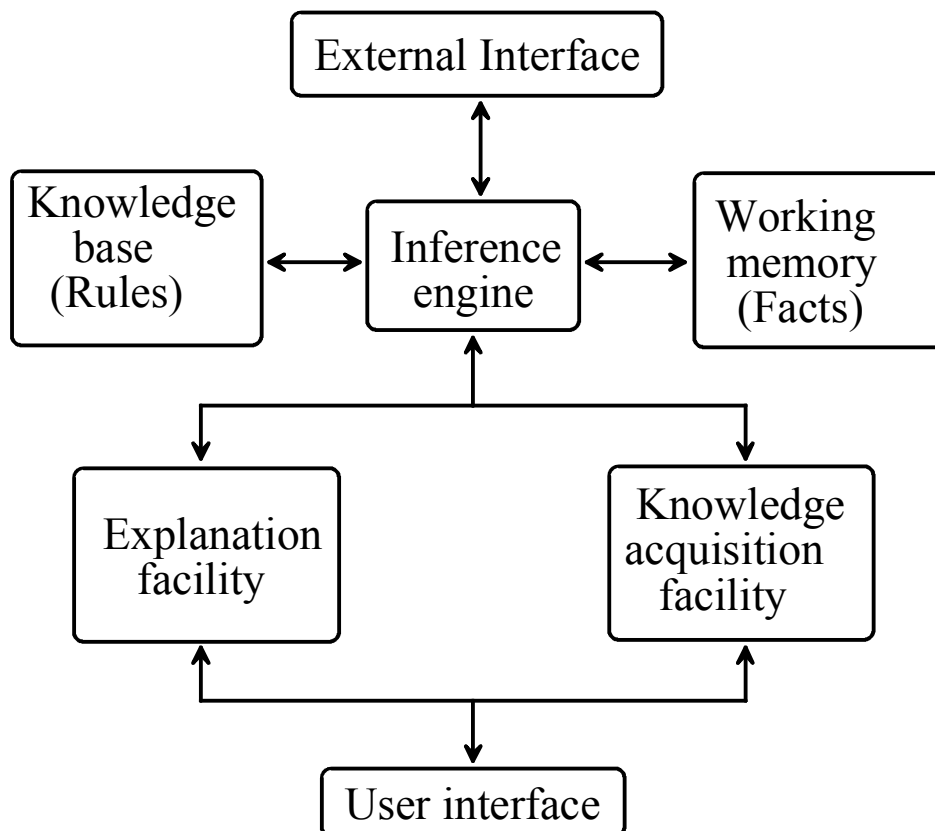
Both of the diagnostic approaches discussed above have limitations when applied to a complex, interrelated system. The symptom recognition reasoning approach knows essentially nothing about the system domain, beyond the explicitly coded information in the rule base. It is generally incapable of dealing with situations which lie beyond these hard coded conditions. Thus these systems may tend to give erroneous diagnostic results, unless exceptions are carefully programmed. In contrast, the structural-reasoning approach can handle a wide range of input conditions, but typically consumes more computational resources in creating a diagnosis.

Recently hybrid diagnostic systems have been created for use in complex systems. These systems tend to combine the ability to make use of functional information and to reduce incorrect diagnosis. It allows isolation of a fault to a specific section of the system, followed by detailed fault detection using heuristics at the unit level.

7.3 EXPERT SYSTEM MODEL

The elements of the expert system are shown in Figure 7.1. The expert system developed is a rule-based system, in which the knowledge base contains the domain knowledge needed to solve problems coded in the form of production rules. The rule-based expert system consists of the following components:

1. user interface: the mechanism by which the user and the expert system communicate.
2. external interface: the mechanism by which the external model (data bases, spreadsheet and other program models) and the expert system communicate.
3. working memory: a global database of facts used by the rules.
4. inference engine: makes inferences by deciding which rules are satisfied by facts.
5. knowledge acquisition facility: an automatic way for the use to enter knowledge in the system rather than by having the knowledge engineer explicitly code the knowledge.
6. explanation facility: explains the processes of inferences to the user.



Error! Switch argument not specified.

Figure 7.1 Expert system architecture.

7.4 KNOWLEDGE REPRESENTATION

7.4.1 PRODUCTION RULE REPRESENTATION

The expert system shell which the factory plans to use, has a rule-based knowledge representation format. It suits domains which require the diagnosis and repair of faults. This is because the operators commonly observe something going wrong, which suggests several possible causes. These can be checked out until one or more breakdowns have been identified. This format of knowledge representation has other benefits because it makes explanations easier and more intelligible. This is hoped to lead to a better trained and more capable workforce.

An operator on one of the machines in the chocolate biscuit factory occasionally notices something going wrong with the chocolate or the biscuits. These observations are called triggers, because they trigger a suspicion in the operator's mind that something is going wrong. Some of the operators are quite good at realising what it is that could have caused these triggers, so a class of rule is constructed which have the following format:

IF <trigger> **THEN SUGGEST** <hypothesis> [ITS]

Hypotheses here are things 'oven too hot', which link a fault in a piece of equipment to an observed effect, such as runny chocolate. Some triggers suggest a single hypothesis, but others could suggest more than one. So, the operator observes a trigger, which suggests one or more possible hypotheses. The hypotheses have to be caused by something in the plant, so another class of rules is needed:

<hypothesis> **CAUSED BY** <fault> [CAU]

For example,

'oven too hot' **CAUSED BY** 'gas temperature too high'

The operator needs a way of checking whether the fault has occurred or not, so a test has to be applied. The proper test is given by a rule of the format:

<fault> **CONFIRMED BY** <test> [CON]

If the test fails, then that is ruled out. But if the test succeeds, then some action must be taken. The action might be a simple repair or, in the case of something more serious,

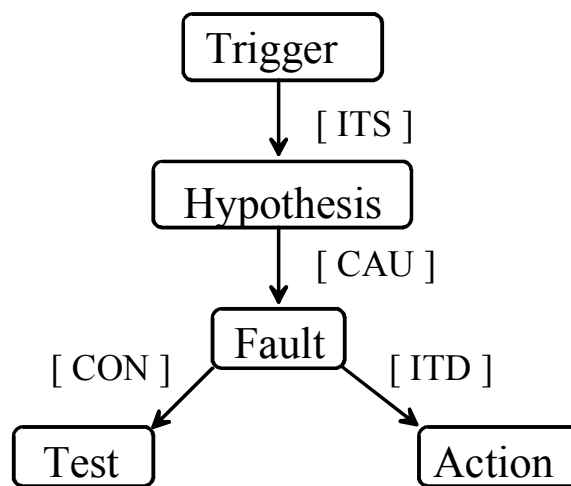
calling a skilled repairman or reporting the fault to the supervisor, before discarding a batch of products. Thus, another type of rule is needed to provide the appropriate action:

IF <test> **THEN DO** <action> [ITD]

The knowledge representation format outlined above uses five different kinds of objects in the knowledge base. The trigger is observation that causes the operator to notice if something is going wrong. This could suggest several hypotheses to the skilled repairman. Each hypothesis could be caused by one more fault. The hypothesis is the mechanism which links a fault on the trigger. If someone asked "Why does high gas temperature make the chocolate runny?", the answer lies in the hypothesis. "Because the oven is too hot". Each fault can be checked by a test and if the fault is confirmed, then some action needs to be taken. These five objects are linked together by four kinds of rules:

IF <trigger> **THEN SUGGEST** <hypothesis>
IF <hypothesis> **CAUSED BY** <fault>
IF <fault> **CONFIRMED BY** <test>
IF <test> **THEN DO** <action>

The four different kinds of rules are needed to represent the different kinds of knowledge that exist. The first rule used data to suggest a hypothesis, but the second uses a hypothesis to suggest data. The third kind of rule provides basic information on how to test for a particular fault, but the fourth says that in the circumstances where a fault is confirmed, then a particular action has to be taken (Figure 7.2).



Error! Switch argument not specified.

Figure 7.2 Rule-based knowledge representation.

7.4.2 KNOWLEDGE ELICITATION

The knowledge engineer feels happy that the knowledge representation format and inference mechanisms are suitable for the chocolate biscuit factory and so embarks in a process of knowledge elicitation. This is not a straightforward task, because so many different people have pieces of relevant knowledge. The operators can describe most of the triggers, but sometimes leave it to the fitters to find out what has gone wrong. The first list of triggers is usually incomplete, but after talking to some of the other people involved, other items can be added.

The next job is to compile a list of possible faults. If a log of faults is kept, then this simplifies the task, but otherwise it is a matter of interviewing the repairmen and fitters to find out what jobs they have to do.

Given these two lists, the rules can be elicited by asking for the links between the two. This can turn out to be an exercise in knowledge refining, because the experts find that having to sit down and think and explain what they do helps them perhaps to find new links or think more carefully about the old ones.

The next stages are fairly straightforward. Each fault can be tested in some way or another. Sometimes it is quite simple, but other times it might involve laboratory tests or queries to suppliers. Whatever the case, some action must be taken. The shift managers can be helpful here. They might have been trying to impose some discipline and consistency on the operator's response to faults and this gives them a chance to review their recommendations and put them down as rules.

The above account of a knowledge elicitation task makes it sound very simple and easy. In reality, all sorts of things can go wrong. Operators might be resentful of a computer program taking over the skills which they are proud to protect. Knowledge might be difficult to elicit from the old hands who feel they have nothing to learn from a computer and, besides, they do it all by feeling anyway. Such obstacles require patience and tact to overcome. Observing the expert at work, if possible, might suggest what the problem solving strategies are. Even if knowledge elicitation is made into an arcade game, expert systems will be no use if people resist and do not use them.

Assuming the knowledge elicitation proceeds successfully, the knowledge engineer could end up with four sets of rules which look like the following tables:

Table 7.1 The trigger-hypothesis rules, [ITS]

IF <trigger> THEN SUGGEST <hypothesis> [ITS]		
Rule number	trigger	hypothesis
ITS 1	biscuits pale	oven too cool
ITS 2	biscuits not risen	oven too cool
ITS 3	biscuits burnt	oven too hot
ITS 4	biscuits burnt	too much cocoa
ITS 5	biscuits burnt	too much sugar
ITS 6	biscuits not risen	too much cocoa
ITS 7	biscuits not risen	too little raising agent
ITS 8	biscuits not risen	faulty flour
ITS 9	biscuits pale	wrong sugar
ITS 10	biscuits pale	faulty flour
ITS 11	chocolate sticky	not enough air in mixture
ITS 12	chocolate sticky	faulty cocoa
ITS 13	chocolate sticky	oven too hot
ITS 14	chocolate runny	rancid butter
ITS 15	chocolate runny	factory too warm
ITS 16	chocolate too dark	too much cocoa
ITS 17	chocolate too light	wrong sugar
ITS 18	bad smell	rancid butter
ITS 19	bad smell	CHECK biscuits burnt

Table 7.2 The hypothesis-fault rules, [CAU]

<hypothesis> CAUSED BY <fault> [CAU]		
Rule number	hypothesis	fault
CAU 1	faulty flour	change source
CAU 2	faulty flour	old state consignment
CAU 3	too little raising agent	someone forgot to add it
CAU 4	too much sugar	nozzle flowing freely
CAU 5	too much cocoa	hopper faulty
CAU 6	too little cocoa	hopper faulty
CAU 7	faulty cocoa	change of source
CAU 8	wrong sugar	faulty batch
CAU 9	not enough air in mixture	dust cover blocked
CAU 10	not enough air in mixture	fan slow or stopped
CAU 11	oven too cool	gases too cool
CAU 12	factory too warm	warm temperature outside
CAU 13	oven too hot	gas temperature too high
CAU 14	oven too hot	cooling failed
CAU 15	too little sugar	nozzle opening restricted
CAU 16	faulty flour	faulty batch
CAU 17	rancid butter	warm temperature outside
CAU 18	faulty cocoa	faulty batch
CAU 19	oven too cool	low temperature gases

Table 7.3 The fault-test rules, [CON]

<fault> CONFIRMED BY <test> [CON]		
Rule number	fault	test
CON 1	dust cover blocked	remove back from dust cover and look
CON 2	fan slow or stopped	shine strobe light and the steady black line is not visible
CON 3	faulty batch	sending sample to laboratory
CON 4	warm temperature outside	taking reading of wall thermometer. Should be less than 80°
CON 5	hopper faulty	measure time for level in hopper to drop by one gradation. Should be 4 minutes
CON 6	change of source	check bags
CON 7	old consignment	check date on bags
CON 8	someone forgot to add it	check log OR ask supervisor
CON 9	nozzle flowing freely	see if sugar flowing steadily from nozzle, without breaking into droplets
CON 10	nozzle opening restricted	see if sugar dripping at less than 10 drops per minute
CON 11	cooling failed	tap turned on AND water jacket cool to touch AND pump not vibrating
CON 12	cooling failed	tap turned on AND water jacket warm to touch
CON 13	gas temperature too high	white color at viewing window
CON 14	low temperature gases	black color at viewing window
CON 15	low temperature gases	red color at viewing window
CON 16	low temperature gases	flickering light at viewing window

Table 7.4 The fault-action rules, [ITD]

IF <test> THEN DO <action> [ITD]			
Rule number	fault	action	lost Production
ITD 1	dust cover blocked	clean and replace	some
ITD 2	fan slow or stopped	call fitter	major
ITD 3	faulty batch suspected	report fault to supervisor	major
ITD 4	warm temperature outside	report fault to supervisor If butter rancid THEN discard batch	some
ITD 5	hopper faulty	call fitter	major
ITD 6	old consignment OR change of source	report fault AND discard batch AND check faulty batch	major
ITD 7	someone forgot to add it	add ingredient AND report fault AND discard batch	major
ITD 8	nozzle flowing freely	turn down sugar supply	some
ITD 9	nozzle opening restricted	switch off sugar supply AND remove nozzle AND replace with spare AND send faulty nozzle for cleaning	some
ITD 10	cooling failed AND water jacket warm to touch	call fitter	major
ITD 11	cooling failed AND pump not vibrating	call electrician	major
ITD 12	cooling failed AND water tap turned off	turn on tap	some
ITD 13	gas temperature too high	turn down oxygen input	some
ITD 14	low temperature gases AND black color	restart override	some
ITD 15	low temperature gases AND flickering light	raise alarm AND evacuate factory	factory shutdown
ITD 16	low temperature gases red color	turn up oxygen supply	some

7.4.3 DECISION TREES

The decision trees knowledge representation (Figure 3) has been used to develop the diagnosis system, because of (1) it provides a visual representation, (2) its structures can be mechanically translated into production rules. [12]

Part of the decision trees which is used to describe the diagnosis processes, is shown in Figure 7.3. There are 5 levels in the decision tree, that is: Trigger, Hypothesis, Fault, Test and Action.

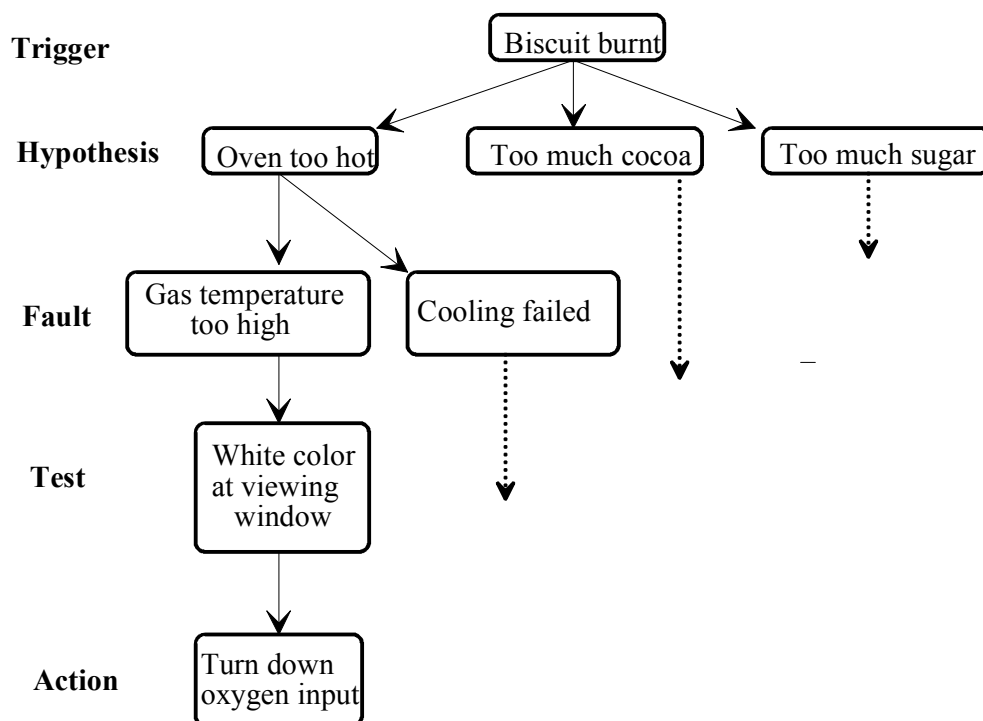


Figure 7.3 Decision tree.

The acquired expertise was summarised in the form of a decision tree from which data could be interpreted to reach conclusions. This decision tree formed the basis of the decision making process in the expert system, enabling the key rules for identifying and locating faults to be developed.

7.4.4 STRUCTURING THE KNOWLEDGE BASE

Using the knowledge base to reach conclusions involves careful systematic search of the decision tree. The decision-making path was carefully planned to ensure that the following requirements were met:

- The system was as fast as possible.
- Data entry errors were detected as early as possible.
- Questions were presented to the user in a logical sequence.
- Intermediate and final conclusions were presented as soon as they were reached.

The resulting knowledge base search procedure is shown in Figures 7.4 to 7.10.

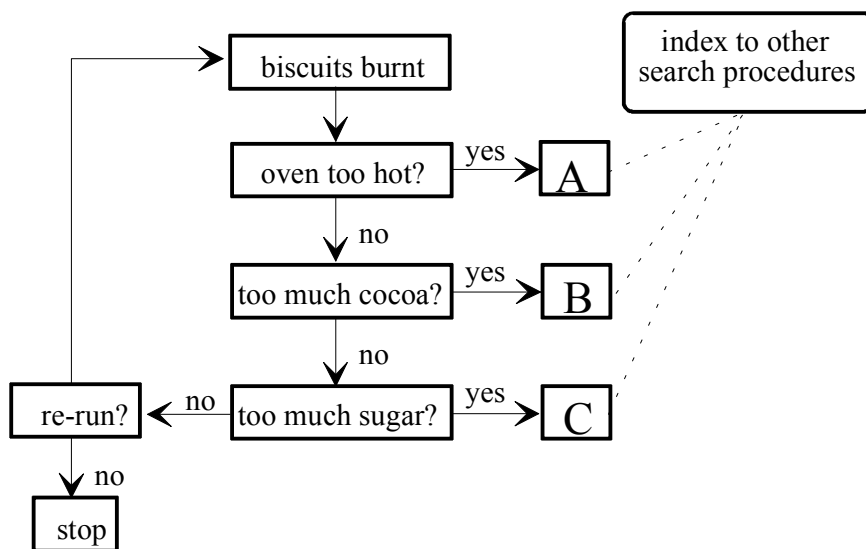


Figure 7.4 Knowledge base search procedure.

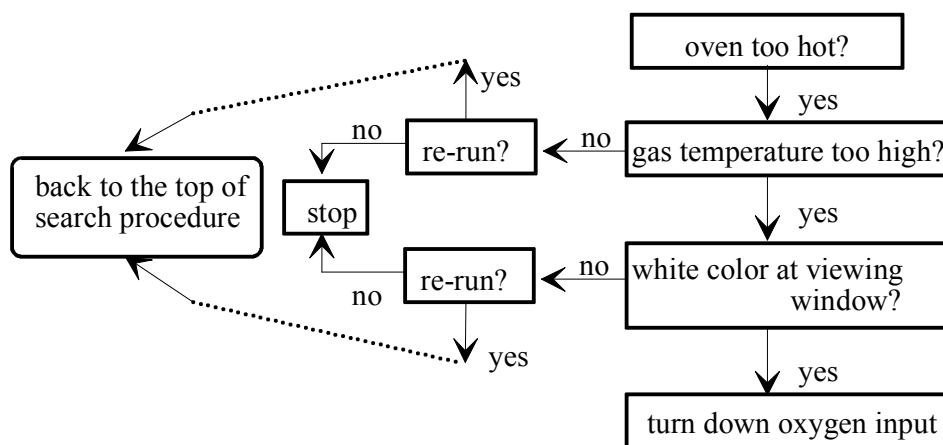


Figure 7.5 Knowledge base search procedure A.

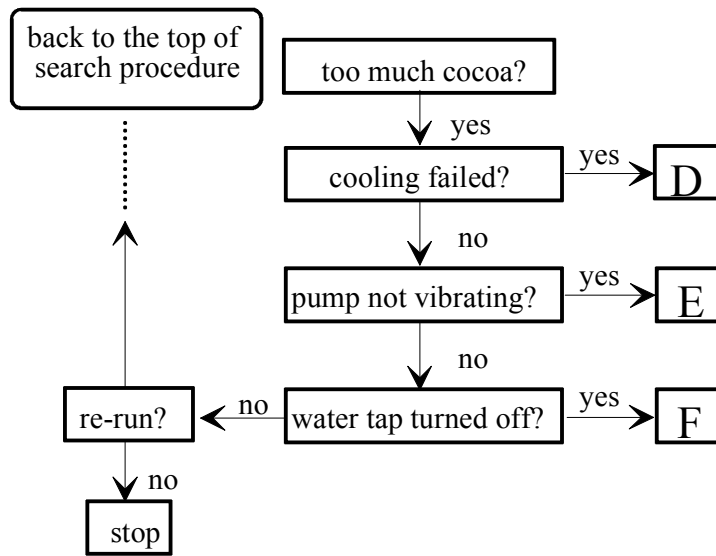


Figure 7.6 Knowledge base search procedure B.

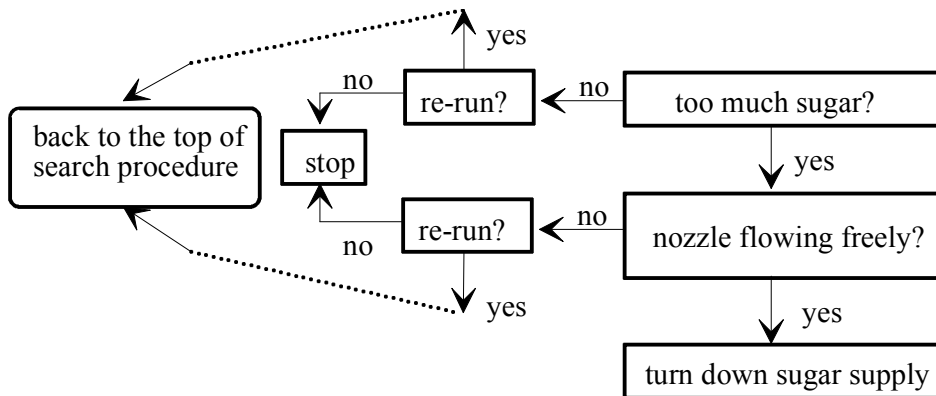


Figure 7.7 Knowledge base search procedure C.

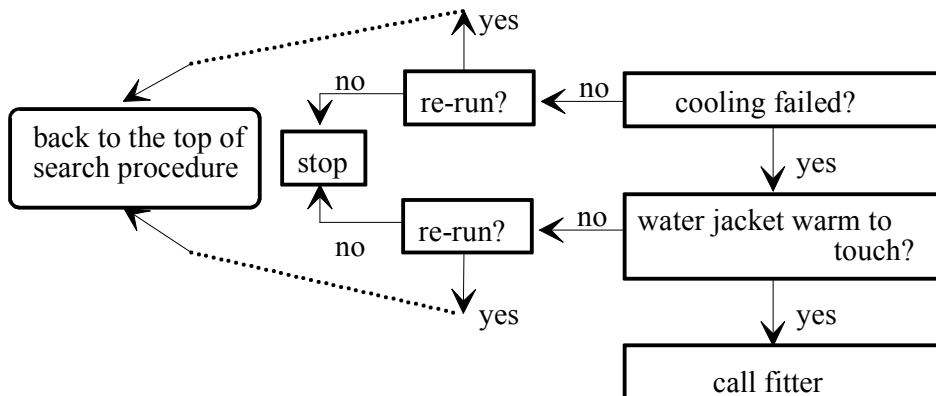


Figure 7.8 Knowledge base search procedure D.

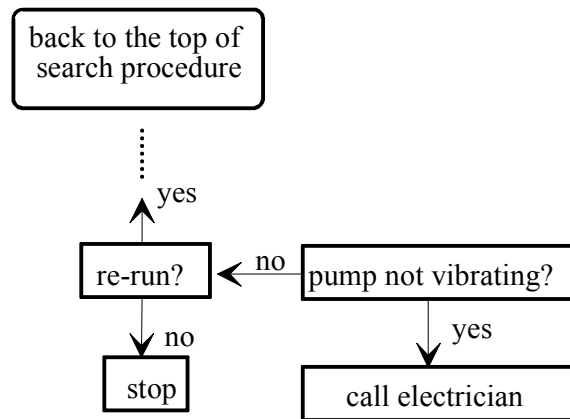


Figure 7.9 Knowledge base search procedure E.

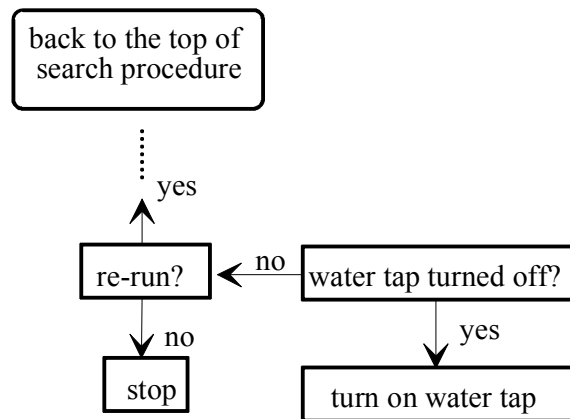


Figure 7.10 Knowledge base search procedure F.

7.5 EXPLANATION FACILITY

The credibility of the expert system is, to some extent, dependent on its ability to explain its reasoning. Explanation facilities can also be used for teaching purposes and for debugging the system. Originally, it was claimed that the expert system could be used by non-experts to help them tackle expert tasks. In practice, most expert system which have been built are used by experts to help them in their work. This is partly due to the poor explanation facilities which are available in most expert systems.

A system, called BLAH, has been developed by Weiner (1980) [13] in an attempt to overcome this problem. A study of explanations used by humans (Weiner, 1979) [14] was used as a basis for the design of BLAH.

The principles which Weiner identified as being important included the following:

1. Explanations should be limited to what is already known by the user.
2. Details should not be given initially.
3. Details should be given in increments.
4. Explanations should be 'marked' in some way so that the underlying structure is more transparent.

7.6 KNOWLEDGE ACQUISITION

Knowledge acquisition is the transfer and transformation of problem-solving expertise from some domain knowledge to a program. In general, the process is one of the most difficult phases of developing an expert system. Fortunately, in the troubleshooting task domain, the knowledge acquisition is quite easier to deal with. At least there are two reasons to support this:

- (1) The faults of equipment are well-defined.
- (2) The use of decision trees captures the expertise in a complete and compact way.

There are three methods which are generally used for obtaining the expert knowledge used in an expert system:

1. Transcription from text - involves all of the problems associated with natural language understanding.
2. Elicitation from experts - involves 'probing' rather than 'mining'. Difficulties have been encountered when experts are regarded as containing chunks of knowledge which can be 'mined' by asking them to write down the rules.
3. Induction from examples.

7.7 BACKWARD CHAINING INFERENCE STRATEGY

Backward chaining inference is usually used in problem diagnosis. In backward chaining, the reasoning process starts from a goal state and backtracks to the paths that have led to the goal. It is also called backward reasoning or goal-driven search. Backward chaining generally is of the form:

Goal State
IF (data condition)

If there is a fault in the chocolate factory, the goal state (hypothesis) is that the oven is too hot. In order to support this hypothesis, you could ask if the biscuit is burnt. If the response is "biscuit burnt", then the hypothesis is proven true and becomes a fact. The hypothesis can then be interpreted as a goal state to be proven. Here we have to pay attention to the fact that backward chaining is often implemented in expert systems in the coding format of a forward chaining rule. In that case, the goal state is specified in the condition part of the rule and the data condition leading to the goal is specified in the conclusion part of the rule. For example, the backward chaining inference can be written in the form of forward chaining in KnowledgePro as the following:

IF biscuit is burnt
THEN gas temperature is too high

To illustrate the usefulness of backward chaining, Figure 7.11 will give a clear understanding. Suppose on the way in to work one day, an operator notices that the weather is unusually warm. This suggests to the alert person that some faults could probably occur because of the heat, and that it might be worth consulting the expert system to find out what to watch out for.

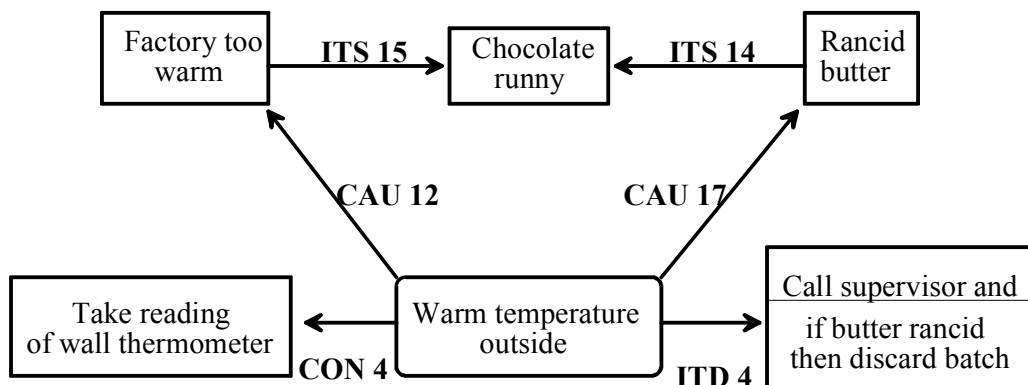


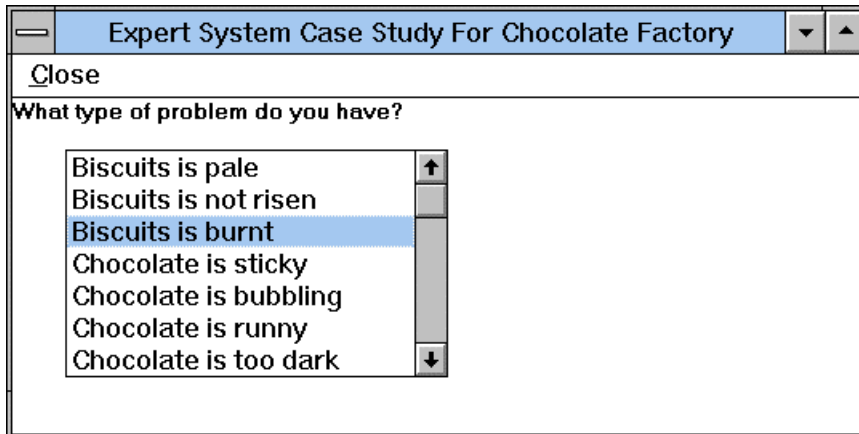
Figure 7.11 Example with backward chaining.

‘Warm temperature outside’ is one of the faults that the system recognises. Rule ITD-4 reasons forward to recommend that the shift supervisor should be advised and the batch discarded if the butter is rancid. This could lead to a major production loss, so it would be a fault worth avoiding if possible. Rules CAU-12 and CAU-17 have ‘warm outside temperature’ as their consequence, so reasoning backwards suggests the hypotheses ‘factory too warm’ and ‘rancid butter’. Reasoning backwards with the ITS rules reveals that the triggers are ‘chocolate runny’ (ITS-14 and ITS-15) and ‘bad smell’ (ITS-18). So, the operator is forewarned to watch out for the chocolate becoming runny or a bad smell developing. Incidentally, reasoning forward with rule CON-4 suggests the test of taking the reading on the wall thermometer. By volunteering the information that the temperature outside is warm and using a mixture of forward and backward reasoning, the user of the expert system is able to discover the seriousness of such a fault developing, and finding out the triggers and which of them should be monitored during the day.

7.8 IMPLEMENTATION

We are using KnowledgePro as a development environment to develop the expert system which is used to diagnose faults in a chocolate factory. KnowledgePro is a high level language because: (1) It is an untyped language; (2) It is interpreted; (3) It has a simple syntax; (4) It does not require the programmer to do any memory management; (4) It provides intelligent defaults whenever possible. Although it is a high level language, it is learned quickly by inexperienced programmers. The strength of the language lies in its flexibility and the power of its combination of object-oriented programming and list processing capabilities.

According to the benefits mentioned above, it allows you to build complex Windows applications rapidly. Samples of the screen display for diagnosis are presented in Figure 7.12 to 7.15.



Error! Switch argument not specified.

Figure 7.12 Consultation window layout screen in KnowledgePro.(cont.)

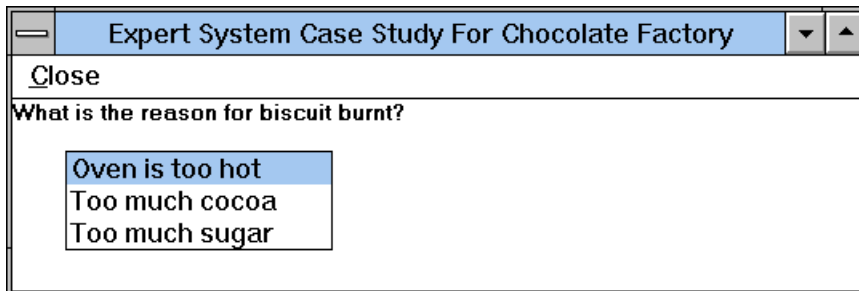
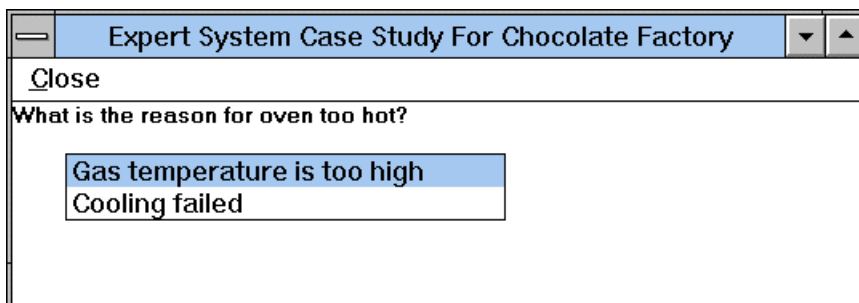


Figure 7.13 Consultation window layout screen in KnowledgePro.(cont.)



Error! Switch argument not specified. Error! Switch argument not specified.

Figure 7.14 Consultation window layout screen in KnowledgePro.(cont.)

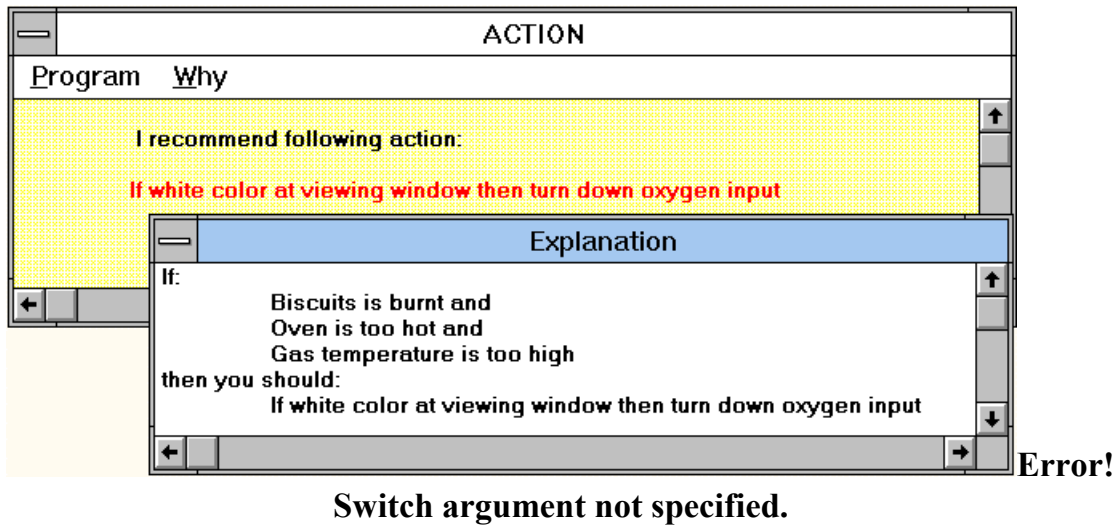


Figure 7.15 Consultation window layout screen in KnowledgePro.(cont.)

7.9 CONCLUSIONS

In this chapter we have looked at an expert system for diagnosing faults in the chocolate factory. The system was constructed with straight forward rule based technology and classic expert system development procedures. By being carefully focused on a well defined application, it is now has a widespread input. Not only are many engineers able to use sophisticated expert system technology, but a major revolution is taking place with regard to how they perform their day to day tasks. The technology is already prompting further development and lessons learned from this application will certainly lead to many new and better capabilities.

The prototype was well accepted by both expert and novice users and was recognized to be of particular value in propagating expertise more widely through their organisation. It will meet the requirements of both teaching and research in university environments and applications building in industry.

The reason why KnowledgePro is chosen as a programming tool to develop an expert system for diagnosing faults at a chocolate factory, is that KnowledgePro combines a number of important technologies into one easy -to-use environment.

7.10 REFERENCES

- [1] Bielawski, L. and Lewand, R., Intelligent systems design - integrating expert systems, hypermedia, and database technologies, John Wiley & Sons, 1991.
- [2] Davis, R., Diagnostic reasoning based upon structure and behavior, *Artif. Intell.*, 24, pp. 347-410, 1984.
- [3] Genesereth, M. R., The use of design descriptions in automated diagnosis, *J. Artif. Intell.* 24, pp. 411-436, 1984.
- [4] Hansen, T., Diagnosing multiple faults using knowledge about malfunctioning behavior, *ACM*, pp. 29-36, 1988.
- [5] Kuipers, B., Qualitative simulation. *Artif. Intell.*, 29, pp. 289-338, 1986.
- [6] Knowledge Garden, KnowledgePro Windows version 2.0 - User manual, Knowledge Garden, Inc. 1991.
- [7] Milne, R., Strategies for diagnosis, *IEEE Trans. Syst. Man Cybernet*, 17, pp. 333-339, 1987.
- [8] Pang, Y. and Reggia, J., A probabilistic causal model for diagnostic problem-solving, *IEEE Trans. Syst: Man Cybernet.* 17, pp. 146-162, 1987.
- [9] Reggia, J., Nau D. and Wang, Y., Diagnostic expert system based on a set covering model, *Int. J. Man-Machine syst.* 19, pp.437-460, 1983.
- [10] Reiter, R. A., Theory of diagnosis from first principle, *Artif. Intell.*, 32, pp. 57-95, 1987.
- [11] Spur, G. and Specht, D., Knowledge-based diagnosis in manufacturing systems, *Manufacturing systems*, Vol. 19, No.2, pp. 179-184, 1990.
- [12] Wang, K., An expert system for fault diagnosis in car engines, Institute Report, NTH, Trondheim, 1993.

APPENDIX 1: LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

(* =====/* AUTOCAD */=====*)

***Comment

Copyright (C) 1986-1992 by Autodesk, Inc.
Version 1.0 for Release 12 (6/1/92)

**Comment

Begin AutoCAD Screen Menus

***SCREEN

**S

[PROCESS]
[PLANNING]
[* * * *]
[= MENU =]
[-----]

[CENTER]^C(load "part") (cenline);
[LINE]^C(line_2);
[HOLE]^C(line_4);
[CURVE]^C(arc_);
[THREAD]^C(line_1);
[KNURL]^C(line_3);
[INTHREAD]^C(line_6);

[MIRROR]^C^C(mirror_);

[MODIFY]^C(modify);
[REDRAW]^CREDRAW;

[OPEN]^COPEN \ (load "part");
[SAVE]^CSAVE;
[Wr *.GDF]^C(feature);

[EXIT]^CQUIT;

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
(* =====/* PART.LSP */=====*)

(setq rr 0)
(setq y0 4.5)

(defun celine()
  (if (= rr 0) (progn
    (command "UCS" "W")
    (command "UCS" "O" "0,4.5,0")
    (command "LINETYPE" "S" "DASHDOT" "" "COLOR" "white")
    (command "LINE" ".5,0" "11.5,0" "")
    (setq y0 4.5)
    (setq xx 0)
  )
  )
)

(defun modify()
  (setq e (car (entsel "Select an object : ")))
  (setq xx (atoi (cdr (assoc 8 (entget e)))))
  (if (> xx 0) (progn
    (entdel e)
    (prompt "\nNext , Please redraw the deleted surface !!!")
  )
  )
)

(defun tol_thr_knu()
  (setq tomin 0)
  (initget 4)
  (setq tomin (getreal "\nThe min tolerance is : "))
  (if (= tomin nil) (setq tomin 0.1))
  (setvar "ELEVATION" tomin)
  (if (= c 3) (progn
    (initget 7)
    (setq tolplus (getint "\nThe knurling turns per-inch are : "))
    (setvar "THICKNESS" tolplus)
  )
  )
  (if (or (= c 1) (= c 6)) (progn
    (initget 7)
    (setq tolplus (getint "\nThe threads per-inch are : "))
    (setvar "THICKNESS" tolplus)
  )
  )
  (if (or (= c 2) (= c 4)) (setvar "THICKNESS" 0))
)

(defun order()
  (if (> xx 0) (progn
    (command "LAYER" "M" xx "")
    (setq xx 0)
  )
  )
)
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
(progn
  (setq rr (+ rr 1))
  (command "LAYER" "M" rr "")
)
)
)

(defun line_type()
  (if (= c 2) (command "LINETYPE" "S" "BYLAYER" "" "COLOR" "yellow"))
  (if (= c 4) (command "LINETYPE" "S" "HIDDEN" "" "COLOR" "cyan"))
  (if (= c 1) (command "LINETYPE" "S" "BYLAYER" "" "COLOR" "red"))
  (if (= c 3) (command "LINETYPE" "S" "BYLAYER" "" "COLOR" "green"))
  (if (= c 6) (command "LINETYPE" "S" "HIDDEN" "" "COLOR" "magenta"))
)

(defun line_(c)
  (initget 1)
  (setq poi1 (getpoint "FROM point : "))
  (initget 1)
  (setq poi2 (getpoint poi1 "TO point : "))
  (if (/= (car poi1) (car poi2)) (tol_thr_knu))
  (order)
  (line_type)
  (command "LINE" poi1 poi2 "")
)

(defun arc_()
  (initget 1)
  (setq poi1 (getpoint "START point : "))
  (initget 1)
  (setq poi2 (getpoint "END point : "))
  (initget 7)
  (setq poi (getreal "RADIUS : "))
  (setq c 2)
  (tol_thr_knu)
  (order)
  (line_type)
  (command "ARC" poi1 "E" poi2 "R" poi)
)

(defun mirror_()
  (setq e (entnext))
  (setq e (entnext e))
  (setq ind 0)
  (while e
    (setq ind (+ 1 ind))
    (setq e (entnext e))
  )
  (setq e (entnext))
  (setq e (entnext e))
  (while (> ind 0)
    (setq l (entget e))
    (setq c (cdr(assoc 62 l)))
  )
)
```


APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
)
(progn
  (setq xc (cadr(assoc 10 l)))
  (setq yc (caddr(assoc 10 l)))
  (setq r (cdr(assoc 40 l)))
  (setq x (cdr(assoc 50 l)))
  (setq y (cdr(assoc 51 l)))
  (setq x1 (+>(* (cos x) r) xc))
  (setq y1 (-(+>(* (sin x) r) yc) y0))
  (setq x2 (+>(* (cos y) r) xc))
  (setq y2 (-(+>(* (sin y) r) yc) y0))
  (setq c (+ 20 c))
)
)
(part (itoa c) 2)
(setq r x)
(part (cdr (assoc 8 l)) 2)
(setq r (strcat r x))
(part (rtos x1 2 4) 7)
(setq r (strcat r x))
(part (rtos y1 2 4) 8)
(setq r (strcat r x))
(part (rtos x2 2 4) 7)
(setq r (strcat r x))
(part (rtos y2 2 4) 8)
(setq r (strcat r x))
(setq r (strcat r " 0.0000"))
(part (rtos tomin 2 4) 6)
(setq r (strcat r x))
(part (itoa toplus) 2)
(setq r (strcat r x))
(write-line r filehand)
(setq e (entnext e))
)
(close filehand)
)

(defun part(s n)
  (setq n (- n (strlen s)))
  (while (> n 0)
    (setq s (strcat " " s))
    (setq n (- n 1)))
  )
  (setq x s)
)
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

(* ===== /* EXTR.PRG */ ===== *)

```
SET TALK OFF
SET COLOR TO R
@9,5 SAY "Reading the Geometric Data File of a part which is saved in AutoCAD"
@11,13 SAY "and extracting feature to *.PDF file in dBASE III !"
@13,24 SAY "Please wait a moment ..."
SET COLOR TO W
CLEAR ALL
USE TEMP%
DELETE ALL
PACK
USE TEMP
DELETE ALL
PACK
APPEND FROM TEMP SDF
REPLACE ALL Y1 WITH 0.0 FOR Y1<0
REPLACE ALL Y2 WITH 0.0 FOR Y2<0
SORT TO TEMP%1 ON SE /A ALL FOR (Y1<0 .OR. Y2<0)
USE TEMP%1
SORT TO TEMP%2 ON Y1 /D ALL
SORT TO TEMP%3 ON Y2 /D ALL
USE TEMP%2
STORE Y1 TO YMAX
USE TEMP%3
IF Y2>YMAX
STORE Y2 TO YMAX
ENDIF
STORE YMAX+.25 TO YMAX
SELECT 2
USE TEMP% ALIAS INF
SELECT 1
USE TEMP%1 ALIAS ORI
COUNT ALL TO N
REPLACE ALL RAD1 WITH YMAX FOR OP<4.OR.OP=22
SORT TO TEMP%4 ON X1 /A,X2 /A ALL
SELECT 3
USE TEMP%4 ALIAS DATA
STORE X1 TO XS
GOTO BOTTOM
STORE X1 TO XE
GOTO TOP
COPY TO TEMP& STRUCTURE FIELD X1,Y1
SELECT 4
USE TEMP& ALIAS XY
APPEND BLANK
REPLACE X1 WITH XE,Y1 WITH 2*YMAX
COPY TO TEMP& SDF
STORE 0 TO I
SELECT 1
GOTO TOP
DO WHILE I<N
STORE TOMIN TO TM
STORE TOLPLUS TO TL
STORE SE TO IND
SELECT 3
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
LOCATE ALL FOR SE=IND
STORE RAD1 TO R
SELECT 1
STORE X1 TO TX1
STORE Y1 TO TY1
STORE X2 TO TX2
STORE Y2 TO TY2
IF (TX1=TX2.AND.TY1<0.AND.TY2<0)
STORE I+1 TO I
SKIP
LOOP
ENDIF
DO CASE
CASE OP=1.OR.OP=3
IF (TY1=TY2)
STORE 1 TO OPERA1
ELSE
STORE 2 TO OPERA1
ENDIF
IF OP=1
STORE 5 TO OPERA2
ELSE
STORE 7 TO OPERA2
ENDIF
DO EXTR1 WITH IND,TX1,TY1,TX2,TY2,R,TM,TL
SELECT 2
APPEND BLANK
IF (TY1>TY2)
STORE R-TY2 TO R
ELSE
STORE R-TY1 TO R
ENDIF
REPLACE OPER1 WITH OPERA1,OPER2 WITH OPERA2,FROMX1 WITH TX1,FROMY1 WITH
TY1,FROMX2 WITH TX2,FROMY2 WITH TY2,DEPTH WITH R,TOMIN WITH TM,TOLPLUS WITH 0
SELECT 1
STORE X1 TO TX1
STORE Y1 TO TY1
STORE X2 TO TX2
STORE Y2 TO TY2
IF OP=1
STORE 5 TO OPERA1
ELSE
STORE 7 TO OPERA1
ENDIF
SELECT 2
APPEND BLANK
REPLACE OPER1 WITH OPERA1,OPER2 WITH 0,FROMX1 WITH TX1,FROMY1 WITH TY1,FROMX2
WITH TX2,FROMY2 WITH TY2,DEPTH WITH 0,TOMIN WITH TM,TOLPLUS WITH TL
CASE OP=2.OR.OP=22
STORE 0 TO OPERA2
IF ((TY1=0.OR.TY2=0).AND.TX1=TX2)
IF SE=1
STORE 3 TO OPERA1
ELSE
STORE 4 TO OPERA1
ENDIF
ELSE
IF (TY1=TY2.OR.OP=22)
```


APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
SELECT 3
LOCATE ALL FOR SE=IND
SKIP -1
DO WHILE OP>3.AND.OP<>22
SKIP -1
ENDDO
STORE 0 TO GR1
IF X1=X2
STORE 1 TO GR1
ENDIF
LOCATE ALL FOR SE=IND
SKIP
DO WHILE OP>3.AND.OP<>22
SKIP
ENDDO
STORE 0 TO GR2
IF X1=X2
STORE 1 TO GR2
ENDIF
SELECT 1
LOCATE FOR SE=IND
IF (OP=22)
IF (GR1=1.AND.GR2=1)
STORE 9 TO OPERA1
ELSE
STORE 12 TO OPERA1
ENDIF
ELSE
IF (GR1=1.AND.GR2=1)
STORE 8 TO OPERA1
ELSE
STORE 1 TO OPERA1
DO EXTR1 WITH IND, TX1, TY1, TX2, TY2, R, TM, TL
ENDIF
ENDIF
ELSE
IF OP=2
STORE 2 TO OPERA1
ENDIF
ENDIF
ENDIF
SELECT 2
APPEND BLANK
IF (TY1>TY2)
STORE R-TY2 TO R
ELSE
STORE R-TY1 TO R
ENDIF
REPLACE OPER1 WITH OPERA1, OPER2 WITH OPERA2, FROMX1 WITH TX1, FROMY1 WITH
TY1, FROMX2 WITH TX2, FROMY2 WITH TY2, DEPTH WITH R, TOMIN WITH TM, TOLPLUS WITH TL
CASE OP=4.OR.OP=6
IF (TX1=TX2.AND.(TY1=0.OR.TY2=0))
STORE I+1 TO I
IF I<N
SKIP
ENDIF
LOOP
ENDIF
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
SELECT 3
IF R=0
LOCATE FOR OP>3.AND.OP<=22
IF X1>X2
STORE X2 TO XS
ELSE
STORE X1 TO XS
ENDIF
CONTINUE
STORE Y1 TO R
DO WHILE (Y1<=0.AND.Y2<=0)
IF Y1<R
STORE Y1 TO R
ENDIF
IF Y2<R
STORE Y2 TO R
ENDIF
CONTINUE
ENDDO
IF X2>X1
STORE X2 TO XE
ELSE
STORE X1 TO XE
ENDIF
IF R>0.4
STORE 0.4 TO R
ENDIF
REPLACE ALL RAD1 WITH R FOR OP>3.AND.OP<=22
SELECT 1
STORE SE TO OPP
REPLACE ALL RAD1 WITH R FOR OP>3.AND.OP<=22
LOCATE FOR SE=OPP
SELECT 2
APPEND BLANK
REPLACE OPER1 WITH 10,OPER2 WITH 0,FROMX1 WITH XS,FROMY1 WITH R,FROMX2 WITH
XE,FROMY2 WITH R,DEPTH WITH XE-XS,TOMIN WITH TM,TOLPLUS WITH 0
ENDIF
IF TX1<=TX2
IF TY1>TY2
STORE TY1-R TO R
ELSE
STORE TY2-R TO R
ENDIF
SELECT 1
STORE OP TO OPP
SELECT 2
IF R<=0
APPEND BLANK
ENDIF
IF OPP=4
IF R<=0
REPLACE OPER1 WITH 11,OPER2 WITH 0,FROMX1 WITH TX1,FROMY1 WITH TY1,FROMX2 WITH
TX2,FROMY2 WITH TY2,DEPTH WITH R,TOMIN WITH TM,TOLPLUS WITH 0
ENDIF
ELSE
IF R<=0
REPLACE OPER1 WITH 11,OPER2 WITH 6,FROMX1 WITH TX1,FROMY1 WITH TY1,FROMX2 WITH
TX2,FROMY2 WITH TY2,DEPTH WITH R,TOMIN WITH TM,TOLPLUS WITH 0
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
ENDIF
APPEND BLANK
REPLACE OPER1 WITH 6,OPER2 WITH 0,FROMX1 WITH TX1,FROMY1 WITH TY1,FROMX2 WITH
TX2,FROMY2 WITH TY2,DEPTH WITH 0,TOMIN WITH TM,TOLPLUS WITH TL
ENDIF
ENDIF
ENDCASE
SELECT 1
LOCATE FOR SE=IND
IF Y1>Y2
IF (OP=4.OR.OP=6)
STORE Y2 TO R
ELSE
STORE Y1 TO R
ENDIF
ELSE
IF (OP=4.OR.OP=6)
STORE Y1 TO R
ELSE
STORE Y2 TO R
ENDIF
ENDIF
SELECT 3
REPLACE RAD1 WITH R FOR SE=IND
STORE I+1 TO I
SELECT 1
IF I<N
SKIP
ENDIF
ENDDO
SELECT 2
COPY TO TEMP% SDF
CLEAR ALL
DELETE FILE TEMP%1.DBF
DELETE FILE TEMP%2.DBF
DELETE FILE TEMP%3.DBF
DELETE FILE TEMP%4.DBF
QUIT
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
(* ===== /* EXTR1.PRG in dBASE III */ ===== *)
**
PROCEDURE EXTR1
PARAMETERS IND,TX1,TY1,TX2,TY2,R,TM,TL
SELECT 3
LOCATE ALL FOR SE=IND
STORE RAD1 TO R
IF TX2>TX1
STORE -1 TO DELTA
ELSE
STORE 1 TO DELTA
ENDIF
STORE 1 TO J
DO WHILE J=1
SKIP DELTA
DO WHILE OP>3.AND.OP<>22
SKIP DELTA
ENDDO
IF (Y1<=TY1.AND.Y2<=TY1.AND.RAD1>TY1)
IF X1>X2
STORE X2 TO TX1
ELSE
STORE X1 TO TX1
ENDIF
REPLACE RAD1 WITH TY1
IF (Y1=0.OR.Y2=0)
EXIT
ENDIF
ELSE
EXIT
ENDIF
ENDDO
LOCATE ALL FOR SE=IND
STORE 1 TO J
DO WHILE J=1
SKIP -DELTA
DO WHILE OP>3.AND.OP<>22
SKIP -DELTA
ENDDO
IF (Y1<=TY2.AND.Y2<=TY2.AND.RAD1>TY2)
IF X2>X1
STORE X2 TO TX2
ELSE
STORE X1 TO TX2
ENDIF
REPLACE RAD1 WITH TY2
IF (Y1=0.OR.Y2=0)
EXIT
ENDIF
ELSE
EXIT
ENDIF
ENDDO
RETURN
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

(* ===== /* PROCESS.KB KnowledgePro Gold */ ===== *)

```

Color is lightyellow.
rgbTextcolor is 16711680.
tInfo is system_info().
Col is first(?tinfo).
Row is element(?tinfo,2).
wCapp is window(quit,1,1,?Col,?Row-3,'Computer-Aided Process Planning',
  [popup,minimizebox,maximizebox,controlmenu,thickframe,titlebar,
  horzscroll,vertscroll,showchildren],,?rgbTextcolor,?Color).
Cappmenu is menu([[&File,&Open...,&Close,&Save,'Save &As...",&Delete,&Print,
  [],'E&xit Kp'],&Autocad,&Extract,&Process,&Help],menutopic).
disable_menu_item(?Cappmenu,[&Close,&Save,'Save &As...",&Delete,&Print]).
Cwidth is element(window_info(?wCapp),10).
Cheight is element(window_info(?wCapp),11).

show_window(?wCapp).

topic quit.
  exit_kp().
end.

topic menutopic(item).
  do(?item).
end.

topic '&Open...'.
  File is [].
  File is file_menu('C:\CAPP','Open',['GDF FILE (*.GDF)',*.GDF,'PDF FILE (*.PDF)',*.PDF,'PPF FILE (*.PPF)'],*.PPF)].
  if ?File is [] then
    exit().
  test is element(string_to_list(?File,' '),2).
  if ?test <> gdf and
    ?test <> pdf and
    ?test <> ppf then
    msg is 'File Type is Error!' and
    title = 'CAPP ERROR INFORMATION' and
    ok_button is 1 and
    lib is load_library('USER.EXE') and
    return is call(?lib,MessageBox,[word(?wCapp),lpstr(msg),lpstr(title),
      word(?ok_button)],int) and
    disable_window(?wCapp) and
    (if exists(return) then
      free_library(?lib) and
      enable_window(?wCapp) and
      exit()).
  Cappfile is read(?File).
  wEdit is edit_window(?CappFile,,1,1,?Cwidth - 15,?Cheight -
5,?File,[child,thickframe,horzscroll,vertscroll,minimizebox,
  maximizebox,siblings],?wCapp).
  disable_menu_item(?Cappmenu,'&Open...').
  enable_menu_item(?Cappmenu,[&Close,&Save,'Save &As...",&Delete,&Print]).
  show_window(?wEdit).
end.

topic &Close.
  wClose is window(,30,10,40,5,'Close',[popup,dialogframe,titlebar]).
  bu1 is button(SAVE,yessave,4,2,8).
  bu2 is button(NOT,notsave,15,2,8).

```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
bu3 is button(CANCEL,cancel,26,2,8).
disable_window(?wCapp) .
disable_window(?wEdit) .
show_window(?wClose).
```

```
topic yessave.
  save_edit_file(?wEdit).
  notsave().
end.
```

```
topic notsave.
  cancel().
  close_window(?wEdit).
  close(?File).
  disable_menu_item(?Cappmenu,['&Close,&Save','Save &As...','&Delete,&Print]).
  enable_menu_item(?Cappmenu,'&Open...').
end.
```

```
topic cancel.
  if exists(wClose) then
    close_window(?wClose).
    enable_window(?wCapp) .
    enable_window(?wEdit) .
    set_active_window(?wCapp).
  end.
end.
```

```
topic &Save.
  save_edit_file(?wEdit).
end.
```

```
topic 'Save &As...'.
  FileText is get_text(?wEdit).
  Savefile is save_as(?File).
  if not(exists(Savefile)) then
    exit().
  new_file(?Savefile).
  write(?Savefile,?FileText).
  close(?Savefile).
end.
```

```
topic &Delete.
  wDelete is window(,30,10,40,5,'Delete',[popup,dialogframe,titlebar]).
  bu1 is button(DELETE,yesdel,10,2,10).
  bu2 is button(CANCEL,notdel,25,2,10).
  disable_window(?wCapp) .
  disable_window(?wEdit) .
  show_window(?wDelete).
```

```
topic yesdel.
  notdel().
  close_window(?wEdit).
  close(?File).
  delete_file(?File).
  disable_menu_item(?Cappmenu,['&Close,&Save','Save &As...','&Delete,&Print]).
  enable_menu_item(?Cappmenu,'&Open...').
end.
```

```
topic notdel.
  if exists(wDelete) then
    close_window(?wDelete).
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
enable_window(?wCapp) .
enable_window(?wEdit) .
set_active_window(?wCapp).
end.
end.

topic &Print.
FileText is get_text(?wEdit).
print(?FileText).
end.

topic 'E&xit Kp'.
exit_kp().
end.

topic &Autocad.
run('C:\CAPP\ACAD.PIF').
end.

topic &Extract.
ExtractFile is [].
ExtractFile is file_menu(*.GDF,'C:\CAPP','Extract',['GDF FILE (*.GDF)',*.GDF]).
if ?ExtractFile is [] then
  exit().
test is element(string_to_list(?ExtractFile,' '),2).
if ?test <> gdf then
  msg is 'File Type is Error!' and
  title = 'CAPP ERROR INFORMATION' and
  ok_button is 1 and
  lib is load_library('USER.EXE') and
  return is call(?lib,Messagebox,[word(?wCapp),lpstr(msg),lpstr(title),
  word(?ok_button)],int) and
  disable_window(?wCapp) and
  (if exists(wEdit) then
    disable_window(?wEdit) and
  (if exists(return) then
    free_library(?lib) and
    enable_window(?wCapp) and
    (if exists(wEdit) then
      enable_window(?wEdit) and
      exit()).
  use_mouse_cursor(?WCapp,WAIT).
  NamePath is first(string_to_list(?ExtractFile,' ')).
  FileText is read(?ExtractFile).
  close(?ExtractFile).
  new_file('c:\dbase\temp.txt').
  write('c:\dbase\temp.txt',?FileText).
  write('c:\dbase\temp.txt','#n ').
  close('c:\dbase\temp.txt').
  run('C:\CAPP\DATABASE.PIF').
  FileText is read('c:\dbase\temp%.txt').
  FileName is concat(?NamePath,'.pdf').
  new_file(?FileName).
  write(?FileName,?FileText).
  close(?FileName).
  close('c:\dbase\temp%.txt').
  delete_file('c:\dbase\temp%.txt').
  FileText is read('c:\dbase\temp&.txt').
  FileName is concat(?NamePath,'.p%').
  new_file(?FileName).
  write(?FileName,?FileText).
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```

close(?FileName).
close('c:\dbase\temp&.txt').
use_mouse_cursor(?WCapp,ARROW).
end. (* extract(). *)

topic &Process.
ProcessFile is [].
ProcessFile is file_menu(*.PDF,'C:\CAPP','Process',['PDF FILE (*.PDF)',*.PDF]).
if ?ProcessFile is [] then
  exit().
test is element(string_to_list(?ProcessFile,!),2).
if ?test <>pdf then
  msg is 'File Type is Error!' and
  title = 'CAPP ERROR INFORMATION' and
  ok_button is 1 and
  lib is load_library('USER.EXE') and
  return is call(?lib,Messagebox,[word(?wCapp),lpstr(msg),lpstr(title),
    word(?ok_button)],int) and
  disable_window(?wCapp) and
  (if exists(wEdit) then
    disable_window(?wEdit) and
  (if exists(return) then
    free_library(?lib) and
    enable_window(?wCapp) and
    (if exists(wEdit) then
      enable_window(?wEdit) and
      exit()).
    select_text(?wCapp,1,1,80,200).
    delete_selected_text(?wCapp).
    step=1.
    Wmater is window(,5,5,50,10,'SELECT RAW MATERIAL',[Popup,Visible,
      DialogFrame,Titlebar,Siblings,Showchildren],?wCapp).
    MaterMenu is menu([[&Steel,free_cutting_steel,[],low_alloy_steel,
      medium_alloy_steel,high_alloy_steel,[],
      tungstan_steel],[&Alloy,copper_alloy,
      Al_alloy,Mg_alloy],[&Iron,cast_iron]],menuTopic).
    if ?menuTopic is []
    then exit().
end. (* Process *)

topic pro_plan.
use_mouse_cursor(?WCapp,WAIT).
FileName is concat(first(string_to_list(?ProcessFile,!)),!%p%).
XYText is read_line(?FileName,1).
close(?FileName).
XYList is list_of_char(?XYText).
length is list_to_string(sublist(?XYList,1,7)).
dir is list_to_string(sublist(?XYList,8,7)).
text(#y1#s#n#n INSTRUCTIONS #n',
  '#n1) The raw material of the part is ',?material,
  '#n2) Take the raw material with',
  '#n Diameter= ',?dir,' inches',
  '#n Length= ',?length,' inches',
  '#n3) Hold the job in the lathe head stock chuck',
  '#n4) Hold the second part of the job in the ',
  '#n lathe tail stock for all operations except',
  '#n the internal operations',
  '#n5) For holding the job in the tails tock use',
  '#n countersunk drill #n#n#n').
FileText is read_line(?ProcessFile,1).
while ?FileText <> number_to_char(26) then

```


APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
    plan(?FileText) and
    step=?step+1 and
    FileText is read_line(?ProcessFile,1).
close(?ProcessFile).
Capptext is get_text(?WCapp).
FileName is concat(first(string_to_list(?ProcessFile,')'),'.ppf').
new_file(?FileName).
write(?FileName,?Capptext).
close(?FileName).
use_mouse_cursor(?WCapp,ARROW).
end. (* pro_plan() *)
```

```
topic menuTopic(item).
    do (?item).
end.
```

```
topic free_cutting_steel.
    material is 'free_cutting_steel'.
    close_window(?Wmater).
    pro_plan().
end.
```

```
topic low_alloy_steel.
    material is 'low_alloy_steel'.
    close_window(?Wmater).
    pro_plan().
end.
```

```
topic medium_alloy_steel.
    material is 'medium_alloy_steel'.
    close_window(?Wmater).
    pro_plan().
end.
```

```
topic high_alloy_steel.
    material is 'high_alloy_steel'.
    close_window(?Wmater).
    pro_plan().
end.
```

```
topic tungstan_steel.
    material is 'tungstan_steel'.
    close_window(?Wmater).
    pro_plan().
end.
```

```
topic copper_alloy.
    material is 'copper_alloy'.
    close_window(?Wmater).
    pro_plan().
end.
```

```
topic Al_alloy.
    material is 'Al_alloy'.
    close_window(?Wmater).
    pro_plan().
end.
```

```
topic Mg_alloy.
    material is 'Mg_alloy'.
    close_window(?Wmater).
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
    pro_plan().
end.

topic cast_iron.
    material is 'cast_iron'.
    close_window(?Wmater).
    pro_plan().
end.

topic plan(LineText).
    reset([oper1,oper2]).
    reset([tool,tool1,tool2]).
    reset([speed,feed]).
    reset([roughcut,roughpass]).
    reset(coolant).
    reset(done).
    LineList is list_of_char(?LineText).
    op1 is list_to_string(sublist(?LineList,1,2)).
    op2 is list_to_string(sublist(?LineList,3,2)).
    fromx is list_to_string(sublist(?LineList,5,7)).
    fromy is list_to_string(sublist(?LineList,12,7)).
    tox is list_to_string(sublist(?LineList,19,7)).
    toy is list_to_string(sublist(?LineList,26,7)).
    depth is list_to_string(sublist(?LineList,33,7)).
    tomin is list_to_string(sublist(?LineList,40,6)).
    tolplus is list_to_string(sublist(?LineList,46,2)).
    xy=(fromy+toy)/2.
    if ?op1=1 then oper1 is 'turning'.
    if ?op1=2 then oper1 is 'taperturning'.
    if ?op1=3 then oper1 is 'facing'.
    if ?op1=4 then oper1 is 'parting'.
    if ?op1=5 then oper1 is 'threading'.
    if ?op1=6 then oper1 is 'inthreading'.
    if ?op1=7 then oper1 is 'knurling'.
    if ?op1=8 then oper1 is 'grooving'.
    if ?op1=9 then oper1 is 'forming'.
    if ?op1=10 then oper1 is 'drilling'.
    if ?op1=11 then oper1 is 'boring'.
    if ?op1=12 then oper1 is 'filleting'.
    if ?op2=0 then oper2 is 'no'.
    if ?op2=5 then oper2 is 'threading'.
    if ?op2=6 then oper2 is 'inthreading'.
    if ?op2=7 then oper2 is 'knurling'.
    ProcessPlan().
end.(* Process Planning *)

topic &Help.
    wHelp is window(,?Cwidth/2+1,1,?Cwidth/2,?Cheight - 3,
    'CAPP HELP',[child,visible,thickframe,controlmenu,horzscroll,vertscroll,
    siblings,titlebar,minimizebox,maximizebox],?wCapp) .
    HelpFile is read('c:\capp\capp.hlp').
    text (?HelpFile).
    close('c:\capp\capp.hlp').
end.

topic ProcessPlan.
    set_number_of_values(tool,1).
    if ?oper1=turning
    or ?oper1=taperturning
    and ?oper2=no
    then tool='single_point' and
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
toolmat='carbide' and  
done='yes' and  
turn_or_taper().
```

```
if ?oper1=turning  
or ?oper1=tapeturning  
and ?oper2=threading  
then tool1='single_point' and  
tool2='v_tool' and  
toolmat='carbide' and  
done='yes' and  
turn_thread().
```

```
if ?oper1=boring  
and ?oper2=inthreading  
then tool1='boring_tool' and  
tool2='boring_v_tool' and  
toolmat='carbide_bit' and  
done='yes' and  
bore_inthread().
```

```
if ?oper1=inthreading  
and ?oper2=no  
then tool='boring_v_tool' and  
toolmat='carbide_bit' and  
done='yes' and  
thread().
```

```
if ?oper1=turning  
or ?oper1=tapeturning  
and ?oper2=knurling  
then tool1='single_point' and  
tool2='knurling_tool' and  
toolmat='carbide' and  
done='yes' and  
turn_knurl().
```

```
if ?oper1=parting  
or ?oper1=grooving  
then tool='parting_tool' and  
toolmat='carbide' and  
done='yes' and  
part().
```

```
if ?oper1=drilling  
then tool='twist_drill' and  
toolmat='HSS' and  
done='yes' and  
drill().
```

```
if ?oper1=filleting  
or ?oper1=forming  
then tool='form_tool' and  
toolmat='carbide' and  
done='yes' and  
form().
```

```
if ?oper1=boring  
and ?oper2=no  
then tool='boring_tool' and
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
toolmat='carbide_bit' and
done='yes' and
bore().

if ?oper1=facing
then tool='parting_tool' and
toolmat='carbide' and
done='yes' and
part().

if ?oper1=threading
then tool='v_tool' and
toolmat='carbide' and
done='yes' and
thread().

if ?oper1=knurling
then tool='knurling_tool' and
toolmat='carbide' and
done='yes' and
knurl().
end. (* done *)

topic thr_speed.
set_number_of_values(rp1,1).
if ?material=free_cutting_steel
then rp1=700 and
sf1=50 and
rp2=400 and
sf2=45 and
rp3=270 and
sf3=40 and
rp4=150 and
sf4=35 and
rp5=80 and
sf5=30 and
rp6=40 and
sf6=25 and
rp7=26 and
sf7=20.

if ?material=low_alloy_steel
or ?material=medium_alloy_steel
or ?material=high_alloy_steel
or ?material=tungstan_steel
then rp1=275 and
sf1=20 and
rp2=160 and
sf2=18 and
rp3=100 and
sf3=15 and
rp4=50 and
sf4=12 and
rp5=30 and
sf5=10 and
rp6=15 and
sf6=8 and
rp7=10 and
sf7=8.

if ?material=cast_iron
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

then rp1=850 and
sf1=60 and
rp2=500 and
sf2=55 and
rp3=340 and
sf3=50 and
rp4=200 and
sf4=45 and
rp5=100 and
sf5=40 and
rp6=50 and
sf6=35 and
rp7=38 and
sf7=30.

if ?material=copper_alloy
then rp1=975 and
sf1=70 and
rp2=600 and
sf2=65 and
rp3=400 and
sf3=60 and
rp4=250 and
sf4=55 and
rp5=130 and
sf5=50 and
rp6=70 and
sf6=45 and
rp7=51 and
sf7=40.

if ?material=Al_alloy
or ?material=Mg_alloy
then rp1=1100 and
sf1=80 and
rp2=700 and
sf2=75 and
rp3=475 and
sf3=70 and
rp4=280 and
sf4=65 and
rp5=150 and
sf5=60 and
rp6=100 and
sf6=55 and
rp7=64 and
sf7=50.

end. (* thr_speed *)

topic drill_pa.
set_number_of_values(feed2,1).
if ?material=free_cutting_steel
or ?material=tungstan_steel
then speed2=50 and
feed2=0.010 and
coolant='lard_or_soluble_oil' and
point=125 and
lip='10_12' and
chisel='125_135' and
helix='24_32'.

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
if ?material=low_alloy_steel
or ?material=medium_alloy_steel
then speed2='110_120' and
feed2=0.020 and
coolant='soluble_oil' and
point=118 and
lip='12_15' and
chisel='125_135' and
helix='24_32'.

if ?material=high_alloy_steel
then speed2=80 and
feed2=0.010 and
coolant='soluble_oil' and
point=118 and
lip='12_15' and
chisel='125_135' and
helix='24_32'.

if ?material=cast_iron
then speed2='80_100' and
feed2=0.020 and
coolant='none' and
point='90_100' and
lip=12 and
chisel='125_135' and
helix='24_32'.

if ?material=copper_alloy
then speed2=200 and
feed2='0.003_0.020' and
coolant='none' and
point=118 and
lip='12_15' and
chisel=0 and
helix='10_22'.

if ?material=Mg_alloy
then speed2='300_2000' and
feed2='0.004_0.050' and
coolant='none' and
point='70_118' and
lip=12 and
chisel='125_135' and
helix='10_30'.

if ?material=Al_alloy
then speed2=60 and
feed2=0.010 and
coolant='0.66_lard_and_0.33oil_and_kerosene' and
point=118 and
lip='7_10' and
chisel='120_130' and
helix='15_30'.
end. (* drill_pa *)

topic cut_speed.
set_number_of_values(sp1,1).
if ?material=free_cutting_steel
then sp1='175_to_350' and
sp2='350_to_450' and
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

sp3='450_to_600' and
sp4='600_to_750' and
sp5='750_to_1500'.

if ?material=low_alloy_steel
and ?tool=single_point
or ?tool1=single_point
then name='yes' and
sp1='150_to_300' and
sp2='300_to_400' and
sp3='400_to_550' and
sp4='550_to_700' and
sp5='700_to_1200'.

if ?material=medium_alloy_steel
then sp1='125_to_250' and
sp2='250_to_350' and
sp3='350_to_450' and
sp4='450_to_600' and
sp5='600_to_1000'.

if ?material=high_alloy_steel
then sp1='100_to_200' and
sp2='200_to_300' and
sp3='300_to_400' and
sp4='400_to_500' and
sp5='500_to_750'.

if ?material=cast_iron
then sp1='100_to_200' and
sp2='200_to_250' and
sp3='250_to_350' and
sp4='350_to_450' and
sp5='450_to_600'.

if ?material=tungstan_steel
then sp1='50_to_150' and
sp2='150_to_200' and
sp3='200_to_250' and
sp4='250_to_325' and
sp5='325_to_400'.

if ?material=copper_alloy
then sp1='300_to_500' and
sp2='500_to_650' and
sp3='650_to_800' and
sp4='800_to_1000' and
sp5='1000_to_1250'.

if ?material=Mg_alloy
then sp1='300_to_500' and
sp2='500_to_600' and
sp3='600_to_800' and
sp4='800_to_1250' and
sp5='1250_to_2000'.

if ?material=Al_alloy
then sp1='100_to_200' and
sp2='200_to_300' and
sp3='300_to_450' and
sp4='450_to_700' and

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
sp5='700_to_1000'.
end. (* cut_speed *)

topic bore_speed.
  set_number_of_values(s1,1).
  if ?material=free_cutting_steel
  then s1='275_to_475' and
  f1='0.010_to_0.020' and
  s2='350_to_625' and
  f2='0.008_to_0.012' and
  s3='500_to_1500' and
  f3='0.002_to_0.008'.

  if ?material=low_alloy_steel
  or ?material=medium_alloy_steel
  or ?material=high_alloy_steel
  then s1='200_to_475' and
  f1='0.010_to_0.020' and
  s2='300_to_675' and
  f2='0.008_to_0.012' and
  s3='500_to_1500' and
  f3='0.002_to_0.008'.

  if ?material=cast_iron
  then s1=250 and
  f1='0.010_to_0.020' and
  s2=275 and
  f2='0.008_to_0.012' and
  s3=300 and
  f3='0.002_to_0.008'.

  if ?material=tungstan_steel
  then s1='150_to_350' and
  f1='0.010_to_0.020' and
  s2='250_to_450' and
  f2='0.008_to_0.012' and
  s3='300_to_800' and
  f3='0.002_to_0.008'.

  if ?material=copper_alloy
  then s1='200_to_500' and
  f1='0.010_to_0.020' and
  s2='300_to_600' and
  f2='0.008_to_0.012' and
  s3='400_to_1000' and
  f3='0.002_to_0.008'.

  if ?material=Al_alloy
  or ?material=Mg_alloy
  then s1='250_to_600' and
  f1='0.010_to_0.020' and
  s2='300_to_1000' and
  f2='0.008_to_0.012' and
  s3='400_to_1500' and
  f3='0.002_to_0.008'.
end. (* bore_speed *)

topic depth_roughpass.
  if ?tomin<=0.005
  then depth2=?depth-0.005 and
  finishcut=0.005 and
```


APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
feed1='0.004_to_0.005' and
speed1=?sp5 and
finishpass=1.

if ?tomin>0.005
then depth2=?depth.

if ?depth2<0.5
then depth1=?depth2 and
roughpass=1.

if ?depth2>=0.5
and ?depth2<=1.0
then depth1=?depth2/2 and
roughpass=2.

if ?depth2>1.0
and ?depth2<=1.5
then depth1=?depth2/3 and
roughpass=3.

if ?depth2>1.5
and ?depth2<=2.0
then depth1=?depth2/4 and
roughpass=4.

if ?depth2>2.0
and ?depth2<=2.5
then depth1=?depth2/5 and
roughpass=5.

if ?depth2>2.5
and ?depth2<=3.0
then depth1=?depth2/6 and
roughpass=6.
end. (* depth_roughpass *)

topic feed1_speed1.
set_number_of_values(feed1,1).
if ?xy<=0.3125
then speed1=?rp1 and
feed1=?sf1.

if ?xy>0.3125
and ?xy<=0.4375
then speed1=?rp2 and
feed1=?sf2.

if ?xy>0.4375
and ?xy<=0.625
then speed1=?rp3 and
feed1=?sf3.

if ?xy>0.625
and ?xy<=1.125
then speed1=?rp4 and
feed1=?sf4.

if ?xy>1.125
and ?xy<=1.75
then speed1=?rp5 and
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
feed1=?sf5.

if ?xy>1.75
and ?xy<=2.75
then speed1=?rp6 and
feed1=?sf6.

if ?xy>2.75
then speed1=?rp7 and
feed1=?sf7.
end. (* feed1_speed1 *)

topic feed_speed.
if ?depth1>0.375
then feed='0.060_to_0.18' and
speed=?sp1 and
roughcut=?depth1.

if ?depth1<=0.375
and ?depth1>0.187
then feed='0.060_to_0.18' and
speed=?sp2 and
roughcut=?depth1.

if ?depth1<=0.187
and ?depth1>0.094
then feed='0.030_to_0.060' and
speed=?sp3 and
roughcut=?depth1.

if ?depth1<=0.094
and ?depth1>0.015
then feed='0.010_to_0.030' and
speed=?sp4 and
roughcut=?depth1.

if ?depth1<=0.015
and ?depth1>0.005
then feed='0.008_to_0.010' and
speed=?sp5 and
roughcut=?depth1.
end. (* feed_speed *)

topic feed_speed_b.
if ?depth1>0.15625
then speed=?s1 and
feed=?f1 and
roughcut=?depth1.

if ?depth1<=0.15625
and ?depth1>0.0625
then speed=?s2 and
feed=?f2 and
roughcut=?depth1.

if ?depth1<=0.0625
then speed=?s3 and
feed=?f3 and
roughcut=?depth1.
end. (* feed_speed_b *)
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
topic coolant.
  set_number_of_values(coolant,1).
  if ?material=free_cutting_steel
  then coolant='water_miscible_dry'.

  if ?material=low_alloy_steel
  or ?material=medium_alloy_steel
  then coolant='water_miscible_dry'.

  if ?material=high_alloy_steel
  or ?material=tungstan_steel
  then coolant='HD_water_miscible'.

  if ?material=cast_iron
  then coolant='GP_water_miscible_dry'.

  if ?material=copper_alloy
  then coolant='water_miscible'.

  if ?material=Mg_alloy
  then coolant='LD_cutting_oil_or_speciality_fluids'.

  if ?material=Al_alloy
  then coolant='water_miscible_dry'.
end. (* coolant *)

topic turn_or_taper.
  cut_speed().
  depth_roughpass().
  feed_speed().
  coolant().
  text('#s##nSTEP ',?step,
    '#nThe operation is :#x35',?oper1,
    '#nThe tool is :#x35',?tool,
    '#nThe tool material is :#x35',?toolmat,
    '#nThe total depth of cut is :#x35',?depth,
    '#nThe rough cut is :#x35',?roughcut,
    '#nThe number of roughpasses are :#x35',?roughpass,
    '#nThe speed is :#x35',?speed,
    '#nThe feed is :#x35',?feed,
    '#nThe coolant to be used is :#x35',?coolant,
    '#n',?oper1,' from :#x35 x=',?fromx,' y=',?fromy,
    '#n to :#x35 x=',?tox,' y=',?toy).
end. (* turn_or_taper *)

topic turn_thread.
  cut_speed().
  depth_roughpass().
  feed_speed().
  coolant().
  text('#s##nSTEP ',?step,
    '#nThe operation is :#x35',?oper1,
    '#nThe tool is :#x35',?tool1,
    '#nThe tool material is :#x35',?toolmat,
    '#nThe total depth of cut is :#x35',?depth,
    '#nThe rough cut is :#x35',?roughcut,
    '#nThe number of roughpasses are :#x35',?roughpass,
    '#nThe speed is :#x35',?speed,
    '#nThe feed is :#x35',?feed,
    '#nThe coolant to be used is :#x35',?coolant,
    '#n',?oper1,' from :#x35 x=',?fromx,' y=',?fromy,
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
'#n to :#x35 x=?,?tox, 'y=?,?toy).
end. (* turn_thread *)

topic bore_inthread.
  bore_speed().
  depth_roughpass().
  feed_speed_b().
  coolant().
  text('#s##n#nSTEP ',?step,
    '#nThe operation is :#x35',?oper1,
    '#nThe tool is :#x35',?tool1,
    '#nThe tool material is :#x35',?toolmat,
    '#nThe total depth of cut is :#x35',?depth,
    '#nThe rough cut is :#x35',?roughcut,
    '#nThe number of roughpasses are :#x35',?roughpass,
    '#nThe speed is :#x35',?speed,
    '#nThe feed is :#x35',?feed,
    '#nThe coolant to be used is :#x35',?coolant,
    '#n',?oper1,' from :#x35 x=?,?fromx,'y=?,?fromy,
    '#n to :#x35 x=?,?tox, 'y=?,?toy).
end. (* bore_inthread *)

topic thread.
  thr_speed().
  coolant().
  feed1_speed1().
  text('#s##n#nSTEP ',?step,
    '#nThe operation is :#x35',?oper1,
    '#nThe tool is :#x35',?tool,
    '#nThe tool material is :#x35',?toolmat,
    '#nThe threads per inch are :#x35',?tolplus,
    '#nThe speed in rpm is :#x35',?speed1,
    '#nThe surface speed is :#x35',?feed1,
    '#nThe coolant to be used is :#x35',?coolant,
    '#n',?oper1,' from :#x35 x=?,?fromx,'y=?,?fromy,
    '#n to :#x35 x=?,?tox, 'y=?,?toy).
end. (* thread *)

topic turn_knurl.
  cut_speed().
  depth_roughpass().
  coolant().
  feed_speed().
  text('#s##n#nSTEP ',?step,
    '#nThe operation is :#x35',?oper1,
    '#nThe tool is :#x35',?tool1,
    '#nThe tool material is :#x35',?toolmat,
    '#nThe total depth of cut is :#x35',?depth,
    '#nThe rough cut is :#x35',?roughcut,
    '#nThe number of roughpasses are :#x35',?roughpass,
    '#nThe speed is :#x35',?speed,
    '#nThe feed is :#x35',?feed,
    '#nThe coolant to be used is :#x35',?coolant,
    '#n',?oper1,' from :#x35 x=?,?fromx,'y=?,?fromy,
    '#n to :#x35 x=?,?tox, 'y=?,?toy).
end. (* turn_knurl *)

topic part.
  coolant().
  text('#s##n#nSTEP ',?step,
    '#nThe operation is :#x35',?oper1,
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
'#nThe tool is :#x35',?tool,
'#nThe tool material is :#x35',?toolmat,
'#nThe coolant to be used is :#x35',?coolant,
'#n',?oper1,' from :#x35 x=',?fromx,' y=',?fromy,
'#n to :#x35 x=',?tox,' y=',?toy).
end. (* part *)
```

```
topic drill.
drill_pa().
text('#s#n#nSTEP ',?step,
'#nThe operation is :#x35',?oper1,
'#nThe tool is :#x35',?tool,
'#nThe tool material is :#x35',?toolmat,
'#nThe tool geometry :',
'#nPoint angle :#x35',?point,
'#nLip clearance angle :#x35',?lip,
'#nChisel point angle :#x35',?chisel,
'#nHelix angle :#x35',?helix,
'#nThe lubricant to be used :#x35',?coolant,
'#nThe speed(sfp) is :#x35',?speed2,
'#nThe feed(ipm) is :#x35',?feed2,
'#n',?oper1,' from :#x35 x=',?fromx,' y=',?fromy,
'#n to :#x35 x=',?tox,' y=',?toy).
end. (* drill *)
```

```
topic form.
coolant().
text('#s#n#nSTEP ',?step,
'#nThe operation is :#x35',?oper1,
'#nThe tool is :#x35',?tool,
'#nThe tool material is :#x35',?toolmat,
'#nThe coolant to be used is :#x35',?coolant,
'#n',?oper1,' from :#x35 x=',?fromx,' y=',?fromy,
'#n to :#x35 x=',?tox,' y=',?toy).
end. (* form *)
```

```
topic bore.
bore_speed().
depth_roughpass().
feed_speed_b().
coolant().
text('#s#n#nSTEP ',?step,
'#nThe operation is :#x35',?oper1,
'#nThe tool is :#x35',?tool,
'#nThe tool material is :#x35',?toolmat,
'#nThe total depth of cut is :#x35',?depth,
'#nThe rough cut is :#x35',?roughcut,
'#nThe number of roughpasses are :#x35',?roughpass,
'#nThe speed is :#x35',?speed,
'#nThe feed is :#x35',?feed,
'#nThe coolant to be used is :#x35',?coolant,
'#n',?oper1,' from :#x35 x=',?fromx,' y=',?fromy,
'#n to :#x35 x=',?tox,' y=',?toy).
end. (* bore *)
```

```
topic knurl.
thr_speed().
feed1_speed1().
coolant().
text('#s#n#nSTEP ',?step,
'#nThe operation is :#x35',?oper1,
```

APPENDIX 1 - LISTING OF INTEGRATED INTELLIGENT PROCESS PLANNING SYSTEM PROGRAM

```
'#nThe tool is :#x35',?tool,  
'#nThe tool material is :#x35',?toolmat,  
'#nThe turns per inch are :#x35',?tolplus,  
'#nThe speed in rpm is :#x35',?speed1,  
'#nThe surface speed is :#x35',?feed1,  
'#nThe coolant to be used is :#x35',?coolant,  
'#n',?oper1,' from :#x35 x=',?fromx,' y=',?fromy,  
'#n to :#x35 x=',?tox,' y=',?toy).  
end. (* knurl *)
```


APPENDIX 2: PROGRAM OF INTEGRATED INTELLIGENT PROCESS SUPPORT SYSTEM [IIPSS]

Integrated Intelligent Process Support System (IIPSS) can be used for advising the optimum operation in batch chemical pulping process.

This program is designed to integrate, meta-system (written in KnowledgePro), a database system (Microsoft Excel) and an application software package (written in C⁺⁺ language).

KnowledgePro constructs the meta-system, which consists of the knowledge-base and inference engine. Microsoft Excel spreadsheet, serves as database and graphical representations, an application software package is a mathematical model of the chemical pulping process.

- Pulp.kb:** This is a main file written in KnowledgePro. This file is play an important role of controlling the whole program.
- Pulp1.xls:** The Microsoft Excel spreadsheet file, which serves as database to store input data and plots the results in the form of chart graphically.
- Pulp_exc.kb:** This file convert the input data from knowledgePro to Excel spreadsheet. Those data can directly be used to create charts of figures through **macro** functions.
- Pulp_inp.kb:** This is an KnowledgePro file which receives the input data from a user. For example, wood chips load, operation temperature,....etc.
- Pulpc.cpp:** Comparable file written in C⁺⁺. This file represent the mathematical model (qualitative model) of the chemical process in the digested.
- Pulpc.exe:** File pulpc.ccp compiled by C⁺⁺ compiler.
- Pulp1.tpx:** This is the hypertext topic file. All of the topics selected from the window processed through this file. For each selected topic, a description is read from the file and placed in the description window.
- Pulp_fig.kb:** This file is to show the pictures in order to describe the fundamental principles of a sulfite pulping process in paper industry. These picture can be represented in bit map file in computers.
- Input:** This file created by KnowledgePro and put all inputs variables given by user to this file.
- Inputcv:** This file created by pulpc.exe and used it self. Purpose of that is in Scandinavia (,) is used to separate decimal. But C⁺⁺ need (.) to make calculation. Time (:) also inverted to decimal. Pulpc.exe do it before the calculation.
- Output:** This file created by pulpc.exe and transferred calculation values and KnowledgePro read the values and print out on the screen.

The program requires the following files for execution:

1. Pulp.kb
2. Pulp_inp.kb
3. Pulp_exe.kb
4. Pulp_fig.kb
5. Pulp.exe
6. Pulp1.tpx
7. Pulp1.xls

Make sure that installed Microsoft Excel under the directory c:\excel\ and file pulp1.xls is copied in the same directory (c:\excex\pulp1.xls.) before you run the file Pulp.kb.

Files 1-6 should be under one directory.

(* ===== PULP.KB =====*)

(*-----LOAD EXCEL SOFTWARE-----*)

w6 is window(,14,3,66,5,,[popup,showChildren,visible,Siblings,DialogFrame],,,
lightgreen,close_event).

text(#n Please wait

Getting the information from spreadsheet file ...').

wait(,4).

close_window (?w6).

ExcelTask is load_program ('c:\excel\excel.exe c:\excel\pulp1.xls',4).

ExcelWindow is first (task_windows (?ExcelTask)).

Macro1 is dde_open (ddeData, Excel, System).

hide_window (?ExcelWindow).

(*-----SETUP-----*)

setup ().

setup1 ().

(*-----CREATE MAIN WINDOW WITH BUTTONS-----*)

wmain is window (quit, 4,1,84,30,'Integrated Intelligent Process Support System [IIPSS]',
[overlapped,thickframe,controlmenu,minimizebox,maximizebox,showchildren,titlebar],
, , green, close_event).

icon is load_icon (kptool_icon).

attach_icon (?wMain, ?icon).

Pulpmenu is menu([[&Situation, 'Chip Load', 'Chip Quality', ph, Temperature],
'&Time-kapa',&Calculation], select).

disable_menu_item(?Pulpmenu,[&Situation]).

w4 is window (,3.5,2,78.4,24.5, ,[childwindow,horzscroll,thinframe,vertscroll],?wmain).
w1 is window (, 3.5,19,78.4,8.5 ,,[childwindow,thinframe,visible],?wmain).
use_font (?bigFont).
text (' #fred Present Item is').
use_font (?mainFont).
bButton is button (Quit,quit,66,7.5,13).

bResults is button (Results,results,53,7.5,13).
bOption is button (Option, option,53,7.5,13).
bclose1 is button (Close1, close1,27,7.5,13).
hide_window(?bOption, ?bclose1).
disable_window([?bResults]).

bInputs is button ('Inputs',inputs,40,7.5,13).

bclose is button (Close, close2,27,7.5,13).
hide_window(?bclose1).
disable_window([?bclose]).

(*-----BUTTON FLOW FIGURE-----*)

bFlowFigure is button ('Flow Figure', FlowFigure,14,7.5,13).
bback is button ('<---' ,back,14,7.5,6.5).
bnext is button ('--->',next,20.5,7.5,6.7).
hide_window(?bback, ?bnext).

(*-----BUTTON INFORMATION-----*)

bTpx is button ('Information',Information,1,7.5,13).
bView is button ('View Inform',CopyTopic,1,7.5,13).
bTopics is button (Inform,toTopics,1,7.5,13).
hide_window (?bTopics,?bView).

curWin is window (,20.5,1.15,65,1.25, ,[child,visible],?w1).
desWin is window (, 1,2.5,78,5, , [childwindow,visible,vertscroll],?w1).

(*-----CREATE TOPIC DISPLAY WINDOW-----*)

w3 is window (, 3.5,1.5,78.4,17, ,[childwindow,thinframe,visible],?wmain).
use_font (?bigFont).
text (' #fred Current topic').
use_font (?mainFont).
libWin is window (, 15.5,1.15,65,1.25, ,[child,visible],?w3).
topWin is window (,1,2.5,78,15.5, ,[childwindow,thinframe,vertscroll,visible],?w3).

(*----- SHOW THE MAIN WINDOW AND PROMPT THE USER-----*)

```
show_window (?wmain).
```

```
(*-----HERE ARE THE TOPICS FOR THIS PROGRAM. -----*)
```

```
topic Information.
```

```
  hide_window (?bTpx).
  show_window (?bTopics,?bView).
  file is ('pulp1.tpx').
```

```
  if ?file <> [] then
    readLibrary (?file).
```

```
end.
```

```
topic readLibrary (file).
```

```
(* This topic reads the topic library file and builds a list of the topics
   included in the file. *)
```

```
msgWin is window ((* eventTopic*),20,10,44,6,,
[popup,showChildren,siblings,DialogFrame]).
text ('
```

```
One moment ... Loading Topic Library').
```

```
  show_window (?msgWin).
  update_window (?msgWin).
  make_modal (?msgWin).
```

```
  set_text (?libWin,[' Library: #s',?file,#n]).
  load (?file, temp).
  topicList is children (temp).
  remove_topic (temp).
  set_display_window (?topWin).
  fullTopicList is [].
  set_file_pos (?file,0).
  apply (ReadLibraryParms, ?topicList).
  set_text (?topWin, concat (?indent, '#m',?fullTopicList,'#m' )).
  mark ( first (?fullTopicList) ).
  close_window (?msgWin).
  if ?inTopicView then
    toTopics ().
```

```
topic ReadLibraryParms (item).
```

```
  if string_where (?item, ' ') <> 0 then
    item is concat ("", ?item, "").
    fullTopic is read (?file, concat (?topicHeader, ?item), '.').
```

```

        if ?fullTopic is ?EOF then
            fullTopic is [].
            fullTopicList gets concat (?item, ?fullTopic).
        end.
    end.

end.

(*=-----READIND OF TOPICS-----=*)

topic CopyTopic (item).
    set_file_pos (?file, 0).
    viewTopic is read (?file, concat (?topicHeader,
        ?current), element (?FullTopicList,
        where (?FullTopicList, ?current) + 1)).
    viewTopic is combine ( concat (?topicHeader,?current, '!'),
        rest (?viewTopic)).
    viewTopic is remove (?viewTopic, ?topicHeader).

        inTopicView is t and
        set_text (?w4, ?viewTopic) and
        show_window ([?w4,?bTopics]) and
        hide_window (?bView).
    end.

topic toTopics.
    inTopicView is f.
    show_window (?bView).
    hide_window ([?w4,?bTopics]).
end.

topic close2.
    disable_window (?bclose).
    enable_window ([?bTpx, ?bView, ?bTopics,?bInputs]).
    hide_window(?bback, ?bnext).
    show_window([?bFlowFigure,?w3]).
    hide_window (?wView).
end.

(*=-----QUIT-----=*)
        (* This topic terminates the application. *)

topic quit.
    hide_window (?wmain).
    dde_execute (?Macro1,['[CLOSE(FALSE)]','[QUIT()]']).
    wait(,5).
    clear ().
end.

```

(*-----HYPERTEXT-----*)

(* This is the default hypertext topic. All of the topics selected from the window are processed through this topic. For each selected topic, a description is read from the file and placed in the description window. *)

topic mark (item).

current is ?item.

set_file_pos (?file, 0).

check is read (?file, concat (?topicHeader, ?item), '*').

topicEnded is f.

apply (checkEnd, ?check).

if ?topicEnded then

describe is ?EOF

else

describe is string_replace (read (?file, '*'), '(',).

if ?describe is ?EOF then

describe is '#fred No Description Available for this topic#d'.

set_text (?curWin, ?item).

set_text (?desWin, ?describe).

topic checkEnd (item).

if string_where (?item, 'end.') <> 0 then

topicEnded is t.

end.

end.

(*-----SETUP-----*)

(* This topic sets up the fonts, colors and general purpose of use throughout the application. *)

topic setup.

topicHeader is 'topic '.

indent is ' '.

EOF is number_to_char (26).

inTopicView is f.

bigFont is create_char_font ([1,1,700,'F','F','F',0,1,34,'Helv']).

hyperFont is create_char_font ([1,1,400,'F','F','F',0,1,34,'Helv']).

boldFont is create_char_font ([1,1,700,'F','F','F',0,1,34,'Helv']).

mainFont is create_char_font ([0.82,0.71428,700,'F','F','F',0,1,34,'helv']).

use_font (?mainFont).

if last (system_info ()) > 2 then

```

    color is green2
    else
    color is black.
    hyper_display (?color,,?mainFont).

```

```

new_file('c:\excel\output').
close('c:\excel\output').

```

```
end.
```

```
(*-----FILES WHICH IS READ BY THE FILE PULP.KB-----*)
```

```
@ pulp_inp
@ pulp_fig
```

```
(*-----INPUT-----*)
```

```

topic inputs.
  hide_window([?w3, ?w4, ?w1]).
  disable_window ([?bTpx, ?bView, ?bTopics, ?bInputs,
  ?bback, ?bnext, ?bResults,?bclose,?bFlowFigure]).
  show_window (?w5).
end.

```

```
(*-----FIGURE-----*)
```

```

topic FlowFigure.
  show_window(?bback, ?bnext).
  enable_window(?bclose).
  show_window(?bclose).
  hide_window([?bFlowFigure,?w3, ?w4]).
  disable_window ([?bTpx, ?bView, ?bTopics, ?bInputs]).
  showIt (.).
  set_focus (?wView).
end.

```

```
(*-----RESULTS-----*)
```

```

topic results.
  show_window(?bclose1).
  enable_window(?bclose1).
  hide_window(?bclose).
  hide_window([?w3, ?w4]).
  disable_window ([?bFlowFigure,?bTpx, ?bView, ?bTopics,
  ?bInputs, ?bButton, ?bResults]).
  @ pulp_exc
end.

```

```

topic close1.
  disable_window (?bclose1).
  dde_close (?value).
  dde_execute (?Macro1,['[CLOSE(FALSE)]',
  '[CLOSE(FALSE)]','[CLOSE(FALSE)]']).
  wait(,5).
  enable_window ([?bFlowFigure,?bTpx, ?bView,
    ?bTopics, ?bInputs, ?bButton, ?bResults]).
  show_window (?w3).
end.

```

(*----- SITUATION BE PROVIDET HERE -----*)

```

topic select (item).
  do (?item).

```

```

topic 'Chip Load'.
  showtext ().
  text (' #e #n#t#t CHIP LOAD is #v ?best8. #v
    #n#n #v ?schipsload. #v #n#n').
  text ('#n #v ?use. #v').

```

end.

```

topic 'Chip Quality'.
  showtext ().
  text (' #e #n#t#t CHIP QUALITY is #v ?best9. #v
    #n#n #v ?schipsquality. #v #n#n').
  text ('#n #v ?use. #v').

```

end.

```

topic 'ph'.
  showtext ().
  text (' #e #n#t#t pH is #v ?best10. #v#n#n#n #v ?spH. #v #n#n').
  text ('#n #v ?use. #v').

```

end.

```

topic 'Temperature'.
  showtext ().
  text (' #e #n#t#t TEMPERATURE is #v ?best11. #v
    #n#n #v ?soperationTem. #v #n#n').
  text ('#n #v ?use. #v').

```

end.

```

topic '&Time-kapa'.
  hide_window([?w3, ?w4, ?w1]).
  show_window (?w9).

```

```

end.

topic &Calculation.
  disable_menu_item(?Pulpmenu,['&Time-kapa',&Calculation]).
  hide_window([?w3, ?w4, ?w1]).
  answer ().
end.

topic showtext ().
  If exists (w7) then
    set_title (?w7, concat ('SITUATION ----> ',?item)) and
    w7Old is ?w7.

    w7 is window (,11.5,3.5,62,22, concat ('SITUATION ----> ',?item) ,
    [titleBar,childwindow,thinframe],?wmain).
    Set_event_topic (clean, sys_char_event).
    use is '#t #fred ---[[[ Press < Esc > to quit #blue SITUATION #fred window ]]]-
--#b '.

  topic clean (info, event, handle).
    if ?info is Esc
    then hide_window (?w7) and
    enable_window ([?bTpx, ?bView, ?bTopics, ?bInputs,
    ?bback, ?bnext, ?bResults,?bFlowFigure]) and
    show_window([?w3, ?w1]).
  end.

  hide_window([?w3, ?w4, ?w1]).
  disable_window ([?bTpx, ?bView, ?bTopics, ?bInputs,
  ?bback, ?bnext, ?bResults,?bFlowFigure]).
  show_window ( ?w7, 5).

  if exists (w7Old) then
    close_window (?w7Old) and
    set_focus (?w7).
  end.
end.

end.

(*==-----THE END OF PULP.KB-----==*)

```


(*===== PULP_EXC.KB =====*)

Macro1 is dde_open (ddeData, Excel, System).
value is dde_open (ddeData, Excel, 'c:\excel\pulp1.XLS').

dde_write (?value,'r3c2', ?chipsload).
dde_write (?value,'r3c3', ?chipsquality).
dde_write (?value,'r3c4', ?chipssize).
dde_write (?value,'r3c5', ?totalSO2).
dde_write (?value,'r3c6', ?pH).
dde_write (?value,'r3c7', ?OperationTem).
dde_write (?value,'r3c8', ?OperationPre).

(*-----CREATION OF CHART1-----*)

dde_execute (?Macro1,['[APP.MOVE(29;5)]','[APP.SIZE(425,260)]']).
wait (,2).
dde_execute (?Macro1,['[SELECT("R2C1:R3C8")'],'[NEW(2)]']).
wait (,1).
dde_execute (?Macro1, ['[GALLERY.COLUMN(3;TRUE)]']).
wait (,4).
dde_execute
(?Macro1,['[SIZE(294;156)]','[FULL(TRUE)]','[ACTIVATE("PULP1.XLS")]']).
wait (,2).

(*----- CREATION OF CHART2 -----*)

dde_execute (?Macro1,['[SELECT("R5C1:R6C4")'],'[NEW(2)]']).
wait (,2).
dde_execute (?Macro1,['[GALLERY.3D.COLUMN(4)]','[FULL(TRUE)]']).
wait (,2).
dde_execute (?Macro1,['[SIZE(294;156)]','[FULL(TRUE)]']).
wait (,2).
dde_execute (?Macro1,['[ACTIVATE("PULP1.XLS")]']).
wait (,1).

(*----- CREATION OF CHART3 -----*)

dde_execute (?Macro1,['[SELECT("R8C1:R9C2")'],'[NEW(2)]']).
wait (,2).
dde_execute (?Macro1,['[GALLERY.3D.COLUMN(4)]','[FULL(TRUE)]']).
wait (,2).

(*-----COMANEDING EXCEL TO DISPLAY THE CHARTS -----*)

show_window (?ExcelWindow).
dde_execute (?Macro1,['[ACTIVATE("chart1")]']).

```

wait (,2).
dde_execute (?Macro1,['[ACTIVATE("chart2")]']).
wait (,2).
dde_execute (?Macro1,['[ACTIVATE("chart3")]']).
wait (,2).

```

```

topic option.
  dde_execute (?Macro1,['[ACTIVATE.NEXT()]]').
  show_window (?ExcelWindow).
end.

```

```

topic ddeData(info,event,handle).
  topic dde_fail_event.
    error_message(dde_fail_event,ddeData).
  end. (* end of dde_fail_event *)

```

```

topic dde_ok_event.
end.

```

```

topic dde_data_event.
end.
end.

```

(*====-----THE END OF PULP_EXC.KB -----====*)

(*===== PULP_FIC.KB =====*)

```

topic setup1.
  bitmapList is [ '1.BMP', '2.BMP', '3.BMP', '4.BMP', '5.BMP', '6.BMP' ].
  timeout is 2.
  color is lightyellow.
  rgbTextColor is 16711680.
  total is list_length (?bitmapList).
  currentItem is 1.
  handlingResize is f.
end.

```

(*----- BACK / NEXT / -----*)

```

topic back.
  if ?currentItem = 1 then
    currentItem is ?total and
    showIt ( )
  else
    currentItem = ?currentItem - 1 and
    showIt ( ).
end.

```

```

topic next.
  if ?currentItem is ?total then
    currentItem is 1 and
    showIt ( )
  else
    currentItem is ?currentItem + 1 and
    showIt ( ).
end.

```

(*-----BMP WINDOW-----*)

```

topic showIt ( ).
  bmpName is element ( ?bitmapList, ?currentItem).
  if exists (wView) then
    set_title (?wView, concat ('Loading...!', ?bmpName)) and
    wOld is ?wView.
  if exists (bmpCurrent) then
    bmpOld is ?bmpCurrent.
    bmpCurrent is load_bitmap ( ?bmpName ).

```

```

wView is window (modifyView, 15, 1.5, 54, , concat ('Loading...!', ?bmpName),
[child,thinframe,siblings,showChildren,titlebar,,maximizebox],
?wMain,,, [resize_event, paint_event]).
bWidth is element (window_info (?wView), 10).
bHeight is element (window_info (?wView), 11).
bitmap ( ?bmpCurrent, 1, 1, ?bWidth, ?bHeight ).
show_window (?wView).
if exists (wOld) then
  close_window (?wOld) and
  delete_bitmap (?bmpOld) and
  set_focus (?wView).
end.

```

(*-----RESIZEING BMP WINDOW-----*)

```

topic modifyView (info, event, handle).
  do (?event).
  topic resize_event.
    if ?handlingResize then
      exit ( ).
    handlingResize is t.
    set_title (?wView, concat ('Resizing...!', ?bmpName)).
    if ?info is 2 then
      bWidth is element (system_info (?wView), 3) and
      bHeight is element (system_info (?wView), 4)
    else

```

```
    bWidth is element (window_info (?wView), 10) and
    bHeight is element (window_info (?wView), 11).
    bitmap ( ?bmpCurrent, 1, 1, ?bWidth, ?bHeight ).
    handlingResize is f.
```

```
    hide_window([?w4, ?w5, ?w9]).
    If exists (w7) then
    hide_window (?w7).
    If exists (w10) then
    hide_window (?w10).
end. (* resize_event *)
```

```
topic paint_event.
    set_title (?wView, ?bmpName).
    collect (.).
end. (* paint_event *)
```

```
end. (* resizeView *)
```

```
(*==----- THE END PULP_FIC.KB -----==*)
```

```
(* ===== PULP_INP.KB ===== *)
(* ----- INPUT WINDOW LAYOUT ----- *)
```

```
w5 is window (3.5,2,78.4,24.5, 'Please give input values here' ,
  [titleBar,childwindow,thinframe],?wmain,, lightyellow).
text ('
```

Wood chips load

pH of initial
cooking liquor

Wood chips quality

Operation
temperature

Wood chips size

Total SO2 of initial
cooking liquor

Operation
pressure

Combined %SO2

Free SO2

Total SO2(10)

S - factor model

Are you sure all the input values are correct?

).

```
comboFavorite1 is combo_box (['22 - 29% gauge', '30 - 34% gauge', '35 - 40% gauge',
  '41 - 46% gauge', '47 - 55% gauge'],,19.5,2.8,19,, '35 - 40% gauge',dropdown).
```

```
comboFavorite2 is combo_box (['Good quality', 'Old quality', '< 10% hard wood',
  'Old chips & < 10% hardwood'],,19.5,6,19,,,dropdown).
```

```
comboFavorite3 is combo_box (['Standard size', 'Over stanard size', 'Under stanard size']
  ,,19.5,9,19,,,dropdown).
```

```
comboFavorite4 is combo_box (['4.00 - 4.80%', '4.81 - 5.44%', '5.45 - 6.15%',
  '6.16 - 6.50%', '6.51 - 7.40%'],,19.5,12,19,, '5.45 - 6.15%',dropdown).
```

freeSO2 is edit_line (['2.95'],,14,16,7,,,dropdown).

comboFavorite5 is combo_box (['2.5 - 2.9%', '3.0 - 3.4%', '3.4 - 3.6%', '3.7 - 4.0%', '4.1 - 4.6%'],,57.5,3,15,, '3.4 - 3.6%',dropdown).

comboFavorite6 is combo_box (['154 - 157 C', '158 - 161 C', '162 - 165 C', '166 - 169 C', '170 - 175 C'],,57.5,7,15,, '162 - 165 C',dropdown).

comboFavorite7 is combo_box (['575 - 624 kPa', '625 - 674 kPa', '675 - 685 kPa', '685 - 735 kPa', '736 - 785 kPa'],,57.5,11.5,18,, '675 - 685 kPa',dropdown).

combinedSO2 is edit_line (['2.80'],,59.5,14.8,6,9,,,dropdown).

'totalSO2(10)' is edit_line (['0.00'],,59.5,17.8,6,9,,,dropdown).

'S - factor model' is edit_line (['0.00'],,14,19.4,7,,,dropdown).

button (Yes,yes,58,20.5,6).

(*----- Finish of windo style -----*)

topic yes.

best1 is get_combo_box (?comboFavorite1).
do (select1).
best2 is get_combo_box (?comboFavorite2).
do (select2).
best3 is get_combo_box (?comboFavorite3).
do (select3).
best4 is get_combo_box (?comboFavorite4).
do (select4).
best5 is get_combo_box (?comboFavorite5).
do (select5).
best6 is get_combo_box (?comboFavorite6).
do (select6).
best7 is get_combo_box (?comboFavorite7).
do (select7).
best8 is get_combo_box (?comboFavorite1).
do (select8).
best9 is get_combo_box (?comboFavorite2).
do (select9).
best10 is get_combo_box (?comboFavorite5).
do (select10).
best11 is get_combo_box (?comboFavorite6).
do (select11).
best12 is get_combo_box (?comboFavorite1).
do (select12).

```

best13 is get_combo_box (?comboFavorite2).
do (select13).
best14 is get_combo_box (?comboFavorite3).
do (select14).
best15 is get_combo_box (?comboFavorite4).
do (select15).
best16 is get_combo_box (?comboFavorite5).
do (select16).
best17 is get_combo_box (?comboFavorite6).
do (select17).
best18 is get_combo_box (?comboFavorite7).
do (select18).

```

```

enable_window ([?bTpx, ?bView, ?bTopics, ?bInputs,
               ?bback, ?bnext, ?bResults, ?bFlowFigure]).
hide_window (?w5).
show_window ([?w3, ?w1]).
enable_menu_item (?Pulpmenu, [&Situation, '&Time-kapa', &Calculation]).
end.

```

```
(*----- RULE1 -----*)
```

```

topic select1.
  if ?best1 is '22 - 29% gauge' then chipsload is ?minusfour.
  if ?best1 is '30 - 34% gauge' then chipsload is ?minustwo.
  if ?best1 is '35 - 40% gauge' then chipsload is ?zero.
  if ?best1 is '41 - 46% gauge' then chipsload is ?plustwo.
  if ?best1 is '47 - 55% gauge' then chipsload is ?plusfour.
end.

```

```

topic select2.
  if ?best2 is 'Good quality' then chipsquality is ?zero.
  if ?best2 is 'Old quality' then chipsquality is ?minusthree.
  if ?best2 is '< 10% hard wood' then chipsquality is ?minusthree.
  if ?best2 is 'Old chips & < 10% hardwood' then chipsquality is ?minusthree.
end.

```

```

topic select3.
  if ?best3 is 'Standard size' then chipssize is ?zero.
  if ?best3 is 'Over stanard size' then chipssize is ?plusthree.
  if ?best3 is 'Under stanard size' then chipssize is ?minusthree.
end.

```

```

topic select4.
  if ?best4 is '4.00 - 4.80%' then totalSO2 is ?plusthree.
  if ?best4 is '4.81 - 5.44%' then totalSO2 is ?plusone.
  if ?best4 is '5.45 - 6.15%' then totalSO2 is ?zero.
  if ?best4 is '6.16 - 6.50%' then totalSO2 is ?minusone.
  if ?best4 is '6.51 - 7.40%' then totalSO2 is ?minusthree.

```

end.

topic select5.

if ?best5 is '2.5 - 2.9%' then pH is ?plusthree.
 if ?best5 is '3.0 - 3.4%' then pH is ?plusone.
 if ?best5 is '3.4 - 3.6%' then pH is ?zero.
 if ?best5 is '3.7 - 4.0%' then pH is ?minusone.
 if ?best5 is '4.1 - 4.6%' then pH is ?minusthree.

end.

topic select6.

if ?best6 is '154 - 157 C' then OperationTem is ?plusfour.
 if ?best6 is '158 - 161 C' then OperationTem is ?plustwo.
 if ?best6 is '162 - 165 C' then OperationTem is ?zero.
 if ?best6 is '166 - 169 C' then OperationTem is ?minustwo.
 if ?best6 is '170 - 175 C' then OperationTem is ?minusfour.

end.

topic select7.

if ?best7 is '575 - 624 kPa' then OperationPre is ?plusthree.
 if ?best7 is '625 - 674 kPa' then OperationPre is ?plusone.
 if ?best7 is '675 - 685 kPa' then OperationPre is ?zero.
 if ?best7 is '685 - 735 kPa' then OperationPre is ?minusone.
 if ?best7 is '736 - 785 kPa' then OperationPre is ?minusthree.

end.

(*----- RULE2 -----*)

topic select8. (* Chip load *)

if ?best8 is '22 - 29% gauge' then Schipsload is ?CL1.
 if ?best8 is '30 - 34% gauge' then Schipsload is ?CL1.
 if ?best8 is '35 - 40% gauge' then Schipsload is ?CL2.
 if ?best8 is '41 - 46% gauge' then Schipsload is ?CL3.
 if ?best8 is '47 - 55% gauge' then Schipsload is ?CL3.

end.

topic select9. (* Chip quality *)

if ?best9 is 'Good quality' then Schipsquality is ?CQ1.
 if ?best9 is 'Old quality' then Schipsquality is ?CQ2.
 if ?best9 is '< 10% hard wood' then Schipsquality is ?CQ3.
 if ?best9 is 'Old chips & < 10% hardwood' then Schipsquality is ?CQ4.

end.

topic select10. (* ph *)

if ?best10 is '2.5 - 2.9%' then SpH is ?PH1.
 if ?best10 is '3.0 - 3.4%' then SpH is ?PH1.
 if ?best10 is '3.4 - 3.6%' then SpH is ?PH2.
 if ?best10 is '3.7 - 4.0%' then SpH is ?PH3.

if ?best10 is '4.1 - 4.6%' then SpH is ?PH3.

end.

topic select11. (* Temperature *)

if ?best11 is '154 - 157 C' then SOperationTem is ?TP1.

if ?best11 is '158 - 161 C' then SOperationTem is ?TP1.

if ?best11 is '162 - 165 C' then SOperationTem is ?TP2.

if ?best11 is '166 - 169 C' then SOperationTem is ?TP3.

if ?best11 is '170 - 175 C' then SOperationTem is ?TP3.

end.

(*----- RULE3 -----*)

topic select12. (* chipsload *)

if ?best12 is '22 - 29% gauge' then Tchipsload is ?|M4|.

if ?best12 is '30 - 34% gauge' then Tchipsload is ?|M2|.

if ?best12 is '35 - 40% gauge' then Tchipsload is ?|0|.

if ?best12 is '41 - 46% gauge' then Tchipsload is ?|2|.

if ?best12 is '47 - 55% gauge' then Tchipsload is ?|4|.

end.

topic select13. (* chipsquality *)

if ?best13 is 'Good quality' then Tchipsquality is ?|0|.

if ?best13 is 'Old quality' then Tchipsquality is ?|M3|.

if ?best13 is '< 10% hard wood' then Tchipsquality is ?|M3|.

if ?best13 is 'Old chips & < 10% hardwood' then Tchipsquality is ?|M3|.

end.

topic select14. (* chipssize *)

if ?best14 is 'Standard size' then Tchipssize is ?|0|.

if ?best14 is 'Over stanard size' then Tchipssize is ?|3|.

if ?best14 is 'Under stanard size' then Tchipssize is ?|M3|.

end.

topic select15. (* totalSO2 *)

if ?best15 is '4.00 - 4.80%' then TtotalSO2 is ?|3|.

if ?best15 is '4.81 - 5.44%' then TtotalSO2 is ?|1|.

if ?best15 is '5.45 - 6.15%' then TtotalSO2 is ?|0|.

if ?best15 is '6.16 - 6.50%' then TtotalSO2 is ?|M1|.

if ?best15 is '6.51 - 7.40%' then TtotalSO2 is ?|M3|.

end.

topic select16. (* pH *)

if ?best16 is '2.5 - 2.9%' then TpH is ?|3|.

if ?best16 is '3.0 - 3.4%' then TpH is ?|1|.

```

if ?best16 is '3.4 - 3.6%' then TpH is ?|0|.
if ?best16 is '3.7 - 4.0%' then TpH is ?|M1|.
if ?best16 is '4.1 - 4.6%' then TpH is ?|M3|.
end.

```

```

topic select17. (* OperationTem *)
if ?best17 is '154 - 157 C' then TOperationTem is ?|4|.
if ?best17 is '158 - 161 C' then TOperationTem is ?|2|.
if ?best17 is '162 - 165 C' then TOperationTem is ?|0|.
if ?best17 is '166 - 169 C' then TOperationTem is ?|M2|.
if ?best17 is '170 - 175 C' then TOperationTem is ?|M4|.
end.

```

```

topic select18. (* OperationPre *)
if ?best18 is '575 - 624 kPa' then TOperationPre is ?|3|.
if ?best18 is '625 - 674 kPa' then TOperationPre is ?|1|.
if ?best18 is '675 - 685 kPa' then TOperationPre is ?|0|.
if ?best18 is '685 - 735 kPa' then TOperationPre is ?|M1|.
if ?best18 is '736 - 785 kPa' then TOperationPre is ?|M3|.
end.

```

(*----- DIFINING OF VALUES -----*)

```

minusnine is 3:00.
minuseighth is 3:05.
minusseven is 3:10.
minussix is 3:15.
minusfive is 3:20.
minusfour is 3:25.
minusthree is 3:30.
minustwo is 3:35.
minusone is 3:40.
zero is 3:50.
plusone is 4:00.
plustwo is 4:05.
plusthree is 4:10.
plusfour is 4:15.
plusfive is 4:20.
plussix is 4:25.
plusseven is 4:30.
pluseight is 4:35.
plusnine is 4:40.

```

```

|M9| is '          - Shorter cooking time by 40 to 50 minutes.
          #t- Required cooking time is 2:55 to 3:05 hours'.
|M8| is '          - Shorter cooking time by 35 to 45 minutes.
          #t- Required cooking time is 3:00 to 3:10 hours'.
|M7| is '          - Shorter cooking time by 30 to 40 minutes.

```

#t- Required cooking time is 3:05 to 3:15 hours'.
 |M6| is ' - Shorter cooking time by 25 to 35 minutes.
 #t- Required cooking time is 3:10 to 3:20 hours'.
 |M5| is ' - Shorter cooking time by 20 to 30 minutes.
 #t- Required cooking time is 3:15 to 3:25 hours'.
 |M4| is ' - Shorter cooking time by 15 to 25 minutes.
 #t- Required cooking time is 3:20 to 3:30 hours'.
 |M3| is ' - Shorter cooking time by 10 to 20 minutes
 #t- Required cooking time is 3:25 to 3:35 hours'.
 |M2| is ' - Shorter cooking time by 5 to 15 minutes.
 #t- Required cooking time is 3:30 to 3:40 hours'.
 |M1| is ' - Shorter cooking time by 0 to 10 minutes.
 #t- Required cooking time is 3:35 to 3:45 hours'.
 |0| is ' - Normal cooking time
 #t- Required cooking time is 3:45 to 3:55 hours'.
 |1| is ' - Longer cooking time by 0 to 10 minutes.
 #t- Required cooking time is 3:55 to 4:05 hours'.
 |2| is ' - Longer cooking time by 5 to 15 minutes.
 #t- Required cooking time is 4:00 to 4:10 hours'.
 |3| is ' - Longer cooking time by 10 to 20 minutes.
 #t- Required cooking time is 4:05 to 4:15 hours'.
 |4| is ' - Longer cooking time by 15 to 25 minutes.
 #t- Required cooking time is 4:10 to 4:20 hours'.
 |5| is ' - Longer cooking time by 20 to 30 minutes.
 #t- Required cooking time is 4:15 to 4:25 hours'.
 |6| is ' - Longer cooking time by 25 to 35 minutes.
 #t- Required cooking time is 4:20 to 4:30 hours'.
 |7| is ' - Longer cooking time by 30 to 40 minutes.
 #t- Required cooking time is 4:25 to 4:35 hours'.
 |8| is ' - Longer cooking time by 35 to 45 minutes.
 #t- Required cooking time is 4:30 to 4:40 hours'.
 |9| is ' - Longer cooking time by 40 to 50 minutes.
 #t- Required cooking time is 4:35 to 4:45 hours'.

(*----- SITUATION WINDOW LAYOUT -----*)

CQ1 is ' COOKING TIME: #n
 #V ?Tchipsquality. #V #n#n
 RESULTS IN THE FINAL PULP:#n
 1. Good brightness #n#n '.

CQ2 is ' COOKING TIME: #n
 #V ?Tchipsquality. #V#n#n
 RESULTS IN THE FINAL PULP:#n
 1. Higher brightness
 2. Lower pulp strength '.

CQ3 is ' COOKING TIME: #n
 #V ?Tchipsquality. #V #n#n
 RESULTS IN THE FINAL PULP: #n
 1. Lower brightness
 2. Lower steam demand '.

CQ4 is ' COOKING TIME: #n
 #V ?Tchipsquality. #V #n#n
 RESULTS IN THE FINAL PULP:#n
 1. Undetermined brightness
 2. Lower steam demand
 3. Lower pulp strength '.

CL1 is ' COOKING TIME: #n
 #V ?Tchipsload. #V #n#n
 RESULTS IN THE FINAL PULP:#n
 1. Lower yield
 2. Higher brightness#n#n
 CAUSE: #n
 1. Chips are not well packed
 2. Steam packer did not work properly '.

CL2 is ' COOKING TIME: #n
 #V ?Tchipsload. #V #n#n
 Desired Conditions'.

CL3 is ' COOKING TIME: #n
 #V ?Tchipsload. #V #n#n
 RESULTS IN THE FINAL PULP:#n
 1. Lower brightness
 2. Reduced acid strength
 3. Higher steam demand#n#n
 CAUSE: #n
 1. Higher moisture '.

PH1 is ' COOKING TIME: #n
 #V ?TpH. #V #n#n
 RESULTS IN THE FINAL PULP: #n
 1. Low brightness #n#n
 CAUSE:#n
 1. High acidity of cooking liquor
 2. Poor cooking acid quality '.

PH2 is ' COOKING TIME: #n

#V ?TpH. #V #n#n
Desired Conditions'.

PH3 is ' COOKING TIME: #n
#V ?TpH. #V #n#n
RESULTS IN THE FINAL PULP: #n
1. Good brightness #n#n
CAUSE: #n
1. Low acidity of cooking liquor
2. Poor cooking acid quality '.

TP1 is ' COOKING TIME: #n
#V ?TOperationTem. #V #n#n
RESULTS IN THE FINAL PULP: #n
1. Good brightness #n#n
CAUSE: #n
1. Steam valve did not open on time
2. Too much condensate in the steam value
3. Poor steam quality
4. Poor temperature control '.

TP2 is ' COOKING TIME: #n
#V ?TOperationTem. #V #n#n
Desired Conditions '.

TP3 is ' COOKING TIME: #n
#V ?TOperationTem. #V #n#n
RESULTS IN THE FINAL PULP: #n
1. Lower brightness #n#n
CAUSE: #n
1. Steam valve did not close on time
2. Too much condensate in the steam valve
3. Poor temperature control '.

(*----- QUANTITATIVE COMPUTATION -----*)

w9 is window (3.5,2,78.4,24.5, 'Please input values for QUANTITATIVE computation',
[titleBar,childwindow,thinframe],?wmain,,lightgray).
text ('

#fbblue What is the total SO2 of cooking acid in percent during
acid filling.....[x.xx] or [x,xx]

Enter the %total SO2 of cooking acid sample collected
10 minutes after the reached a MAXIMUM.[x.xx] or [x,xx]

Enter the TIME the sample was collected
(i.e. time from the start of the cycle.[x:xx] #fblack (h:min)

#fblue Enter the S - factor at that time the sample
was collected.....[xx,xx] or [xxx,x]

Operation Temperature (*C) Operation Pressure (kPa)

Enter TIME into cooking cycle..... [x:xx] #fblack (h:min)

#fred Are you sure all the input values are correct?

).

ed1 is edit_line ('5,38',63,2.8,7).
ed2 is edit_line ('1,89',63,5.9,7).
ed3 is edit_line (2:10,,45,9.2,7).
ed4 is edit_line (124,,45,12.5,7).
ed5 is edit_line (163,,33,15.2,7).
ed6 is edit_line (680,,63,15.2,7).
ed7 is edit_line (3:23,,45,17.6,7).

button (Yes,ok,58,20.5,6).

topic ok.

TS0 is get_text (?ed1).
TS10 is get_text (?ed2).
t1 is get_text (?ed3).
SMF1 is get_text (?ed4).
TE is get_text (?ed5).
P is get_text (?ed6).
t is get_text (?ed7).

new_file ('c:\excel\input').
write ('c:\excel\input', ?TS0).
write ('c:\excel\input', '#n').
write ('c:\excel\input', ?TS10).
write ('c:\excel\input', '#n').
write ('c:\excel\input', ?t1).

```

write ('c:\excel\input', '#n').
write ('c:\excel\input', ?SMF1).
write ('c:\excel\input', '#n').
write ('c:\excel\input', ?TE).
write ('c:\excel\input', '#n').
write ('c:\excel\input', ?P).
write ('c:\excel\input', '#n').
write ('c:\excel\input', ?t).
write ('c:\excel\input', '#n').
close('c:\excel\input').
hide_window (?w9).
show_window([?w3,?w1]).
(*enable_menu_item(?Pulpmenu,[&Situation]).*)

```

end.

(*----- ANSWER OF QUANTITATIVE CALCULATION-----*)

```

topic answer ().
  If exists (w10) then
    w10Old is ?w10.

    w10 is window (,18.5,5.5,47,18, 'QUANTITATIVE ANSWER',
    [titleBar,childwindow,thinframe],?wmain, ,lightgray).
      if exists (w10Old) then
        close_window (?w10Old) and
        set_focus (?w10).
      show_window (?w10).
      run('pulpc.exe',0).
      SMF is read_line('output',1).
      RCT is read_line('output',1).
      KP is read_line('output',1).
      close('output').
      text ('###n###n#fblue  S - FACTOR is ..... #v ?SMF. #v ').
      text ('###n###n#fblue  Estimated Remaining Cooking Time is..... #v ?RCT. #v
hrs').
      text ('###n###n#fblue  Estimated Current Kappa Number is..... #v ?KP. #v ').

      button ( OK,ok1,22,14.5,6).

    topic ok1.
      hide_window (?w10).
      show_window([?w3,?w1]).
      enable_menu_item(?Pulpmenu,['&Time-kapa',&Calculation]).
    end.
  end.
end.

```

(*==----- THE END PULP_INP.KB -----==*)

```
(* ===== PULPC.CPP ===== *)
```

```
#include <math.h>
#include <stdio.h>
#include <fstream.h>
#define NEWLINE '\n'

void main(void)
{
double TS0,TS10,t1,SMF1,TE,P,t,SMF,CT,TS,K,RCT,SMTF,KP,dmin;
int hour,min,iKP;
char c;
FILE *fp;
FILE *fg;
FILE *op;
ifstream ifst;
ofstream ofst;
/* convert , to . and : to .fraction */

min=0;
op=fopen("inputcv","w");
fp=fopen("c:\\excel\\input","r");
c = getc(fp);

while (c != EOF)
{
if (c == ',')
c = '!';
else if (c == ':')
{ c = getc(fp);
while (c != NEWLINE)
{ if ((c >= '0') | (c <= '9'))
min = 10 * min + (c-'0');
c = getc(fp);
}
dmin = min/60.0;
dmin = dmin * 1000.0;
dmin = dmin + 0.5;
min = dmin;
putc('.',op);
fprintf(op,"%d",min);
min = 0;
}
putc(c,op);
c = getc(fp);
}
fclose(fp);
```



```

fclose(op);

/* use converted file */
ifst.open("inputcv");

ifst >> TS0;
ifst >> TS10;
ifst >> t1;
ifst >> SMF1;
ifst >> TE;
ifst >> P;
ifst >> t;

ofst.open("output");

TS=((8.336*TS0)+(7.425*TS10));
SMF=-1*((-75.5+TS)/(0.0217));
K=exp(27.25-(10166.08/(273+TE)));
CT=((SMF-SMF1)/(P*0.01*K));
RCT=CT-t+t1;
hour=RCT;
dmin=RCT-hour;
min=dmin*60.0;
SMTF=SMF1+((P*0.01*K*(t-t1)/30));
KP=108.2-SMTF+TS;
iKP=KP;

// *****
//  fprintf(fg,"%f\n%f\n%f\n",SMF,RCT,KP);

ofst.precision(2);
ofst << SMF;
ofst.precision(6);
ofst << "\n";
ofst << hour << ':'<< min;
ofst << "\n";
ofst << iKP;
ofst << "\n";

// *****
//  fclose(fg);
//  fclose(fp);

ifst.close();
ofst.close();
}
(=====THE END OF PULPC.CPP=====)

```

APPENDIX 3: LISTING OF CASE STUDY PROGRAM FOR CHOCOLATE FACTORY

(* =====FACTORY.KB=====*)

(*-----CREATE MAIN WINDOW WITH CHOICES-----*)

```
windowStyle = [ popup, visible, titleBar, controlMenu, thickFrame,maximizeBox,
minimizeBox ].
```

```
hyperStyle = [ popup, visible, titleBar ].
```

```
mainFont is create_char_font ( [0.82,0.71428,700,'F','F','F',0,1,34,'helv']).
```

```
use_font (?mainFont).
```

```
hyper_display (green2,,?mainFont).
```

```
explain = [].
```

```
hovedprogram ( ).
```

```
topic hovedprogram.
```

```
    window (,10,5,60,15,'Expert System Case Study For Chocolate
Factory',?windowStyle).
```

```
    menu( [&Close], menuHandler).
```

```
    diagnose = ?factory.
```

```
    close_window ( ).
```

```
    diagnoseWindow = window ( , 10,10, 70, 10,'ACTION', , , black, 11665407,
close_event ).
```

```
    meny = menu( [ [&Program, &Restart, [ ], &Close],&Why ], menuHandler ).
```

```
    text( '#n#t I recommend following action:#n#fred',concat( #t,?diagnose) ).
```

```
topic menuHandler( item ).
```

```
    do (?item).
```

```
    topic &Restart.
```

```
        close_window( ?diagnoseWindow ).
```

```
        new_kb('factory1').
```

```
    end.
```

```
    topic &Close.
```

```
        clear ( ).
```

```
    end.
```

```
    topic &Help.
```

```
        help( ).
```

```
    end.
```

```
    topic &Why.
```

```
        window( , 15,5, 60, 15,'Explanation' ).
```

```
        why is list_to_string( ?explain,' and#n#t' ).
```

```

                text( ' If:',concat(#t,?why),' then you should:',concat(#t,?diagnose) ).
            end.
        end.
    end.
end.

(*****RULE BASE *****)

topic factory.
    IF ?'ask factory' = 'Biscuits is pale'
    THEN factory  = ?'pale'.

    IF ?'ask factory' = 'Biscuits is not risen'
    THEN factory  = ?'risen'.

    IF ?'ask factory' = 'Biscuits is burnt'
    THEN factory  = ?'burnt'.

    IF ?'ask factory' = 'Chocolate is sticky'
    THEN factory  = ?'sticky'.

    IF ?'ask factory' = 'Chocolate is bubbling'
    THEN factory  = ?'oven'.

    IF ?'ask factory' = 'Chocolate is runny'
    THEN factory  = ['Report fault to supervisor.',
                    'If butter rancid then discard batch'].

    IF ?'ask factory' = 'Chocolate is too dark'
    THEN factory  = ['Measure time to level in hopper to drop by',
                    'none gradation should be 4 minuts.',
                    '#fbblue If necessary call fitter'].

    IF ?'ask factory' = 'Chocolate is too light'
    THEN factory  = 'Report fault to supervior'.

    IF ?'ask factory' = 'Bad smell from biscuits'
    THEN factory  = 'Check biscuit burnt'.

topic 'ask factory'.
    ask (#eWhat type of problem do you have?,'ask factory',
        [ 'Biscuits is pale','Biscuits is not risen','Biscuits is burnt','Chocolate is
sticky',
        'Chocolate is bubbling','Chocolate is runny','Chocolate is too dark',
        'Chocolate is too light','Bad smell from biscuits' ] ).
    explain gets ?'ask factory'.
end.

```

end.

topic pale.

```
IF ?'ask pale' = 'Oven is too cool'
THEN pale = ?'color'.
```

```
IF ?'ask pale' = 'Faulty flour'
THEN pale = ?'faulty'.
```

```
IF ?'ask pale' = 'Wrong sugar'
THEN pale = 'Report fault to supervisor'.
```

topic 'ask pale'.

```
ask('What is the reason for biscuit pale?', 'ask pale',
    ['Oven is too cool', 'Faulty flour', 'Wrong sugar']).
explain gets ?'ask pale'.
```

end.

end.

topic color.

```
IF ?'ask color' = 'Black color at viewing window'
THEN color = 'Restart override'.
```

```
IF ?'ask color' = 'Red color at viewing window'
THEN color = 'Turn up oxygen supply'.
```

```
IF ?'ask color' = 'Flickering light at viewing window'
THEN color = 'Rasie alarm and evacuale factory'.
```

topic 'ask color'.

```
ask ('Which color do you observing at viewing window?', 'ask color',
    ['Black color at viewing window', 'Red color at viewing
window', 'Flickering light at viewing window']).
explain gets ?'ask color'.
```

end.

end.

topic risen.

```
IF ?'ask risen' = 'Oven is too cool'
THEN risen = ?'color'.
```

```
IF ?'ask risen' = 'Too much cocoa'
THEN risen = ['Measure time to level in hopper to drop by',
    'one gradation should be 4 minuts.',
    '#fbblue If necessary call fitter'].
```

```

IF ?'ask risen' = 'Too little raising agent'
THEN risen  =['Add ingredient and report fault',
              'and discard batch'].

IF ?'ask risen' = 'Faulty flour'
THEN risen  =?'faulty'.

topic 'ask risen'.
    ask('What is the reason for butter not risen?', 'ask risen',
        ['Oven is too cool', 'Too much cocoa', 'Too little raising agent', 'Faulty
flour']).
    explain gets ?'ask risen'.
end.
end.

topic faulty.
IF ?'ask faulty' = 'Change of source'
THEN faulty  = ['Report faulty and discard batch and',
               'check faulty batch'].

IF ?'ask faulty' = 'Old stale consignment'
THEN faulty  = ['Report faulty and discard batch and',
               'check faulty batch'].

IF ?'ask faulty' = 'Faulty batch'
THEN faulty  = 'Report fault to supervisor'.

topic 'ask faulty'.
    ask('What is the reason for faulty flour?', 'ask faulty',
        ['Change of source', 'Old stale consignment', 'Faulty batch']).
    explain gets ?'ask faulty'.
end.
end.

topic burnt.
IF ?'ask burnt' = 'Oven is too hot'
THEN burnt  = ?oven.

IF ?'ask burnt' = 'Too much cocoa'
THEN burnt  = ['Measure time to level in hopper to drop by',
               'one gradation should be 4 minuts.',
               '#fblue If necessary call fitter'].

IF ?'ask burnt' = 'Too much sugar'
THEN burnt  = ['See if sugar flowing steadily from nozzle',

```

```
'whitout breaking ioto droplets.',
#fblue Turn down sugar supply.'].
```

```
topic 'ask burnt'.
```

```
ask('What is the reason for biscuit burnt?','ask burnt',
    ['Oven is too hot','Too much cocoa','Too much sugar']).
explain gets ?'ask burnt'.
```

```
end.
```

```
end.
```

```
topic oven.
```

```
IF ?'ask oven' = 'Gas temperature is too high'
THEN oven = 'If white color at viewing window then turn down oxygen input'.
```

```
IF ?'ask oven' = 'Cooling failed'
THEN oven = ?failed.
```

```
topic 'ask oven'.
```

```
ask('What is the reason for oven too hot?','ask oven',
    ['Gas temperature is too high','Cooling failed']).
explain gets ?'ask oven'.
```

```
end.
```

```
end.
```

```
topic failed.
```

```
IF ?'ask failed' = 'Water jacket warm to touch'
THEN failed = 'Call fitter'.
```

```
IF ?'ask failed' = 'Pump is not vibrating'
THEN failed = 'Call electrician'.
```

```
IF ?'ask failed' = 'Water tap is turned off'
THEN failed = 'Turn on the tap'.
```

```
topic 'ask failed'.
```

```
ask('What is the reason for cooling failed?','ask failed',
    ['Water jacket warm to touch','Pump is not vibrating',
    'Water tap is turned off']).
explain gets ?'ask failed'.
```

```
end.
```

```
end.
```

```
topic sticky.
```

```
IF ?'ask sticky' = 'not enough air in mixture'
THEN sticky = ?mixture.
```

```
IF ?'ask sticky' = 'faulty cocoa'  
THEN sticky = ['Report faulty and discard batch and',  
              'check faulty batch'].
```

```
topic 'ask sticky'.  
    ask('What is the reason for chocolate sticky?', 'ask sticky',  
        ['Not enough air in mixture', 'Faulty cocoa']).  
    explain gets ?'ask sticky'.  
end.
```

```
end.
```

```
topic mixture.
```

```
IF ?'ask mixture' = 'dust cover is blocked'  
THEN mixture = 'Clean and replace the mixture'.
```

```
IF ?'ask mixture' = 'fan is slow or stopped'  
THEN mixture = 'Call fitter'.
```

```
topic 'ask mixture'.  
    ask('What is the reason for not enough air in mixture?', 'ask mixture',  
        ['Dust cover is blocked', 'Fan is slow or stopped']).  
    explain gets ?'ask mixture'.  
end.
```

```
end.
```

APPENDIX 4: LISTING OF IIPMS PROGRAM

```
(* ===== IIPMS.KB =====  
                                Integrated Intelligent Systems Application.  
This program performs break-even analysis for 2 competing alternatives in a production  
environment. Meta-system written in KnowledgePro to work together with Excel 4.x.  
and a mathematical model written in C.  
  
                                1994.2.  
=====*)
```

```
setup().  
w1 is window(,14,3,66,26,,  
    [popup,showChildren,Siblings,DialogFrame],,lightblue,close_event).
```

```
text('
```

```
    < Introduction >  
    -----
```

```
    This IIPMS system performs break-even analysis for 2  
    competing alternatives in a process of production planning.
```

```
    The meta-system first reads the values of the coefficients from an  
    EXCEL spreadsheet file.
```

```
    It then sends the coefficients to a mathematical model witten in C  
    for computation of break-even points.
```

```
    The mathematical model sends the results back to the meta-system.
```

```
    The meta-system then makes the decision based on the rules.
```

```
    ).
```

```
b1 is button(OK,next,30,24,6,t).  
show_window(?w1).
```

```
topic next.
```

```
hide_window(?w1).
```

```
w2 is window(,10,5,73,22,,[popup,showChildren,Siblings,DialogFrame],,lightyellow,  
    ,close_event).
```

```
text('
```

```
    The program requires the following files for execution:
```


-

-
1. Proj.exe (The C executable file) & Proj.pif.
 2. IIPMS.xls (The Excel spreadsheet file).

In addition, the following three files are created:

1. Coeff (Created by KnowledgePro and used by C).
2. Result (Created by C and used by KnowledgePro).
3. Consult (Created by KnowledgePro to store the consultation if required).

).

```
b2 is button(Begin,IIPMSbegin,22,20,10,t).
b3 is button(Cancel,IIPMSexit,42,20,10,f).
show_window(?w2).
end. (* end of next *)
```

```
topic IIPMSbegin.
  hide_window(?w1).
  hide_window(?w2).
  w3 is window(14,3,66,26,,[popup,showChildren,visible,Siblings,DialogFrame],,,
    lightgreen,close_event).
  text('#n Please wait .....
  Getting the coefficients of two functions from spreadsheet file ...').
  wait(1).
  load_program('c:\excel\EXCEL.EXE c:\excel\IIPMS.XLS').
  channel is dde_open(ddeData,EXCEL,'c:\excel\IIPMS.XLS').
  dde_request(?channel,'r2c1:r3c4',CSV).
end. (* end of IIPMSbegin *)
```

```
topic ddeData(info,event,handle).
  do(?event).
```

```
topic dde_fail_event.
  error_message(dde_fail_event,ddeData).
end. (* end of dde_fail_event *)
```

```
topic dde_data_event.
  poly_coeff0 is string_to_list(first(?info),' ,').
  new_file('c:\excel\coeff').
```

```

write('c:\excel\coeff',?poly_coeff0).
close('c:\excel\coeff').
coeff11 is read_line('c:\excel\coeff',1).
coeff12 is read_line('c:\excel\coeff',1).
coeff13 is read_line('c:\excel\coeff',1).
coeff14 is read_line('c:\excel\coeff',1).
coeff21 is read_line('c:\excel\coeff',1).
coeff22 is read_line('c:\excel\coeff',1).
coeff23 is read_line('c:\excel\coeff',1).
if ?coeff14 is cost
then function_type is 0.
if ?coeff14 is profit
then function_type is 1.
text('#e#n#s The function is',?coeff14,'function.').
new_file('c:\excel\coeff').
write('c:\excel\coeff',?coeff11).
write('c:\excel\coeff',#n').
write('c:\excel\coeff',?coeff12).
write('c:\excel\coeff',#n').
write('c:\excel\coeff',?coeff13).
write('c:\excel\coeff',#n').
write('c:\excel\coeff',?coeff21).
write('c:\excel\coeff',#n').
write('c:\excel\coeff',?coeff22).
write('c:\excel\coeff',#n').
write('c:\excel\coeff',?coeff23).
write('c:\excel\coeff',#n').
text('#n#n Please specify the number of units').
ed1 is edit_line(x).
set_focus(?ed1).
end. (* end of dde_data_event *)
end. (* end of Data_event *)

topic x.
x is get_text(?ed1).
disable_window(?ed1).
write('c:\excel\coeff',?x).
close('c:\excel\coeff').
if ?function_type is 0
then function_type is cost.
if ?function_type is 1
then function_type is profit.
text('#n#n Computing the breakeven points,please wait...').
run('c:\excel\proj.pif',0).
breakeven1 is read_line('c:\excel\result',1).
breakeven2 is read_line('c:\excel\result',1).
value_or_not is read_line('c:\excel\result',1).

```

```

poly1 is read_line('c:\excel\result',1).
poly2 is read_line('c:\excel\result',1).
value_or_not is (?value_or_not+0.5) div 1.
close('c:\excel\result').
text(' Done.').
if ?value_or_not is 1
then value_or_not is yes and
    text('#n#n The type of the functions is#s',?function_type,'functions.') and
    text('#n#n The breakeven points are#s',?breakeven1,'and',?breakeven2) and
    do(answer) and
    do(the_altern) and
    text('#n#n What we suggest to choose is alternative:#s',?the_altern) and
    string is 'ALTERN_' and
    do(formula) and
    text('#n#n Do you want to store the consultation? ') and
    l1 is list_box([yes,no],store_or_not).
if ?value_or_not is 0
then value_or_not is no and
    text('#n#n The breakeven points do not have real roots!!
    Please modify the coefficients in the spreadsheet file!') and
    b4 is button(Exit,IIPMSexit,28,22,8,t) and wait().
if ?value_or_not is 2
then value_or_not is no and
    text('#n#n The two competing alternatives are the same!!
    Please modify the coefficients in the spreadsheet file!') and
    b5 is button(Exit,IIPMSexit,28,22,8,t) and
    wait().
end. (* end of x *)

topic IIPMSexit.
clear().
exit_kp().
end. (* end of IIPMSexit *)

topic answer.
if ?function_type is cost
then answer is lower.
if ?function_type is profit
then answer is higher.
end. (* end of answer *)

topic the_altern.
if ?answer is lower and ?poly1>?poly2
then the_altern is 2.
if ?answer is lower and ?poly1<=?poly2
then the_altern is 1.
if ?answer is higher and ?poly1>=?poly2

```

```

then the_altern is 1.
if ?answer is higher and ?poly1<?poly2
then the_altern is 2.
end. (* end of the_altern *)

```

```

topic formula.
if ?coeff12<0
then :sign12=""
else :sign12='+'.
if ?coeff22<0
then :sign22=""
else :sign22='+'.
if ?coeff13<0
then :sign13=""
else :sign13='+'.
if ?coeff23<0
then :sign23=""
else :sign23='+'.
if ?the_altern is 1
then formula is 1 and
    text("#n#n The alternation function that you choose is:') and
    text("#n#s ',?coeff11,*X*X',?sign12,?coeff12,*X',?sign13,?coeff13).
if ?the_altern is 2
then formula is 2 and
    text("#n#n The alternation function that you choose is:') and
    text("#n#s ',?coeff21,*X*X',?sign22,?coeff22,*X',?sign23,?coeff23).
end. (* end of formula *)

```

```

topic store_or_not(item).
store_or_not is ?item.
do(storing).
disable_window(?11).
b5 is button(Exit,IIPMSexit,28,22,8,t).
end. (* end of store_or_not*)

```

```

topic storing.
if ?store_or_not is yes
then storing is yes and
    text("#n#n Storing your consultation...') and
    new_file('c:\excel\consult') and
    write('c:\excel\consult',?string) and
    write('c:\excel\consult',?the_altern) and
    close('c:\excel\consult') and
    text(' Done. ')
else storing is no.
end. (* end of storing *)

```

```

topic setup.
new_file('c:\excel\result').
close('c:\excel\result').
new_file('c:\excel\consult').
close('c:\excel\consult').
end. (* end of setup *)
(*=====THE END OF IIPMS.KB=====*)

```

```

(*===== PROJ.C =====*)

```

```

/*
    This program completes the break-even points computation of two competing alternatives
    based on their function models.
    The program is one part of IIPMS system, which is used to make break-even analyse and
    decisions.

```

```

*/

```

```

#include <stdio.h>
#include <math.h>
main()
{
float a1,b1,c1,a2,b2,c2,x1,x2,x,f1,f2;
int s;
FILE *fp;
FILE *fg;
fp=fopen("c:\\excel\\coeff","r");
fg=fopen("c:\\excel\\result","w");
fscanf(fp,"%f%f%f%f%f%f%f",&a1,&b1,&c1,&a2,&b2,&c2,&x);
if((pow((b1-b2),2)-4*(a1-a2)*(c1-c2))<0)
{
s=0;
goto abc;
}
s=1;
if( (a1-a2)<0.000001 && -(a1-a2)<0.000001 &&
( (b1-b2)>0.000001 || -(b1-b2)>0.000001) ) /* a1==a2 and b1!=b2 */
{
x1=-(c1-c2)/(b1-b2);
x2=x1;
goto next;
}
if( (a1-a2)<0.000001 && -(a1-a2)<0.000001 &&
(b1-b2)<0.000001 && -(b1-b2)<0.000001 ) /* a1==a2 and b1==b2 */

```

```

{
if( (c1-c2)<0.000001 && -(c1-c2)<0.000001 ) /* c1==c2 */
{
s=2;
goto abc;
} /* The two alternatives are the same! */
else /* c1!=c2 */
{
s=0;
goto abc;
} /* Do not have real roots! */
}
x1=-((b1-b2)+sqrt((b1-b2)*(b1-b2)-4*(a1-a2)*(c1-c2)))/(2*(a1-a2));
x2=-((b1-b2)-sqrt((b1-b2)*(b1-b2)-4*(a1-a2)*(c1-c2)))/(2*(a1-a2));
next:
f1=a1*pow(x,2)+b1*x+c1;
f2=a2*pow(x,2)+b2*x+c2;
abc:
fprintf(fg,"%f\n%f\n%d\n%f\n%f\n",x1,x2,s,f1,f2);
fclose(fg);
fclose(fp);
}
□
(*==-----THE END OF PROJ.C-----==*)

```


GLOSSARY

Application Software Programs that can performs specific, user-oriented tasks.

Artificial intelligence (AI) The science of making machines behave in a way that would generally be accepted as requiring human intelligence. Important sub fields of artificial intelligence include robotics, computer vision, speech synthesis and recognition, automated reasoning and theorem proving, natural language processing, automatic programming, automated learning, neural networks, and expert systems.

AI paradigm A mechanism that can be used to represent knowledge in an expert system program; for example: production rule, frames, and object oriented programming techniques.

AI programming language A programming language specifically designed for use in artificial intelligence. Such specialised language provide mechanism and structures that facilitate symbolic reasoning. The two most common of these language are LISP and PROLOG.

ASCII American Standard Code for Information Interchange. This is a standard code used to represent letters, numbers, and special functions as a series of zeros and ones.

Assembly language A low-level language in which each instruction is assembled into one machine-language instruction.

Automatic programming An area of AI research involved in creating AI software that can generate programs from a programmer's specifications.

Backward Chaining Search technique used in production (IF-THEN rule) systems that begins with a possible goal or hypothesis (the action clause of a rule) and works "backward" through a chain of rules in an attempt to find a verifiable set of condition clauses. It is a goal-driven procedure.

Binary code System of numbering in base 2, which uses only 1s and 0s as digits. Because of the simple correspondendence to on-off states of electronic switches, the binary number system is used to "code" information and instructions inside a computer.

Bit-mapped display A display screen that allows a programmer to run each individual pixel on or off.

Blackboard A system architecture that uses multiple accessible processes, called knowledge sources, within its data base.

Blackboard Architecture Architecture that enables independent knowledge sources (and representations) to communicate through a central database called a blackboard.

Cell The structure used in a computer to represent a list. Each cell has two fields for storing data and pointing to other cells in the list.

Certainty Factor A number attached to a rule of fact that denotes the degree of certainty that is assigned to it. The use of certainty factors is a common approach for representing uncertainty in production rule systems.

Chaining A technique for reproducing or approximating part of an expert's reasoning processes by utilising a sequence of rules from a set of production rules. This can involve forward, backward, or mixed chaining.

Chip Single device consisting of transistors, diodes, and other components forming a complex circuit on a 1/4 by 1/4 inch section of a wafer sliced from a crystal of silicon.

Class Term used in object-oriented programming to designate a group of items with the same characteristics. (For example, the car Mustang in a class of transportation).

Cognitive Science The field that investigates the details of the mechanics of human intelligence to determine the processes that produce intelligence in a given situation.

Compiler A program that converts an entire high-level language program into machine language.

Computer vision An area of AI research that is attempting to enable computers to understand visual images.

Conclusion Consequent of a production rule.

Condition Antecedent of a production rule.

Conflict resolution The mechanism in forward or backward chaining that determines which rule should actuate or fire when there is more than one rule in the conflict set.

Control Procedure(s) that affects the order of problem-solving tasks in expert systems.

Control knowledge Facts that influence the selection of the control strategy.

Control strategy Selecting the next course of action given many problem solving tasks.

Database The organisation of files into related units that are then viewed as a single storage concept. The data is then made available to a wide range of users.

Database Management System (DBMS) Software that establishes, updates, or queries a database.

Decision Support System (DSS) Computer-based information system that combines models and data in an attempt to solve non structured problems with extensive user involvement.

Decision Table Table that is used to represent knowledge and prepare it for analysis.

Decision Tree Graphic presentation of a sequence of interrelated decisions to be made under assumed risk.

Deductive Reasoning In logic, reasoning from the general to the specific. Conclusions follow premises. Consequent reasoning.

Default value A value that is used if no other value is specified.

Degree of Freedom (DOF) The "joints" in a robot arm.

Development environment That part of the expert system that is used by the builder. It includes the knowledge base, the interface engine, knowledge acquisition, and improving reasoning capability. The knowledge engineer and the expert are considered a part of this environment.

Distributed AI Splitting of a problem to multiple cooperating systems for deriving a solution.

Domain The application area of an expert system--the problem area of interest; or, the application area in which an expert system is being developed; or, a person with expertise in the domain of the expert system being developed. The domain expert works closely with the knowledge engineer.

Domain Expert Person with expertise in the domain in which the expert system is being developed. The domain expert works closely with the knowledge engineer to capture the expert's knowledge in a knowledge base.

Domain knowledge The facts and rules of thumb of a problem area of application.

Dynamic Explanation Explanation that fits the execution pattern of the rules.

Dynamic programming Path-finding algorithm in speech recognition.

End effector Another name for a robot hand. It also is called a gripper.

Event driven Same as forward chaining.

Expert system An advanced computer program that can, at an acceptable level of competence, solve difficult problems requiring the use of expertise and experience.

Expert system shell A building kit to aid in the construction of expert systems; also referred to as an expert system application generator.

Expert systems A sub field of AI involving the development and application of expert system programs.

Explanation The process of describing how and why an expert system reached a particular conclusion.

Explanation facility The component of an expert system that can explain the system's reasoning, such as how a conclusion was reached or why a particular question was asked.

Fifth Generation The generation of computers that will be build on knowledge-based systems and natural language concepts.

Fifth-generation language Artificial Intelligence languages such as LISP and PROLOG and their variants.

Firing a Rule Obtaining information on either the IF or THEN part of a rule, which makes this rule an assertion.

Forward chaining A search control strategy that starts from facts to arrive at a conclusion.

Frame-based CAI A computer-assisted instruction technique based on the method used in a programmed instruction text. The material presented to the student depends on how the questions asked are answered.

Fuzzy Logic Used imprecise or possibility knowledge, based on fuzzy set theory, to handle uncertainty in expert systems.

Generate and test Heuristic search procedure that generates solutions and tests them for acceptability.

Hardware The computer(s) on which an expert system (or any computer program) is developed or deployed.

Heuristics Information, judgmental knowledge of an application area that constitutes the "rules of good judgement" in the field. Heuristics also encompass the knowledge of how to solve problems efficiently and effectively, how to plan steps in solving a complex problem, how to improve performance, and so forth. (A rule of thumb usually developed through professional experience.)

Hierarchical planning A search technique that produces a hierarchy of abstraction spaces, in each of which preconditions at a lower level of abstraction are ignored.

High-level language A language in which the computer instructions closely resemble English. One high-level language instruction is normally converted into several machine-language instructions.

Human factors This refers to all interfaces between man, machine, and the environment in which they operate.

Icon Visual, graphic representation of an object, word, or concept.

Image acquisition Translation of visual information such that it can be interpreted by brain or by a computer (picture digitisation).

Image Analysis process of determining the major characteristics of a digitised picture.

Image Understanding Final interpretation of a scene by the computer.

Implementation Introduction of a change; putting things to work.

Inference Engine The component of an expert system that controls its operation by selecting the rules to use, accessing and executing those rules, and determining when a solution has been found. This component is known also as the control structure or rule interpreter.

Inheritance A mechanism in a frame or object-oriented system that allows all the information known in general about all members of a class to be considered true for each individual member of the class.

Integrated Circuit An electronic circuit containing multiple electronic components fabricated at the same time in steps on a single slice or wafer of semiconductor material. When separated into individually packaged integrated circuits, they are known also as IC's or chips.

Intelligence Robot A robot that includes AI techniques to allow it to understand its environment and change its action on the basis of external situations. An intelligent robot is known also as a sensor-controlled robot.

Interface Portion of a computer system that interacts with the user, accepting commands from the computer keyboard and displaying the results generated by other portions of the computer system.

Justifier Explanation facility in an expert system.

Knowledge Understanding, awareness, or familiarity acquired through education or experience. Anything that has been learned, perceived, discovered, inferred, or understood. The ability to use information.

Knowledge Acquisition The process of extracting knowledge from the domain expert for developing the knowledge base. This is typically performed by interviewing, scenario-building, and questionnaires.

Knowledge base Collection of facts, rules, and procedures organised into schemes. The assembly of all of the information and knowledge of a specific field of interest.

Knowledge Engineer AI specialist responsible for the technical side of developing an expert system. The knowledge engineer works closely with the domain expert to capture the expert's knowledge in a knowledge base.

Knowledge implementation The process of taking the knowledge found during knowledge acquisition and translating it into an operational expert system program.

Knowledge Representation The process of defining the approach that will be used in an expert system program to represent the domain knowledge found during knowledge acquisition.

Knowledge System (An expert system) Computer system that embodies knowledge; includes inexact, heuristic, and subjective knowledge; the results of knowledge engineering.

LISP (List Processor) An AI programming language that is especially popular in the United States.

Logic programming Language, like PROLOG, based on first-order predicate calculus.

Machine Language A language for writing instructions in a form to be executed directly by the computer. The language is composed of two values; zeros and ones.

Mathematical Model System of symbols and expressions representing a real situation.

Metaknowledge knowledge in an expert system about how the system operates or reasons. More generally, knowledge about knowledge.

Metarule A rule about a rule. Metarules are a type of production rule used in expert systems to specify the conditions under which certain rules should be followed instead of others.

Mixed chaining A reasoning technique used in a production rule system that allows both forward and backward chaining to be used for different parts of the same problem.

Mouse A small, sliding, handheld pointing device that controls the movement give an individual that is a member of more than one class the attitudes of each class.

Natural language An application of artificial intelligence in which the focus is programming the computer to understand language and linguistics.

Neural Network A collection of interconnected neurons, either biological or artificial.

Neuron Nerve cell in a biological nervous system.

Nodes Objects in a linked graph.

Numerical Processing Traditional use of computers to manipulate numbers.

Object-oriented language A programming language that manipulates objects used for declarative knowledge.

Object-oriented programming A set of techniques that allows programs to be built using object as the basic data-items and actions on objects as the active mechanisms.

Object A data structure that contains all the information related to a particular entity. It might be considered a frame with additional features allowing it to contain and invoke methods and to send and receive messages.

Parallel Processing The computer technique of performing several processing actions at the same time.

Pattern-matching An AI technique that recognised and patterns in objects, events, and processes. Computing with several processors working simultaneously, each on a sub problem; then the results are combined.

Physical Integration Packaging of hardware, software, and communications required for functional interaction.

Planning and decision support An area of AI research that is applying AI techniques to the planning and decision-making process to help managers who have decision-making responsibilities.

POPLOG AI language that combines aspects from LISP and PROLOG.

Presentation Language The information displayed; output.

Problem Solving Process in which one starts from an initial state and proceeds to search through a problem space to identify a desired goal.

PROLOG High-level computer language designed around the concepts of predicate calculus. PROgramming in LOGic.

Prototyping Strategy in system development in which a scaled down system or portion of a system is constructed in a short time, tested, and improved in several iterations.

Random Access Memory (RAM) A memory into which data can be placed (written) and from which data be retrieved (read)

Rapid Prototyping In expert system development, quick development of an initial version of an expert system, usually a system with 25 to 200 rules, to test the effectiveness of the overall representation and inference mechanisms being employed to solve a particular problem.

Ready-made Expert System Mass-produced package that may be purchased from a software company. Very general in nature.

Real-time In synchronisation with the actual occurrence of events; results are given rapidly enough to be useful in directly controlling a physical process or guiding a human user.

Robot Re programmable, multifunctional manipulator that is designed to move parts, material, and so on, or that performs assembly, welding, or spraying activities.

Robotics An area of AI research involved in developing intelligent robots.

Rule Formal way of specifying a recommendation, directive, or strategy, expressed as IF premise THEN conclusion.

Rule-based System An expert system made up of production rules; also called a production system.

Rule Interpreter The part of a production system that executes the rules.

Scheduler The program in a blackboard architecture that selects the most likely processing event that will lead to a complete problem solution.

Semantics Meaning in language. The relationship between words and sentences.

Shell A complete expert system stripped of its specific knowledge. In rulebased systems, it is a kind of expert system development tool consisting of two stand-alone pieces of software: a rule set manager and an inference engine capable of reasoning with the rule set built with the rule set manager.

Simulation An AI technique that uses a model of intelligent human behaviour to determine if the computer will exhibit the same intelligent behaviour as a human.

Software Tools (for knowledge engineering) A software package that provides facilities to aid in expert system development.

Symbolic Processing Use of symbols, rather than numbers, combined with rules of thumb (or heuristics) to process information and solve problems.

Symbolic Reasoning (*See* Symbolic Processing)

Syntax Manner in which words are assembled to form phrases and sentences. Putting words in a specific order.

Tool A software package that facilitates the creation of other software.

Uncertainty In the context of expert systems, a value that cannot be determined during a consultation. Many expert systems can accommodate uncertainty; that is, they allow the user to indicate if he or she does not know the answer.

Uncertainty Avoidance Strategy in approaching problems where it is assumed that everything is known with certainty.

User-Friendly Term used to describe a facility designed to make interaction with a computer system easy and comfortable for the user.

User interface The component of an expert system that allows bidirectional communication between the expert system and its user.

Virtual memory A system of managing RAM and disk space so that a computer appears to have more memory than it really does.

Windowing A means of dividing the computer screen into several areas so that a variety of information can be displayed simultaneously.

Workplace (or blackboard) A globally accessible database used in expert systems for recording intermediate, partial results of problem solving.

Working memory the part of an expert system program that contains the data the system has received about the current problem. In addition, any information that the expert system derives about the present problem is stored in the working memory.

ISBN 82-7119-863-7
ISSN 0802-3271