12

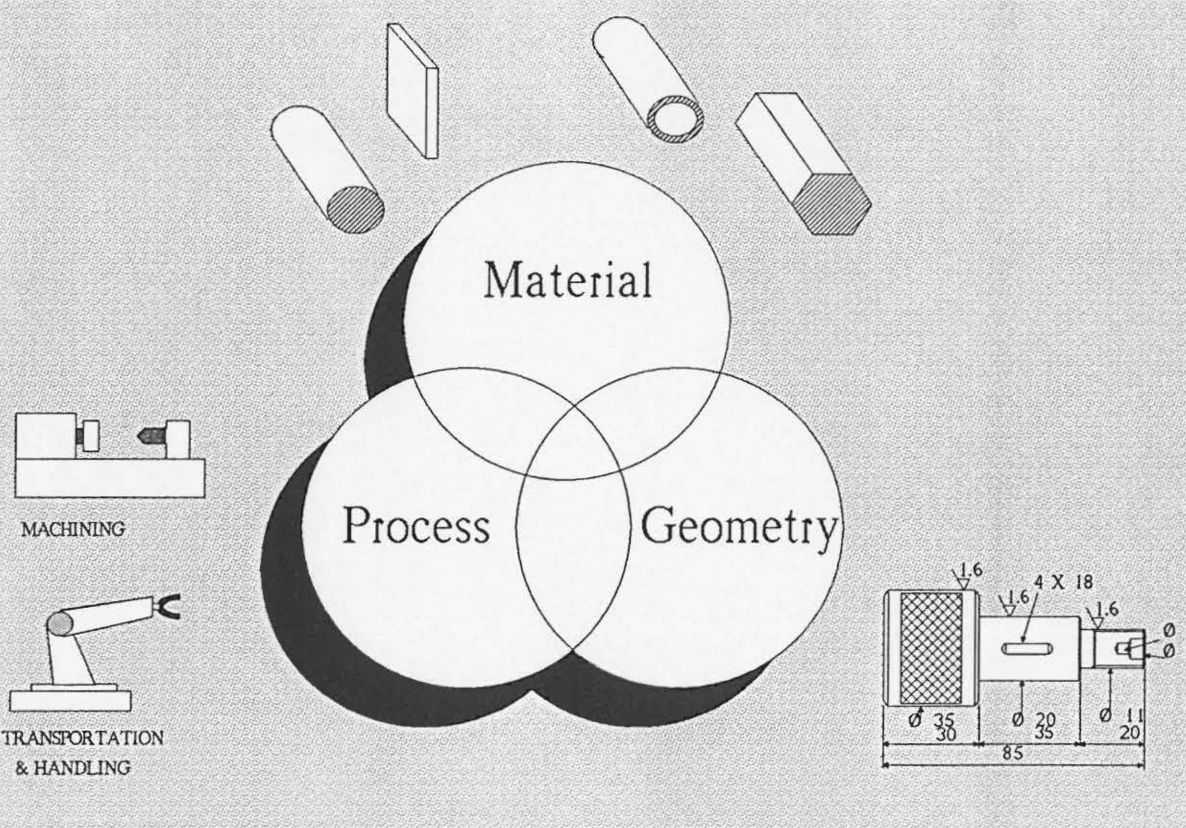
PROCESTEKNISK INSTITUT

DANMARKS TEKNISKE HØJSKOLE

Laboratoriet for Almen Procesteknik

Torben Lenau

KNOWLEDGE BASED ENGINEERING DESIGN AND MANUFACTURING PROCESS PLANNING



Publication No. PI.89.08 – A / AP.89.07

Institute of Manufacturing Engineering, Technical University of Denmark Bldg. 425, DK – 2800 Lyngby, Denmark
Laboratory of Process and Production Engineering

Torben Lenau

KNOWLEDGE BASED ENGINEERING DESIGN AND MANUFACTURING PROCESS PLANNING

Laboratory of Process and Production Engineering
Institute of Manufacturing Engineering
Technical University of Denmark
Bldg. 425, DK-2800, Lyngby, Denmark.

Publication No. PI.89.08-A / AP.89.07

This dissertation is written using the Macrostar word-processor and printed on a HP Laserjet printer. Figures are made in the GEM DRAW program

Laboratory of Process and Production Engineering
Institute of Manufacturing Engineering
Technical University of Denmark
Bldg. 425, DK-2800, Lyngby, Denmark.

ABSTRACT

This Ph.D. dissertation presents the results from the research work carried out under the tittle "Knowledge Engineering within the Mechanical Area - Expert Systems".

The computer techniques known as knowledge engineering or artificial intelligence are described, with emphasis on techniques relevant for selected application areas within manufacturing. Those areas include computerized process planning and intelligent support systems for engineering design. Both areas are vital integrating elements in the work of reaching Computer Integrated Manufacturing (CIM). It is investigated how these areas can be computerized and in this way make them more efficient.

A prototype system for generative process planning (XPLAN) and an intelligent design support system for selection of surface treatment processes (ESOP) are described.

During the research work, the importance of systematic description and development methods was realized, especially for cooperative work. Different methods for description and modelling of engineering activities were therefore investigated and are described.

KEYWORDS

Manufacturing Process Planning, CIM, CAD, CAM, CAPP, Expert Systems, Design Support Systems.

PREFACE

This dissertation is part of the requirements for obtaining the Danish Ph.D. degree (den tekniske licentiatgrad), and documents the work in the project "Knowledge engineering within the mechanical area - expert systems" ("Knowledge engineering indenfor det mekaniske område - expertsystemer").

The research work has been conducted over the past three years, and was performed as part of the manufacturing systems integration efforts of the CIP group within the Laboratory of Process and Production Engineering, at the Technical University of Denmark. Professor Ph.D. Leo Alting has supervised the research.

I would like to express my heartfelt thanks to Professor Leo Alting for creating an inspiring working environment and for his enthusiastic support throughout the research work.

Furthermore I would like to thank my colleague Ph.D. students Arne Bilberg, Steen C. Christensen, Niels Erik Larsen, Morten Als Pedersen and Ole Rindom for very good cooperation and many inspiring discussions. Also I would like to thank Ph.D. Jørgen Jørgensen for his engagement in the initial phases of the project. My thanks also goes to employees at the institute for good cooperation. Special thanks goes to Dr. Peter Sackett for his willingness to serve as an examiner of this Ph.D. work.

I am grateful for the economic support from STVF - The Danish Technical Research Council (the FTU program), without which the research would not have been possible.

As practical remarks it should be mentioned that all the words in the index in the back of the report are shown in *italics* within the text. List of references and bibliography are placed after each chapter as well as in the back of the thesis.

Lyngby, February 1989

Torben Lenau

CONTENTS

CHAPTER 1. INTRODUCTION, HYPOTHESIS AND OBJECTIVES

This thesis work is basically initiated by the need to assure competitiveness in the future for Danish companies. Many other countries make a large effort to improve their position in the international competition, which becomes harder due to larger markets and easy transportation. For several years the Danish competitiveness has been assured through factors like specialized high quality products (e.g. advanced test equipment, large turn-key production equipment), relatively small flexible productions, high reliability/quality and through advanced industrial design (Danish design). There are many competitive factors and to survive a company has to rise those factors to the best within the area.

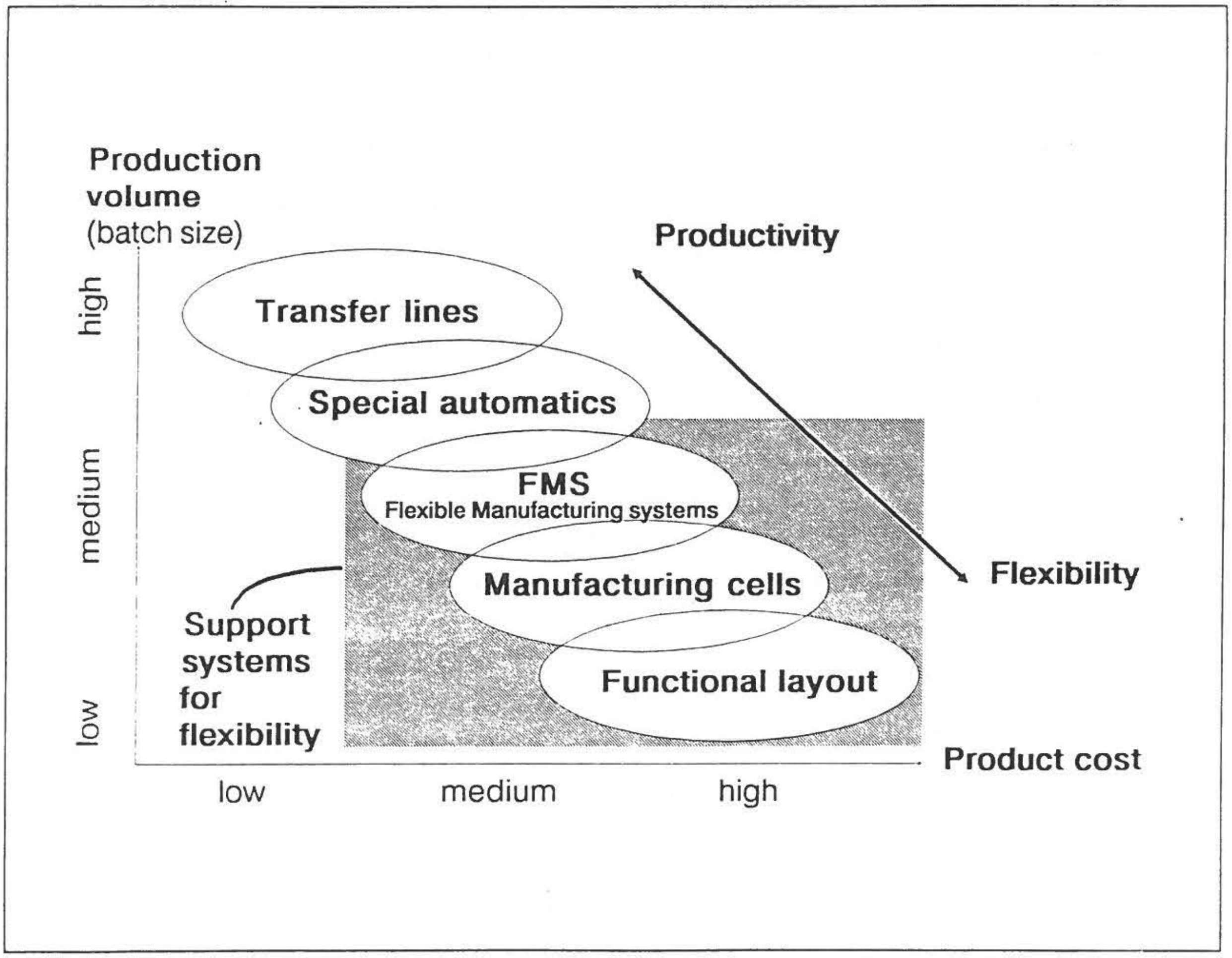


Figure 1.1 Type of production related to production volume and product cost. The broken line indicate the area where the systems described in this thesis are most applicable.

Low volume		High volume	
low cost	**high cost**	**low cost**	**high cost**
Handmade goods Handcraft articles (furniture, clothes)	Adv.measurement equipment Machine tools Adv. medical equipment Ships Airplanes	Calculators Refrigerators Cameras Audio & Video Electronics	Cars Computers Copying machines

Figure 1.2 Product examples related to production volumes and product cost.

The changes that in one company have increased the competitiveness may be useless or have no effect in other companies. The type of product and production have to be taken into consideration before attempts are being made to make improvements. Besides, production volumes and product costs are important factors (see Figure 1.2 for examples). Product complexity is a third important dimension that could be added to Figure 1.1. The figure is primarily valid for relatively simple parts. For more complex parts NC machines will almost always be preferable to conventional manufacturing.

General Objectives:	General production goals:	General production actions:
- Low product prices - High, uniform and documented quality - Fast delivery - Good after delivery support (service, repair) - Specialized high technology products - Knowledge intensive products - Good industrial design (aesthetics, function) - Customer specific products (many variants)	- Faster design, planning and production. - Short lead times - High production flexibility - Minimize work in progress (small product prices) - Easy distribution of product and spare parts	- Faster design, planning and production. - Standardized products and production - Few variants - Few components in products - Smaller production batches

Figure 1.3 General objectives, goals and actions for manufacturing.

For low cost products the four first objectives in Figure 1.3 are most important. To fulfil those objectives it is important to keep the number of product and component variants low. The fewer the parts the easier it is to obtain lower prices, high and uniform quality, fast delivery and a good support after delivery.

Customer specific products are another way to become competitive, especially for low volume high cost products. The consequence of producing individualized products can be a larger number of variants and a more complex production.

This thesis examines some of the means that will assure a leading position through improving the production flexibility, better utilization of the production equipment and through improved quality, and it focuses on production preparation and design activities in the company. The improvements will give the company the advantage of faster delivery and improved quality, and the possibility of more order based production and smaller product volumes. The work is oriented toward mechanical manufacturing characterized by low to medium size production volumes and a medium to high production cost, but the results may also be used within other manufacturing areas.

The **hypothesis** of this work is that design and production preparation activities have a major impact on *production flexibility*, *production utilization* and *product quality*, and that improvements can be reached through an integration of design, production and production preparation, i.e. a better and faster access to and exchange of information about production. The integration can be facilitated through organizational changes (e.g. product development groups) and through information and planning tools (on computers or on paper). The thesis focuses on computer information and planning systems.

The overall **objective** of this thesis work is to investigate how advanced computer techniques (especially knowledge engineering) can be used to facilitate the information flow between design, production and production preparation in order to improve the company competitiveness. The goals are to get an overview of the different techniques and tools in knowledge engineering and to develop prototype systems for selected areas. Those areas are manufacturing process planning and design support systems for process selection. The areas are central elements in the work of reaching computer integrated manufacturing, and they are considered well suited for knowledge engineering because of their knowledge intensive, ill-structured nature.

In the remaining part of this chapter knowledge engineering, process planning and design support systems are introduced. A more comprehensive description can be found in the following chapters.

Knowledge engineering

Knowledge Engineering (*KE*) is a part of the broader research field *Artificial Intelligence* (*AI*), which also includes tasks like advanced motion (robotics) and voice recognition. One of the aims of knowledge engineering is to make it possible to tell the systems <u>what</u> to do rather than <u>how</u> to do it. This should make it possible to concentrate the effort on the problem itself rather than on program details, and therefore use more time to experiment with prototypes and in that way reach a better result.

Generally KE should help to preserve knowledge and make it accessible for more people. It is a tool that can be used to formalize rules of thumb and other ill-structured knowledge (*heuristic knowledge*). This formalization can furthermore be used to uncover other areas or sub areas that need to be systematically analyzed.

Process Planning

In process planning production facilities are selected and the manufacturing operations planned. Computer systems that facilitate automation of process planning are called *computer aided process planning systems* (*CAPP*). CAPP systems can help to speed up the process planning activity and to ensure a higher degree of standardization. Both traditional process planning and advanced CAPP systems produce a set of manufacturing instructions based on information about the finished design.

Often process planning is considered as an isolated task where a trained person looks on an blue print of the part, and then based on his/hers experience plans how the part shall be manufactured. This is only half the truth, since many of the manufacturing decisions for a part is actually taken by the designer. When the designer for example decides what material he will use for the part, he has also set up restrictions for which processes that possibly can be used. Many times he has actually a specific process in mind when designing the part e.g. die casting which influences the geometry of the part. In this way limits are set up for which processes that can be used, and what is even more interesting, limitations for which alternative processes that a process planner can choose from. Since the designer usually creates a design/ shape with a certain process in mind and the design therefore to some degree is tailored to this particular process, process planning and CAPP systems can only produce as good plans as the input allows. This is one of the reasons for considering design support systems.

Design support systems

When discussing integrated production it is important to focus on how design and production information correlate. It is generally believed that the designer with his decisions is responsible for 70-80% of the product costs [*Suh*.88], since raw materials, processes, production methods incl. assembly more or less are determined in this phase. CAD systems enable modelling the product and storing this model in a database, but there is no access to production knowledge, that could contribute to a more optimal design. There is for this reason a need for advanced information systems that can supply the designer with the relevant information, relevant design rules and company standards.

Through the use of knowledge engineering techniques it is possible and realistic to represent knowledge about production processes, possibilities, limitations, advantages and consequences, and use this in intelligent consulting systems for designers or process planners.

In order to get the right product at the lowest price it is necessary that the designer can make qualified manufacturing decisions. To do this the designer needs supporting tools that can assist him in selecting manufacturing processes and in investigating the requirements that those processes imply. Quick and reliable answers are needed to questions that arise in the design phase, e.g. can a certain material be welded, can it be used in acid environments, which processes can produce a special geometry or information on design rules for die-cast parts. While the designer is using the CAD system he should be able to get such information from an expert system, which is tailored to be a natural part of the design procedure. Such a system can be called a *design support system*.

The thesis describes how systems for process selection can be built. Artificial intelligence and knowledge engineering offer new possibilities for automation of process selection and many other areas within manufacturing, and it is discussed how production knowledge can be represented and handled in a knowledge based system. Different methodologies for design of design support systems are described.

A system for selection of surface treatment processes that is developed as a joint program with major Danish companies is also described. Surface treatment is a fast developing area with many new and improved processes, and it is therefore an excellent area for investigating the possibilities of design support systems. The system is based on an programming environment for expert system development, where knowledge is represented as frames and rules.

The term process

The term *process* is used in many contexts with different meaning, and it shall therefore be explained how it is used here. A process can generally be described as an activity where material, energy and information are transformed into the desired part. In this context a process is used as a synonym for a manufacturing method, e.g. turning, drilling, milling, forging or surface treatment. A machine can normally accomplish a single process (e.g. turning on a lathe), but in some cases a single machine covers more processes (drilling and milling on a milling center). In other words, a process is a general term that implies a group of machines or manufacturing methods with common characteristics.

References - Introduction

[Suh.88] Suh, Nam P.: Basic concepts in Design for Producibility, Keynote paper, Annals of CIRP vol 37/2/1988, p.559-567.

CHAPTER 2. COMPUTER INTEGRATED MANUFACTURING

In this chapter the different activities and functions in a manufacturing environment are described and the role of process planning and design support is discussed from an integration point of view.

2.1 Manufacturing

The term manufacturing is very broad and covers all of the activities in the modern company as shown in Figure 2.1. In the following these company functions are very briefly described in order to illustrate the complexity of a manufacturing facility.

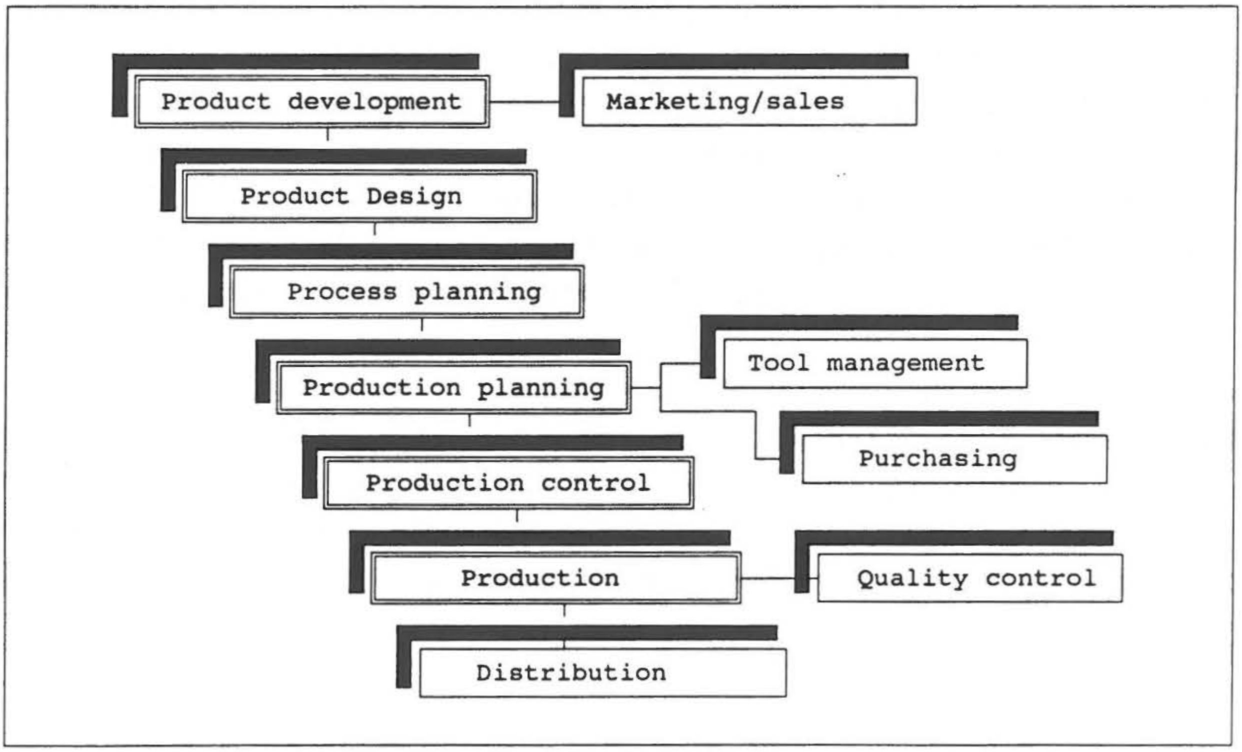


Figure 2.1 Functions in manufacturing, the double line indicates functions in CIP - Computer Integrated Production.

The product development work can be initiated from several places but most often the input comes from the sales/marketing department. They have either received a request for a product, or through market research realized the need for a new product. Product development launches many alternative product ideas and compares them with respect to initial requirements, feasibility, manufacturing possibilities, company policies, etc. The best alternative is then handed over to product design where the more detailed considerations are done and blue print drawings and other product documentation made. In process planning suitable production equipment is selected and planned in detail.

Production planning determines when the various parts should be produced or bought in order to minimize number of parts in storage and to obtain the best utilization of the production equipment. The rough planning is often made by specialized personnel on MRP-systems (Manufacturing Resources Planning), while the more detailed planning at the most is done by an experienced person in the workshop. In order to make the detailed planning more efficient production simulation techniques on computers can be a significant help. Tool management ensures that the necessary production tools are ready when required, and plans the production tool maintenance. The purchase department buys materials and components at the right time to the (hopefully) best price. Production control ensures that the production plans are kept and takes action when something goes wrong. In the production the material is shaped and the products made. This can be done in manual workshops, or on more automatic NC-machines with advanced transport and handling equipment like AGVs or robots. Quality control checks if the product fulfils the requirements and the distribution handles the packaging and transport of the products to the wholesaler or the end consumer.

2.2 Computer Integration

Within the past many years emphasis has been on improving the efficiency of each of the company functions described in the previous section, but these improvements do not necessarily improve the company efficiency as a whole. Large efforts are therefore put into integration work in these years.

CIM - Computer Integrated Manufacturing denotes the integration of the functions involved in specifying and producing products. CIM do not only involve integration of all the functions in a specific company, but also functions in other companies can be relevant, e.g. subcontractors. Most often the emphasis is on computers and not on integration. Several machine tool vendors today offer CIM systems, but those systems mainly focus on the hardware integration. Integration of the different decision making and planning functions is very complex and company dependent and accordingly difficult to implement in a standard system. At The Institute of Manufacturing Engineering (*IME*) different research programs within CIM are carried out jointly under the title *Computer Integrated Production* (*CIP*). This includes research on intelligent CAD, computer aided process planning and production simulation with a miniature manufacturing cell (*CIM Minilab*) as a test environment. CIP covers the company functions shown in Figure 2.1 with a double line. A more detailed description of the different CIP areas can be found in [*Alting*.86.a], [*Alting*.86.b], [*Christensen*.88], [*Pedersen*.88], [*Larsen*.89] and [*Bilberg*.89].

Companies are usually split up into a number of separated functions (e.g. as in Figure 2.1), that each can be overviewed and controlled. Each of the company functions can be optimized isolated, but due to interface problems between the different company functions the company activities are not optimized as a whole. The traditional functional organization structure in a company can therefore be a significant obstacle for the attempts of reaching an integrated factory. True integration does therefore require both an enhanced information structure and changes in the organizational structure.

Computer Integrated Manufacturing involves not only an automation and computerization of the different manufacturing functions, but also how the functions can be integrated. Here it is important to have tools and methodologies for describing and modelling such an integrated system. Examples of methodologies/tools are the IDEF methods developed in the ICAM project, the cognitive engineering framework, and the three level methodology described in chapter 4.

Process planning and design support systems for integration

Both design support and process planning play a central role in the work with integrated manufacturing. The development within both activities seek to integrate product design and production activities. When planning computer integrated manufacturing it is important to remember the decision making at different levels and in different activities in the company. The integration does not only concern the exchange of data but also how and where decisions are taken.

Justification of Computer Integrated Manufacturing

Normal accounting practice can be difficult to apply on investment analyses for most of tools in Computer Integrated Manufacturing. The reason for this is that the benefits are difficult to capitalize, and that most of the tools are "state-of-the-art" (new to everyone) with almost no previous financial experiences. Costs are normally a good measure for the judgments of the soundness of new investments (see section 5.11). For example new production equipment can be justified from cost savings on reduced labor and faster production. Many of the computer tools, e.g. CAD, CAPP, design support systems, bring benefits like improved company flexibility and more standardized products. Even though these benefits are obvious, they can be very difficult to measure economically.

[*Evans* & *Sackett*.85] describe the problem and present a method for justifying a "state-of-the-art" computer tool. The method supplies practical guidelines for how to justify a project to the financial department or to the board of directors. Economic effects should be set up whenever possible and illustrated with optimistic and pessimistic figures. To explain the benefits, advantages can be listed and compared with former employed techniques or the situation in rival companies.

References - Computer integrated manufacturing

[Alting.86.a] Alting Leo: Integration of Engineering Functions/Disciplines in CIM, Annals of the CIRP Vol. 35/1/1986, p.317-320.

[Alting.86.b] Alting L., Lenau T.: Integration of the Engineering Functions in the CIM Philosophy, ISATA, Flims, Switzerland, 6-10 oct. 1986.

[Bilberg.89] Bilberg Arne: Simulation as a tool for developing FMS control programs, Ph.D. thesis, Lab. of process and production engineering, Technical university of Denmark, Lyngby, Denmark, January 1989, 258 pages.

[Christensen.88] Christensen Steen C.: Computer Integrated Manufacturing Miniature Laboratory (CIM-ML), Overall structure, database and communication interface, Ph.D. thesis, Institute of Manufacturing Engineering, The Technical University of Denmark, Publ.no. AP.88.16, July 1988, 160 pages.

[Evans & Sackett.85] Evans Steven & Sackett Peter J.: Financial justification of state-of-the-art investment : a study using CAPP, CAPP computer aided process planning, CASA/SME, Editor Joseph Tulkoff, 1985, p.47-50.

[Larsen.89] Larsen Niels Erik: Simulation - a concept for production scheduling, Ph.D.-thesis, Lab. of process and production engineering, Technical university of Denmark, Lyngby, Denmark, January 1989, 398 pages.

[Pedersen.88] Pedersen Morten Als: Computer Integrated Manufacturing Miniature Laboratory (CIM-ML), Cell control, robot programming, communication, Ph.D. thesis, Institute of Manufacturing Engineering, The Technical University of Denmark, Publ.no. AP.88.17, July 1988, 116 pages.

Bibliography - Computer integrated manufacturing

[Gibbons.88] Gibbons Albert J.: A standards framework for the computer-integrated enterprise, CIM review, Vol 4, No. 3, p.10-17.

[Kochlar.79] Kochlar A.K., Development of computer-based production systems, Edward Arnold Ltd., UK, 1979, 250 pages.

[Solberg.85] Solberg J.J., Anderson D.C., Barash M.M., Paul R.P.: Factories of the future: defining the target, CIDMAC, Purdue University, In. USA, 66 pages.

[Yeomans.87] Yeomans R.W., Choudry A., Hagen P.J.W.ten: Design rules for a CIM system, North Holland, Amsterdam, 1987, 454 pages.

CHAPTER 3. KNOWLEDGE ENGINEERING

In this chapter the basic concepts for *knowledge engineering (KE)* are explained, and techniques for *knowledge based systems (KBS)* are described.

Knowledge engineering is a fairly young scientific discipline which is illustrated by the unclear terminology and the lack of clear definitions. A few definitions are therefore appropriate. Artificial intelligence is the broadest term and covers various disciplines that imitate human activities like thinking, advanced motion, vision, etc. Knowledge engineering is concerned with the brain activities and the resulting computer systems are called knowledge based systems or *expert systems*.

A number of different definitions for knowledge engineering exist ranging from functional to technical definitions. A *functional definition* concentrates on the task the system is going to solve and on the system behavior. A computer system that acts like an expert within a certain area and solves problems that normally require an expert will be called an *expert system (ES)*. A computer system is an expert system if it solves complicated problems that else require expert assistance.

The *technical definition* looks on the system structure and the systems are referred to as knowledge based systems (KBS). The basic elements in a KBS are a knowledge base, a control structure, a temporary memory and a user interface as shown in Figure 3.1. Some people consider the temporary memory as a part of the knowledge base but it will here be treated separately. A KBS does furthermore usually have procedures for acquiring new knowledge and for updating of the existing. In the knowledge base there are two basically different types of knowledge called procedural and declarative knowledge.

Since knowledge engineering and expert systems are not unambiguously defined, it is also difficult to state what the advantages and limitations are. Characteristics that often are stressed include:

- a computer system that contains the knowledge of an expert, and can simulate the reasoning of the expert.
- a computer system that can reason on the basis of knowledge about cause/effect relations (*heuristic knowledge*) rather than on knowledge about deeper functional relations.
- dynamic knowledge structure that is easily changed.
- a computer system that can recognize and formulate a problem on the basis of observations (generalize from examples).
- a computer system that can increase its own knowledge (learn).

The majority of present expert systems concentrate on the first three characteristics, while the latter two to a large degree are wishful thinking.

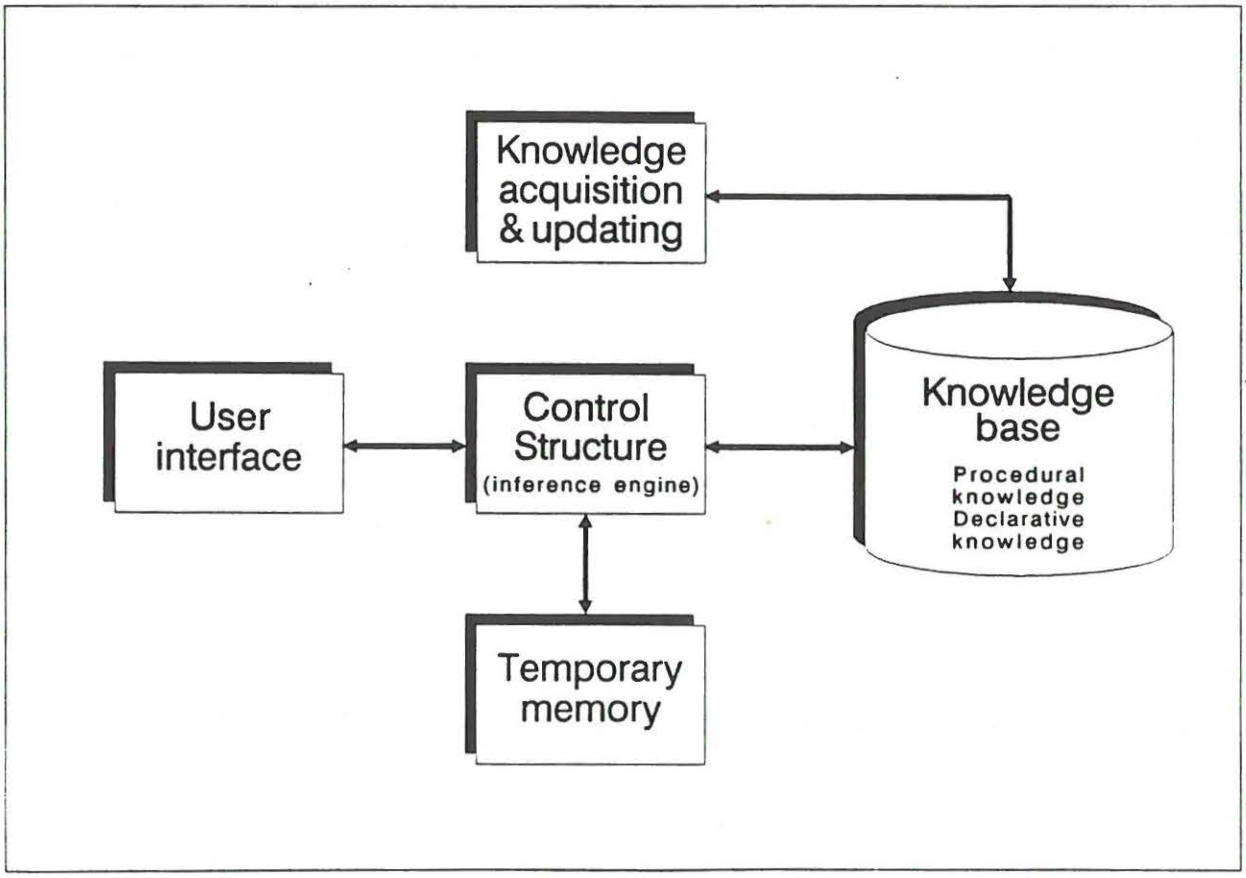


Figure 3.1 Structure of a knowledge based system.

3.1 Knowledge representation

The central element in computer processing has for many years been **data** like numbers and facts. The computers were good at calculating and manipulating large amounts of data. Today it is realized that the computer has to do more than just manipulating data. Instead of data the central element in computer processing today is **knowledge.** Knowledge is a broader term that covers both data and how to manipulate those data.

Knowledge is roughly divided into declarative and procedural knowledge [*Nilsson*.82], and together they form the knowledge base. *Declarative knowledge* is the factual part of the knowledge like data and relations between the data, while *procedural knowledge* concerns how to alter or change the declarative knowledge. Declarative knowledge deals with objects and events while procedural knowledge has to do with actions.

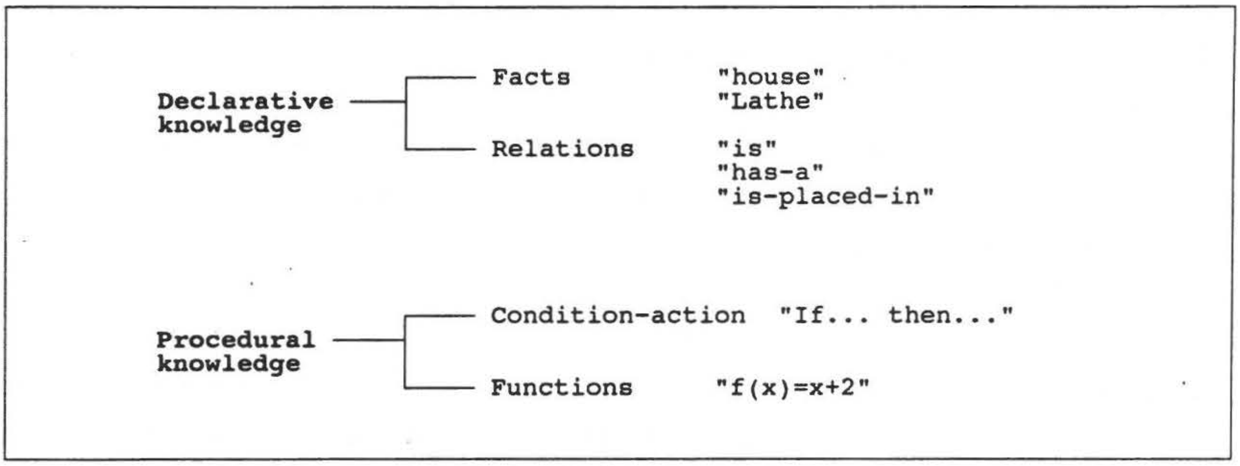


Figure 3.2 Declarative and procedural knowledge.

In traditional programming languages (called *sequential languages*) declarative and procedural knowledge are separated since the procedural knowledge is represented as the program itself and the declarative knowledge is kept on files as data. The consequence is that the programs are static and hard to change. In knowledge engineering declarative and procedural knowledge are represented together and this gives a number of advantages and possibilities.

Declarative knowledge is sometimes levelled in classes and class members [*Winston*.88]. The idea of levelling knowledge about objects in classes and class members is that some knowledge is more common and therefore not needs to be represented individually for each object. Reasoning is often done on a general level of knowledge, after which the assumption is that the results also apply to the specific knowledge.

In order to communicate knowledge to other people verbally, on paper, through a computer or some other media, it is necessary with some commonly agreed upon conventions on how to represent the knowledge. A knowledge representation can be described through its syntax and its semantics. In other words a representation is a set of syntactic and semantic conventions. A *syntax* defines the symbols in a representation and how those symbols can be arranged. The *semantics* specify how to represent knowledge in symbol arrangements allowed by the syntax. In languages like English the syntax defines the different word classes like verbs and nouns and how they can be arranged according to each other. The semantics concern the meaning in the sentences. A piece of data usually consists of a label that identifies what type of data it is and one or more values. The semantics are specified by a description of how particular syntactic constructions relate to get something done.

Representation of declarative and procedural knowledge can be either very similar or quite different or something in between. In the following a number of different representation techniques will be described.

Predicate calculus

Predicate calculus is a formal language of symbol structures i.e. a syntax, and is described by [*Hayes-Roth*.83] and [*Clocksin & Mellish*.84]. The syntax is built up from objects (called terms) and predicate symbols, where an object holds the name of a thing and the predicates represent the relations between the things. A simple example of a predicate calculus statement is

(is-a shaft bar)

that in other words expresses the fact that a shaft is a bar. The words "shaft" and "bar" are objects while "is-a" is a predicate. Logical connectors like "and", "or", "not", "implies" and "is equivalent to", and quantifiers like "for all" and "there exists", make it possible to express declarative knowledge in a useful way. A more advanced example of predicate calculus that uses a quantifier is the following statement (called a *clause*)

for all x: [shaft(x) -> bar(x)]

The clause expresses that all shafts are bars.

The syntax called first order predicate calculus is the basis for the so-called *logic programming* where *Prolog* is the most well known language [*Clocksin & Mellish*.84]. It is called first order because the variables can only hold objects and not predicates.

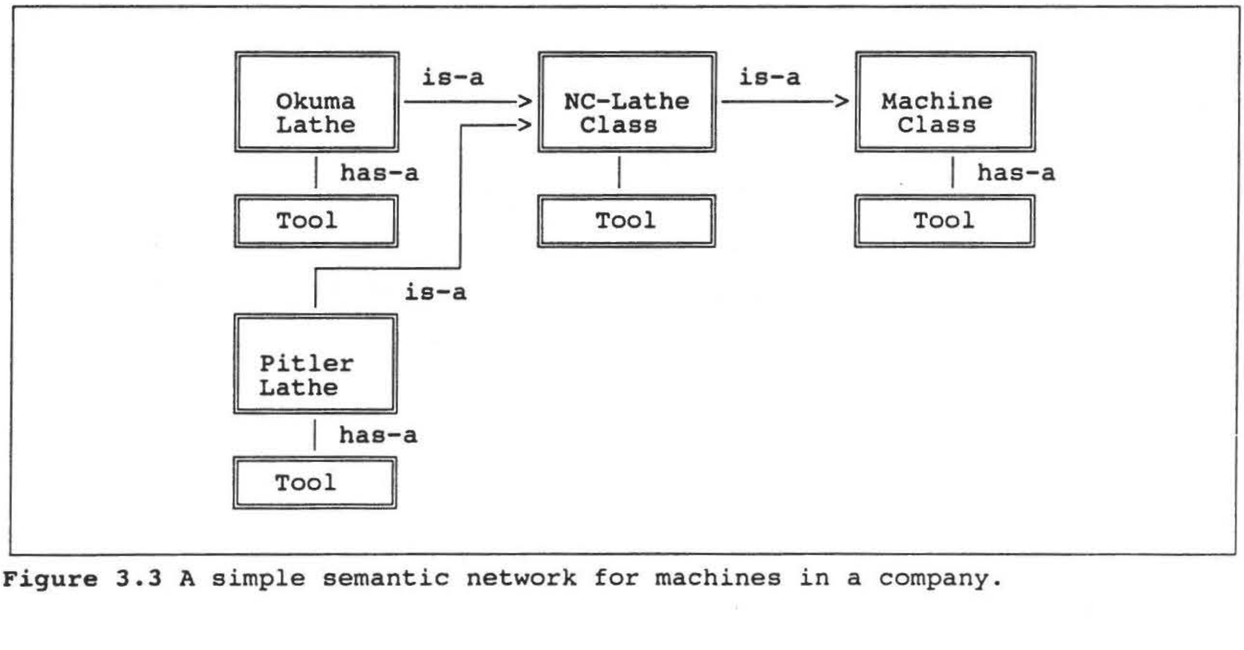


Figure 3.3 A simple semantic network for machines in a company.

Semantic networks

Semantic networks are network structures that represent the semantics (the meaning) of the structures in the described subject [*Waterman*.86], [*Winston*.84]. A semantic network can be represented as a number of points (called nodes or objects) connected by links (called arcs or slots) that describe the relations between the nodes. Where predicate calculus is a formal syntax, semantic networks are a method to describe semantics, i.e. the meaningful content of the thing that is described. An example of a simple semantic net is shown in Figure 3.3.

As mentioned previously knowledge can be expressed on a general level for a class of things and on a more specific level for the individual class members. In order to get from the general level description to the specific level description a number of mechanisms (called facets) can be used. Three basic mechanisms are *inheritance*, *demons* and *defaults* [*Winston*.84].

Knowledge on the general level does in the most cases apply on the specific level as well. The inheritance mechanism makes it possible for each of the class members to inherit slot values. For example it can be assumed that the NC-lathe in Figure 3.3 has a tool since it is known that the more general class "machine" has it.

Demons can be used to express procedural knowledge since they hold an if-needed procedure that can perform some action. A demon is placed in an slot and does normally calculate a value that is placed in the node that the slot is pointing at. Demons are not limited to numerical functions but can hold any function. A simple calculation example for cutting parameters is shown in figure 3.4.

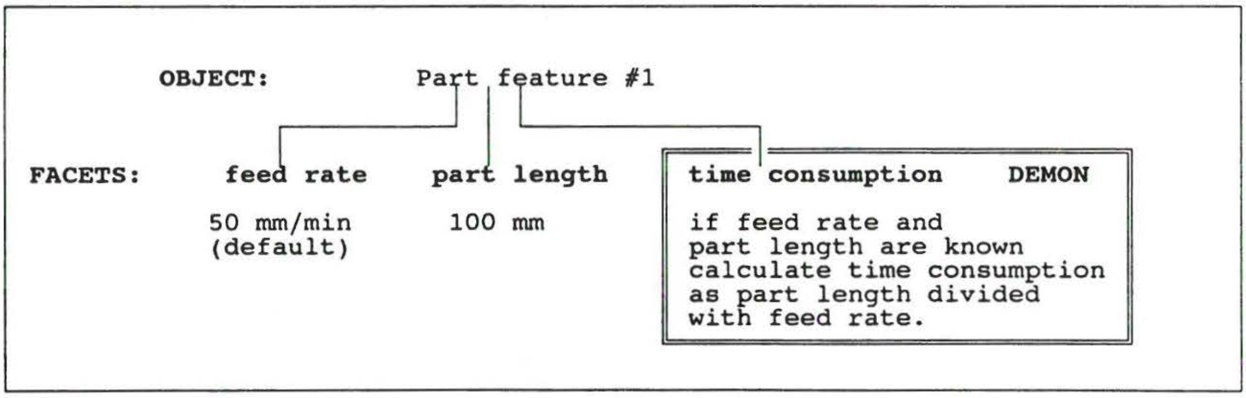


Figure 3.4 A demon (if-needed procedure) for time calculation.

In many cases only some of the necessary information is known and a likely assumption can therefore be used. The default facet serves this purpose. In Figure 3.4 the default value 50 mm/min. will be used for time calculation if no other value has been added.

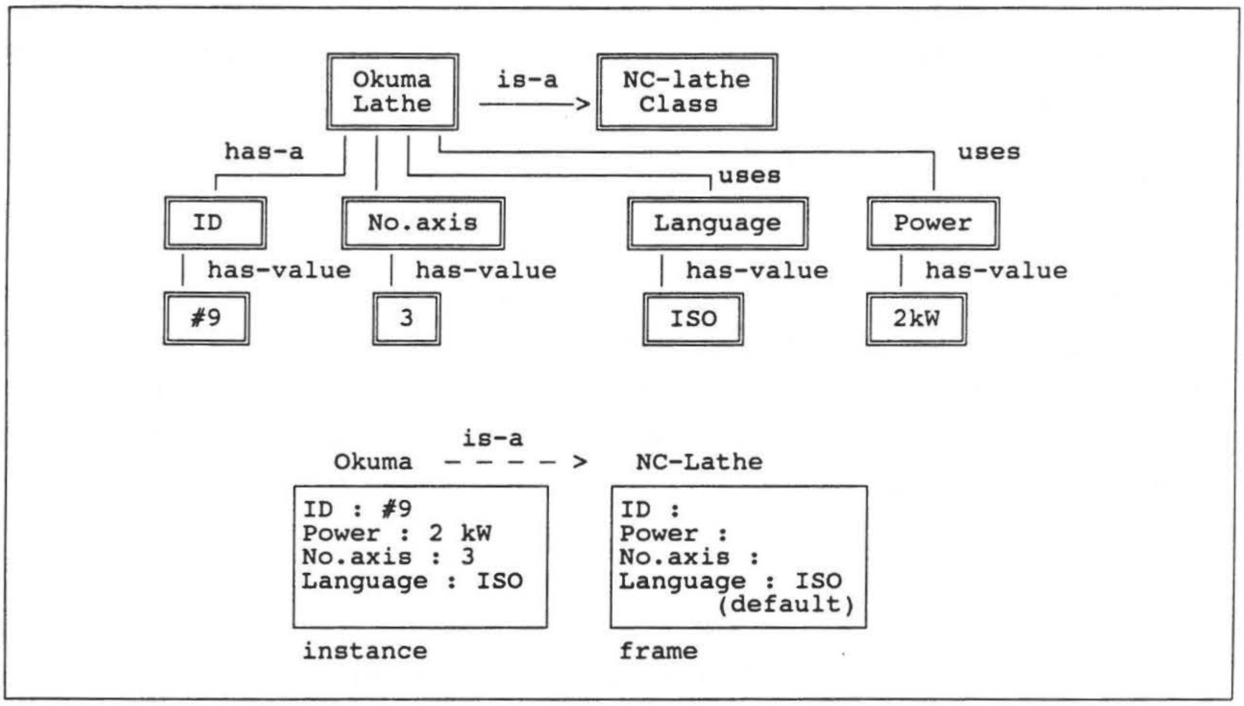


Figure 3.5 Semantic net and frame representations for an NC machine.

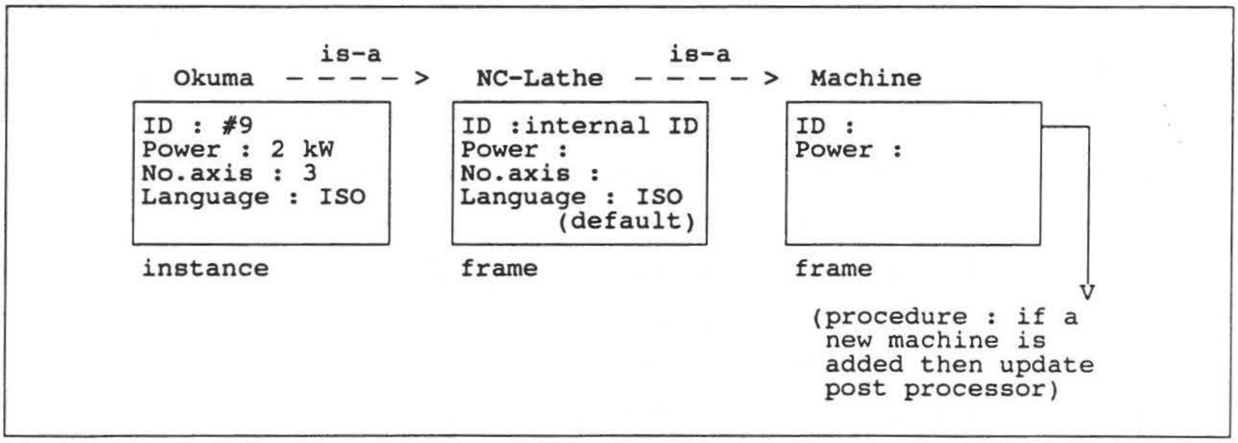


Figure 3.6 A frame representation for machines in a company.

Frames

A more general way of representing knowledge is a structure called *frames* [*Waterman*.86], [*Winston*.84]. A frame is a collection of semantic net nodes and slots that together describe a subject (Figure 3.5). The subject can be a physical object with properties like shape, weight, dimensions, color, etc. or it can be a concept like an act or an event. General class knowledge is handled in the frames while specific knowledge for class member is handled in *instances* of the frames. The frames can be arranged in hierarchies or networks and frames on lower levels can inherit from higher level frames. An example of a frame system is shown in Figure 3.6.

Rule representation

Rules in rule based systems are used to express conditional declarative knowledge and procedural knowledge about actions to be performed and their prerequisite conditions. A rule based system is also called a production system and the rules therefore *production rules*. Production rules have the form

IF (condition) THEN (action).

When the IF portion of a rule is satisfied by the facts, the action specified by the THEN portion is performed.

3.2 Control structure (Inference engine)

An important element in a knowledge based system is the *control structure* that also is called the *inference engine*. The control structure contains procedures for specification of problems to be solved and how to reach solutions. This means that there are two major tasks performed by the control structure, goal definition/reduction and problem solving. In different applications the tasks may have different importance. Sometimes it is clear what the problem is and the system will therefore mainly be concerned with problem solving. In other cases it can be very difficult to determine what the problem actually is, but when it is found the solution follows easily.

Both control tasks involve the need for basic techniques to exploit constraints, efficient search and exploration of alternatives.

Generally a control structure consists of a set of control options like where knowledge about procedures are stored, which process decides that a procedure can act, how computer resources are allocated and how procedures communicate.

Temporary memory

The inference engine keeps track of its solutions in a temporary common memory (also called an an *assertion base*). The temporary memory holds all the variables and conclusions that only is used in the actual session. This memory can be used to examine what happens if some answers are changed, and to backtrack if the user realizes that he has given a wrong answer. The assertion base can also be used to show the current status, i.e. the conclusions and lines of reasoning found by the system so far. By separating the knowledge and the assertion base it is possible to "start over" simply by initializing the assertion base.

3.3 Basic control options

Storage of knowledge

A central element in the control structure is where the procedural knowledge is stored. There are 3 basic ways to store the knowledge called action, object and request centered control. In *action-centered control* the procedures know what subprocedures to use to perform actions. In most of the sequential program languages this is the case, since each procedure knows the names of a number of other procedures it can use. In *object-centered control* the class descriptions specify how to deal with objects in their own class. This is the case in object oriented programming. In *request centered control* the procedures know their own purpose so that they can respond to requests and tell about their qualifications in the given situation, after which the calling procedure can select the best suited.

Communication between procedures

Communication between procedures can be carried out in two basically different ways. A procedure can send a message (in form of a number, text or some other format) directly to the other procedure that the message is intended for. This is called the *private line method* because only the two involved procedures have access to the message. Action centered and object centered systems do usually communicate through private lines.

Another possibility is that the message is placed in some common place called a blackboard where other procedures can see the message, react and place new results on the blackboard. This is referred to as the *blackboard method*. In most cases it is not convenient that all procedures can read all messages and a variant of the blackboard method called the *reserved-spot* method can then be used. Here a procedure puts messages at places where only a subset of the other procedures can read them.

Constraint propagation

In many cases things are described through their constraints. A real estate property is defined by its fences, a car by the limits for its performance and a computer by storage and speed limitations. Constraints can be expressed in several ways. One very often used is sets of equations, that can be solved by substitution. Such a set of equations can also be represented by a network where the nodes are either operators (+,-,*,:,etc.) or constants, and the links are the variables as shown in Figure 3.7. *Constraint propagation* can solve the set of equations by finding the nodes with only one unbound link (variable) and calculating their values. The nodes represent constraints for the variables and by calculating values for the unbound variables the constraints are propagated through the net. Spread-sheet programs make extensive use of constraint propagation.

```
A = 5
B = 7
C = A * B
D = C - A
E = F + D

Result:
5*7=35=C, 35-5=30=D.
```

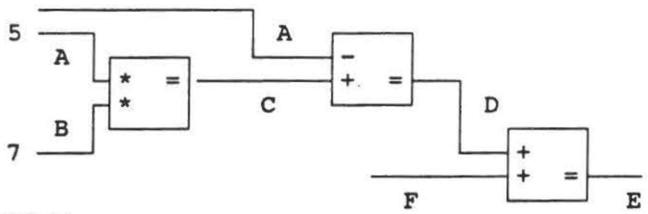


Figure 3.7 Constraint propagation for a set of equations.

Dependency directed backtracking

To get from a problem to its solution many subproblems have to be solved. Each subproblem leads to one or more alternative subsolutions and the total problem/solution space forms a tree where some of the branches lead to solutions and other are blind alleys. When ending in a blind alley in the search for a path through the tree there are different techniques to find new paths. The simplest is chronological backtracking that backtracks to the last untried alternative while it withdraws the selections made in the blind alley. This can result in a very comprehensive search and another technique can therefore be used. *Dependency directed backtracking* keeps accounts of the dependencies between alternative paths and the goal. When backtracking from blind alleys it selects the alternative that is most likely to lead to the goal.

Dependency directed backtracking is very often used in planning problems where a number of goals and requirements have to be fulfilled with the consequence that there are many blind alleys.

Exploration of alternatives

In knowledge engineering there often is a need to determine one or more alternative solutions to a problem. It is therefore necessary to have search procedures that find paths through a search space from an initial state to a goal state. This is a logistic problem and a few of the basic logistic procedures will be explained.

A search problem can be illustrated by a network with nodes and connecting lines (links). To each link can be attached a cost factor that tells how good the link is compared to other links. One node is the starting point and denotes the initial situation and another node is the goal node and it stands for the desired situation. The alternative possible paths through this network form a tree as illustrated in Figure 3.8.

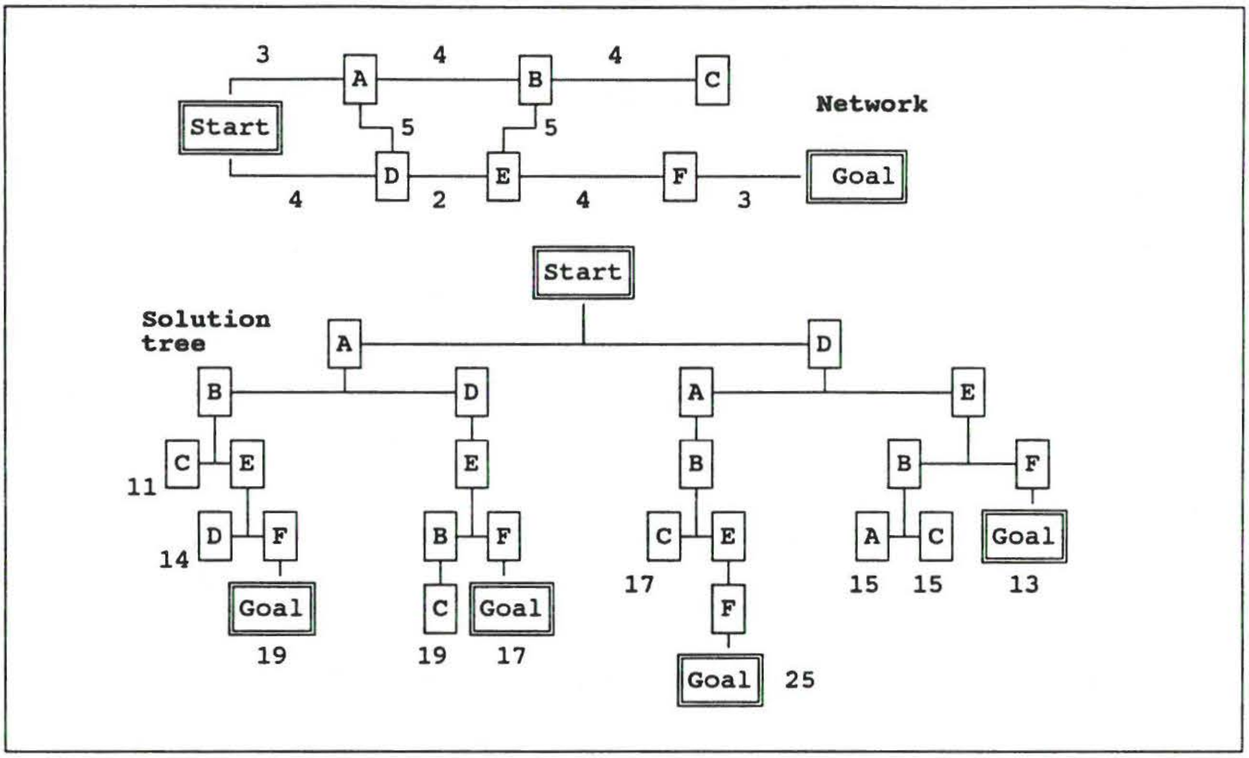


Figure 3.8 A search network and the corresponding solution tree.

Finding paths

Finding paths are important when determining how to get from an initial state to a goal state. In many cases it is sufficient just to find a path that not necessarily is the shortest one. In those cases one of the following *search procedures* can be used: depth first search, breadth first search, beam search, hill climbing and best-first search.

Depth first search picks an alternative at random at each ramification and continues that way until a leaf is encountered.

If the leaf is identical to the goal node then the solution is reached and the procedure stops. In other cases do the procedure backup through the solution tree until it finds an alternative path and this path is then examined. In the example in Figure 3.8 the search order for dept first search would be:

Start-A-B-C--B-E-D--E-F-Goal.

Breadth first search examines all nodes at each level in the solution tree to see if it is the goal node. Unless there is a short (few levels) path from the start node to the goal node, this method will be quite time consuming. The search order for the example is:

Start-A-D-B-D-A-E-C-E-E-B-B-F-D-F-B-F-C-E-A-C-Goal

Beam search is like breadth first search because it progresses level by level. The difference is that beam search only examines the best X nodes at each level. The search order for X=2 for the example is:

Start-A-D-B-D-A-E-C-E-B-F-A-C-Goal

In *hill climbing* the procedure at each level selects the node with the smallest cost. When a leaf that is not the goal is reached the procedure backs up to the last level with alternatives and from here chooses the link with the lowest cost except for the links already tried. The search order for the example is:

Start-D-E-F-G

Best first search chooses the node with the lowest cost among those nodes that can be reached from the nodes examined so far. The search order for the example is:

Start-A-B-C-D-E-F-Goal

Finding shortest paths

When it is important to find the shortest path there are two basic different search algorithms: exhaustive search and branch and bound. The logistic problem with the travelling salesman is a classic example of a problem that involves finding the shortest path.

The simplest procedure to find the shortest path is *exhaustive search* where all paths are tried and the best one chosen. This means that all paths in the solution tree are examined. [*Winston*.84] calls the procedure the *British Museum method*.

Branch and bound also finds the shortest path since it in the search always selects the shortest partial path (counting from the starting point). For the example in Figure 3.8 the search order will be (the number is the cost of the partial path):

```
Start-A(3)     -B(7)
               -D(8)-E(10)
      -D(4)-E(6)          -F(10)
           -A(9)
```

The branch and bound procedure can be improved by estimating remaining distance to the goal and then choose the path with lowest expected cost (cost of partial path + estimated cost to the goal). Another improvement could be to eliminate redundant paths. This latter is called discrete dynamic programming. If a node can be reached through several partial paths all but the one with the lowest cost are ignored. A procedure that combines discrete dynamic programming and estimation of shortest path is called A*. Branch and bound can only be applied to so-called cost problems where it is possible to estimate a cost (or another value factor) for each alternative path in the solution tree. In all other cases exhaustive search must be used if an optimum solution is desired.

Forward/backward chaining

To reduce the difference between a starting point and a goal point there are two possibilities; either to begin with the starting situation or the desired goal situation and then determining how to get to the other situation. Beginning at the starting situation and then finding a path to the goal is called *forward chaining* and the opposite is called *backward chaining*. Forward chaining is well suited in situations where input conditions are known and consequences need to be explored. An example is process control where measured process parameters are known and the question is which other parameters to change in order to control the process. Forward chaining is sometimes said to be *data driven*.

Backward chaining works the other way around by defining the goal or the criteria function and then exploring the demands to the starting conditions. Backward chaining is said to be *goal driven*.

Uncertainty

Often knowledge is not exact but associated with *uncertainty* or incompleteness to a smaller or larger degree. When the knowledge in a KBS is uncertain the conclusions made based on those must be it as well. Uncertainty can be handled by using a factor that tells how certain the knowledge or information is. This factor would ideally follow a probability theory (e.g. *Bayes*) but this is only rarely the case. A number of techniques have therefore been developed and here certainty factors and fuzzy theory shall be mentioned.

Certainty factors express the belief of how certain a statement is. It is used in many rule based systems in various forms. There exist many different versions of certainty factors with different ways of calculating accumulated uncertainties.

Bayes statistics uses a combination of a priori probabilities and probabilities calculated from observations.

Fuzzy theory can be used to express inexact or vague statements like "large volumes", "approximately equal to" and "good quality" [*Hayes-Roth*.84], [*Negoita*.87]. A fuzzy set describes such inprecise statements through a membership function. A membership function defines an interval [0,1] where 0 is false and 1 is true. Operations like union, intersection and complement are used to calculate compound fuzzy sets. Figure 3.9 shows a fuzzy set for the inprecise statement "young people".

Statement: Young people

Membership function:
$$u(x) = \begin{cases} 0 & \text{for } x >= 35 \text{ years} \\ 1/(|x/10|) & \text{for } 35 > x >= 10 \\ 1 & \text{for } 10 > x \end{cases}$$

Fuzzy set: The resulting set of numbers from u(x)

Union:	u(x) or v(x)	= max [u(x),v(x)]
Intersection:	u(x) and v(x)	= min [u(x),v(x)]
Complement:	not u(x)	= 1 - u(x)

Figure 3.9 Fuzzy set for the statement "Young people".

3.4 Goal definition and reduction

A knowledge based system can usually solve various aspects of a problem domain and it is therefore necessary to encircle what the problem exactly is before solving it. Defining and formulating the problem is very important and a consequence of saving time in this phase is often that the programs made are never used since they do not satisfy the need. When the problem has been defined it must be broken down into smaller subproblems that can be solved. This is called problem reductions since large problems that are difficult to solve are substituted by a number of smaller problems where solutions easier can be found. Another way of reducing a large solution space is to limit the goals and sub-goals that were set up in the first place.

3.5 Problem solving

Problem solving can be defined as how to get from a current state to a goal state, or said in another way how to reduce the difference between the two states. This can be done through *means-ends analysis* where procedures are selected according to their ability to reduce the difference between the two states. The *general problem solver* (*GPS*) is a particular control strategy that controls how the selection of procedures in the means-ends analysis is done.

Problem solving can be viewed as finding possible solutions and to determine which one is the best. In most cases the number of possible solutions are very large and some kind of reduction of solution space is required. In many cases it is sufficient to find a possible solution. Such problems are called non comparing problems and can be solved using the search techniques described earlier in "finding paths". In other cases the best solution is required, and in the general case they can only be solved using exhaustive search. A special kind of comparing problems that are most common in practical cases are the cost problems. Here it is possible to calculate a cost for each of the decisions taken when reaching for the solution. The cost increases monotonous for each decision that is taken and it is therefore possible to apply the methods described in "finding best paths", e.g. branch and bound.

An often used problem solving paradigm is called *generate and test* or *hypothesize and test* [*Winston*.84]. The paradigm includes two different types of procedures. The one type - the generator - generates possible solutions to the problem and the other type - the tester - tests if the solutions are acceptable. The generator can either generate all possible solutions, after which the tester takes over or each solution can be tested after it is generated. Knowledge about valid solutions can be build into the generator so it only generates solutions that are likely to succeed.

Rule based systems

Rules in *rule based systems* are used to express procedural knowledge about actions and conditional declarative knowledge. Production rules have the form

IF (condition) THEN (action).

When the IF portion of a rule is satisfied by the facts, the action specified by the THEN portion is performed. When this happens the rule is said to fire or execute. To express uncertain knowledge it is possible to use certainty factors on both the condition and the action side of the rule. Simple rule based systems seem to be difficult to overview as the number of rules increases [*Allen*.86]. This disadvantage can to a large extent be avoided by organizing the rules logically in sets.

Conclusions in rule based systems are reached through forward or backward chaining. Forward chaining is also called data driven search because it starts in the condition part of the rule. The inference engine checks the condition parts of all rules and when conditions are satisfied, the associated actions may satisfy the condition part of other rules, and so on until a desired action (goal) has been reached. In backward chaining the inference engine compares the goal with the action part of the rules. If a match is found the condition part of the rule becomes a subgoal. This subgoal is then compared with the rules again, and if no match is found a question is asked to the user. The inference engine also controls in what order the rules are searched, and what to do when rules conflict.

Object oriented programming

Object oriented programming is a method where logical units of knowledge called objects can communicate with each other by sending *messages* (messages are also called *handlers*). Each object has knowledge about its own behavior, characteristics and limitations and can react on input from the outside world (other objects). Declarative knowledge in object oriented programming can be represented by frames.

An object is a logical unit and it can be very simple with only a few slots or it can be quite complicated. Objects can represent physical things like a car or a lamp. When a message is sent to an object it invokes a procedure (called a handler) attached to the object which gives some kind of response. The same message sent to different objects can cause different effects. If for example the message "work" is sent to the object "car", a procedure in "car" will interpret the message as "drive". If the same message is sent to the lamp it would understand it as "switch on the light". Advantages of object oriented programming are that data is easily organized in the same logically units that one thinks of, and that the definition of a procedure is connected with the object. Examples of object oriented programming languages are *Smalltalk* and *Goldworks*.

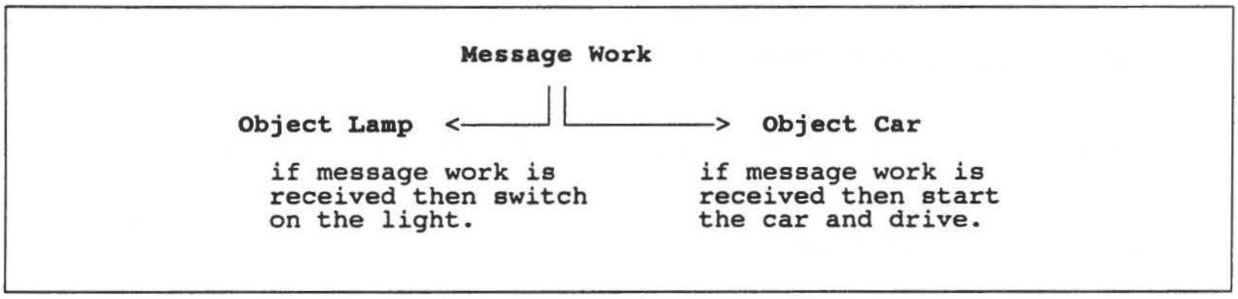


Figure 3.10 Objects and messages in object oriented programming.

Planning Problems

A planning problem can be defined as selection of a sequence of actions where each action only can be performed when certain prerequisites are fulfilled. A very simple planning problem is the previous mentioned travelling salesman problem (finding the route between a number of cities). Process planning (see chapter 5) involves the selection of a sequence of manufacturing processes that can transform a raw material into a finished product. Process planning can be very complex since many factors have to be taken into consideration (e.g. geometrical changes, economy, production capacity).

Complex planning problems can be solved by planning at several abstraction levels. This can be illustrated by process planning. First step is to plan at the most abstract level, i.e. to find a sequence of general activities that are likely to solve the problem. On a more concrete level each of these activities are planned in more detail. Search techniques can be used to select a best sequence. If it in the detailed planning is found that one of the general activities are inexpedient the planning at the more abstract level must be carried out again.

[*Tengvald*.84] describes how to use this type of abstraction and search to perform operation planning. The *XPLAN* system described in chapter 6 performs planning in three abstraction levels: selection of processes, selection of machines and operation planning.

3.6 Decision tree programming

Another way to represent knowledge is through *decision trees*. In the decision tree method data relations are systematized in a hierarchical structure (a tree). A set of branches represents a set of conditions, and the corresponding action to each condition is the part of the tree structure that is connected to the branch. A decision tree contains both all the condition/action relations (the rules) and how they are interrelated. Decision tree programming can be regarded as a rule based system that uses forward chaining. Rules are expressed as shown in Figure 3.11 where an action is selected depending on which conditions that are fulfilled. In decision tree programming, unlike knowledge engineering, the order in which the rules can fire (execute) is fixed to their position in the tree structure, and this representation is therefore less flexible (a rule can only be applied from its position in the tree). This means on the other hand that decision trees are a very well structured representation where it is easy to get an overview of the program structure. Another major difference is that KBS operates with networks (many-to-many relations) while decision trees operate with tree structures (single-to-many relations).

The *DCLASS* system [Dclass.87] is an example of a decision tree programming system.

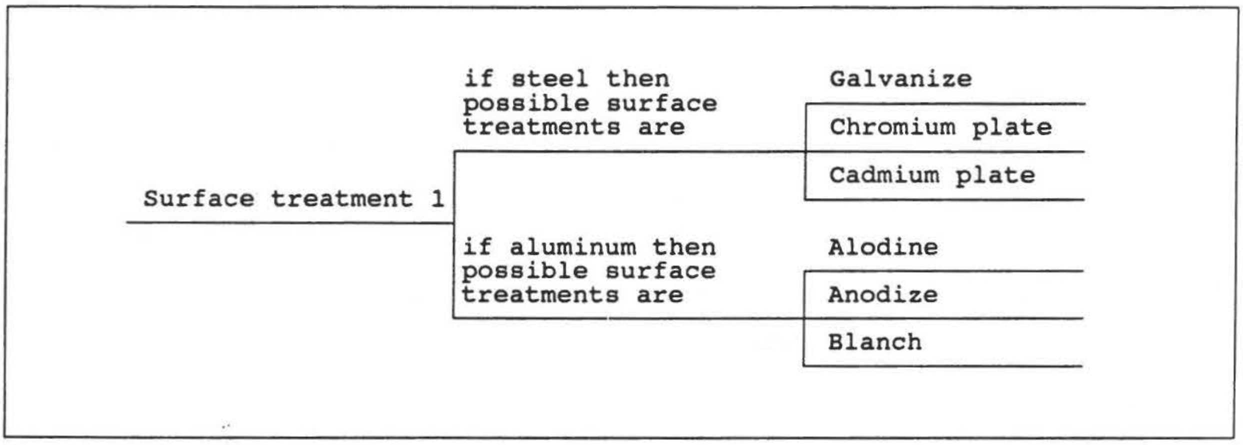


Figure 3.11 A decision tree for surface treatment.

3.7 Other AI techniques

Inductive systems

The basic idea in conventional programming and in knowledge engineering is that problem solving can be described by a number of rules. It can be difficult to capture all the rules and the many exceptions. This is the reason why the idea with generalized knowledge and induction has been introduced to knowledge engineering. If the computer programs could learn from examples they would be easier to make. Many rules and exceptions do also mean that there is a need for more efficient ways of storing knowledge.

Inductive knowledge based systems learn through examples. The knowledge is said to be introduced to the system through *induction*. Such systems generate general rules on the basis of examples [*Genesereth* & *Nilsson*.87]. Having learned their rules from examples they are to a certain degree able to solve problems they have not seen before. This fact can be viewed upon as both an advantage and a limitation. It seems like a programmers dream that a computer system will be able to adjust itself and in that way correct errors and bugs, but it can on the other hand be difficult to predict how the system will act in unforeseen situations. The inductive systems are therefore accused of being difficult to control.

Neural networks

Neural networks is an analogue parallel computer architecture that reminds of the architecture of our own brain. One of the reasons that it is so difficult to capture the rules in problem solving is that we are not really aware of what the rules look like - "we just use them" [*Brunak*.88].

In neural networks knowledge is acquired through examples, in the same way that we learn ourselves. A neural network generalizes from examples and is therefore also capable of solving problems it has not seen before (but the problems must be similar to previous experienced problems and knowledge).

One of the very large problems in building computer tools today is the data acquisition. To find the relevant knowledge and to formulate it in computer structures is a very challenging problem. An expert usually makes very precise and correct decisions and the basis for making expert systems is that knowledge from the expert can be formulated precisely, for instance in a number of rules. Anyone that has been involved in building expert systems or other computer systems has realized how difficult this is.

It is therefore a very tempting idea to duplicate the way human beings perform knowledge acquisition. It is possible that neural networks are a way to do this, but they are still only a research object. Apart from learning from examples neural networks are an architecture for parallel processing, i.e. a possible answer to another bottle-neck in computing today.

3.8 User interface

For a knowledge based system like for any computer based system it is important to have a user friendly, consistent and easy-to-use interface. It is important that the system seems transparent to the user, so he/she understands the basic idea of how the program works. If the user sees a system as a black box where only input and output are known , a consequence might easily be a system that is not used, particularly if there are non-explained errors or wrong conclusions. It is also important that the user easily can correct errors and supplement information. Another important issue is that systems must be easy to use. If for instance the user has not used the system for a long time it must not take too long to get acquainted with it again. On the other hand the system must have advanced features that will satisfy the experienced user.

The user interface involves a dialogue type, help and explanation facilities and some input devices. It is important that the dialogue is informative so the user has no doubt in how to use the system. On the other hand the system must be fast in use so the experienced user is not bored with to many unnecessary explanations. Many systems fail on their user interface in spite of good intentions. It is important to remember that many people only use the systems occasionally and therefore easily forget most of the elementary knowledge of how to use the system.

Basic dialogue types are *command language* and *multiple choice menus*. A dialogue based on a command language is usually very fast to use but it can be difficult to remember all the commands. The commands are usually short abbreviations with only a few characters and can in many cases be little informative and difficult to understand. Command languages are good for experienced users that use the command syntax often, but less suited for other users. Systems using multiple choice menus are on the other hand self explanatory and easy to use but may be punctilious and slow for the experienced user.

A very advanced interface is a *natural language interface*. The user here enters questions and answers in the same way he/she would have asked other people. In the very sophisticated systems the interface is combined with a microphone and voice generator so the system can communicate through speech. This type of interface is very difficult to build because the syntax and the semantics in natural language are much more complicated than in any command language. The same word can for example have different meaning depending on the context, and every sentence do often also depend on what was said before. The most advanced natural language interfaces that presently have been developed do only work on a subset of a normal conversation vocabulary.

With the more powerful computers the possibilities of using advanced graphic interfaces have increased. Where most of the communication previously went through the keyboard as textual questions and answers it is now possible to explain through pictures and use advanced pointing options like a mouse or by pointing directly with the finger on the screen.

3.9 Knowledge acquisition

To find the knowledge, to determine what form it has and how it can be represented are difficult tasks referred to as *knowledge acquisition*. Usually there are two persons involved in knowledge acquisition, the expert that has the knowledge and a knowledge engineer that knows about computers and representations. When one person has to understand the thinking of another person and write it down there will always be a loss of knowledge. On the other hand if the expert and knowledge engineer are the same person there is a chance that the system will be his own private system less understandable for other people.

Knowledge acquisition will normally involve an interview of the expert, model/prototype building and criticism of the prototype by the expert.

3.10 Choice of representation methods

The choice of representation methods depends among other things upon the task the system is going to perform. Tasks that are interesting in connection with Computer Integrated Production are consulting and planning. Where consulting systems result in simple answers like which process to use, planning systems end up with more complex answers like a sequence of machines.

A lot of data and relations are subject to frequent changes within manufacturing and design knowledge. The knowledge concerns information about machines, tools, materials, properties, a.o., and for this reason there is a need for a dynamic knowledge structure. Another need is that several people in different departments must have access to the same knowledge. A solution for declarative knowledge to both those needs is the use of an integration between relational database technology and knowledge engineering. Using standardized database techniques for the storage and retrieval of knowledge have two major advantages. One is the handling of multiuser access right problems, and another is the use of a common data format that can be utilized by other programs. Compared with decentralized data storage, central databases have the well known advantages of consistent and up to date data.

[*Schmeltz*.86] describes an taxonomy for the description of a KBS as shown in figure 3.12. The taxonomy describes and classifies the different elements in an KBS. The taxonomy has been used to describe two ES (appendix 2) and gives a very good overview of the capabilities of a system. On the other hand the taxonomy is rather detailed and therefore difficult to use when comparing different systems.

In appendix 1 is shown a more simplified schema for description of KBS together with two examples. Both examples were made in connection to the ESOP project in order to select a suitable development tool.

TAXONOMY for design and use of knowledge based systems

1. Inference machine

A. Rule based (forward, backward, likelihood)
B. Logic based (Horn clause, first order predicate)
C. Object oriented (inheritance in trees or networks)

2. Description of a KBS

A. Domain/knowledge base description (description of conditions and transformations, inference control)
B. User interface (windows, menus, functions buttons, mouse,question/answer)
C. System interface (to operating system, to database systems, to programming languages)

3. Use of KBS

A. Form of execution (interpreted/compiled, inter/extern knowledge base, session save)
B. Execution facilities (explanations, knowledge base browse, question answering)

4. Development of a KBS

A. Knowledge acquisition (induction from examples, systematic questioning, intelligent help)
B. Editor (word processing, dedicated editor, browser)
C. Development help (on-line manual, tutorial system)
D. Documentation of KBS (domain description, cross reference)
E. Implementation (consistency check - data types - knowledge consistency - loops - unreachable parts of the program, testing - automatic generation of test data - verification of results - trace, modules)
F. Self learning system (what can be learned - conditions - transformations - control knowledge, on what basis - existing knowledge - experience, inference method - induction - deduction - analogy)

Figure 3.12 Taxonomy for design and use of knowledge based systems [Schmeltz.86].

3.11 Development tools

The previous mentioned representation techniques can be programmed directly in a basic language like *LISP*, *PROLOG* or *Smalltalk*. Prolog (PROgramming in LOGic) is developed as a language to handle logic programming and predicate calculus. Smalltalk is made for object oriented programming. High level programming languages are called *expert systems shells* and *expert system developing environments* (*ESDE*). The basic languages (low level) are very flexible and almost anything can be programmed. The price for the flexibility is that the programming can be a very large job. In the *shells* and in the ESDE many of programming techniques are implemented in a way, so they can be used more or less generally. Most shells only have a subset of the representation techniques and are usually closed systems, where there only is a limited access to add user defined programs. ESDE are larger programming systems that until recently were very expensive and only worked on dedicated expensive hardware (the so-called *LISP-machines*). Now it is possible to get ESDE at reasonable costs and to use more general workstations or some of the more powerful PC's. The ESDE offers most of the programming techniques and it is very easy to add user defined programming at many levels. [*Bastlund*.87] and [*Schmeltz*.86] give descriptions and overviews of a number of expert system shells and ESDE's.

3.12 Hardware

Most of the conventional programming languages are sequential languages (only one instruction is processed at any given time). Those languages are designed to fit the traditional *Von-Neumann* computer architecture, where a single processor does all the processing. The knowledge engineering techniques are made to fit the problems they will solve but not necessarily to ensure efficient utilizations of the computer. This means that there is a need for more efficient computer architectures and a very promising one is parallel processing. A parallel processing architecture uses a number of processors that can communicate with each other.

It is important also to consider the type of hardware for KBS. Most often there are different hardware needs for the development of a KBS and the use of the final program. For development work the requirements are normally satisfied by a powerful workstation. For the execution of the final programs it is most often sufficient with less powerful computers. For the end user a more important need is multiuser access to the programs and knowledge bases. This can be obtained on mainframe computers with terminals or on workstations (PC's) connected in local area networks.

Lisp sets up special demands to the hardware due to its list structures and its interpreted mode. For this reason some of the computer vendors offer dedicated computers for Lisp processing, the so-called *Lisp machines* [*Schwartz*.85]. Lisp machines are large personal computer workstations with very large RAM memory (20-100 MB) and huge disk storage (1 GB or more). Some of the Lisp language primitives are sometimes implemented in hardware to increase performance. The user interfaces are very good with high resolution bitmapped monitors and advanced graphic facilities. The Lisp machines are rather expensive compared with other personal workstations but this will probably change with the rapid hardware development. Presently it is possible to install special Lisp coprocessors in conventional workstations in order to speed up the program execution.

References - Knowledge Engineering

[Brunak.88] Brunak Søren, Lautrup Benny: Neurale netværk, Nysyn, Munksgaard, 1988, 160 pages, (in Danish).

[Clocksin & Mellish.84] Clocksin W.F., Mellish C.S.: Programming in Prolog, Springer-Verlag, Berlin, 1984, 293 pages.

[Bastlund.87] Bastlund C., et al.: Programmeringsværktøjer til ekspertsystemer, Datalogisk Institut, Københavns Universitet, DIKU rapport nr. 87/2, Februar 1987, (in Danish).

[DCLASS 87] DCLASS - The expert logic processor, CAM Software inc. Provo, Utah,USA, 1987.

[Genesereth & Nilsson.87] Genesereth Michael R., Nilsson Nils J.: Logical foundations of artificial Intelligence, Morgan Kaufmann Publishers, Inc. 1987, 405 pages.

[Hayes-Roth.83] Hayes-Roth, F. et al.: Building Expert Systems, Addison-Westley Publishing Company Inc., 1983, 403 pages.

[Kvist.88] KVIST - Konstruktion af videnbaserede systemer, Nyhedsbrev nr.1, 2, 3, Dansk Datamatik Center, 1988, (in Danish).

[Nilsson.82] Nilsson, Nils J.: Principles of Artificial Intelligence, Springer Verlag, NY, 1982, 427 pages.

[Negoita.87] Negoita Constantin V., Ralescu Dan: Simulation, Knowledge-based Computing, and Fuzzy Statistics, Van Nostrand Reinhold Company, New York, 1987, 158 pages.

[Schmeltz.86] Schmeltz Pedersen Gert (editor): Erfaringer med videnbaserede systemer, Dansk Datamatik Center, DDC 219/1986-10-31, Lyngby, Denmark, 1986 (in Danish), 153 pages.

[Schwartz.85] Schwartz David C.: The Lisp Machine Architecture, Artificial Intelligence - Toward practical application, Elsevier Science publishers B.V. 1985, p.161-165.

[Tengvald.84] Tengvald E.: The design of expert planning systems, Linköbing University, Linköbing, Sweden, 1984, 162 pages.

[Waterman.86] Waterman, Donald A.: A guide to expert systems, Addison-Westley Publishing Company Inc., 1986.

[Winston.84] Winston, Patrick Henry: Artificial Intelligence, Addison-westley Publishing Company, Reading, Massachusetts, USA, 1984, 524 pages.

Bibliography

[Alty.84] Alty J.L., Coombs M.J.: Expert systems, concepts and examples, NCC Publications, Manchester, England, 1984, 194 pages.

[Chandrasekaran.83] Chandrasekaran B.: Towards a Taxonomy of problem solving types, The AI magazine, Winter/spring 1983, p.9-17.

[CIM.85] Int. Conf. on Manuf. Science, Technology and Systems of the Future, Ljubljana, Yoguslavia, 1985, Published in Robotics & Computer-Integrated Manufacturing, 1988, Vol 4, No. 1/2, 295 pages.

[Henderson.84] Henderson P.: Functional programming, Application and implementation, Prentice-Hall International, 1984, 333 pages.

[Lenau.86.a] Lenau T.: Knowledge Engineering indenfor det mekaniske område (Ekspert Systemer), Indlæg på FTU-temadag, DTH, Lyngby, August 1986, (in Danish).

[Lenau.88.c] Lenau T., L.Alting, Artificial Intelligence for Process Selection, CIRP Int. Sem. on Manuf.Technology, Singapore, 18-19 August 1988. Published in Journal of Mechanical Working

Technology, August 1988, Elsevier, p.33-51.

[Olson.87] Olson Judith R., Rueter Henry H.: Extracting expertise from experts: Methods for knowledge acquisition, Expert Systems, August 1987, Vol.4, No.3, p.152-168.

[Viden.87] Videnbaserede systemer i Danmark, Teknologistyrelsen, Formidlingsrådet, September 1987, (in Danish).

CHAPTER 4. SYSTEM DEVELOPMENT METHODOLOGY

Before building an expert system it is important to describe the domain that the system will work within, and specify goals and requirements for the system. This requires techniques for how to describe the domain, for how to setup requirements and find the goals and a methodology for the system development.

Generally it is important to analyze the situation where initiatives to make design support systems are taken. One must look on how the designer works and in what way he makes his decisions, in order to determine what type of system is needed. It can be disastrous to apply computer tools without having made a thorough investigation of the area where the computer techniques shall be used.

Several system description and development techniques are already developed and used widely and a few of the principal ones will be described here.

4.1 The system as a process

[*Stahl*.77] describes a system as a process with input, output and a number of conditional factors as illustrated in Figure 4.1. The process can be divided into a number of sub-processes where output from one process can be input or a conditional factor for other processes (Figure 4.2).

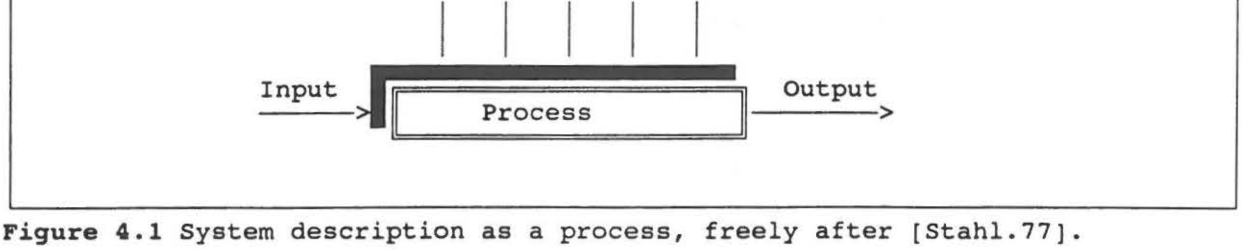


Figure 4.1 System description as a process, freely after [Stahl.77].

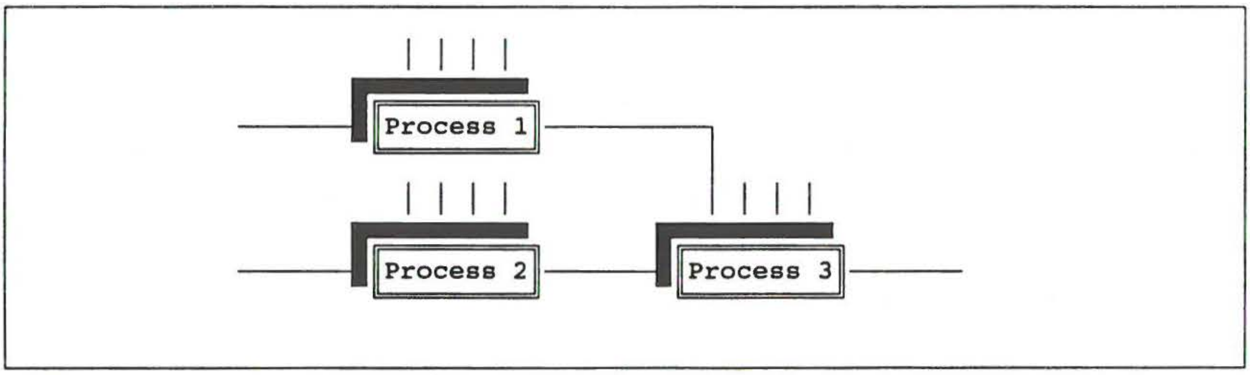


Figure 4.2 System description for a number of sub-processes.

4.2 The IDEF method

Another system description method is developed by the American ICAM project (Integrated Computer-aided Manufacturing) and is called the IDEF method (ICAM DEFenition method). In this context only a brief introduction will be given to the IDEF methods. A more comprehensive description can be found in [*Ibsen*.88].

The IDEF method has 4 modelling tools called IDEF0, IDEF1, IDEF2 and SDM.

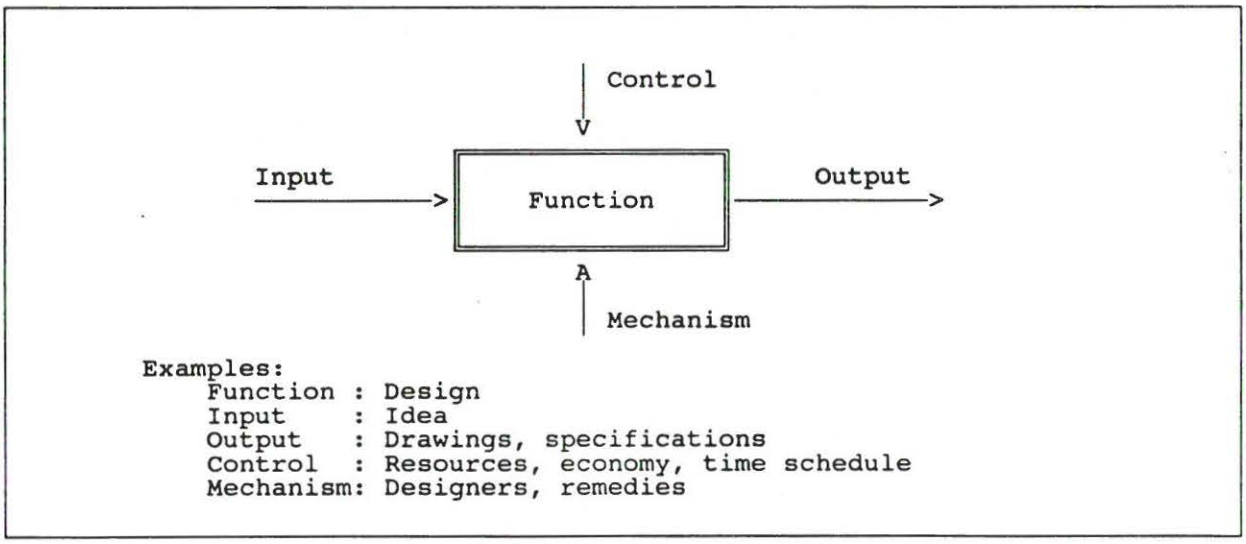


Figure 4.3 Basic components of an IDEF0 diagram [Ibsen.88].

IDEF0 is a tool for modelling the functions, the informations and objects which chain those functions together. IDEF0 is derived from the SADT technique (structured analysis and design technique). The basic components of an IDEF0 diagram are illustrated in Figure 4.3. and Figure 4.4. Like in the process diagram a function can hierarchically be broken down into subfunctions. In this way it is possible to describe a system on both general and specific levels. There are rules for nomenclature, e.g. for how to number diagrams at different levels.

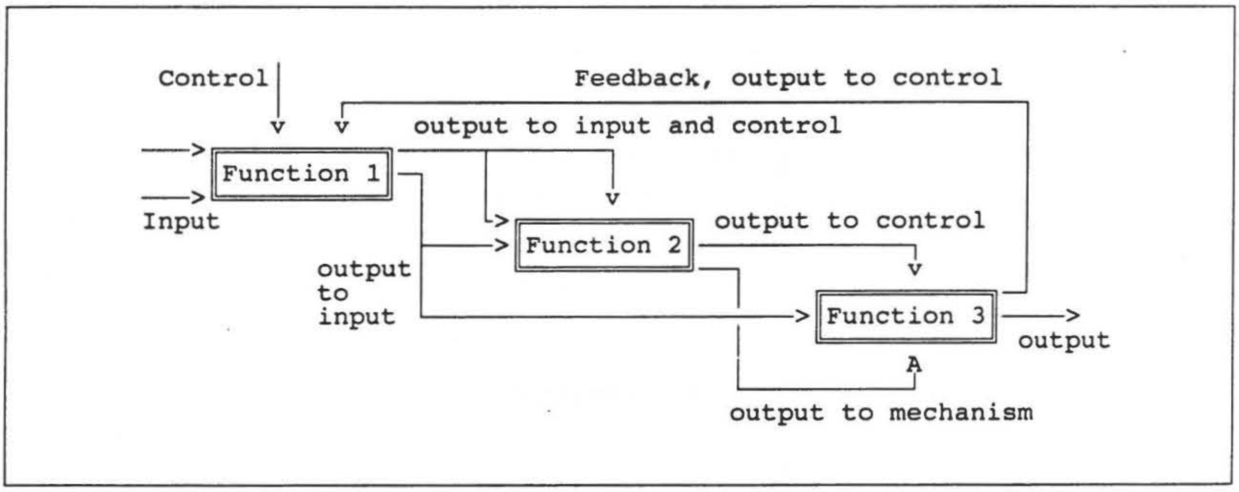


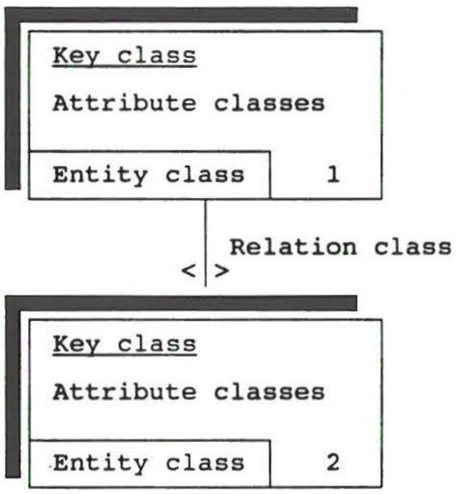
Figure 4.4 An IDEF0 diagram [Ibsen.88].

The IDEF0 method suggests a 8 step general procedure for how to carry out the modelling:

0. Problem identification (formulation)
1. Determination of modelling basis (detailed problem)
2. Context, viewpoint and purpose
3. Collection of data
4. Sorting of data (into data and functions)
5. Creation of a provisional model
6. Review of model
7. Documentation of final model

The result of a IDEF0 modelling is a documentation partly of the existing system partly of the desired system.

IDEF1 is a tool for modelling the information structure that supports the functions. There are four building elements in IDEF1 called classes; classes for entities, relations, attributes and keys. The four classes can be represented in a diagram as shown in Figure 4.5. Relations can be one of four types: a non-specific one-to-one relation, specific one-to-one, one-to-many, and many-to-many relations. Another element in IDEF1 is a procedure in four phases for the creation of the models (see Figure 4.6). The IDEF1 models can be transformed directly to relational database tables.



Examples:
Entity class : Worker, working week, department.
Relation class : Has, works in.
Attribute class: Worker name, date, wage.
Key class : Worker number, department number.

Figure 4.5 The four building elements (classes) in an IDEF1 diagram [Ibsen.88].

IDEF2 is a dynamic modelling tool for simulation of a systems behavior. SDM is system development methodology for planning, organization and control of larger system development projects.

In this context only the IDEF0 and IDEF1 techniques have been studied. The two techniques can be used to describe the data that is exchanged between the different company functions, but not the knowledge. The techniques help to describe data structures when building design support systems, but do not give any support with respect to the procedural knowledge - the decision modelling.

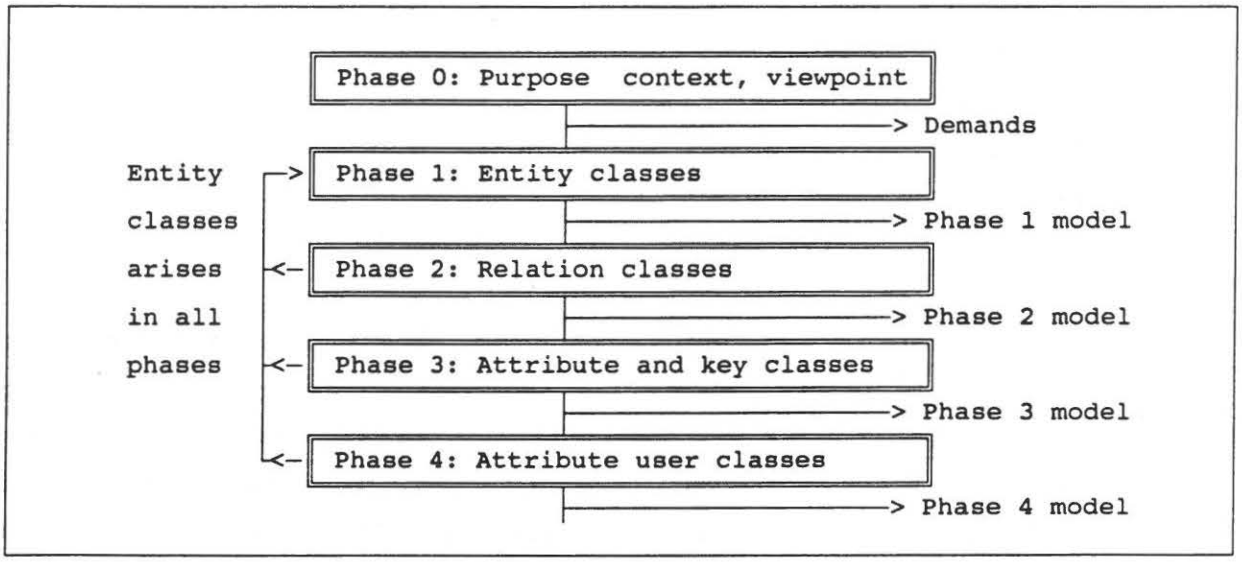


Figure 4.6 Phases in IDEF1 modelling [Ibsen.88].

4.3 A cognitive engineering framework

[*Rasmussen*.88] suggests a methodology, referred to as framework, that can be used to describe and model technical systems. Rasmussen considers the interaction between the user, the task and the system as being of vital importance for the design of computerized technical systems. The role of the user, his interpretation and interaction with his surroundings have a central position in the design of a successful system. The word "cognitive" refers to the fact that people can solve problems they have not seen before because they can generalize and use techniques that they know apply to similar problems. A cognitive system will similarly be able to solve new problems on the basis of applied general principles. The method is also based on the important point of view, that people solve similar problems in different ways, and a system therefore must be open for different ways of use. Rasmussen's method is oriented toward how decisions are taken, i.e. the procedural knowledge, rather than toward information modelling.

The framework can be used either to describe existing systems (for analysis purposes) or as a design tool, and it describes technical systems through the following five model dimensions:

- The problem domain
- The decision task
- The role and organization of decision makers (users)
- Mental strategies
- The users cognitive mechanisms and preferences

The problem domain is a description of the actual problem (that the system will solve) and the knowledge required to solve the problem. The decision task concerns how decisions and choices are made. The three other dimensions deal with the users and how they think and function. In the following the five dimensions will be described in more detail.

The problem domain

The problem domain is the total sum of knowledge needed for solving problems within the domain. It represents all the information that surrounds the decision maker.

Means-end levels	Properties of the system
Purpose/ Constraints	Requirements of the environment Performance of the system Reasons for design Refers to properties of the environment
Abstract Function	Abstract relations between intention and system Refers to abstract terms, not system, not environment
Generalized function	Black box / input-output models (irrespective of the underlying implementation) Refers to recurrent, familiar, input-output relations
Physical function	The use of the object Refers to the underlying physical processes
Physical form	Classification and representation of material objects

Figure 4.7 The means-end abstraction hierarchy for description of the problem domain. Freely after [Rasmussen.88].

It is useful to describe the problem domain through abstraction/decomposition, where each function in a system is described in a number of abstraction levels. A system is described in various levels of decomposition (*part-whole*), and in five abstraction levels varying from the general to the specific (*means-end hierarchy*). The description equals a map of the domain in question where each of the activities forms a trajectory on the map.

Figure 4.7 describes the levels in a means-end abstraction hierarchy. The figure can be read from the top, where the purpose based properties (the reasons for the design) are propagated from the top and down through the hierarchy. It can also be read from the bottom as physically based properties (causes of performance) that are propagated from the bottom and up. Figure 4.8 illustrates how abstraction and decomposition is combined.

	Part-whole (decomposition)				
Means-end levels	Whole system	Sub system	Functional Circuit	Circuit Stage	Component
Purpose Constraints					
Abstract Function					
General function					
Physical function (method)					
Physical form (tool)					

Figure 4.8 A means-end/part-whole diagram [Rasmussen.88].

Rasmussen claims that it is very difficult if not impossible to create a procedural model that can capture the variability of real-life performance. A model will always be a simplified replica of the real life. The assertion is supported by several case studies. One case study is based on several observations of the activities performed by different maintenance technicians while they repair circuit boards with identical malfunctions. The trajectory in the system diagram for each technician was different. This underlines that there are several aspects in a decision task that all must be remembered in order to get a complete model. A single description of a decision task only represents one path through a solution space and is therefore in itself insufficient for systems design.

Another difficulty lies in the formulation of a description diagram. For the means-end/ part-whole diagram it can be difficult to find a suitable and proper division for the axes. For example for the part-whole axis it can be difficult to determine what should be decomposed. The description task becomes even more complex since different people make different formulations and divisions. Still the diagram is a good mean to become aware of all the information that is required in a model, and for what purpose the information is used by different people. Such a diagram can be a good help in the analysis work and is likely to trigger new ideas.

Rasmussen draws some conclusions from the description of the problem space.

- There are two types of representations of environments, one that can be expressed by means-end relations and one that is characterized by intuitive judgments. The former can be handled through rational analysis while the latter needs field studies.
- System design should not only be focussed on familiar tasks with procedural support, but it is important also to focus on problem solving and improvisation at higher levels.
- The user must be able to retrieve information from his point of view, i.e. the setup of the search profile must be very flexible.
- Information has two different sources: physical properties can be collected and calculated while purpose based informa-

tion (goals and intentions) is difficult to collect in a formalized manner.

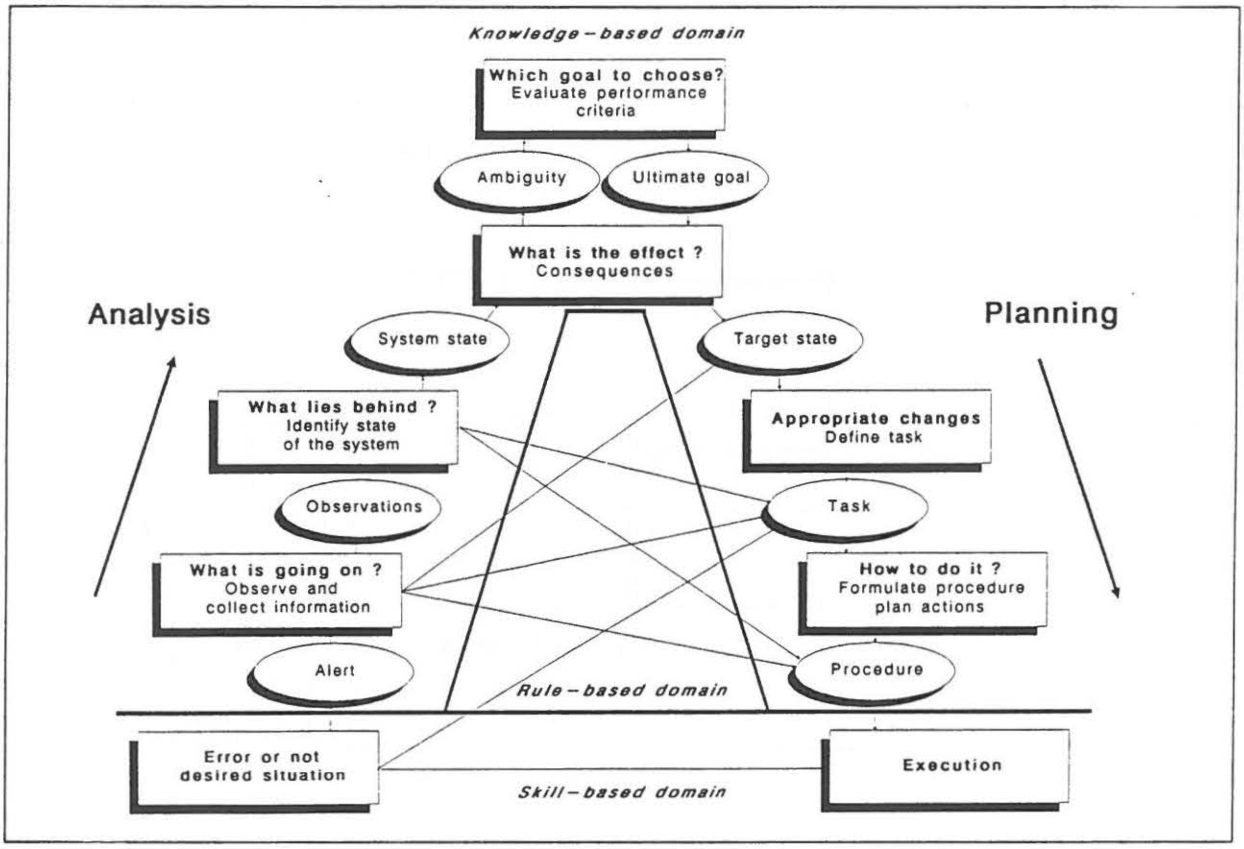


Figure 4.9 Elements in decision making. Freely after [Rasmussen.88].

The decision task

Decision making can generally be represented by a number of typical decision processes, like problem analysis and diagnosis, evaluation, rating and choice of goals, planning of resources, execution and monitoring. Decision making is made either as rational reasoning or as intuitive judgments. When analysing the decision task the following subjects should be considered :

- Decision subfunctions and role allocation between the system designer, the user and the system concerning the division of basic knowledge, state data and processing capacity.
- Are messages neutral messages, advice, recommendation or orders? Who has responsibility in case of mistakes.
- Interface design: Intelligent systems that build models of

users and their strategies, advanced display formats (support of the novice without frustrating the expert), direct manipulation of knowledge, menus, windows, thesauri, etc.
- Generalization of data.

Information receivers

Transmitters	A	B	C	D	E	F	G	H	I	J	K
A	O										
B		O	V	V	V						
C		V	O	V	V						
D		V	V	O	V						
E		V	V	V	O	V	V	V	V	V	
F					V	O	V	V	V	V	
G					V	V	O	V	V	V	
H					V	V	V	O	V	V	
I					V	V	V	V	O	V	
J		V								O	V
K											O

Figure 4.10 Description of communication between decision makers [Rasmussen.89].

The role and organization of decision makers (users)

A general interface between a user and a system can only be made to a certain degree. The actual environment in which it has to function must be taken into consideration. Integration between different functions can be made at various levels of generalization. Integration between production control and planning for example, needs information transfer at a detailed level, while it is sufficient with information on a more general level for integration between management and production. It is also important to consider who makes the decision and where it is taken. Decisions can be made on an individual, a collective or a cooperative decision making basis. When there are several decision makers, a more general level of decision making is required. The diagram in Figure 4.10 is used to describe which people that communicate with each other. By moving columns and rows it is possible to get an overview of the actual organizational structure. The highlighted boxes indicate people that communicate and therefore form an organizational unit.

Mental strategies

Different users use different *mental strategies* to similar decision problems, and it is therefore important that a system for decision support takes this into consideration in order to address a broad spectrum of users. Mental strategies concern many different subjects like how to identify a problem, how information is acquired, how detailed an answer should be, the extent of help needed by the user, etc. An intelligent decision support system should be able to analyze the user queries in order to identify the strategy the user is trying to apply and then supply the required support in displays and messages.

In diagnosis for example two different strategies can be applied. Diagnosis can generally be looked upon as a search to identify the deviations from a normal situation. One type of search representing one strategy will compare a set of observations of the malfunctioning system (the symptoms) and compare them with known symptoms that could identify the malfunction. This kind of search can be called *symptomatic search*. Another type of search or strategy is the *topographic search*. Here the mal-operating system is compared with a normal operating system and mismatches are located in the system. The two types of strategies are quite different and computer implementations will also be different.

The users cognitive mechanisms and preferences

The information <u>contents</u> of a decision support system can be determined through analyses of the problem space and the mental strategies. The display <u>form</u> depends on the users cognitive control mechanisms. Whether the user is familiar with the way in which questions and information are presented to him is important for correct interpretation and understanding. Among other things the layout of the user interfaces (screens, graphics, etc.) is part of this dimension.

4.4 The cognitive engineering framework applied to design

The cognitive engineering framework for technical systems described in the previous section will here be used on design support systems. The methodology is general and can be applied to design of information systems. It emphasizes the importance of the individual characteristics for each system, and a large effort is placed in the description of the present situation, information and how decisions are taken. The model separates the description of the problem domain, the decision mechanisms and user related topics, since it describes technical systems by five basic characteristics (called dimensions): The problem domain, the decision task, the role and organization of the decision makers, mental strategies, the cognitive mechanisms of the users and their preferences. The framework will in this paragraph be used on selection of surface treatment processes and examples from the work with the ESOP system (see the next chapter) will be used. The framework was not used in the development of the ESOP system, but serves here merely as an example.

The problem domain

The problem domain covers all the knowledge and solutions that the designer uses when he solves a design problem. Either the designer possesses the knowledge himself or he obtains it from other sources like text books, advisory experts or production employees. Maybe the designer in cooperation with the management sets up requirements for the product. Some of those requirements could concern the surfaces of the product (appearance, wear, corrosion, etc.) and surface treatment processes can therefore be considered. How requirements are set up and alternative processes explored can be described in an abstraction/decomposition (means-end/part-whole) diagram. The diagram forms a map where the paths illustrate how the analysis/synthesis/selection work is carried out. The abstraction levels (the vertical axis in the diagram) are useful since they clarify what type of decisions that are taken and what level of information that is needed. The horizontal axis in the diagram helps to describe where the decisions are taken and how they affect other decision makers.

Figure 4.11 describes how surface treatment processes are selected and where in the company the decisions are taken. The Figure also describes the abstraction level of the decisions. There are four company activities involved in process selection: The company management, product development and design, process and production planning and quality control. The decisions made by the company management are strategic and very general and could for example be a specification of the type of processes that should either be preferred or avoided. The goals for the management decisions are to maximize the company turnover and to secure the strategic position of the company.

In product development and design it is decided which processes to use. Within the given time and effort constraints, a number of processes are investigated and compared with the product requirements. Relevant product requirements are identified on a more abstract level (need for corrosion and wear resistance, attractive appearance, etc.). On a lower abstraction level those requirements are detailed further and process selection is done using methods like search and comparison. The tools used in the selection include various search and comparison techniques.

Process and production planning describe how the production should be carried out in detail and when it should be done. The goal is to make a plan that utilizes the production capacity and produces the parts at the lowest price taking quality into consideration. Planning is done within economic and capacity limitations. The selection of production equipment is an example of an abstract level decision while the decisions in operation sequencing belong to the general function level. Decisions at the physical function level include detailed planning of each operation, estimation and calculation of parameters. This is done using various planning techniques and calculation algorithms (The physical form level).

The purpose of quality control is to ensure that products are actually produced according to their specifications. This is done by measuring the products and comparing the results with the specifications. By using statistical methods only a subset of the parts need to be tested.

	Company functions			
Means-end levels	Company management	Product development & design	Process & production planning	Quality control
Purpose Constraints	Max.company turnover Strategic position Legal & environmental constraints	Functional specifications and limitations Time & effort	Select and plan production, Capacity & economic limitations	Control produced parts & products, economic & technical limitations
Abstract Function	Priority Economy Information flow	Category of product parameters. Surface quality - corrosion, wear, appearance, electrical, etc. Process evaluation.	Selection of technology, in-house, capacity, reliability, environmental considerations.	Measurement methods, Comparisons
General function	Administration of office and personnel	Detailed product parameters. Surface lifetime - corrosive media, forces, temperature, etc. Parameter evaluation	Operation planning sequences, facilities, retrieval.	Measurement, Statistical selection
Physical function	Office and planning procedures	Search for and comparison of surface treatment processes, Calculation of consequences	Parameter estimation and detailed calculation, detailing operations	Measurement processes
Physical form	Equipment	Search and comparison techniques	Algorithms & planning techniques	Measuring techniques, Statistical methods

Figure 4.11 Abstraction description of a system for selection of surface treatment processes.

The decision task

The previous section described the problem domain and analyzed the information basis for the decisions. The analysis of the decision task involves description and classification of the decisions themselves. The decision task is normally rather complex but can be classified in a number of general decision types. Selection of surface treatment processes involves the following types of decisions:

- Are material properties adequate according to product requirements ?
- Changes in product requirements ?
- Which requirements can not be fulfilled ?
- Can those requirements be satisfied by surface treatment ?
- Which surface treatment processes are relevant ?
- What are the implications of using a surface treatment process ?
- etc.

The role allocation between the designer, the user and the computer can be considered here with respect to where the decisions are taken. Will it be most appropriate to build an automatic system that itself makes many of the decisions or should the system suggest a number of alternatives that the user can choose from.

The role and organization of decision makers (users)

The designer communicates with specialists and people from the production in order to make the best design. It is therefore important that the conclusions, informations and formulations in the system can be approved by all those people.

Mental strategies

The different users have different ideas about how to encircle usable processes and which questions should be answered in what order. Some users would for example prefer material as a first search criteria while others would consider application as most important. The system should satisfy both groups.

The users cognitive mechanisms and preferences

The designers will not use a system like ESOP every day and it is therefore important that he can use the system without remembe-ring special commands or what questions he has to answer. One way of achieving this is by using an interview dialogue where the system presents the questions to the user. The dialogue with the computer must be immediately understandable for the user, and it should be considered if the screen layout should resemble the layout of existing information on paper.

4.5 Three level development methodology

A three level development methodology involving three increa-singly detailed levels of specification, a knowledge level, a function level and a program level is described in this section (also described in [*Lenau*.88.c]). Similar methodologies are reported by [*Tong*.87] and [*Kim*.87]. [*Kim*.87] adds maintenance of the finished system as an extra level, and emphasizes that the selection of the initial problem for a knowledge based system must be evaluated from a cost/benefit criteria. In the case of process selection the benefits of a well functioning system are difficult to measure directly, but the expected benefits from a better flexibility and a shorter development cycle are obvious.

The knowledge level

On the knowledge level type of problem, domain knowledge, solution types, and their interrelations are described. For process selection the main problem is to determine which process or which sequence of processes that can produce a given design considering its special characteristics (part geometry, basis material, functional requirements, etc.).

Domain knowledge for process selection includes on the one hand different types of manufacturing knowledge about processes and machines (e.g. capabilities, availability, cost), and on the other hand product experiences (e.g. history of good and bad designs). The domain knowledge comes from several sources, and it is one of the advantages in knowledge engineering that those different sources are combined into one (utilization of several representation techniques) . Process knowledge is found in books and files and among experienced people in production departments. Product knowledge can normally be found in the departments for design and quality control.

Requirements to solutions include contents (basic type of answers, e.g. number of solutions found, information about processes), form (input and output, e.g. graphic interface, menus, mouse) and hardware consideration (central/de-central solutions).

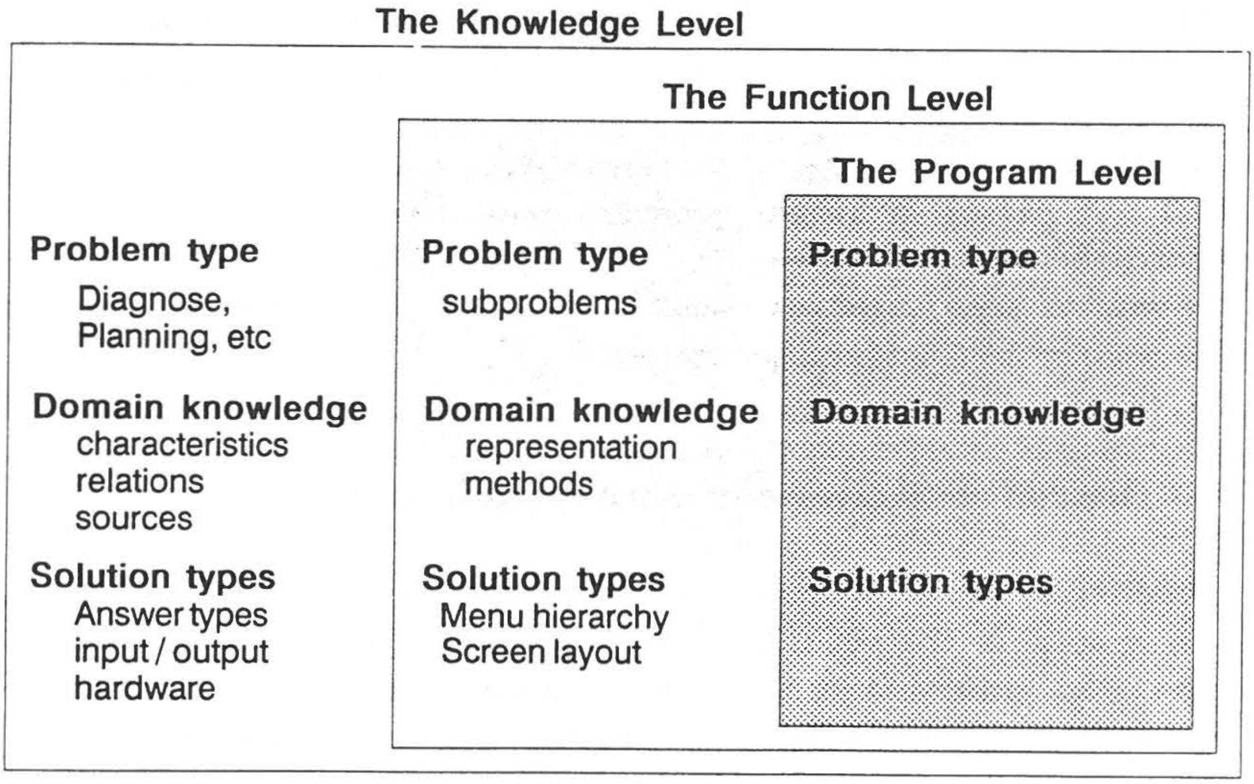


Figure 4.12 Three level system development methodology.

The function level

At the function level problems, solutions and domain knowledge are detailed further. Problem solving was on the knowledge level considered as a black box with problems as input and solutions as output. On the function level the black box is divided into subproblems and subsolutions. At this point different representation techniques are considered in order to find the best way to describe the desired system. It is among other things analyzed whether the problem can be characterized as having a single clear defined goal that can be used to backtrack solutions, or whether different types of input can lead to different solution types. How to describe a system using different KE representation techniques is described in chapter 3.

The program level

At this level the system description made at the function level is translated into structures that can be used on the computer. Depending on the type of software this can be a relatively small task if an advanced software tool is used. On the other hand it can be quite a large job using more primitive software tools.

It is important to divide system specifications into levels like this to assure efficient program development. If for instance system development is done mainly at the program level, overview is easily lost, and too much time spent on programming details, debugging and last minute changes.

4.6 General considerations about system design

Generally it is important to analyze thoroughly the domain that the system is intended for, and setup demands for what the system shall do. This includes technical specifications like the type of information, screen layouts, safety, etc. It is also important to keep in mind when building design support systems that other factors than the pure technical can be very important for a successful system. Organization and responsibility will have a large impact on the success of the system. For example when the design support systems can make decisions, it must be considered who is responsible for wrong decisions, and what implications it has. It is also important to consider if the instructions given by the designer are actually followed when the part is produced.

In some cases the production preparation changes minor details in the design and this limits the advantages of the better decisions that designer can make using a design support system.

Another important subject is the analysis of how communication is carried out and which channels it follows. Some of the communication is very formal and can be found in drawings, plans, guidelines, etc. Analysis work is often based on the formal communication, but the informal communication e.g. a chat in the lunch room or next to the coffee machine may have large impact on how a product is produced. The lack of integration between company functions can to some extent be a result of bad informal communication due to personal relations or communication problems due to physical distances between people.

In this chapter three different system development methodologies have been described. They each have their different purpose. The IDEF method is suited for describing the functional structure in a company and for modelling the information flow. The cognitive engineering framework focuses on the decision making and on how organizations and persons affect the decisions. The three level methodology is a guideline for how to divide the system development work into phases.

References - System Development Methodology

[Ibsen.88] Ibsen Jess Julin, Karsholdt Peter, Vesterager Johan: Introduktion til IDEFO & IDEF1, Driftteknisk Institut, DTH, 1988, Vol.1 (51 pages) + Vol.2 (75 pages).

[Kim.87] Kim, S.H.: Managing the development of advanced knowledge-based systems, CIRP Seminar on Computer Aided Process Planing, Pensylvania State University, PA, USA, June 1987.

[Lenau.88.c] Lenau T., Alting L., Artificial Intelligence for Process Selection, CIRP Int. Sem. on Manuf.Technology, Singapore, 18-19 August 1988. Published in Journal of Mechanical Working Technology, August 1988, Elsevier, p.33-51.

[Rasmussen.88] Rasmussen Jens: A cognitive engineering approach to the modeling of decision making and its organization in: process control, emergency management, CAD/CAM, office systems, and library systems, Advances in Man-Machine systems research, JAI press Inc., vol.4. p.165-243, 1988.

[Rasmussen.89] Rasmussen Jens: Analysis of Cooperative Work, Discussion paper for planning of field studies, Risø National Laboratory, Roskilde, Denmark, February 1989, 31 pages.

[Tong.87] Tong, Christopher: Toward an engineering science of knowledge-based design. Artificial Intelligence in Engineering, 1987, Vol.2, No.3.

CHAPTER 5. PROCESS SELECTION AND PLANNING

This chapter discusses how manufacturing processes are selected and planned for mechanical products. The word "*process*" is used in several connections, but in this context it is solely used for discrete part manufacturing (see chapter 1 for a more detailed definition).

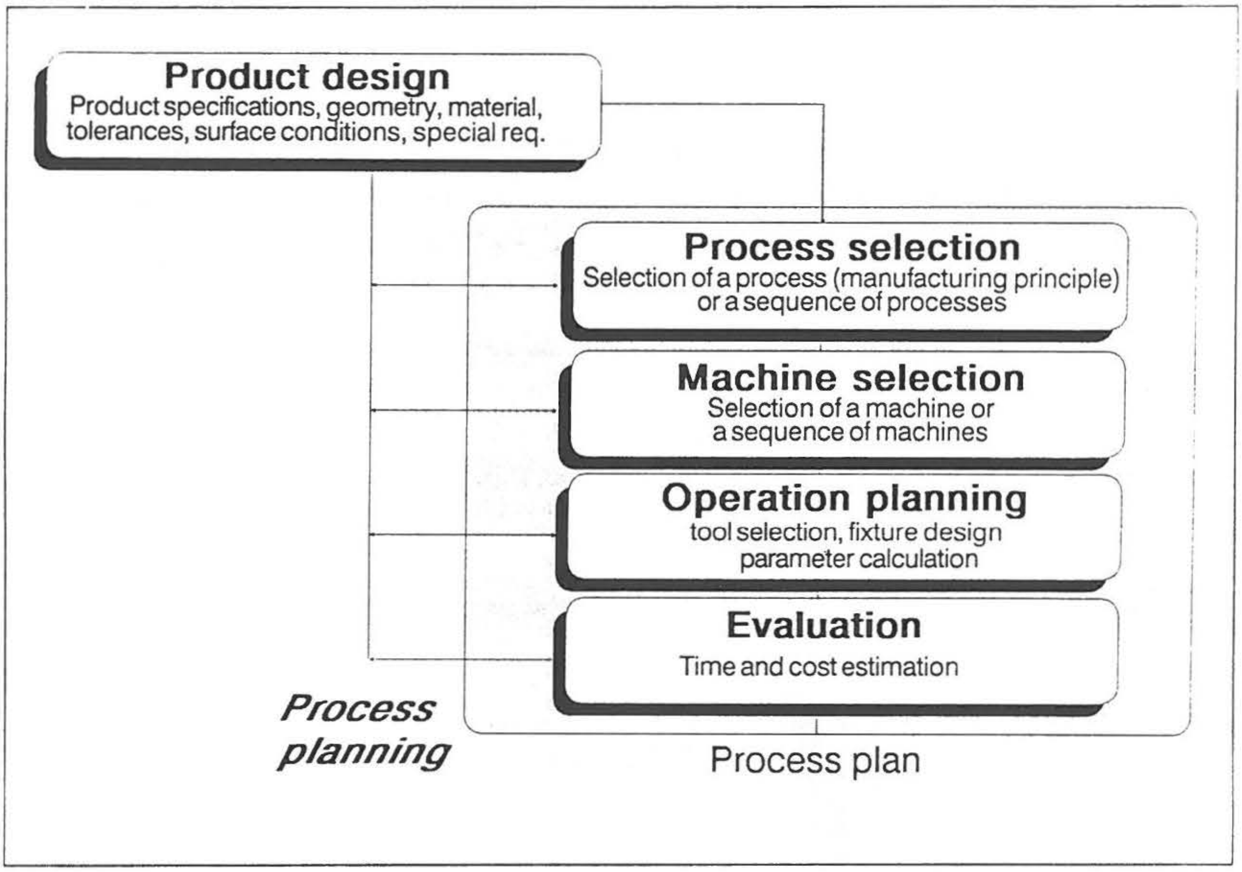


Figure 5.1 Activities in process planning.

5.1 What is Process Planning

Process planning is a rather complex area including a number of different tasks, starting with the initial selection of manufacturing processes and ending up with the detailed information for how to produce the part (see Figure 5.1). Process planning is also called manufacturing planning, materials processing, process engineering and machine routing. The process plan is also referred to as a route sheet, an operation plan, an operation planning summary and an operation sheet.

5.2 Selection of manufacturing processes

First step in process planning is to determine which process (type of machines) or sequence of processes that can produce the part. In many cases some of the processes are already selected by the designer, and the selected processes are then usually indicated on the part drawing. Often the designer has made the design assuming that a certain process is used but without giving special instructions about the use of the process.

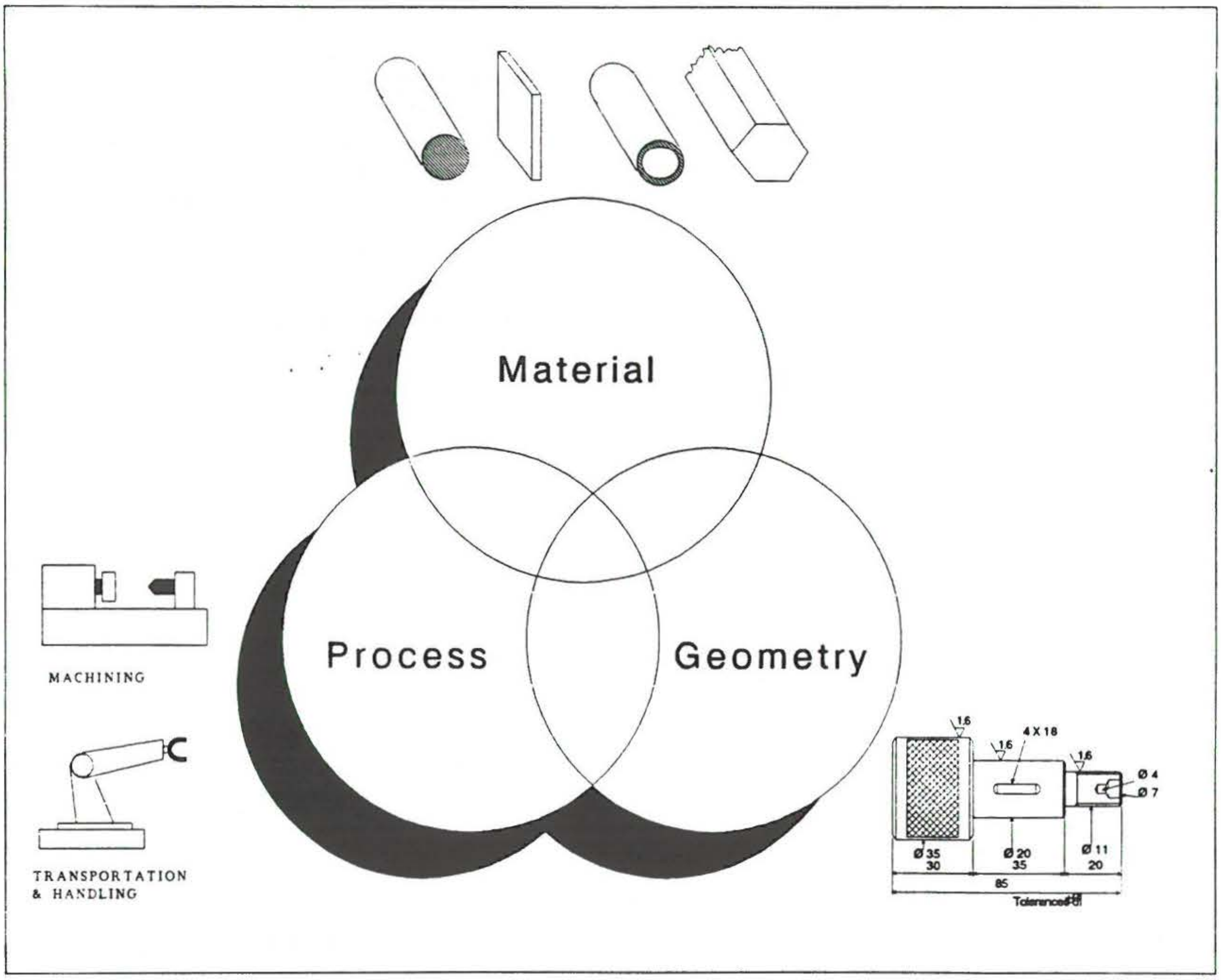


Figure 5.2 Selection of processes, material and geometry are closely related.

Process selection is a very complex matter that closely interferes with design, and this is one of the main considerations in *design for producibility*. The primary task for the designer is to create a product that fulfils the functional demands, but most often there are a number of design alternatives. In the evaluation of those alternatives it is important also to take the manufacturing implications into consideration. Doing so the designer actually selects the manufacturing processes. Process selection is done by comparing part requirements with the capabilities of the different processes. The process selection depends on parameters like type of geometry, dimensions, tolerances, how the part can be clamped, material properties, batch size, time constraint, costs, etc. One of the factors that make process selection complex is the strong dependency between product characteristics and manufacturing possibilities. The part geometry for example depends on how the part is produced, i.e. the selected manufacturing processes, but the selection of the manufacturing processes does also depend on the selected part geometry. The process selection is therefore iterative. Figure 5.2 illustrates that selection of manufacturing processes, material and geometry are very close related. A selection within one of the areas sets up limitations for the selection in the other areas.

An important point of view when talking process selection is the difference between fixed well established processes and many non-conventional processes. The fixed well established processes cover for example discrete metal processes like machining or forging. Here the design consideration mainly concerns how to make the part so it fits the process. For many parts made of non-metallic materials by non-conventional processes the design of the process itself can be as important as the product design.

Information about the different manufacturing processes can be found in various handbooks [*Alting*.82], [*Metals Handbook*.69], [*DeGarmo*.69], [*Bolz*.63], [*Manufacturing Consortium*.85]. In the handbooks it is easy to find comprehensive information about a process, but it can be difficult to find information about which processes that can produce a given geometry or machine a certain material. In other words the books are limited to a single index (in section 7.3 other relevant indexes are listed). The index is most often the process name. *Value Control Guide* [Value.63] is an American system for process selection that uses another index. On the basis of a simple part classification system (10 geometry groups) possible manufacturing processes are proposed. For every process there is a description in schematic form (interval for batch sizes, obtainable tolerances and surfaces, production cost, etc.) so it is possible to compare the processes. Both the Value Control Guide and the handbooks are best suited for the initial exploration of the processes, but less suited for the detailed process planning.

[*Jepsen*.78.b] lists some of the factors that influence the process selection and therefore must be taken into consideration by the person that selects the manufacturing processes (often the designer):

- Product demands (geometry, tolerances, surface conditions, material, appearance, properties)
- Market demands (time pressure, price)
- Technical feasibility
- Resources (Production equipment, capacity, skilled labor, money for new investments, reliability of material supply, energy supply)
- Legislation (environmental, responsibility)
- Company policy (goals, in-house processes)

Manufacturing processes can be divided into main processes, helping operations and pre-/post-operations. The main process is defined as the process that gives the part its main shape. Examples are casting, hot forging, powder compaction, etc. Helping operations are the activities that are necessary to make the main process work correctly, e.g. heating before forging or lubrication in machining. Pre- and post-operations are activities that have to be done before or after the main process e.g. cut-off of bar material, deburring. When selecting manufacturing processes different alternative main processes are considered, and a few ones chosen for closer consideration. This involves evaluation of helping, pre and post operations. Figure 5.3 illustrates how selection of main process and the different operation are related.

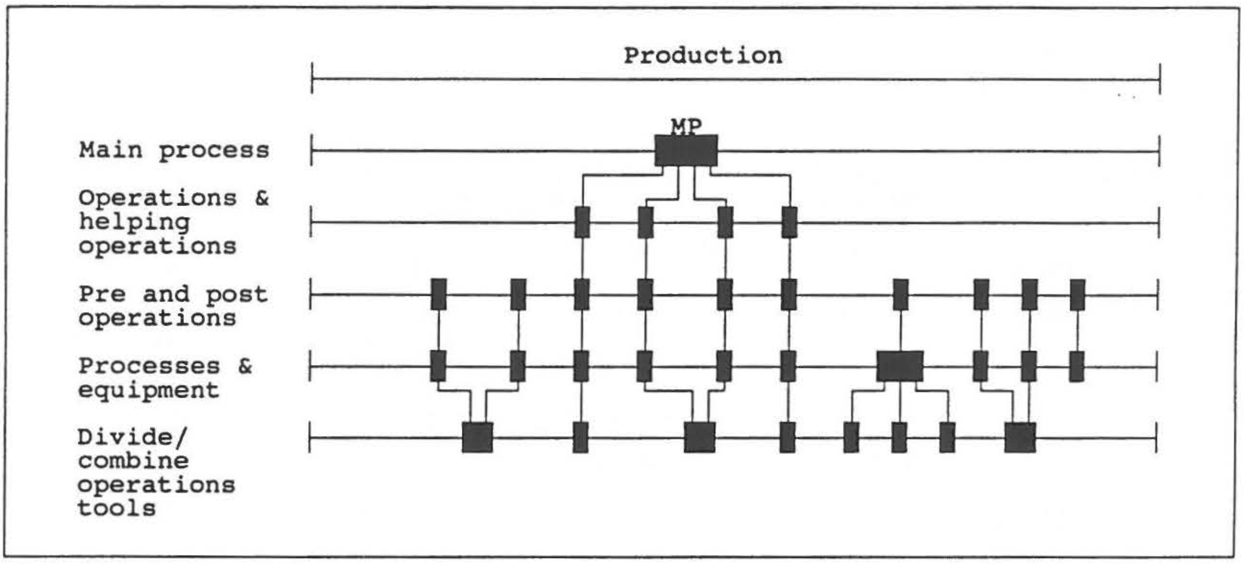


Figure 5.3 Stepwise specification of manufacturing processes and operations for a specific production method [Jepsen.78.b].

5.3 Manufacturing taxonomy

A part can usually be produced by several different manufacturing methods. A cylindrical part for instance, can be turned from a bar, or it can be forged, or maybe welded together from different pieceparts. In order to be able to overview all the possibilities, the different available processes can be classified according to their characteristics. A system for classification of the manufacturing processes is also called a *manufacturing taxonomy*. A lot of work has been done within this area to develop a general taxonomy [*Jørgensen* & *Alting*.81], [*Alting*.82], [*Manufacturing Consortium*.85].

[*Alting*.82] describes how a process can be defined as a transformation of raw material, energy and information into the desired part. The different manufacturing processes are then grouped according to whether they are mass conserving, mass reducing or joining processes (Figure 5.4). Metal cutting processes like turning, milling and grinding are examples of mass reducing processes, forging is a mass conserving process, while welding is a joining process. Each of these groups is then divided into a number of subgroups. The different groups in such a taxonomy have common characteristics, and this fact can be used when selecting processes.

Process or flow type	State of material	Category of basic process	Primary basic process	Process examples
Mass-conserving processes (dM=0)	Solid	Mechanical	Plastic deformation	**Forging** and **rolling**
	Granular	Mechanical	Flow and plastic deformation	**Powder compaction**
	Fluid	Mechanical	Flow	**Casting**
Mass-reducing processes (dM<0)	Solid	Mechanical	Ductile fracture and brittle fracture	**Turning milling,** and **drilling**
		Thermal	Melting and evaporation	Electrical discharge machining (**EDM**) and cutting
		Chemical	Dissolution	Electro chemical machining (**ECM**)
			Combustion	**Cutting**
Joining processes				
Atomic bonding	Solid	Mechanical	Plastic deformation	**Friction welding**
	Fluid (vicinity of the joint)	Mechanical	Flow	**Welding** (fusion)
Adhesion	Solid (fluid filler material)	Mechanical	Flow	**Brazing**

Figure 5.4 Classification of manufacturing processes (only typical process examples are mentioned) [Alting.82].

It is important to have in mind, that the designer when constructing a part actually performs some of the process planning, which again affects the design of a process planning system. Especially when discussing integrated production it is important to focus on how design and production information correlate. The designer is with his decisions responsible for a major part of the product costs, since raw materials, processes, production methods including assembly more or less are determined in the design phase. This is the background for the need of advanced information systems that can supply the designer with the relevant information, relevant design rules and company standards.

5.4 Process selection strategy

A process selection strategy is a plan for how the general activities in process selection are carried out. [*Jepsen*.78.a] describes a process selection strategy as a nine step plan:

- Analyses of objectives and prerequisites
- Temporary setup of goals and criteria
- Encircle material groups that can be used
- Setup of alternative processes/process-types for alternative component divisions
- Setup of possible combinations of materials and processes
- Gathering of information and specifications of a criteria function
- Gathering of information and specifications of production methods
- Evaluation/selection.
- Determination of the final part specifications

Jepsen has followed how a refrigeration compressor and its different sub-components are designed and planned for production. Due to the large production volume there was a basis for a long term design phase where a design team could examine many design and manufacturing alternatives. The observations were compared with the above mentioned process selection strategy.

The conclusion was that a clearly defined scope and the organizational relations for the process selection are very important. The process selection strategy was followed and there was not observed any needs for changes in the strategy. The strategy is therefore considered as a good guideline for how to perform process selection.

5.5 Example of process selection

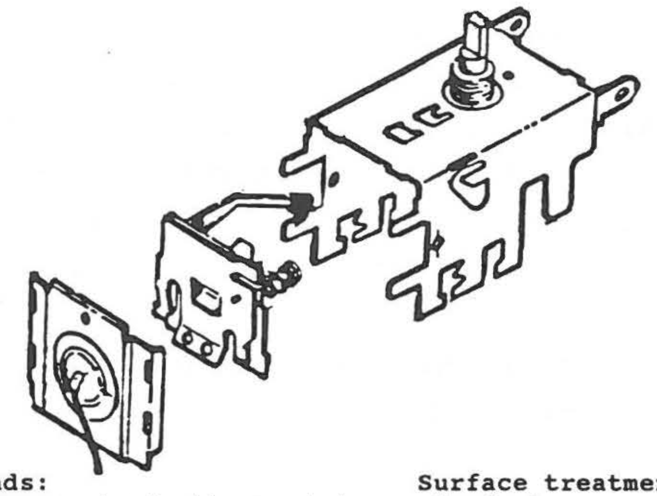
In the following an example will be used to describe the design consideration concerning manufacturability of a part. The example is taken from the one of the companies involved in the ESOP project that actually produces the part (ESOP is described in chapter 7). The part is a thermostat box made of sheet metal and it is for instance used in refrigerators. The part was planned to be produced in very large numbers and it was a requirement that it could be assembled automatically. The functional demands included low production costs, resistance toward corrosion and wear, reasonable strength and sufficient ductility to ensure that assembly can be carried out. Furthermore the material must be able to conduct the heat.

It was decided to make the part in sheet metal produced by punching and bending, apparently because these processes are known in the company and cost effective. The properties of low cost sheet metal were not sufficient concerning corrosion, wear and strength. Improvements of those properties for the sheet metal are therefore required and they can be obtained through surface treatment. The question is which treatments that can be used without loosing other properties like heat conductivity and ductility and without rising the cost of the product too much.

When the thermostat is used in a refrigerator it is not exposed to mechanical stress that exceeds the critical stress for the material. But there is a risk that the "legs" on the box will bend so much that automatic assembly can not be done. To avoid this the strength of the part can be increased through heat treatment (hardening) before the surface treatment.

Within the box there is a suspension arm that activates the thermostat switch. The bearing for the suspension arm is a knife-edge in the box. Wear in this bearing can cause bad adjustment of the thermostat. The thermostat must function correctly for a number of years and wear resistance should therefore be as good as possible.

The component is not visible when mounted in a refrigerator, and some rust stains do not affect the function of the thermostat. On the other hand some kind of corrosion protection is required since the part must look attractive (when sold as a spare part). During operation the thermostat is exposed to fluctuating temperatures and therefore condensed water. In some cases the condensed water contains fruit acid, and the thermostat must therefore be resistant to this exposure for a number of years.



Functional demands:
- base for sub components in thermostat
- low cost
- corrosion resistant
- wear resistant
- strength
- ductility
- heat conductivity

Surface treatment processes
- Hardening
- Saltbath nitration
- Annealing
- Chromating

Figure 5.5 Thermostat box and design considerations.

The functional demands were met through the use of four succeeding surface treatments. First the punched and bended sheet metal part is hardened to ensure strength. Next a surface treatment is used to improve wear resistance, and a tempering process then ensures ductility. Finally another surface treatment is used to enhance corrosion resistance and the appearance of the part. There are more inexpensive alternatives to this solution but it is then necessary to weaken some of the functional demands. This illustrates also that the process selection is iterative.

In this case it was found that a design support system could help the designer set up the functional demands and in the search and examination of surface treatment alternatives.

5.6 Machine selection

Having determined a sequence of manufacturing processes, next step is to select the actual machines to be used. Turning for instance can be performed on many different lathes, and milling on different types of machining centers or milling machines. Machines that combine several processes like the 4 axis lathe (the fourth axis can be used for milling) have become more common and place special requirements on the machine selection. Machine selection does also depend on the degree of automation since the machines are often designed for automated production.

Machine selection is made based on consideration about technical specifications, production capacity, batch sizes, set up time and costs, etc.

5.7 Operations planning

Operations planning covers the selection of a sequence of basic machining operations on a machine, so that the design requirements can be met. It also includes calculation of the different machining parameters, like cutting depth, feed and speed.

Sometimes the more general term process planning is used exclusively for operations planning, i.e. not including the process and machines selection. It can be difficult to separate the operations planning from the selection and design of tools, dies and fixtures.

5.8 Tools, dies and fixtures, selection and design

Another important task in process planning is selection/design of tools to be used on the different machines. The word tools are here used in a broad sense covering both machining tools and fixturing tools. In some cases it is possible to reuse existing tools, for example cutting tools for lathes and mills, and the task is here search and selection. In other cases new tools are designed for each part. Examples are dies, forging tools, special fixtures for cutting processes, etc. To increase flexibility and reduce cost modularized tools are often emphasized.

[*Jørgensen*.78] describes how to systematize and rationalize tool and die design and lines up what is needed in the different design and production phases. The needs in the product design phase are easy access to retrieval of similar tools and general guidelines for the design of tools and dies. The needs in the process planning phase concern guidelines for tool design and efficient search methods for standard tool elements. Also graphical design support (CAD/CAM), calculation and evaluation support are needed. In the production phase the needs are easy access to experiences with previous productions and guidelines for systematic maintenance.

[*Jepsen Jensen*.84] describes how to support the design of tools and dies for die casting. Jepsen Jensen has developed a system that assists the tool designer with guidelines, calculations and graphic facilities.

5.9 Process plan evaluation

Often there are a number of alternative process plans and it is necessary to rank them in order to select the right plan. There are several ways to do so, and one way is to use time or cost estimates to evaluate process plans and compare them with other plans. Alternatively production capacities or preferences can be used as selection criteria.

Estimation of time consumption and cost are important for several reasons. It can be used as a measure of the quality of the process plan and to compare different process plans for the same part. Time consumption is also used in the production planning, in order to utilize the machines efficiently. Both time consumption and cost are used for wage calculation for workers and for product cost estimates to costumers.

Work has been done in setting up standard estimation methods. The manual standard systems are usually easy to use, but also quite time consuming. Computerized versions of the systems tend to be "canned" where it is difficult to check if the system has made the calculations correctly and to correct errors.

It is very difficult (if not impossible) to select an optimal process plan since several of the process planning parameters varies over time. A plan that is optimal at one time will usually not be optimal at other times. A way to select process plans is to evaluate all process plans for the production in a given time period identifying bottlenecks. This can be done using production planning techniques, where the bottlenecks can be removed by moving production to other machines, i.e. by selecting alternative process plans. A very promising production planning method that can be used for this purpose is production simulation as described by [*Larsen*.89] and [*Bilberg*.89].

5.10 Time estimation

The purpose of estimating time consumption can be to use it in:

- calculation of wages for workers in the production
- estimation of delivery deadlines
- selection criteria
- production planning

A time estimate is calculated as the sum of a fixed time (preparation time) and a variable time (machining time). The preparation time covers activities that is performed one time for each production batch, e.g. mounting and adjustment of tools and fixtures, loading of NC-information, etc. The machining time is proportional with the number of parts produced, and covers load and unload time, cutting time, moving and waiting time, tool change, etc.

One method of estimating time is called *synthetic time calculation* (STC) [Intro.86]. STC is based on a calculation of the time consumption for the single operations in the production of a part, e.g. the drilling of a hole, clamping of the part, etc. In other words the time consumption is calculated as a sum of a number of time elements that each is easy to estimate. All the production work is divided into a number of standard operations where formulas are made based on observations of similar operations in the previous production. The quality of the results from a STC mainly depends on the measurement of time consumption and on how general the formulas can be made. When a STC is designed right it produces fairly good time estimates. STC can be used for calculation of both preparation and machining time.

A number of time calculation systems have been developed and they cover various manufacturing processes and operations. One of the more well-known systems is the "Method Time Measurement" (*MTM*). MTM covers mainly manual work like in assembly and is based on 10 types of motion (reach, move a part, twist the hand, etc.) that each has a formula with a number of parameters (e.g. length of the movement) [Intro.86]. With the growing automation in production today the time consumption becomes more deterministic and the use of computer tools therefore more appropriate.

A time estimation system that has been used in the Scandinavian countries for many years is the *Bringby* system [Bringby.65]. It is a manual method for turning, drilling and milling.

Whatever method is preferred it is important to determine how detailed and accurate the time estimates have to be, since the effort put in the development work is proportional with the complexity of the system, and it is important to keep the purpose of the system in mind.

5.11 Cost estimation

Cost estimation can be made for different purposes:

- Calculation of product price
- Comparison between different manufacturing processes
- Make/buy analyses
- Investment analyses
- Cost of design changes

The procedures and the parameters used to calculate the cost are different depending on the purpose. A cost estimate for process plan evaluation can for example not be used in an investment analysis. Figure 5.6 illustrates the important parameters and differences between cost calculations for different purposes.

Expenses in a company are often divided into direct and indirect costs. The direct expenses include the costs that directly relates to a product, e.g. material cost, production wages, machining and assembly costs, etc.. Design are in some cases regarded as an direct expense (especially for larger development work). Indirect expenses cover all other costs and are sometimes referred to as overhead expenses. Examples are the expenses from administration (personnel, accounting), planning, management, etc.. A cost estimate is encumbered with uncertainty and sensitivity analyses must therefore be made before decisions are made.

For both time and cost calculations computerization offers new possibilities. Calculations can be made easy and can therefore give a better basis for decision making. A detailed description of cost estimation principles can be found in [*Jepsen*.75].

	Purpose for cost estimation				
Cost parameters	Product price	Process selection criteria	Make/buy analyses	Investment evaluation	Cost of design Changes
Direct expenses (material, labor cost,	X	X	X	X	X
Indirect expenses (overhead)	X				
Depreciation for prod.equipment (Writing off)	X			X	
Profit	X				

Figure 5.6 Cost estimation - parameters for different applications.

5.12 Different types of process planning

How process planning is carried out depends among other things upon the type of production and the size of the company. The documentation level needed depends on how many different people there are involved in design/process planning/production and whether the activities are performed close to each other geographically and in time. In large companies where many different people are involved in production planning and design it is necessary with detailed specifications in order to ensure correct production, tracability and placement of responsibility. In such companies the process planning function becomes quite large.

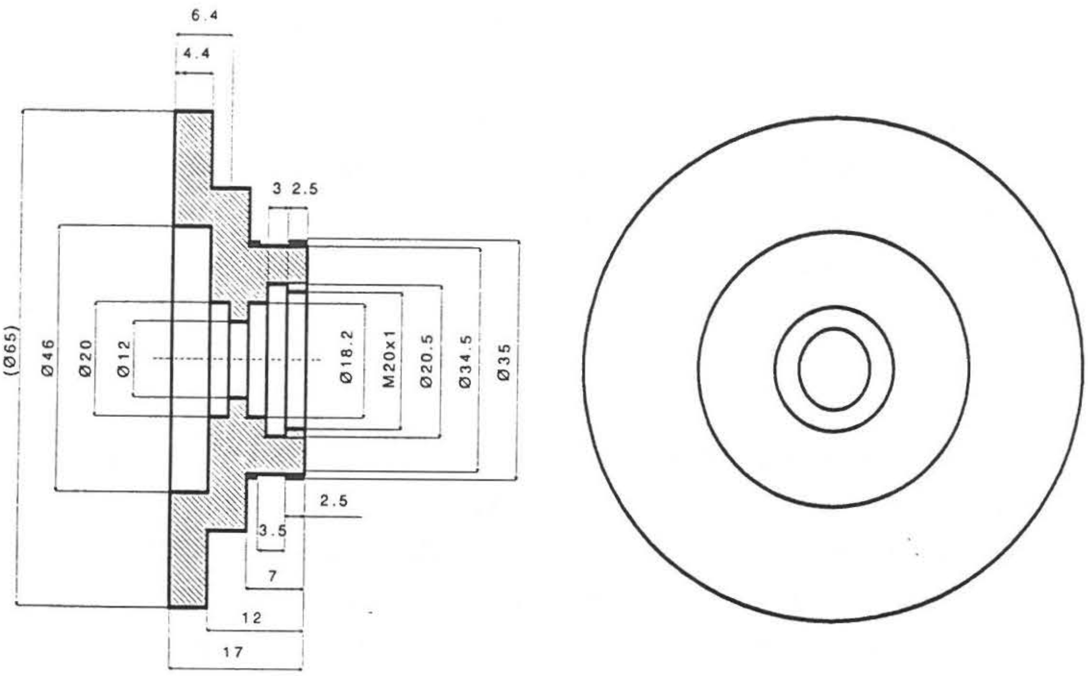
In smaller companies (like the most companies in Denmark) it is much easier for people to talk with each other and clear up misunderstandings or unclear details. Only a few people are involved in the various activities from design to production and this makes it easier to trace errors and place responsibility without a comprehensive written documentation. A large amount of the process planning is done by the staff in the workshop.

A special group of companies mainly produce parts for other companies, i.e. are subcontractors. Here of course it is necessary to have unambiguous documentation for the parts and the required production methods.

The size of the process planning function also depends on the type of production. The size of the production batches and the number of variants have a large impact on how often process planning is carried out. When the batch sizes are relatively small and there are many variants the planning is often done by the operator or the NC-programmer. When the batch sizes are larger it becomes more important to optimize the process planning. With many variants and larger batch sizes (or expensive production) it is necessary to make a more distinct division of the tasks and the personnel is often specialized for planning the production and for operating the machines.

5.13 Process planning example

The different considerations taken in process planning are best illustrated by an example. The part shown in Figure 5.7 is a cylindrical house that holds a bearing. The material is aluminium and it is produced in medium batch sizes. The designer used aluminium to keep weight low and assumed that it could be produced by conventional machining. The design was documented in a drawing like the one shown in the figure. The production prepa-



Process plan

Part number 123456   Product number 33445
Name House   Date Jan 30, 1989

#	Description				Setup time	Machining time
10	Machine saw				0.00	
11	Saw blank from Aluminum bar material, d=65, l=19					0.000
20	Mazak NC-lathe				3.00	0.265
21	Clamp blank			Chuck		0.015
22	Face turn to l=18			facebit f2		0.020
23	Turn	d=50	l=12	sidebit s1		0.050
24	Turn	d=35	l=7	sidebit s1		0.030
25	Groove	d=34.5	l=3.5	groovebit g1		0.010
26	Drill	d=12	l=17	drill d12		0.030
27	Drill	d=18.9	l=3	drill for M20(18.9)		0.015
28	Int. turn	d=18.2	l=2	int.groovebit i3		0.040
29	Int. groove	d=20.5	l=3	int.groovebit i3		0.015
31	Turn thread	M20x1	l=2.5	tap M20		0.040
40	Okuma NC-lathe				3.00	0.105
41	Clamp part at d=50			Chuck		0.015
42	Face turn to l=17			facebit f2		0.020
43	Int. turn	d=46	l=4.4	facebit f2		0.050
44	Int. turn	d=20	l=2	facebit f2		0.020
50	Cortini NC milling center				3.5	0.275
51	Clamp part in fixture			f24778		0.020
52	Mill key seats nv=60			mill d10		0.080
53	Drill 8 holes d=2.3			drill M8(d2.3)		0.050
54	Chamfer 8 holes 45			chamferdrill		0.025
55	Drill thread in 8 holes M3			tap M3		0.100
60	Anodize white				0.0	0.000
70	Store				0.0	0.000

Figure 5.7 Example for process planning: House for bearings.

ration department receives the drawing and starts the process planning by considering the process selection.

Process selection includes how to machine the blank part: turning of the concentric cylindrical surfaces (in two clampings), drilling of holes, milling of the key seats and a surface treatment to improve the material properties. Depending on the expected production size pressure die casting may be considered as an additional process, that will reduce material waste.

Machines that can be selected for turning can either be a single lathe that in two clampings turn all the cylindrical surfaces, or it can be two lathes where the one lathe machines the part on one side from bar material and the other lathe machines the part from the other side. A milling center can drill the holes and machine the key seats. Alternatively a four axis lathe with equipment for material handling can do all the machining in two setups.

In the operation planning the sequence of the operations is planned, tools are selected and the cutting parameters are calculated. Finally estimates are made for setup time on each machine and machining time for each operation.

References - Process Planing

[Alting.82] Alting Leo: Manufacturing Engineering Processes, Marcell Dekker, Inc. N. Y. 1982

[Bilberg.89] Bilberg Arne: Simulation as a tool for developing FMS control programs, PhD thesis, Lab. of process and production engineering, Technical university of Denmark, Lyngby, Denmark, January 1989, 258 pages.

[Bolz.63] Bolz Roger W.: Production Processes, The Producibility Handbook, Inductrial Press Inc., New York, 1963.

[Bringby.65] Bringby Einar: Tidsformler 4, Sveriges Verkstadsförening, 1965, 57 pages.

[DeGarmo.69] Degarmo E.P.: Materials and processes in Manufacturing, 3 ed., London Macmillan, 1969, 949 pages, (also published in 1984).

[Intro.86] 8371 Introduktion til driftteknik, note 5, Metodestyring, Driftteknisk Institut, DTH, 1986, (in Danish).

[Jepsen.75] Jepsen Erik Lund: Principper for omkostningsberegninger og forkalkulationer, Afdelingen for Mekanisk Teknologi, Danmarks Tekniske Højskole, Publ.nr. AP.75.15, 1975, 80 pages, (in Danish).

[Jepsen.78.a] Jepsen Erik Lund: Udvikling af en hensigtsmæssig strategi for procesvalg, Lab. for Almen Procesteknik, AMT, Technical University of Denmark, Publ.no.78.10, 1978, 148 pages, (in Danish).

[Jepsen.78.b] Jepsen Erik Lund: Valg af industrielle fremstillingsmetoder, PhD thesis, Lab. for Almen Procesteknik, AMT, Technical University of Denmark, Publ.no. AP.78.04, 1978, 111 pages, (in Danish).

[Jepsen Jensen.84] Jensen Lars Jepsen: Computer aided tool and die design, PhD-thesis, Lab. for Almen Procesteknik, Danmarks Tekniske Højskole, Aug. 1984, Publ.no. AP.84.18, 146 pages.

[Jørgensen.78] Jørgensen Jørgen: Systematic tool and die design, Lab. for Almen Procesteknik, Danmarks Tekniske Højskole, Sept. 1978, Publ.no. AP.78.111, 139 pages.

[Jørgensen & Alting.81] Jørgensen Jørgen & Alting Leo: Manufacturing Process Taxonomy, Lab. of Process and Production Engineering, Technical University of Denmark, Publ. no. AP.81.07, 1981, 10 pages.

[Larsen.89] Larsen Niels Erik: Simulation - a concept for production scheduling, PhD-thesis, Lab. of process and production engineering, Technical university of Denmark, Lyngby, Denmark, January 1989, 398 pages.

[Manufacturing Consortium.85] Manufacturing Consortium, Edited by Dell K. Allen & Leo Alting, Brigham Young University, Provo, Utah, USA, 1985, 17 volumes.

[Metals Handbook.69] Metals Handbook, American Society for Metals, Metals Park, Ohio, volumes 1-7, 1961, 1064, 1967, 1969 and later.

[Value.63] Value Control Guide, Value Analysis Inc., N.Y. USA. 435 pages, 1963.

CHAPTER 6. AUTOMATED PROCESS PLANNING

This chapter describes how the process planning activity can be automated, and how computers can be used for this purpose. A generative process planning system (XPLAN) that has been developed by the author is described.

6.1 Automating process and machine selection

Process and machine selection are carried out by setting up critical requirements, by searching for suitable solutions and by comparing these solutions to find the best. Automation of process and machine selection involves finding efficient ways of encircling critical requirements and effective search/comparison techniques.

The designer plays as mentioned previously a very important role in selecting manufacturing processes. Often he/she has one or more specific processes in mind when designing the part. So the question is what type of computer system that will be most useful in order to make a better process selection. One way is to integrate the process selection in a process planning system, which integrates process/machine selection, operations planning and time/cost estimation. In such a system though, the number of possible alternative processes is limited, due to the designers choice of part geometry, material, etc. Automation/computerization of the manufacturing decisions made by the designer might therefore have a larger effect. Design support systems that are suited for this purpose are described in chapter 7.

Next step in process planning is more specific and deals with the selection of the exact machines to use. Sometimes the designer even has a specific machine in mind and at other times he/she just specifies the type of process e.g. lathe or milling machine. The machine selection is then made by the process planner and it is based on information about part characteristics and process capabilities.

6.2 Automated operation planning

Operation planning can be carried out in three basically different ways called forward planning, backward planning and macro planning. In forward planning the operations are selected one by one in order to get from the raw material to the finished part. Backward planning works the other way around and starts with the finished part. Inverse operations are then added one by one until the raw material is reached. The operations sequence is then the reverse of this list. Macro planning uses a number of standard macros, that each is a predefined operation sequence for a standard feature like a pocket, a hole or a curved surface. [*Chang & Wysk*.85] give a more comprehensive description of forward and backward planning.

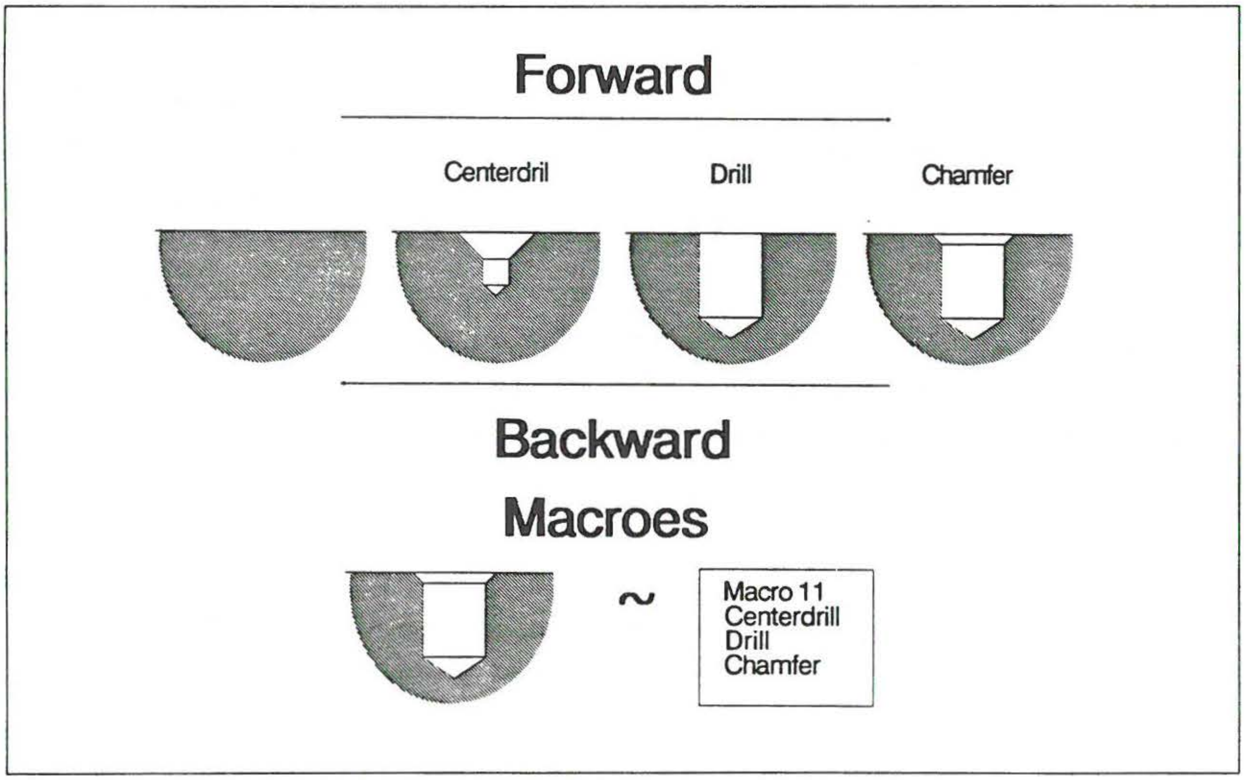


Figure 6.1 Forward, backward and macro operation planning.

When designing a process planning system, the question comes up, whether it is possible to find an optimal operations sequence. It is difficult, if not impossible, to define an absolute optimum, since it depends upon so many interrelated factors in a larger perspective. One operations sequence might seem optimal, but if for instance the lot size is changed, or if the tools do not match the requirements (e.g. the tool breaks), the optimum sequence would be another. In many cases it will be sufficient to have a "good" solution, which not necessarily is optimal. The calculations and knowledge handling for finding a "good" solution, are also easier than what the optimal case demands. The parameters are calculated as "optimal" based on qualified assumptions and considerations about tool shift, machining time and the risk of tool breakage.

Many new production machines can calculate the parameters they need on the basis of only a few informations and the need for calculations in advance is therefore limited. Many NC-machines include some operation planning in their macros, e.g. pocket macros.

6.3 Automated time and cost estimation

Time and cost estimation are very company dependent like other process planning activities, and it is therefore necessary to tailor automatic systems to fit the specific company needs. Various programs offer facilities that can be of significant help in the development of estimation systems.

In the USA the Metcut Association has developed programs for estimation of cutting parameters and time [*Metcut*.87]. *Sanviken Coromant* is marketing a program called *Corocut* [Corocut.87], that helps calculating cutting parameters and machining time (for Sanviken tools). XPLAN contains a good basic framework for the development of such systems, but in the present version only time calculation for turning is included. XPLAN estimates setup and machining times.

6.4 Computerizing process planning

Process planning can be computerized on different levels. The simplest way is to use the computers text editing facilities, and in this way limit the amount of paperwork. Though simple, this use of the computer can be very timesaving for the process planner, and thereby profitable. Computerized process planning systems can be classified in three categories namely the constructive, the variant and the generative approach. Figure 6.2 illustrates the economic regions for different process planning systems. An overview and description of a number of CAPP systems can be found in [*Zhang & Alting*.89].

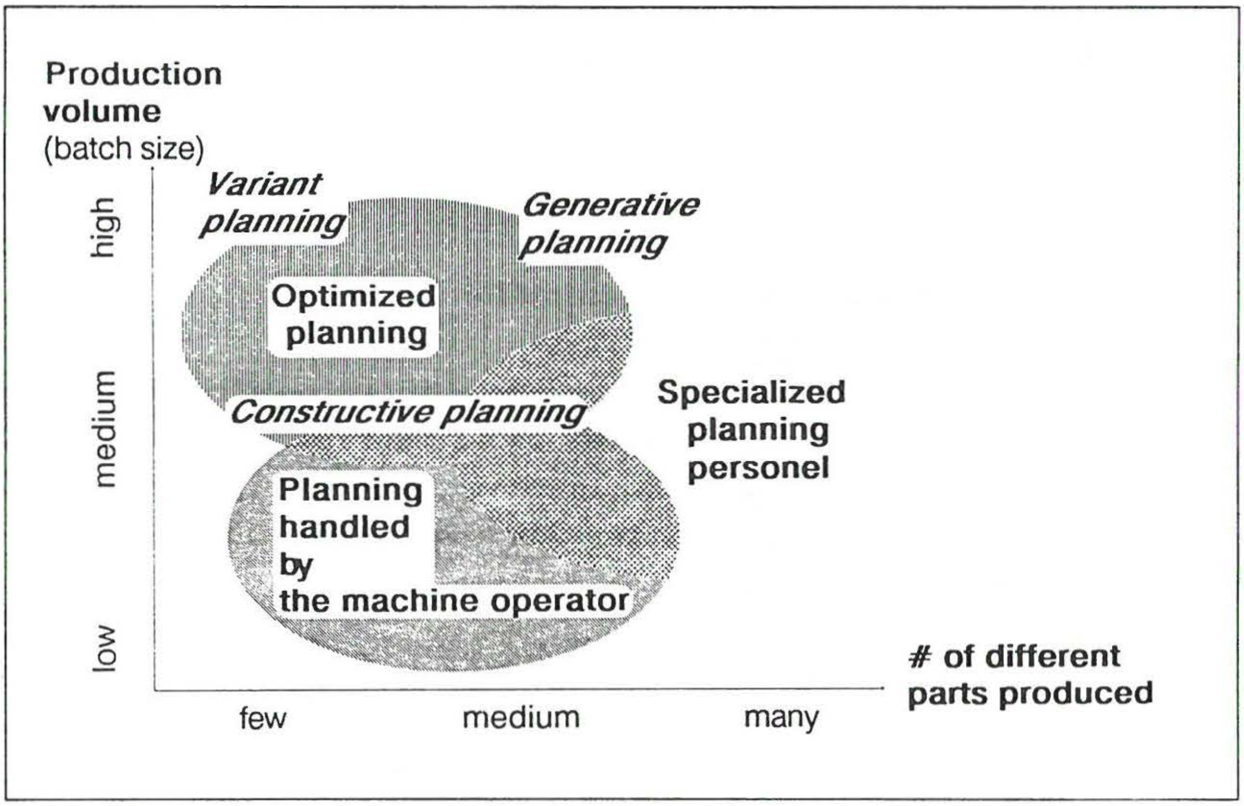


Figure 6.2 Economic regions for different process planning systems.

The effort in building computer aided process planning systems has until now been concentrated on machining processes and especially the turning process. But there is also a need for CAPP systems within other process areas, like sheet metal processes and welding.

6.5 Constructive approach

Constructive CAPP uses an advanced dedicated editor where all the standard operations in a company are described. Various calculations can also be done by the computer. Constructive CAPP systems supply only a low level of automation and only little process planning knowledge is stored in the system. On the other hand it is possible in such a system to cover all the process planning in a company and it is therefore realistic to implement such systems in companies today [*Evans & Sackett*.84]. Benefits that are obtained using such systems are standardized formats and reduced planning time. Commercial systems that use the constructive approach include *Centreplan*, *Multiplan* and *Supercapes* [*Lyons*.86] (some of the mentioned systems do also include variant and generative planning). The *Locam* system from PAFEC does also include constructive CAPP [*Locam*.88]

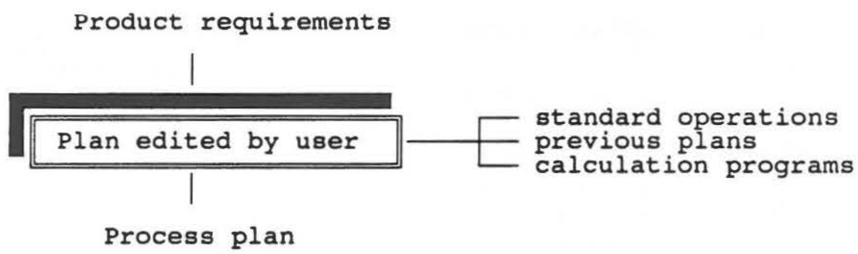


Figure 6.3 Constructive process planning.

6.6 Variant approach

A more sophisticated way of computerization is called the *variant approach*. In the variant approach parts are grouped into a number of part families, characterized by similarities in manufacturing methods, and thus closely related to *group technology*. For each part family a standard process plan, that includes all possible operations for the family, is stored in the system. Through classification and coding, a code is build up by answering a number of predefined questions. This code is then used to identify the part family and the standard plan for the part. The standard plan is retrieved and edited so it matches the specific part. The variant approach is widely used [*Ewersheim*.82], [*Zhang & Alting*.89] and has both advantages and disadvantages. Advantages compared with the manually performed process planning are easy plan editing which significantly reduces the tedious paperwork, and a certain degree of standardization due to the standard plans. Disadvantages are that only a low level of automation can be obtained and that it is a static system with respect to changes in products and machinery. It is not necessarily a simple task to make changes in the system.

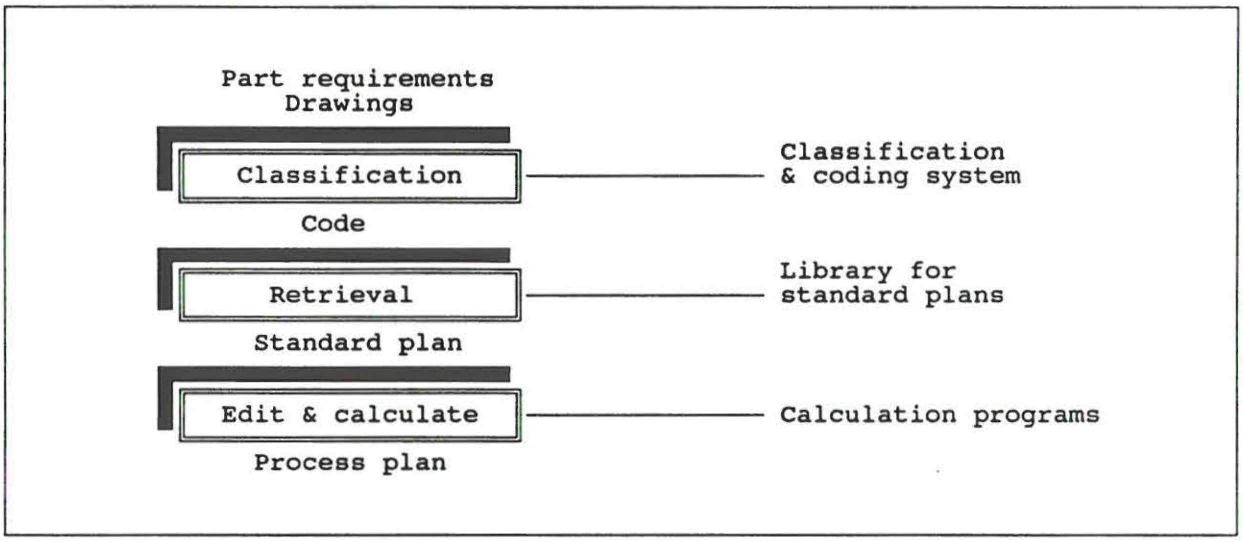


Figure 6.4 Variant process planning.

6.7 Generative approach

A third and more advanced way to computerize process planning is the *generative approach*. In the generative approach most of the planning decisions are made by the computer. The objective for generative systems are to produce individual plans for individual parts automatically. A generative system can automatically select an "optimal" sequence of machines, perform the operation planning for each machine, select tools and fixturing, and perform various calculations. In the most advanced systems the user only has to input part data (see next section on input to CAPP systems) and select between a few good alternative plans produced by the system. The system uses knowledge about the capabilities and limitations for each machine, and about the production of the different details on the part (features) for part analysis and for generating process plans. This knowledge can be represented by different decision logic methods as described in chapter 3.

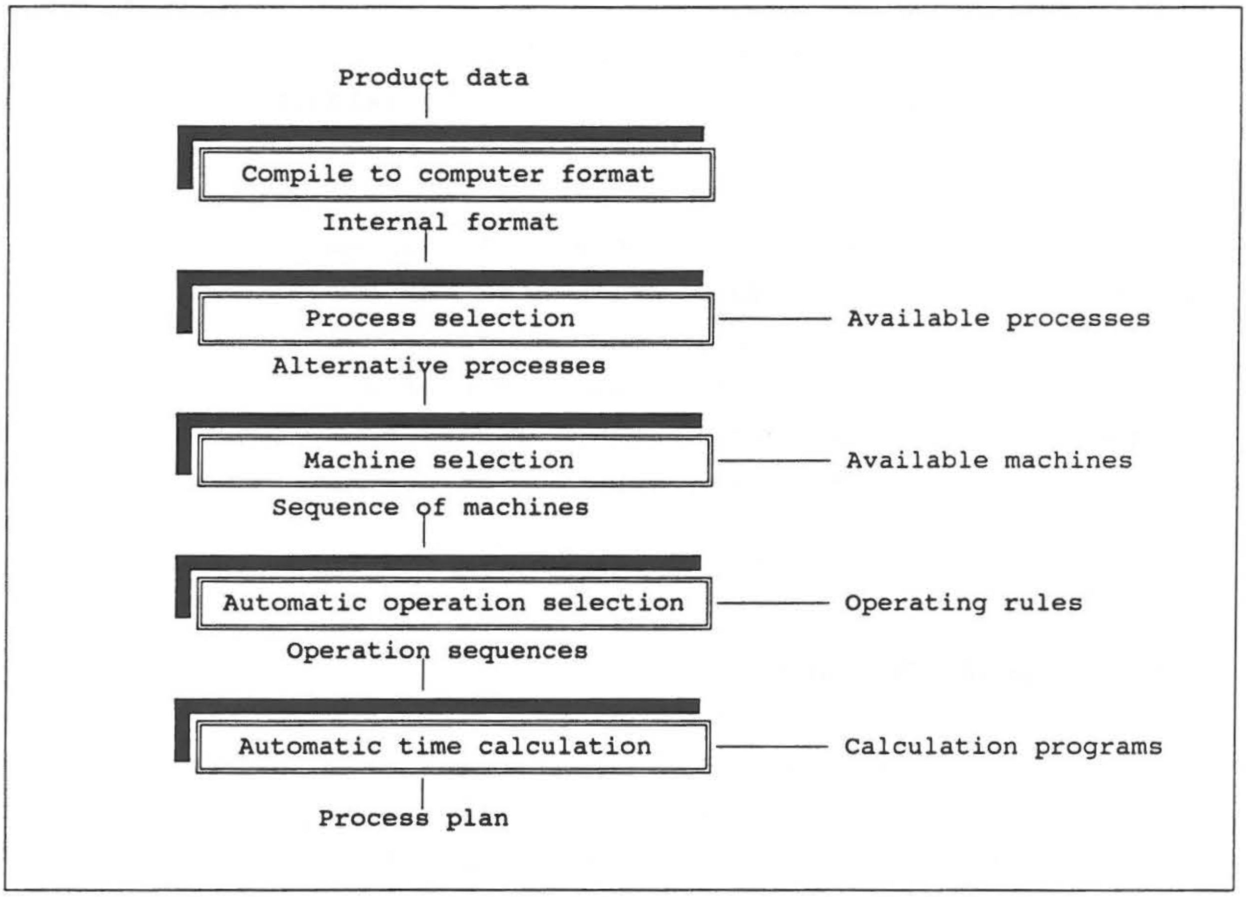


Figure 6.5 Generative process planning.

Generative systems have many advantages like standardized plans, faster plan generation and the possibility of adding many automatic calculations (time, parameters, cost). The main disadvantage is the large initial job with knowledge acquisition. Since the generative systems can do the detailed planning more or less automatically, a large amount of company specific knowledge is required. The company can not buy a turn-key program, but has to invest both manpower and money for the development. Especially investment of manpower can be a problem since the experienced process planners required are hard to spare, since they often are few in number and are very busy in their jobs.

6.8 Input to CAPP systems

Input to CAPP systems can come either as a text input where the user answers a number of questions (dialogue input) or as graphic input where the part data is gathered from a CAD model (CAD input). The former is the most common in existing systems, while the latter is still a fairly undeveloped area due to its complexity. Classification of part characteristics (using *group technology* principles) can be a good help as a front-end to a generative system. It simplifies both the CAD model recognition in CAD interfaces and reduces the number of questions in dialogue input systems. The part classification can be done by the designer or later by the process planner and results in a code that is used as part of the input to the process planning system.

6.9 Design/CAD interface

Interpretation of a CAD model is a problem that not yet is solved satisfactory [*Chang* & *Wysk*.85], [*Zucherman*.87]. The reason for this is that CAD systems use simple geometrical primitives like arcs and lines or solids to represent a model, and a data structure that only uses such primitives can not represent relations between part elements. In CAD systems today the designer translates his ideas about a part, which include the functional relations between the different primitives into the CAD system representation where the relational part is lost. When a process planner or computer program needs to extract information from the CAD model there are problems interpreting the information right, since the relational information is either not there or only given implicitly.

Much work has been put into systems that can interpret part data from a CAD database. [*Choi*.82] presents a system that can produce process plans for simple prismatic parts. The system receives input from a CAD database where *elementary machined surfaces* (e.g. simple hole, chamfer) can be recognized for holes, slots and pockets.

The fact that the CAD database does not have a complete description of the part means that such type of systems can only solve some of the interface problems.

Feature based design

The shortcomings of conventional CAD representations are the reason why another approach to CAD called the *feature based design* (*FBD*), is being considered. FBD uses a different type of primitives called features that relate to product functions and to manufacturing information. Examples of features are slots, holes and chamfers. FBD is limited to use the predefined basic features, and is therefore best suited for part families, i.e. families of parts with a number of geometric, production or other similarities (e.g. gear shafts or pump houses). Feature based design is discussed in more detail in chapter 7.

6.10 XPLAN

In [*Lenau*.85] the ground was laid for the present research work, and a first prototype CAPP system was developed. This prototype has now been further developed into the generative process planing system called XPLAN. The main objectives of XPLAN are to:

- generate process plans at an expert level (in industrial environments)
- enable automatic plan generation for alternative machines
- allow machine availability and other priorities to be included

in the process selection and the the priorities to be externally updated
- include individual operation specifications, tooling selection and calculation of process data and time
- be user friendly both in use and adaptation

XPLAN is based on the *DCLASS* tree processor. The user dialogue is handled in hierarchical tree structures and decisions are made through logical operators and variable values attached to the branches in the trees. The decision logic is a combination of decision trees and rule based expert systems.

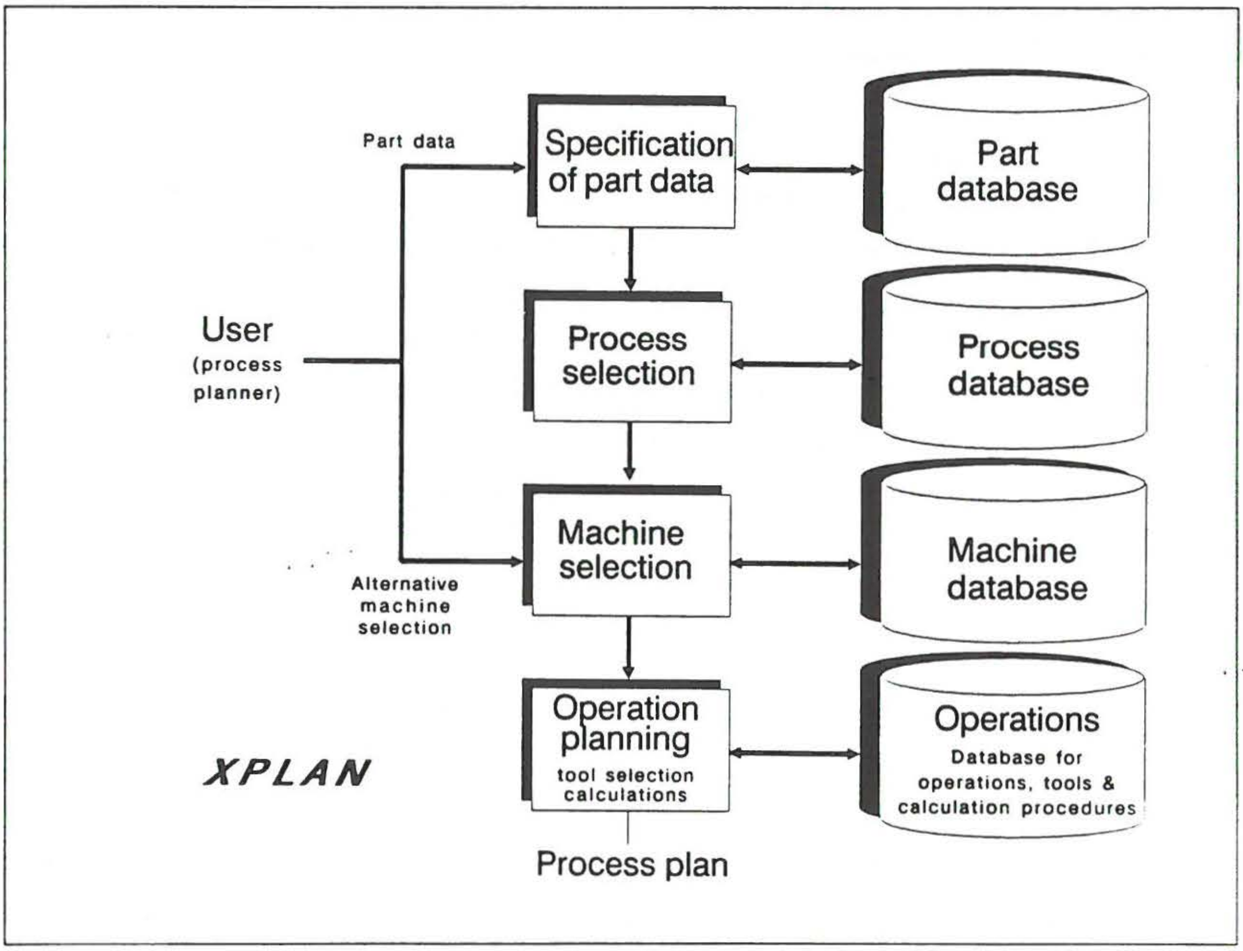


Figure 6.6 The XPLAN structure.

Figure 6.6 shows in principle the different elements in the XPLAN decision making. For each of the boxes in Figure 6.6 there is defined one or more tree structures. A set of keywords are triggered automatically when the user interactively defines the actual product requirements (in the part specification tree) and these keywords (output flags) will match other keywords (input flags) that defines the solution/plan/action in the planning trees (process selection, machine selection and operation planning). Many trees may be involved in the solution generation.

XPLAN works in four steps as illustrated in Figure 6.6. First step is the part specification where the user has to answer a number of questions about the part. It is questions about geometry, dimensions, material, batch size, etc. The questions are either of the multiple choice type or the variable value type. The questions and their sequence are build into a tree structure, where it also is possible to control whether a question is asked at all. If for example the user has told the system that the part is solid then it would be meaningless to ask about internal features. Since the sequence of questions is controlled in the tree structure it is possible to backup to previous questions and change the answer. To each question it is also possible to add extra information (graphics and text) which only will be displayed if the user asks the system to do so.

Decision making is controlled by the flags (input/output keys) which can be added to any branch in the tree. Those keys can be used to make automatic selection of other branches in the same or in other tree structures. The keys can be combined logically (AND, OR, NOT) and provide a flexible way of designing rules. Figure 6.7 describes the tree structures in XPLAN and the use of input/output keys. DCLASS offers a very strong debugging help in the TREEDRAW facility. It simply draws the tree structures with questions, keys, etc.

Second step in XPLAN is the process selection where the relevant manufacturing processes are selected. The process selection is also formed in a tree structure which is traversed automatically by using the keys and variables generated during the part specification. Each process has its own tree structure where the characteristics of the process is described.

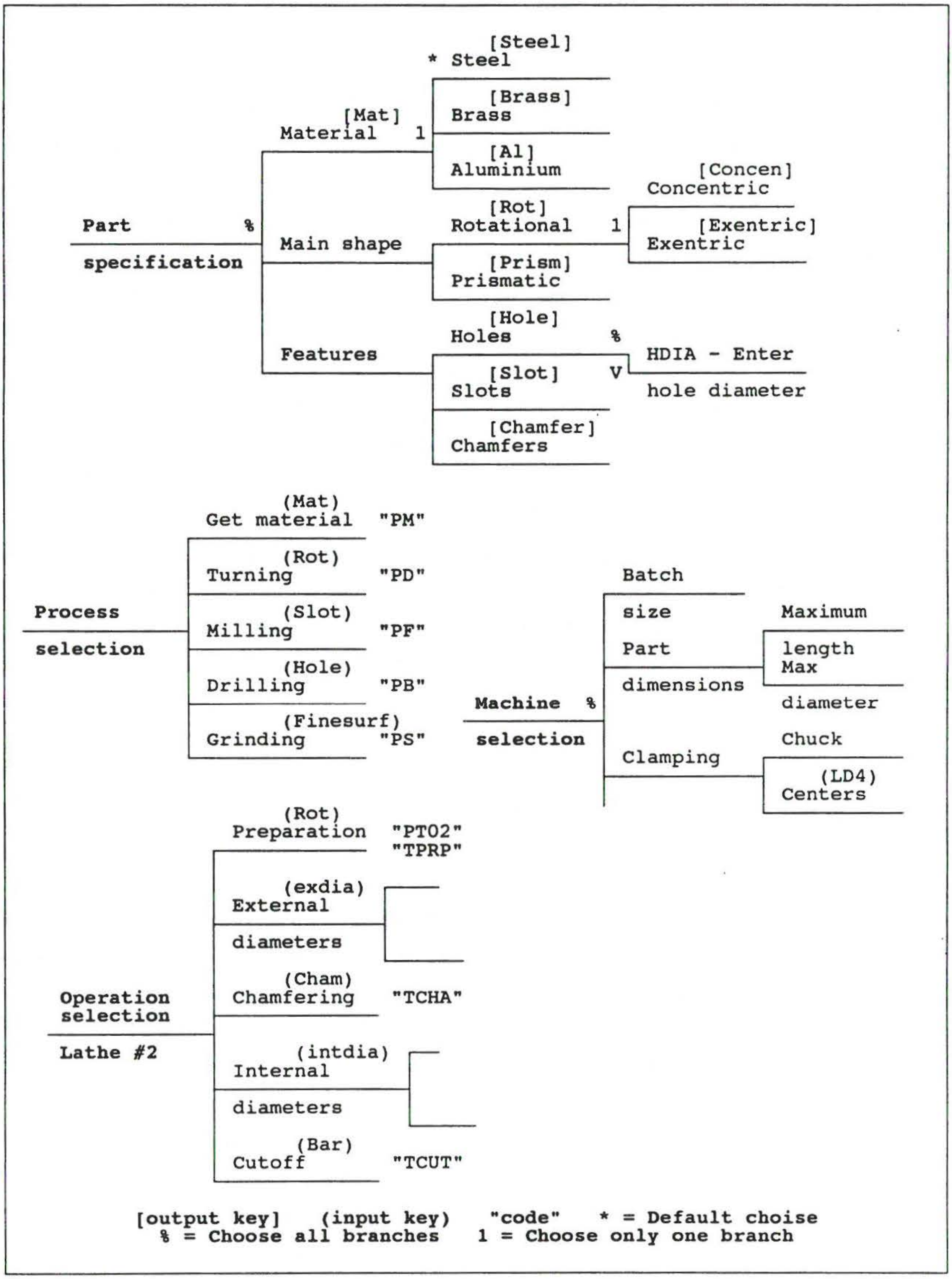


Figure 6.7 XPLAN tree structures and input/output keys.

Third step is to traverse a tree structure for each of the selected processes. The traversal results in a search profile. This profile is compared with each machine capable of performing the process and candidate machines are identified. The candidate machines are sorted according to the process planning strategy. A strategy could be to choose a faster machine before a slower machine or to choose a low cost machine.

The fourth step is to plan the operation sequence and to calculate expected operation/process time for the machines with the highest priority. There is a separate tree structure for each machine so it is possible to individually tailor the sequence of operations and time calculation.

```
------------------------------------------------------------------------------
                              CIM - MINILAB                        1988- 4-20
                         XPLAN    PROCESS   PLAN
Part ID:PN005                                      Planner:SCC
------------------------------------------------------------------------------
 P#  O#      Description                                 ID
------------------------------------------------------------------------------

 10 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
          Storage/retrieval System
     10     2 Brass  Plate(s)  L=76.0 B=20.0 T=2.0    MS 58 Pb
     20     Get Pallet                                A04
     30     Get fixture                               F02
     40     Fix part(s) in fixture                    FPN005A
     50     Load Input Conveyor                       LPN005A
 20 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
          ASEA IRB/L6 Robot
     10     Inp. Conv. --> Mill                       EE#3
 30 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
          Storage/retrieval System
     10     Store Pallet
 40 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
          CORTINI H105 NC milling machine
     10     Drill 4  holes  D=4                       22
     20     C'Sink Holes.                             30
     30     Engrave Text.                             11
 50 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
          ASEA IRB/L6 Robot
     10     Mill --> Inspect. St.                     EE#3
 60 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
          Inspection Station I
     10     Deburr and polish 2 face(s)               OPN005A
     20     Inspect Part.                             IPN005A
     30     Fix part(s) in fixture                    FPN005B
 70 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
          ASEA IRB/L6 Robot
     10     Inspect. St. --> Outp. conv.              EE#3
 80 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
          Storage/retrieval System
     10     Unload Conveyor
     20     Unload Part(s).
     30     Store Part
     40     Store Fixture
------------------------------------------------------------------------------
```

Figure 6.8 A process plan generated by XPLAN.

All necessary data are now created and a plan can be generated (Figure 6.8). This plan includes the best machines according to the chosen strategy. It is possible to ask for alternative machines and get a new plan with operations sequences and time calculations for the new machines. XPLAN is integrated in the CIM miniature laboratory that was mentioned in chapter 2 [*Christensen*.88], [*Pedersen*.88], where it generates plans like the one shown in Figure 6.8.

The format of the plan can be changed very easily, since it only requires adjustments in an ASCII file dedicated to that purpose. It is thus possible to get an output that will fit the company specific needs. XPLAN is also described in [*XPLAN*.86].

6.11 Test of XPLAN in a company

A program based on XPLAN that generates simple routings and calculated times were developed in a short time (two weeks) in cooperation with a major Danish company. The main purpose was to build a synthetic time calculation system. The calculated times were in the company used for production scheduling, wage calculation and investment evaluation. The program was based on the routings and times from 50 different process plans. The plans were reviewed to find common characteristics that could serve as parameters in the time calculation program and the following parameters were used:

Preparation time	: Introduction, get tools, get material
Setup time	: Change clamping, setup tools, adjust tools, personal time.
Machining time	: Cutting volume (length * thickness of each cut) for vertical and horizontal surfaces, number of chamfers, grooves and knurls, cutting volume and length for thread, cutoff.

The system was tested on 40 other parts from the company. The result of the test was that the routings were acceptable (with minor exceptions) and the time estimates were regarded close to what a planner would have estimated for 88% of the parts. These results are fairly good considering the short development time and the weak time data that was used to make the time calculation (similar parts could have very different times in the existing plans).

References - Automated Process planning

[Chang & Wysk 85] Chang Tien-Chien, Wysk Richard A.: An introduction to automatic process planning. Prentice-Hall 1985.

[Choi.82] Choi Byoung Kyu: CAD/CAM compatible, tool-oriented process planning for machining centers, PhD-thesis, Purdue Univ., Dec. 1982, 277 pages.

[Corocut.87] Corocut Handbook, Sanviken Coromant, Abildager 26-28, 2605 Brøndby, 1987.

[Christensen.88] Christensen Steen C.: Computer Integrated Manufacturing Miniature Laboratory (CIM-ML), Overall structure, database and communication interface, Ph.D. thesis, Institute of Manufacturing Engineering, The Technical University of Denmark, Publ.no. AP.88.16, July 1988, 160 pages.

[Evans & Sackett.84] Evans S., Sackett P.J.: Realized performance in a novel approach to computer aided process planning, Int. Conf. on Computer aided Engineering, Univ. of Warwick, Coventry, UK, Dec.10-12 1984, p.48-52.

[Ewersheim.85] Eversheim W.: Survey of Computer Aided Process Planning Systems. CIRP annals 1985, vol 34/2/1985.

[Lenau.85] Lenau T.: Datamatstøttet Procesplanlægning, Eksamensprojekt (Master thesis), Procesteknisk Institut, DTH, 1985, 99 pages, Publ.nr. PI.85.04.B/AP.85.03, (in Danish).

[Locam.88] Locam CAPP, Brocure material from PAFEC, Kampsax data, Dagmarhus, 1553 København V.

[Lyons.86] Lyons J.W.: The role of process planning in computer aided manufacturing, Knowledge engineering and computer modelling in CAD conference (CAD'86), London 2-5 Sept. 1986, p.379-384.

[Metcut.87] Brocure material for MetCAPP process planning software and Cutdata calculation software, Metcut Research Assocoates, 3980 Rosslyn Drive, Cincinnati, Ohio 45209.

[Pedersen.88] Pedersen Morten Als: Computer Integrated Manufacturing Miniature Laboratory (CIM-ML), Cell control, robot programming, communication, Ph.D. thesis, Institute of Manufacturing Engineering, The Technical University of Denmark, Publ.no. AP.88.17, July 1988, 116 pages.

[XPLAN.86] Lenau Torben: XPLAN generative process planning system, Reference Manual, Institute for Product development, The technical University of Denmark, December 1986, 39 pages.

[Zhang & Alting.89] Zhang Hongchao, Alting Leo: Computer aided process planning: the state-of-the-art survey, Int. Journal of Production Research, april 1989, Vol.27, no.4, 31 pages.

[Zucherman.87] Zucherman, M.I. A knowledge base development for producibility analysis in mechanical design, CIRP Seminar on Computer Aided Process Planing, Pensylvania State University, PA, USA, June 1987, p.301-307.

Bibliography - Automated Process planning

[Allen.86] Allen D.K., Computer aided process planning: Software Tools, Integrated and Intelligent Manufacturing, Anaheim, CA, USA, 1986.

[Alting.86] Alting L., Lenau T.: Integration of the Engineering Functions in the CIM Philosophy, ISATA, Flims, Switzerland, 6-10 okt. 1986.

[Alting & Zhang.88] Alting L., Zhang H.C., Lenau T.: XPLAN - an expert process planning system and its further development, Proc. of the 27 int. MATADOR conference, Manchester, UK, 20-21 april

1988. Publ.nr. AP 88.26

[CAPP.87] Computer-aided Process Planning, 4 papers, CIM review - The Journal of Computer-Integrated manufacturing, Fall 1987, Vol.4, No.1.

[CIRP.87] Computer Aided Process Planning, CIRP seminar at Pensylvania State University, PA, USA, June 1987.

[Halevi.80] Halevi Gideon: The role of Computers in Manufacturing Processes, John Wiley & Sons, New York, USA, 1980, 495 pages.

[Lenau.86.b] Lenau T., Alting L.: XPLAN - An expert process planning system. The Second International Expert Systems Conference, London, UK, Sept/Oct. 1986. Publ.nr. AP 86.26

[Lenau.86.c] Lenau T., Alting L.: Tutorial on Computer Aided Process Planning The Autofact Conference, Detroit, Michigan, USA, Nov.1986. Publ.nr. AP 87.02.

[Lenau.87] Lenau T.: The role of Process Planning within the factory, Lab. for Almen Procesteknik, PI, DTH, May 1987, Publ.nr. AP 88.06.

[Lenau.88.a] Lenau T.: Datamatstøttet Procesplanlægning, ATV-SEMAPP Seminar on Computer aided Process Planning, The Technical University of Denmark, 7 April 1988.

[Lenau.88.b] Lenau T.: Computer Aided Process Planning, 50 min. Video for the European PACE educationprogram, June 1988.

[Major.87] Major F., Grottke W.: Knowledge engineering within integrated process planning systems, Robotics & Computer-Integrated Manufacturing, 1987, Vol.3, No.2, p.209-213.

[Marks.87] Marks Peter C.: Expert System-Based Generative Process Planning, Autofact Conference, Detroit, USA, November 1987.

[Tengvald.84] Tengvald E.: The design of expert planning systems, Linköbing University, Linköbing, Sweden, 1984, 162 pages.

[Vedel.88] Vedel M., Siegumfeldt J.: Procesplanlægning for prismatiske emner, Procesteknisk Institut, DTH, marts 1988, Publ.nr. AP.88.05 / PI 88.06, (in Danish).

[Weill 85] Weill R.D.: Present tendencies in computer aided process planning. PROLAMAT 85, Paris, France, June 1985.

CHAPTER 7. DESIGN SUPPORT SYSTEMS

In this chapter the design process is analyzed and the objectives, the phases in and the techniques for *design support systems* (*DSS*) are described. DSS can help to increase the company competeability through better products and more efficient productions due to better information to the designers. A major area for application of design support systems is *design for manufacturing* (*DFM*). DFM involves considerations about how the product can be produced (*design for producibility - DFP*) and about how it can be assembled (*design for assembly - DFA*). DFM is also called *simultaneous engineering* because the aim is that both design and production engineering are carried out simultaneously. Other terms used for DFM include *concurrent product development*, *life-cycle engineering*, and *design fusion*.

DFM is described as a mean of integrating design and production, and especially selection of manufacturing processes is explored. An efficient selection of the right processes is vital to the competitive situation for the company. Design support systems can assist the designer in the selection of manufacturing technology, but can also present the designer with alternatives that he otherwise may not have considered, i.e. work like an idea generator. The three level methodology described in chapter 4 is used to describe how to design process selection systems.

7.1 Modelling the design process

In order to create design support systems it is important to understand the design process and to realize where the considerations about manufacturing and other important factors come in. The design process is different from person to person, from company to company and for different products, and consequently there is a need for a general model (a framework) that can describe various design processes. In spite of the differences there are a number of features that characterize the design process in general, and they can therefore be included in the general model. In this paragraph <u>four</u> different approaches to model the design process are described, and they are used to investigate how and where the manufacturing decisions are taken.

It can be discussed if it is possible to model the design process. Some will argue that due to its innovative and ill structured nature it is hard if not impossible to establish a useful model. It is true that many of the most bright inventions are made in unpredictable situations like in a swimming pool or on a picnic tour, and that different designers work very differently, but when the design work is analyzed many common characteristics show up. Those characteristics form a model. In a teamwork where several designers have to work together and maybe even cooperate with people from other company functions, it is important that there is a common agreement on the sub-activities and their interrelations in design, i.e. a model of the design process.

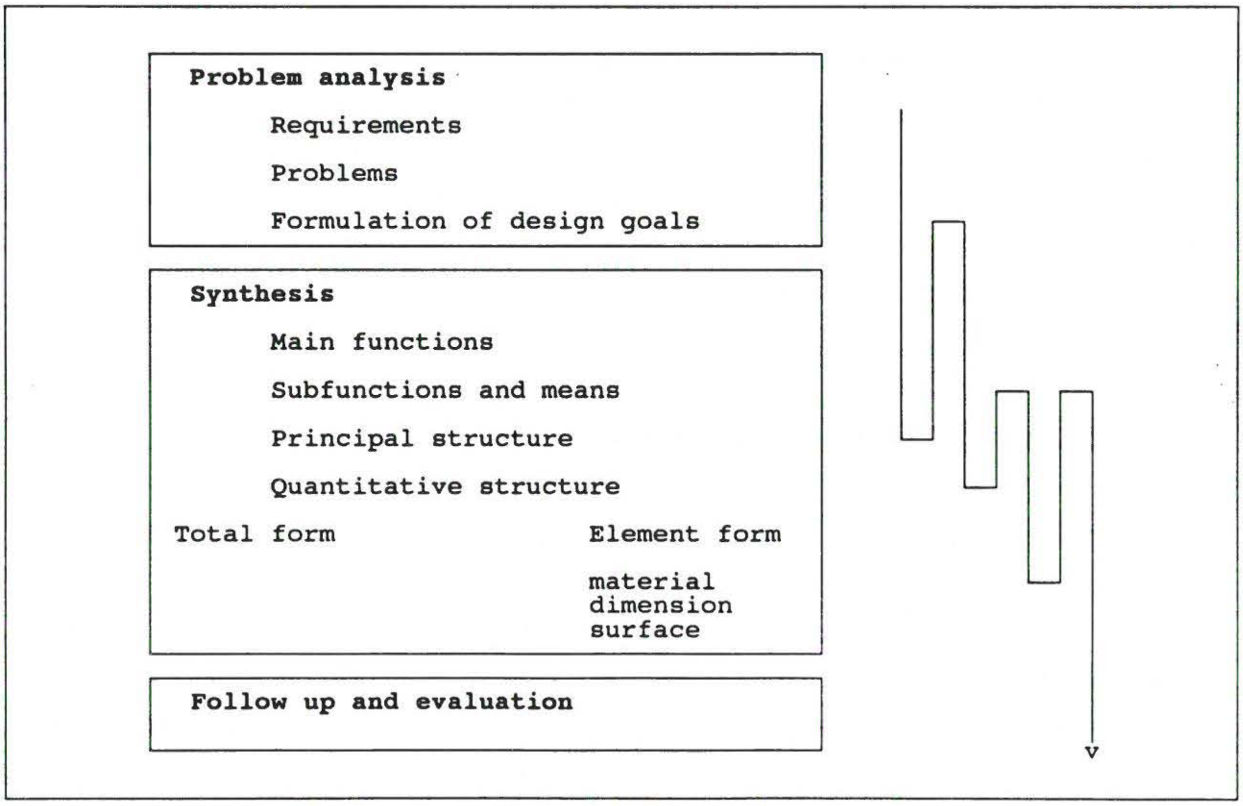


Figure 7.1 Elements in product design [Stahl.77].

Stahls design model

[*Stahl*.77] describes the design process as shown in Figure 7.1. The design process is usually initiated by recognition of a need where the product in mind hopefully can be a solution. First phase in the design process is an analysis of the need and the problem derived from here. Stahl emphasizes the difference between a need and a problem. The need represents an unsatisfactory situation while the problem is formulated by the company on the basis of the need. The problem does therefore represent a subset of the total space of the means that can be used to fulfil the need. A problem can be encircled by looking at the need from various abstraction levels. This is illustrated by the example shown in Figure 7.2.

Observation: Isolation mats that dust during assembly

Abstraction level	need
1	A machine for mat rapping
2	Dust removal equipment
3	Mats that do not dust
4	Easy-to-assemble mat

Figure 7.2 Abstraction levels in need comprehension, freely after [Stahl.77].

The example illustrates formulation of a problem based on an observation of isolation mats that dust during assembly. On a low abstraction level the problem is "to much dust in the mats" and a solution could therefore be some kind of a mat rapping machine that can remove the dust. More generally the solution is dust removal equipment. Thinking on a more abstract level it can be realized that the problem is the mat itself, and the solution therefore is a new type of mat that does not dust. More generally the solution is mats that are easy to assemble.

Having realized the need and formulated the problem the design goals must be formulated. The goals set up limits and criteria for the suitability of the different solutions. Goals can include demands for price, expected lifetime, in-house production, conventional or new technology, etc. It is worth while noticing that set up of the goals involves considerations and selection of manufacturing technology at a more superior level.

The setup of the main functions for a product is a description of how the formulated problem can be solved. Most often there exist several alternative sets of main functions. Splitting those functions into subfunctions and means give a more detailed description. The principal structure shows how the principal means are related, and the quantitative structure shows how it is practically done. When the functional structure is determined the product can be dimensioned and shaped. In the follow up and evaluation phase manufacturing considerations come in again but at a more detailed level.

Asimows design model

[*Asimow*.62] sees the designer in a very central role where the viewpoints and interests of many different people are synthesized into a whole. The designer has to think of how the product will

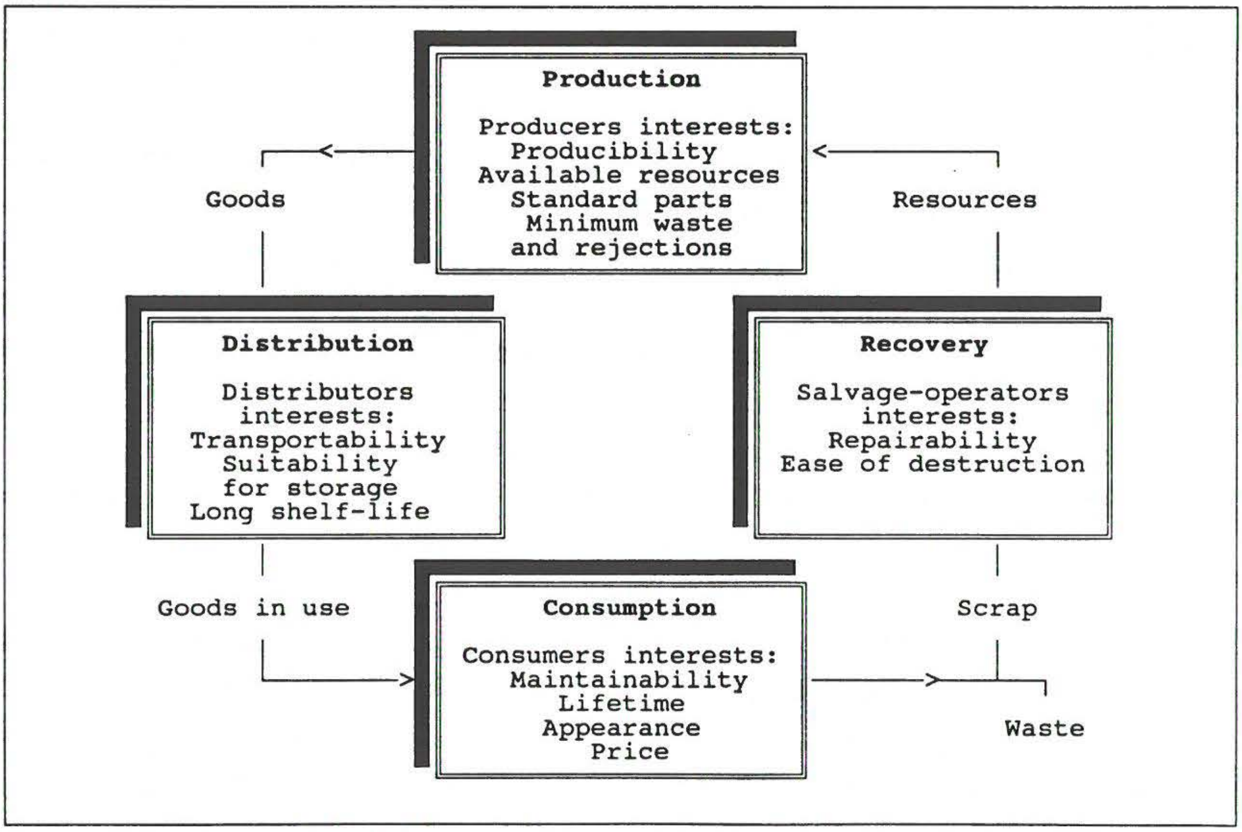


Figure 7.3 The production consumption cycle [freely after Asimow.62]

function in the production consumption cycle (see Figure 7.3) and the design is therefore a "compromise" between the interests from the consumer, the producer, the distributor and the "salvage-operator" (repair, destruction).

Creativity in design is a talent for discovering combinations of principles, materials or components, which are especially suitable as solutions to the problem on hand. The individual elements need not be new, in fact development of new elements is more the object of research than design.

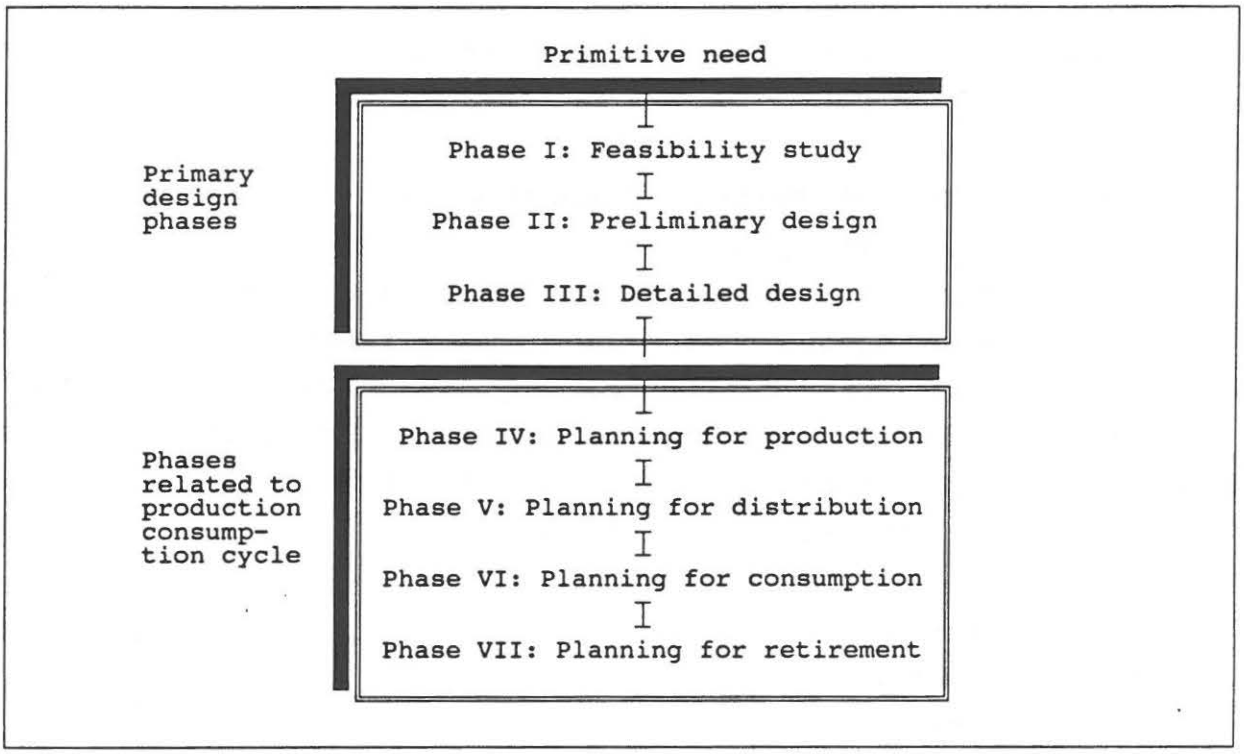


Figure 7.4 The phases of a complete design project [Asimow.62]

Even though each design project has its own individual story, the phases shown in Figure 7.4 are on the whole common to all projects. In the feasibility study it is investigated if a current or a potential need exists. Design parameters, constraints and criteria are determined, and potentially useful solutions are sorted out on the bases of physical realizability, economic worthwhileness, and financial feasibility. The preliminary design phase starts with the set of useful solutions which were developed in the feasibility study. Each solution is then explored and the best one chosen. The best solution is then the basis for phase III, the detailed design. Here the provisionally design is developed further, analyses and experimental test work are carried out. In phase IV production possibilities are explored and in phase V the viewpoint of distribution is taken into consideration (packaging, storage, marketing). Phase VI involves considerations of the consumers use of the product. It can be ease of maintenance, reliability, safety, convenience in use, aesthetics, etc. Finally in phase VII the "retirement" of the product is planned and this brings in environmental considerations, replacement of old with new products with a minimum of inconvenience and whether it is possible to reuse components or materials (e.g. like recycled glass bottles).

Model for technical systems - Domain theory

In the German speaking part of the world much work has been done in order to model how machine systems (also called technical systems) are being designed. One of the directions in this work is called the *WDK-school* [*Myrup*.88], [*Hubka*.82]. [Myrup.88] describes machines and technical systems in four domains or phases where the design is characterized in increasing concretization levels. The four domains are:

- Process domain
- Function domain
- Organ domain
- Building domain

The domains will shortly be described in the following using an water pump as an example. In the process domain the basic transformations performed by the machine are described in a blackbox form and alternative solution principles are suggested. The process domain description for a waterpump is an aggregate that can move a liquid from a lower to a higher position and solution principles include centrifugal force, Archimedes snail, one way vent, etc. The abstract functions that together form the product are described in the function domain as "abilities" in general form e.g. "sealing device" or "absorb forces from rotating device". In the organ domain are the organs in the machine described and modelled. An organ is a physical arrangement that serves a function e.g. a filter or a bearing. In the building domain all the parts in the machine are modelled in detail, i.e. dimensions, materials, tolerances, etc.

Engineering design as a social process

[*Bucciarelli*.88] claims that design can be regarded from several very different viewpoints, and improvement actions are accordingly different. On one hand design can be regarded as a process that can be controlled, and where flowcharts can be used to describe and model the design process (the three former described models represents this point of view). On the other hand do people engaged in management of design regard the organizational aspects as most important when considering improvements. Improvements in design work do according to this point of view include more effective ways of communication and more effective organizations for collaborative work. It is not enough to improve the work of the individual designer. Usually more designers have to work together, and the cooperation and communication are therefore also important aspects to take into consideration when improving design work. Bucciarelli emphasizes that the different viewpoints must be considered before improvement work is carried out.

Process selection in design

Process selection is made in almost all the design phases but the most comprehensive manufacturing considerations are in the phases following the conceptual design, i.e. in the more detailed design. The different phases have different requirements to design support systems, and when creating design support systems it is therefore important to consider which design phase the system primarily will support. The process selection in the early design phases is of a more general nature and the informations required are accordingly more general than what is needed later on. Considerations will here typically be type of geometry that can be produced, surface conditions, reliability, cost, etc. The later design phases need more detailed information about the processes and maybe about the specific production equipment.

7.2 Design support

Design of products is based on the designers knowledge and experience, and it is therefore common that he sticks to a few well-known materials and design principles in his work. There are probably several reasons for this and one very likely is that it is time consuming to explore new possibilities and that there is a chance that the effort is wasted. If the designer has faster and easier access to the information he needs, he might more often try new solutions.

In order to enhance design quality and flexibility it is necessary not only to look on the various design phases but also to analyze the sub-activities in design and determine what can be done differently. Sub-activities in design include (the list is not complete):

- Modelling/drawing
- Descriptions
- Design retrieval
- Search for related parts/components, BOM
- Information search for standard components
- Complex calculations
- Material selection
- Design for production (Process selection)
- Design for assembly

Modelling/drawing

A very important activity in design is the documentation of the design considerations. The designer usually has a lot of ideas in his head, but in order to explore the feasibility of the different solutions and to be able to discuss them with other people, he has to document the thoughts in some way. This is usually done through sketches and drawings on paper. Drawing on paper has the advantage that it is very fast to make, but on the other hand the paper is limited to two dimensions. It can be difficult to imagine the part in three dimensions on the basis of a two dimensional drawing. Another shortcoming of the paper representation is that it is difficult to reuse existing work (previous models) directly.

Improvements of the modelling/drawing sub-activity are usually based on CAD systems. CAD systems are today fairly common but a major shortcoming is the lack of a more complete product description. The CAD model includes geometry and dimensions but not other relevant information like tolerances and surface roughness. Work is being done with standardized *object oriented representations of CAD models* and this can be a possible solution to some of those problems [*Donohue*.87].

Descriptions

Documentation of the design thoughts does also include textual information in the form of descriptions, specifications, instructions, BOM and special considerations. This documentation is also usually written on paper. The main disadvantages with documentation on paper is that it is time consuming to write and make changes in it, and that it may be uncertain if it is the the latest revision of the document. Word processing and centralized data storage can usually help here.

Design retrieval

When designing a new product or a component for an existing product, it must be compatible with existing products and parts. Retrieval of existing similar designs can help keeping the number of variants low and it also introduces standardization to some degree. On the other hand the designer must be careful that he does not limit his creativity by reusing existing outworn designs. Retrieval is normally limited to search on a part ID number and the search for similar parts can be very time consuming. Classification/retrieval systems based on *group technology* and *database management systems* are then interesting opportunities.

Search and support for similar mechanisms

When the designer has established the main functions in a product that can satisfy his requirements, specific solutions can be created in different ways and by using different components. In some cases the number of different solutions is large and it can be time-consuming to explore them all. In many cases the designer is unaware of how many different solutions there are and this does of course limit the work of finding alternative solutions.

A design support system can here be used to give good advice concerning design of the different features and mechanisms. If for example the designer needs a slot on a shaft, the system could show the different possible slots and present recommendations for how to position the feature on the part and how to calculate minimum sizes.

Another example where the designer needs to consider many different mechanisms in order to find a solution are assembly methods, where two or more parts have to be mounted together. This can be done in many different ways for example with

- Screws
- Snap-lock mechanisms
- Rivets
- Glue
- Welding
- Soldering
- Close fit
- Nails

Having considered these possible assembly methods, the designer may reevaluate if it is necessary to divide the product into several parts (*DFA*). Another example of a problem with many solutions is mechanisms for changes in momentum or revolutions. One solution is to use a gear, but there are many different gears each with their advantages and limitations. Different gear types are listed below.

- Mechanical gears:
  - Toothed wheel gear
  - Angle toothed wheel gear
  - Friction gear
  - Worm reduction gear
  - Planet gear
  - V-belt transmission
  - Belt transmission
  - Chain
- Hydro-mechanical transmission
- Electrical transmission
- Hydraulic transmission
- Pneumatic transmission

There is not only a need for fast and easy access to information about similar mechanisms, but there is also a need for information about standard components like screws, bolts, sealing rings, etc. This must also be taken into consideration when building process selection systems since the mechanisms and components many times influence the material and process selection.

Complex calculations

The designer often has to re-design complex mechanisms (e.g. bearings and gears) where design rules and calculations already exist. These rules and calculations can be handled in computer systems which will give significant improvements concerning time, errors and number of alternatives that can be explored. One example of such a system is the *CADOPS* system for design of bearings [*Ferreirinha*.87]. Another example is the Asterix system for design of gear transmissions [*Christiansen*.84].

Material selection

Material selection involves exploration and selection of materials on the basis of part geometry, properties, characteristics and manufacturing possibilities. A simplified search for materials based on properties can be done using commercial material databases. They exist both as general databases covering a large spectrum of materials or as specific databases for a single material supplier. More advanced material selection that takes other criteria into consideration should be integrated in a process selection system since the criteria for the selection of materials are so closely related to the selection of processes.

Advanced material selection is made on the basis of specifications about strength, appearance of the product, corrosion resistance, wear resistance, electric and magnetic properties, ductility, and how the part shall be produced.

Design for producibility - process selection

Since the designer sets up limits for how to produce the part and thus for the process planning, it is important to supply the designer with up to date manufacturing information. Figure 7.5 illustrates how minor design changes could improve the producability. This communication can be carried out in many ways . One very good and simple, but unfortunately seldom used way, is that the designer talks directly with the process planners and the workers who later will produce the parts. The designer gets valuable information about details and features that are inexpedient, and maybe good ideas for modifications. This of course only can be done in companies where the design and production departments are placed on the same geographic spot.

The problem of lack of integration between design and production knowledge has lead to a desire for changes in the designers methodologies, so he more easily can take manufacturing possibilities into consideration. This field has many names and one of them is *design for producibility (DFP)*. DFP covers mainly the integration with production, but some people regard DFP as covering both production and assembly (design for assembly).

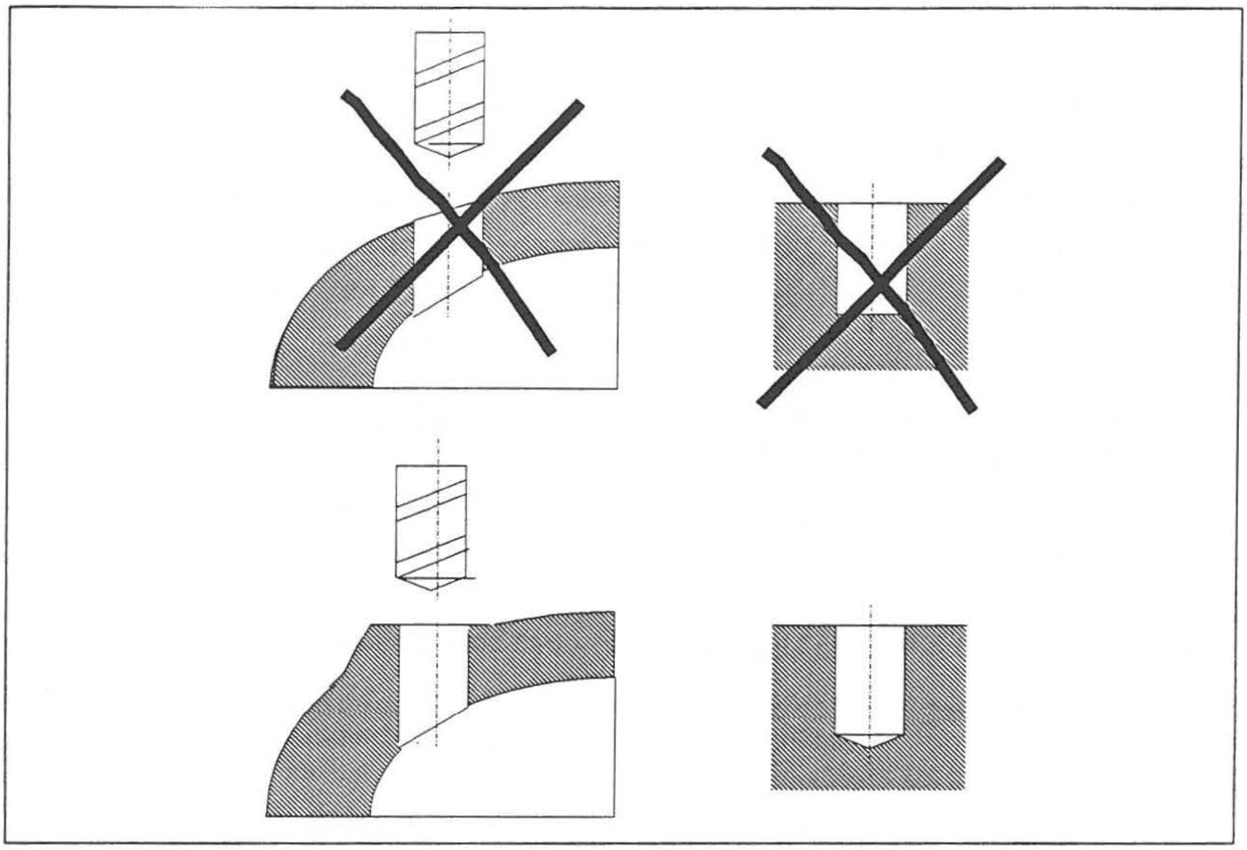


Figure 7.5 Design for manufacturing - good and bad examples.

[*Suh*.88] classifies the efforts for DFP in two basic approaches - an algorithmic approach and an axiomatic approach. In the *algorithmic approach* the work of the designer is described in step-by-step procedures on an ad-hoc basis. The *axiomatic approach* assumes that there exist some generalized fundamental principles for different design situations and that they can be applied to other similar design problems. Generally the algorithmic approaches can be derived from the more general axiomatic approaches, but the algorithmic approaches are more suited for practical use. The axiomatic approaches can be used as guidelines when building design support systems. In many cases though, well-defined algorithmic methods exist, but it can very difficult to determine what the more general principles are.

An example of what Suh calls the algorithmic approach is the Boothroyd methods described in the next paragraph "Design for Assembly". [Jepsen.78.b]'s advice about avoiding semi-products and in-between-processes can also be considered as an algorithmic approach.

Suh describes an axiomatic approach where he from two basic axioms (universally accepted principles) derives eight corollaries (propositions that are indirectly proved by proving another propositions) and fifteen theorems (propositions that can be deduced from the premises of a system) that can be used as general rules. The two axioms are described as

- Axiom 1 (The Independence Axiom): Maintain the independence of functional requirements.

- Axiom 2 (The Information Axiom): Minimize the information contents (associated with the task of fulfilling the functional requirements).

Examples of the corollaries that follow those axioms are:

- Decoupling of coupled designs: Separate parts (or aspects of a solution) if the functional requirements become dependent.

- Minimize the number of functional requirements.

- Integrate design features in a single physical part if the functional requirements can be independently satisfied.

- Use standardized parts if consistent with the functional requirements.

- Use symmetrical shapes and parts if consistent with the functional requirements.

- Specify the largest allowable tolerances.

- Seek an uncoupled design that requires less information than the coupled design.

DFM has as mentioned earlier many advantages but there are also a few problems. There are three basic problems in DFM :

- How can good design guidelines be collected and formulated.
- What suitable means should be selected or developed for

recording these guidelines.
- How can it be ensured that the guidelines are followed by the designer.

Design for assembly

Apart from considering the production alternatives for the product the designer also has to plan how the product can be assembled. The designer must have a good insight in how assembly is done and what type of problems that can occur in order to avoid them. One of the main design for assembly considerations concerns minimizing the number of parts in a product. More generally DFA concerns the principal component division, i.e. whether the part should be made in one piece, assembled from several sub components or be integrated with other components.

[*Boothroyd*.87] describes a general method for design for assembly (DFA). The DFA technique involves to steps: i) minimizing of the number of separate parts in an assembly; ii) improvements in the "assemblability" of the remaining parts. Boothroyd considers the first step as the most important and he suggests a procedure to part count reduction that involves asking three questions:

- During the operation of the product, does this part move bodily with respect to all other parts already assembled?
- For fundamental reasons, does the part have to be of a different material from all other parts already assembled?
- Does the part have to be separated from all other parts already assembled because otherwise assembly or disassembly of other separate parts could not be carried out?

Though simple Boothroyd claims that significant savings can be obtained using the method.

There are many general design rules for assembly but the majority is only true in special well-defined cases [*Bancroft*.88]. One example is the rule about uni-axis assembly. It says that all parts must be placed on the base subassembly from the same direction. In this way the product should be easier to assemble. In many cases this is true, but for "hard automation" (production lines) it is not so. The reason is that only one assembly can be done at the same time. But one of the main ideas in hard automation is that several operations can be performed at the same time in order to increase productivity. Another rule says that the number of assembly steps should be minimized, but sometimes it is more convenient to add several easy steps rather than a few complicated ones.

7.3 Design support techniques

Knowledge sources

The designer does apart from his own knowledge use information from various sources:

- General or specific books and other written material
- Manufacturing courses
- Manufacturing consultants
- Design for manufacture groups
- Organized and unorganized feed-back from production and planning (lunch room, coffee-machine).

Figure 7.6 is an example of a comparison of the capabilities for different processes (in this case obtainable surface roughness). A process is characterized by one or more of the following parameters:

- Geometry (shape) that can be produced
- Size limitation
- Tolerances
- Surface finish
- Processing time
- Power (energy) consumption
- Information consumption (flexibility, ease of use)
- Price

- Reliability

The geometrical capabilities of a process is difficult to describe and therefore to compare with part requirements. The geometry is therefore often considered in generalized categories using classification principles. Figure 7.6 shows a diagram that gives an overview of obtainable surface roughness for different manufacturing processes. Similar diagrams can be found in various standards (Danish Standard - DS 941, German standard - DIN 4766, Swedish standard - SMS 674, General Motors Standard [*Chang & Wysk*.85, p.93])

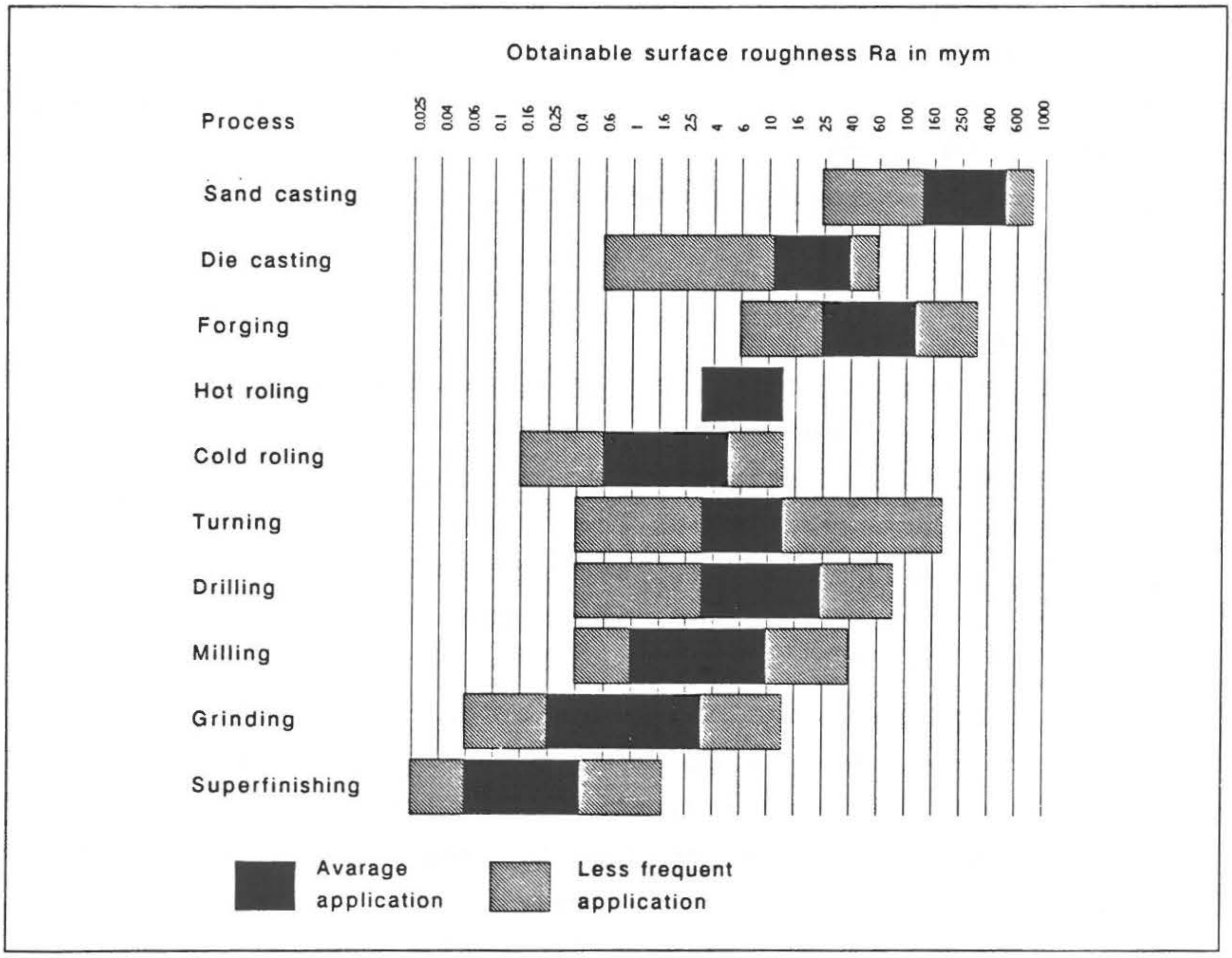


Figure 7.6 Obtainable surface roughness for different processes.

One way of achieving producible parts are through the use of design groups with people from both design and production departments. The group has regular meetings (maybe once a week) where design possibilities and their producibility are discussed.

Another way of obtaining products that are optimized for production is to let the designer create a number of alternative solutions and pass them over to another person or group of persons with manufacturing knowledge which will select one of the alternatives and make minor changes in the design.

The designer has different needs and requirements regarding sources of information depending on the situation. In some cases he will read a book to obtain the needed information and in other cases a computer based retrieval system will be the answer. What to use in which situation depends on the type of information and on how often the decisions are made. Computer based systems are best suited for situations with many often repeated decisions, complex situations (many facets) and where the decisions depend more on specific knowledge rather than on general knowledge.

Techniques

The design support techniques can be a methodology that people must learn to use in their work or it can be some kind of a computer tool. The methodologies include brainstorming and the use of design morphologies. The computer tools can be graphic programs like CAD that facilitate the modelling of the product, or it can be some kind of an information system.

With a computer tool the designer can get information about different processes that can produce the part and about part features that will be difficult or inexpedient to produce. This information helps the designer to create a better design seen from a manufacturing point of view, and the communication is carried out while the designer is creating his/hers design.

What should such systems look like ? One type of system is a consulting system, that based on user input can tell which processes that can be used. Input information could be part geometry, dimensions, raw material, desired tolerances, surface finish and other product requirements. System knowledge should concern the capabilities of the different processes and a way to indicate how well the processes can produce certain features, for example with estimates for cost and time consumption.

Another way to transfer manufacturing information to the designer, is through a support system integrated with a CAD system, so the designer is told immediately when he/she makes mistakes. This can be done by transferring some of the control of the CAD modelling from the designer to the computer system. Instead of giving the designer free hands when building the CAD model, the system leads him/her through the design while it performs various checks. Such a system does of course have its limitations and it is best suited for well-defined part families, i.e. families of parts with a number of geometric, production or other similarities (e.g. gear shafts or pump houses).

There are a number of different design modelling techniques on CAD systems where the wire-frame, surfaces and solids representations are the more well-known. [*Krouse*.82] and [*Bradford*.88] give a good overview of different modelling techniques, and [*Chang & Wysk*.85] give an overview over commercial CAD/CAM systems and the representations. Wire-frame systems are the simplest and the most widely used. They have a few drawbacks where the major ones are visible "hidden lines" and the lack of a consistent and complete product description. The latter drawback is shared by the two other modelling techniques and for this reason other modelling techniques and representations are being developed. An interesting work is the attempt to create a standardized representation of product information [*Donohue*.87]. The representation can be used for both data exchange between CAD systems and as a good foundation for integration with production and planning related activities.

Intelligent CAD - Feature based design

An important topic is the interfacing of intelligent support systems with CAD and other existing computer systems. Many products are today modelled on CAD systems and it would therefore be natural to extract product information directly from the CAD database and use it in expert consulting systems. Graphics are also a very powerful tool to illustrate process limitations and capabilities in design support systems, and CAD could be utilized naturally for this purpose.

Interpretation of a CAD model is a problem that not yet is solved satisfactory. The reason is that CAD systems use simple geometrical primitives like arcs and lines or solids to represent a model, and a data structure that only uses such primitives can not represent relevant relations between the primitives. In CAD systems today the designer translates his ideas about a part including the functional relations between the different primitives into the CAD system representation, where the relational part is lost. When a process planner or a computer program needs to extract information from the CAD model, there is problems interpreting the information right, since the relational information is either not there or only given implicitly.

This is one of the reasons why another CAD representation called *object oriented CAD* is being considered. A special type of object oriented CAD is the *feature based design* approach (*FBD*). In traditional CAD systems the model is represented by geometrical primitives like lines, circles, cubes and cylinders. FBD uses a *feature based representation* where the primitives are called features. A feature relates to product functions and/or to manufacturing informations. It is a higher order form and can for example be a manufacturing related unit like a chamfer, a hole, a slot or a thread. The designer most often thinks in terms of such units and the idea is that the designer saves this information together with the CAD model, so it is easier to interpret it later on for production and planning oriented purposes.

FBD has both advantages and limitations compared with traditional CAD systems. Among the advantages are that information is easily extracted and that it is possible to include checks and design rules. The limitations include the fact that only parts with the pre-defined features can be modelled, and FBD are therefore best suited for part families, i.e. families of parts with a number of geometric, production or other similarities (e.g. gear shafts or pump houses).

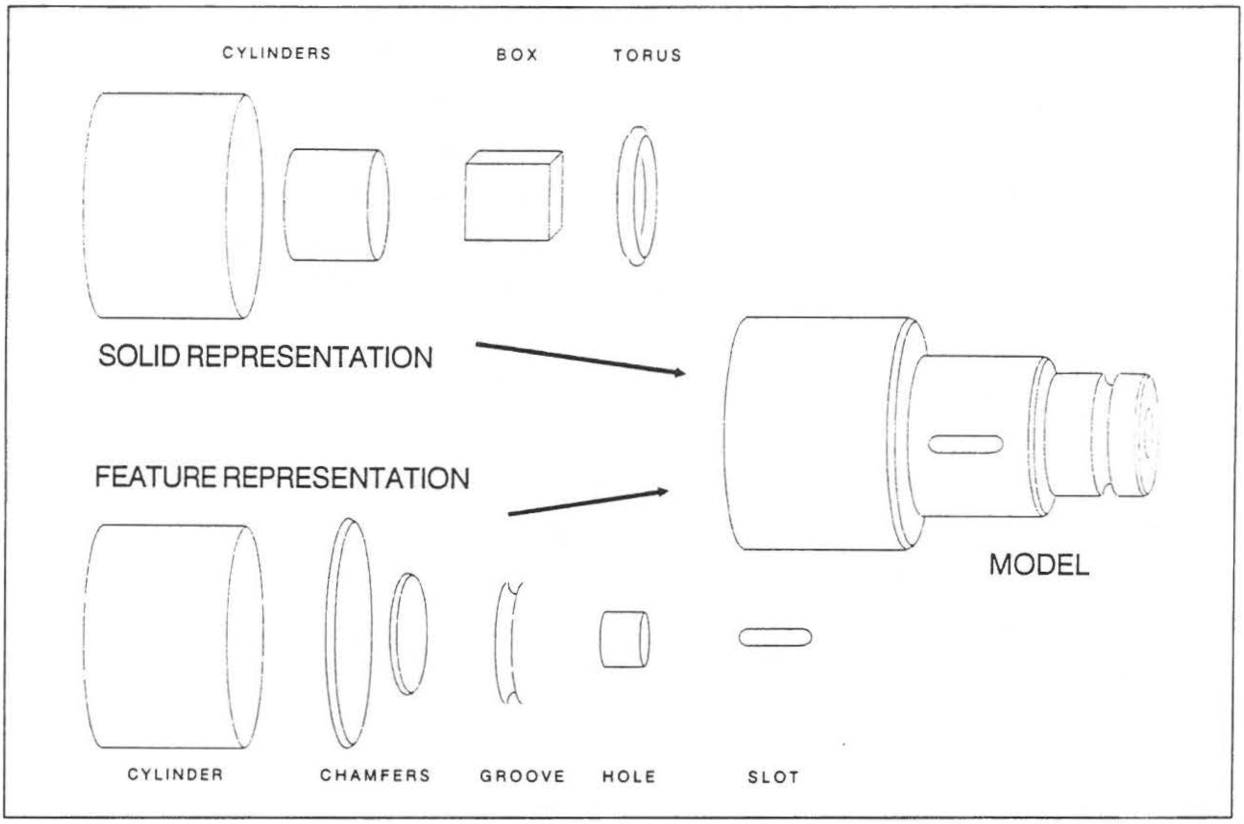


Figure 7.7 CAD representation techniques - Solids and features.

Another limitation in the feature based representation is that the features are process dependent. A given type of process can produce a given set of geometries that can be represented by a certain number of features. These features may be suitable for this specific process but it is not certain that they fit other processes. For example do the features for the turning process include slots, holes, cylinders, etc, and they all have their special characteristics (dimensions, tolerances, geometry). Another process like powder compaction would have different features or maybe the same features with other characteristics. If the features are used as representation units, one must therefore be aware of that this representation most likely only fit the process that they are made for.

Examples of work with feature based representation are the *PFDM* system, the structured design environment described by [*Lu*.88] and the *Cimplex system* [*Klein*.88].

The Part Family Design and Manufacturing system (PFDM) is developed partly by the author and it is based on *Autotrol S7000* CAD/CAM software and the *DCLASS* tree processor from Brigham Young University in Utah. Design procedures and checks are build into DCLASS decision trees, which also are used to control the graphic routines in S7000. The design is done by first defining a main shape and thereafter a number of geometric features. Currently the system works with two part families, a rotational and a prismatic part family. PFDM is also described in [*Lenau*.86.b].

Another approach to integrate design and manufacturing using CAD systems is to make specialized programs for product families. [*Peters*.88] describes a system called *DOPS* for design of pinion shafts.

7.4 Design of process selection systems

In the following it will be described how to specify and build systems for selection of manufacturing processes using the 3 level methodology described in chapter 4 for design of knowledge based systems.

Knowledge level

On the knowledge level demands regarding problem, knowledge and solutions are set up. Here it is realized that the problem of selecting a manufacturing process is more complex than it first seems to be. If it was possible to set up a final profile with all specifications/requirements for the desired process at one time, it would also be relatively simple task to find a process that matches those requirements. But the problem is iterative since the designer at first hand establishes a simple search profile that probably matches many processes. Looking into those different processes the designer finds out that he has more requirements and he therefore changes the profile. The system must therefore be able to handle an iterative type of consulting.

One way of handling this kind of iteration is to divide the system into two subsystems. The first subsystem handles a rough and fast search for possible solutions, with few questions and easy access by "trial and error". This procedure will limit the number of solutions to a few, after which the second subsystem takes over. Here more detailed analyses are carried out to identify each process that can be used. One of the advantages of dividing the problem into two steps is that only knowledge relevant to the actual process will be used in step 2.

Another system requirement could be that it must interface to graphics, both concerning input (e.g. from a CAD system) and output (illustrations of questions, curves, etc.). Requirements of how questions are presented to the user and how he may answer those are also important. Often it will be convenient that questions can be answered by selecting one or more options displayed on the screen. This helps the user since he is told what alternatives he can select from, and invalid answers are avoided.

Much knowledge is common to a group of processes and it would therefore be reasonable to store this knowledge at one place. For example all nickel-processes share the properties of the metal nickel, and those properties should therefore be stored at one place. This leads to a need for a hierarchical representation where data can be represented depending on its generality.

General views on hardware also belong to the knowledge level. It is important that the designer has easy and fast access to the expert system and to the knowledge. It is therefore essential with a central knowledge base and a fast workstation or terminal.

Function level

At this level proper representation methods and inference techniques (control structures) are designed or selected.

The need for hierarchical representation can be accomplished by both semantic nets and frames. Frames are the most advanced representation where it is possible to represent both detailed information and procedures for an object.

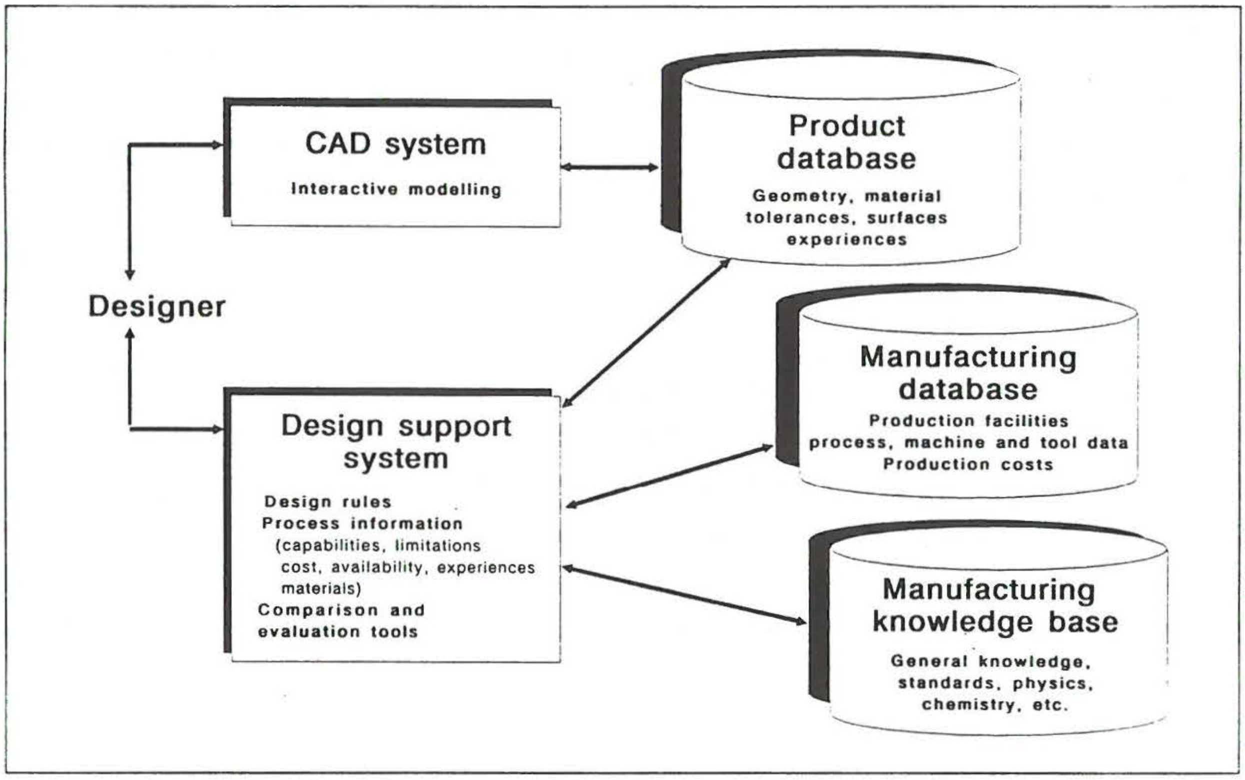


Figure 7.8 A model for a support system for process selection.

Procedural knowledge for process selection can be handled by production rules, object oriented programming or a combination of both. Production rules are particularly suitable to knowledge that can be expressed in the form IF <condition> THEN <action>. This can be general knowledge about many processes or it can be knowledge specific to a certain process. Object oriented programming can be used where there is a need for using a function that logically belongs to an object, e.g. a function that displays special extra information about a specific process. Another use is to place different functions with the same name at different objects, and when the function is called, it will behave differently depending on the object. For example, a display function that displays relevant information for a process can be made individually for each process, but they are all invoked by the call "show process information".

For subsystem one where the user should be able to input many different requirements, forward reasoning seems like a reasonable solution. After each input from the user the system checks if it has any rules where the condition part matches the input. This can trigger other rules, and eventually select a possible process. Forward reasoning gives the designer the possibility of trying different input to see what happens, and can in other words be used iteratively.

In subsystem two where the designer is interested in whether a certain process can be used, backward chaining may be more suitable. A goal, e.g. "check if process #x can be used" can be set up. Where the consequence part of a rule is matched, the condition part may rise questions, that are put to the user. Subsystem two asks the questions it needs to determine whether the process can be used or not. In some cases forward chaining can also be used for subsystem two. This can be done by grouping the rules according to the processes they belong to. A process is then examined by activating the relevant group of rules.

Program level

At this level relevant computer structures and languages are selected. The function level demands were frame representation for the declarative knowledge, rules and object-oriented programming for the procedural knowledge and an inference engine that can handle both forward and backward reasoning. It would be convenient if the development system has an open architecture, where it is possible to modify and include user defined low level programming directly and in this way make changes in the development system itself. Only the expert system development environments (*ESDE*) can fulfil all those demands.

References - Design Support Systems

[Asimow.62] Asimow Morris: Introduction to Design, Prentice Hall, N.Y., 1962, 135 pages.

[Bancroft.88] Bancroft Carol E.: Design for manufacturability - half speed ahead, Article in Manufacturing Engineering, September 1988.

[Boothroyd.87] Boothroyd G., Design for Assembly - The key to design for manufacture, International Journal of advanced Manufacturing Technology, Vol 2 (3), p.3-11.

[Bradford.88] Bradford David: Through the labyrinth of solids modeling, Mechanical Engineering, March 1988, p.30-36.

[Bucciarelli.88] Bucciarelli Louis L.: An ethnographic perspective on engineering design, MIT, Cambridge, USA, 40 pages.

[Chang & Wysk 85] Chang Tien-Chien, Wysk Richard A.: An introduction to automatic process planning. Prentice-Hall 1985.

[Christiansen.84] Christiansen Jens Juul, Christensen Steen Colliander: PFP systemer, Master thesis, Lab. of process and production engineering, Technical University of Denmark, 1984, Vol.5. Publ.no. AP.84.14, 75 pages, (in Danish).

[Donohue.87] Donohue J.F., Excellence is standard for CAD/CAM data exchange, Mini-Micro Systems, June 1987, p.51-52.

[Ferreirinha.87] Ferreirinha P., Konstruktionswissenschaft und rechnerunterstützte modellierung des konstruktionsprozesses, Int. Conf. on Engineering Design, ICED 87, Boston MA, 17-20 August 1987, p.455-462.

[Hubka.82] Hubka V.: Principles of Engineering Design, Butterworth Scientific, London, 1982, 113 pages.

[Jepsen.78.b] Jepsen Erik Lund: Valg af inductrielle fremstillingsmetoder, PhD thesis, Lab. for Almen Procesteknik, AMT, Technical University of Denmark, Publ.no. AP.78.04, 1978, 111 pages, (in Danish).

[Klein.88] Klein Arthur: A solid Groove - Feature-based programming of parts, Mechanical Engineering, March 1988, p.37-39.

[Krouse.82] Krouse J.K., What every engineer should know about Computer-aided Design and Computer-aided Manufacturing, Marcel Dekker Inc., New York, 1982, 144 pages.

[Lenau.86.b] Lenau T., Alting L.: XPLAN - An expert process planning system. The Second International Expert Systems Conference, London, UK; Sept/Oct. 1986. Publ.no. AP 86.26

[Lu.88] Lu S.C-Y., Subramanyam S.: A computer-based environment for simultaneous product and process design, Advances in manufacturing systems engineering, 1988, p. 35-46.

[Myrup.88] Myrup Andreasen M., Olesen Jesper, Rosenberg Ryan, Ferreirinha Pedro: CADOBS konstruktionslogik, Institute for Engineering Design, Technical University of Denmark, 1988., 46 pages, (in Danish).

[Peters.88] Peters J., Campenhout D.V., Manufacturing oriented and functional design, Annals of the CIRP, Vol. 37/1/1988, p.153-156.

[Stahl.77] Stahl Henrik, Tjalve Eskild: Konstruktionsteknisk Problemløsning, Lab. for Konstruktionsteknik, AMT, Technical University of Denmark, 1977, Publ.no. AP.78.01, 118 pages, (in Danish).

[Suh.88] Suh, Nam P.: Basic concepts in Design for Producibility, Keynote paper, Annals of CIRP vol 37/2/1988, p.559-567.

Bibliography - Design Support Systems

[Kinoglu.85] Kinoglu Faith, Riley Don, Donath M.: Expert System Model of the Design Process, Autofact Conference, Detroit, Mi, USA, November 4-7, 1985, p.7-1/15-27.

[Lim.87] Lim Beng Siong: An IKBS for integrating component design to tool engineering, Expert Systems, 1987, Vol.4, No.4, p.252-267.

[Meguid.87] Meguid S.A.: Integrated Computer-aided Design of Mechanical Systems, Elsevier Applied Science, London, UK, 1987, 193 pages.

[Ray.87] Ray Stéven R.: Process reasoning, IFIP WG5.7 Working Conference on Information Flow in Automated Manufacturing Systems, NBS, Gaithersburg, MD, August 6-7, 1987, 12 pages.

[Warnecke.88] Warnecke H.J., Bässler R., Design for assembly - Part of the design process, Annals of the CIRP, Vol. 37/1/1988, p.1-4.

[Zucherman.87] Zucherman, M.I. A knowledge base development for producibility analysis in mechanical design, CIRP Seminar on Computer Aided Process Planing, Pen. State Univ., USA, June 1987, p.301-307.

CHAPTER 8. ESOP - an expert design support system

An expert support system for selection of surface treatment processes (*ESOP*) has been developed in cooperation with major Danish companies (reported in [*Lenau*.88.c], [*Lenau*.88.d]). The work is partly sponsored by The National Agency of Industry and Trade and The Technical Research Council in Denmark. Surface treatment is a rapidly developing area with many new and improved processes, and is therefore an excellent area for investigating the possibilities of design support systems. The designer inputs desired characteristics for the design and the system will then based on its knowledge about surface treatment processes suggest suitable processes together with advantages and disadvantages for each process. The system is based on an advanced programming environment for expert system development, where knowledge is represented as frames and rules.

Background for the project

Surface treatment processes are frequently used to obtain special product properties. The properties can be enhanced wear resistance, corrosive, welding, soldering or electric properties, etc. It is characteristic that the properties can be obtained for more inexpensive materials when using surface treatment, and that properties can be obtained which otherwise were not possible. The surface treatment area is very complex and the number of surface treatment processes are large with many different applications with different results. It is therefore difficult for the designer to overview all the possible choices he has in a given situation, and it takes a long time to explore them. Furthermore the complexity is increasing since there within the last few years has been a strong development and research in new methods.

On this background there is an increasing need for development of a support system, where the designer in an easy and efficient way can be presented with the alternative processes that can be applied for a given product considering its special characteristics, geometry and material. Such a system will increase the possibilities of a faster and better optimization of the design, and at the same time attract the attention to the consequences regarding economy and environment.

A system for selection of relevant surface treatment processes can be developed using knowledge engineering techniques. These techniques give the opportunity of handling the logical relations there are between product demands on the one side and process capabilities on the other side in an efficient way.

8.1 System development methodology and ESOP

The three level methodology described in chapter 4 will be used to describe the system development work in ESOP. The methodology includes three phases referred to as a knowledge level, a function level and a program level. At the knowledge level the problem domain and the environment for the system are analyzed and principal solutions and objectives are set up. At the function level functions and procedures that can connect initial situations with goal situations are designed, and at the program level the functions are translated into structures that can be used on the computer.

The methodology was not actually used in the ESOP project, but the phases in the development work fit well with the methodology. The methodology is used to describe the development work for three reasons. First there is a good accordance between the model and the work carried out, second it serves as an example of how the methodology can be used, and third it will be of help for the future development work with the ESOP project and similar projects.

In the ESOP project the knowledge level includes analyses of the designers work and how selection of surface treatment processes is carried out. The considerations about the more detailed analyses of the sub-tasks in design, the use of knowledge engineering and how to select a proper software tool belong to the function level. At the program level the design and selection of specific program structures are done and prototypes built. The prototypes can be used to reevaluate the decisions at the two other levels.

8.2 The knowledge level

Analyses of how the designer works and how the decisions about surface treatments are made belong to the knowledge level. Through interviews of a number of designers the following steps in the design work were realized:

1. Setup of functional demands
2. Creation of design concepts
3. Selection of material and surface treatments
4. Design considerations for assembly
5. Quality and reliability considerations
6. Standardization and reduction of number of variants

The design work is initiated through a realization of a market need that the company is capable of fulfilling. It can be decided either to design a completely new product or to reuse and change an existing design. The design steps are the same in both cases but the extent of the work is different.

Functional demands to a part include all desired part characteristics and can for example be: Wear resistance, corrosion resistance, strength, attractive appearance, welding and soldering possibilities, etc. The demands to a part are usually a combination of those. It is common that not all demands are realized from the start and it is therefore important that it is easy to make changes in the input to the support system and explore the consequences (iteration). Functional demands can also be justified from a marketing point of view (e.g. fancy colour, smooth surface).

For each functional demand a number of parameters must be considered. For corrosion for example the parameters could be corrosive media (type, concentration, oxygen content, temperature, flow), part geometry and for how long time the part can resist corrosion. Other more general parameters include batch sizes, price, production experience and reliability.

Selection of material and surface treatments

On the basis of the interviews with designers at one of the involved companies a flow diagram for selection of materials and surface treatments were made (see Figure 8.1).

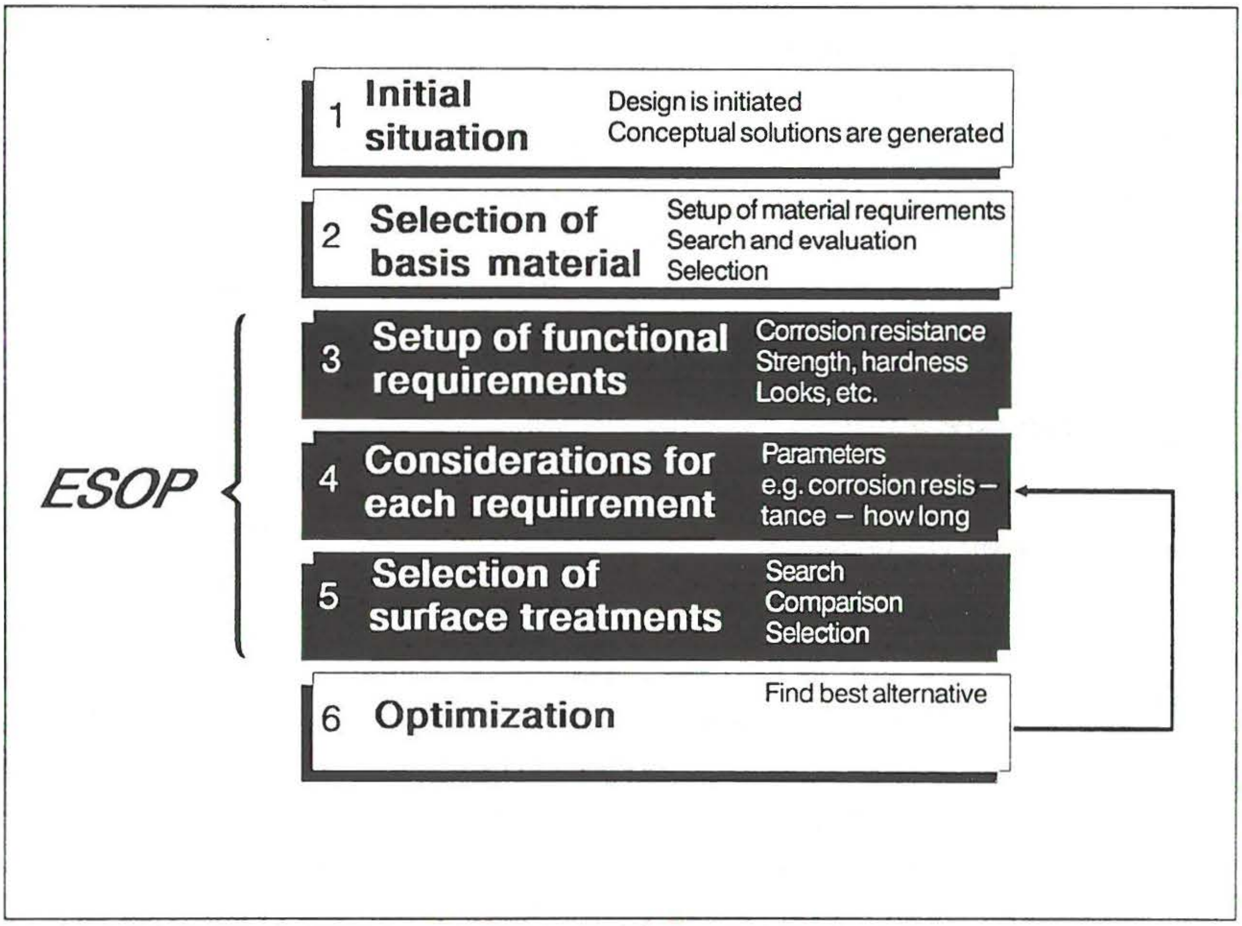


Figure 8.1 Phases in selection of materials and surface treatments.

Consideration about the initial situation includes gathering of information about the product and its scope. For revision of an existing product the previous design consideration must be reviewed.

Base material is selected by the designer on the basis of functional requirements, production and economical considerations. Most often there is a need to consider several alternative materials. In the present project material selection has deliberately been left out of consideration in spite of its close interference with selection of surface treatments. This is simply done in order to have a more practical and limited task, that could be carried out within the given time limit.

The functional requirements to the part include special qualities in the final product and examples are

- Wear resistance
- Corrosion resistance
- Strength
- Attractive look
- Welding and brazing properties
- etc.

Most often several product qualities are required. It can vary how many of the qualities the designer is aware of from the beginning, and in some cases does the enhancement of some product qualities lead to the need for other qualities, e.g. enhanced strength does often cause reduced ductility. Realizing product qualities and requirements and finding ways to fulfil them is therefore often an iterative process. For every requirement there are several factors or parameters that must be considered. Figure 8.2 gives some examples.

For each alternative material relevant processes can be found through a comparison of functional requirements and process characteristics. Apart from the parameters that directly relate to a functional requirement, a number of general parameters like batch sizes, experience, reliability, make/buy analysis, company policies, must be considered as well.

FUNCTIONAL REQUIREMENT	PARAMETERS
Corrosion resistance	Corrosive media: Type, concentration, oxygen content, temperature, flow. Geometry: Corners, cracks, holes, difference in material thickness. Durability: Surrounding environment, lifetime, fluctuating actions.
Strength	Geometry: Notches, surface condition. Forces. Vibrations and fluctuating actions.
Appearance	Color and brilliance. Surface finish: Rough, dull or smooth.

Figure 8.2 Examples of functional requirements and parameters.

Optimization is done by comparing alternative set of solutions with part geometries, materials, production tools, market situation and economy. Often the information is not sufficient to make a decision and prototypes can then be made and tested. If a satisfying result can not be obtained, product requirements and parameters must be reevaluated in step 4 in order to find a compromise.

Knowledge about surface treatment processes was in the ESOP project to a large extent accessible from a catalogue made by one of the participating companies. The catalogue gives a comprehensive description of a large number of surface treatment processes. The fact that a large amount of the needed data was already written down on paper was a tremendous help in the development work. Figure 8.3 gives an impression of the number of processes.

A number of different people and company activities are directly involved in the process selection, but others may have different needs for information about surface treatment processes. The designer is directly involved in process selection. Since the designer is not an expert on surface treatment he uses technical consultants in his search for the right process. Those consultants would also need a computer information system and they would probably use it in the same way as the designer.

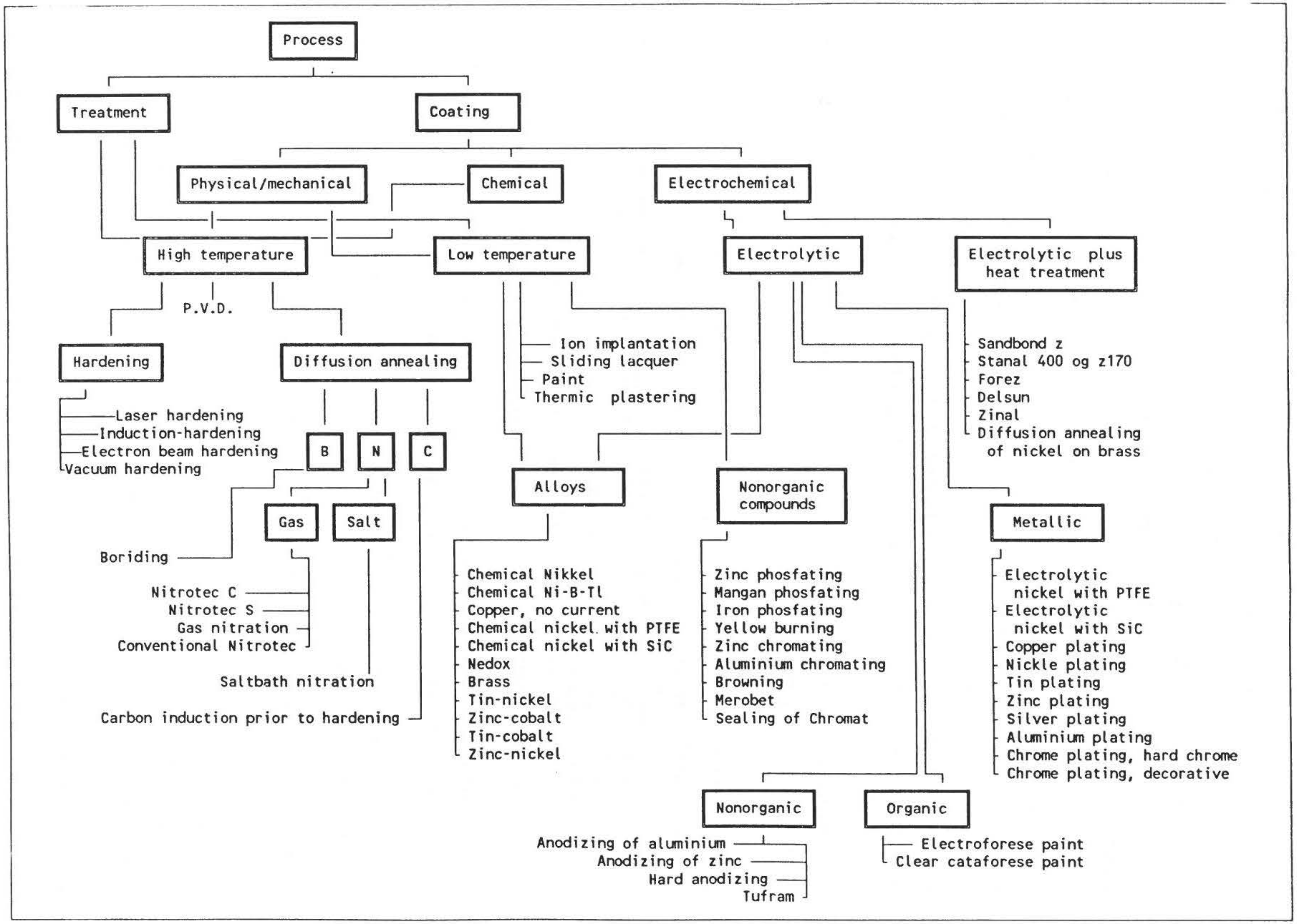


Figure 8.3 The process hierarchy in ESOP.

The people from quality control mainly need information about experiences from previous use of a surface process (the parts the process was used on, production problems, product errors or complaints). The production staff need access to the same information about the surface treatments that the designer has based his selection on. Furthermore they need access to previous experiences and knowledge about how to carry out the production. The different people have different requirement to a computer system and in the ESOP project it was decided to focus on the designers needs, and to place much of the general information about the surface treatment processes in a database that could be accessed by different people and different programs.

The analyses of the designers work led to the requirements to the ESOP program shown in Figure 8.4.

A. Search for surface treatment processes based on
- Basis material used in the part (preferably also combinations).
- Properties (Strength, hardness, appearance, corrosion resistance, wear resistance, max. temperature, welding and brazing properties, etc.).
- Previous use of a process.
- Part geometry.
- Random words or part of words in the knowledge base.

B. Special remarks concerning experiences, advantages, disadvantages, pitfalls.

C. Explanation of results (why was a process selected).

D. Show relations between different processes (which processes are prerequisites for others).

E. Availability of the process and production facilities.

F. Cost estimation.

Figure 8.4 General requirement to the ESOP program

Material and properties were selected as search criteria (A) because they were accessible from the process catalogue. Product and part information were not easy accessible and geometry was considered difficult to handle within the time limits. Availability of the production facilities (E) requires an registration that can not be obtained at the moment. It was chosen to handle cost estimation (F) as a price index attached to each process. In this way it is possible to compare relative production expenses.

Design support systems can be made using various computer techniques. Knowledge engineering was considered well suited for process selection due the iterative way of working and because much of the procedural knowledge about processes is due to changes.

8.3 The function level

The iterative process selection performed by the designer can be handled in a two step procedure. First a few general questions can limit the number of processes to examine further. The user can enter and change the requirements and immediately see the result in the form of how many processes that now match the profile he has set up. When the designer wants to examine one of these processes further, this can be done in a second step, where the expert system in more detail can ask questions to find out if the process can be used in the actual case.

The requirements to the ESOP program were described in the previous section, and they were met in the following way. Search (A) can be carried out on an arbitrary number of properties and materials. The search will find processes that match all the specified properties and materials. By specifying several materials it is possible to handle parts that are made from different materials, e.g. both steel, copper and brass. The special remarks (B) can either be handled as a text attached to each process or they can be formulated as rules that only are activated under special conditions. Explanation of results (C) can be obtained by recording for each process the properties that either advocates or talk against the use of the process. Relations between processes (D) can be handled as text comments attached to each process.

The ESOP system is structured as shown in Figure 8.5. The system has a knowledge base where the knowledge about the different processes is stored, a database where the pure data about processes can be saved, a working storage that contains conclusions and lines of reasoning and three user interfaces to ensure the best possible communication between the different users and the system. During the design of the ESOP system, structuring has been a major concern to ensure the dynamics that would allow structural changes to be made with a reasonable effort.

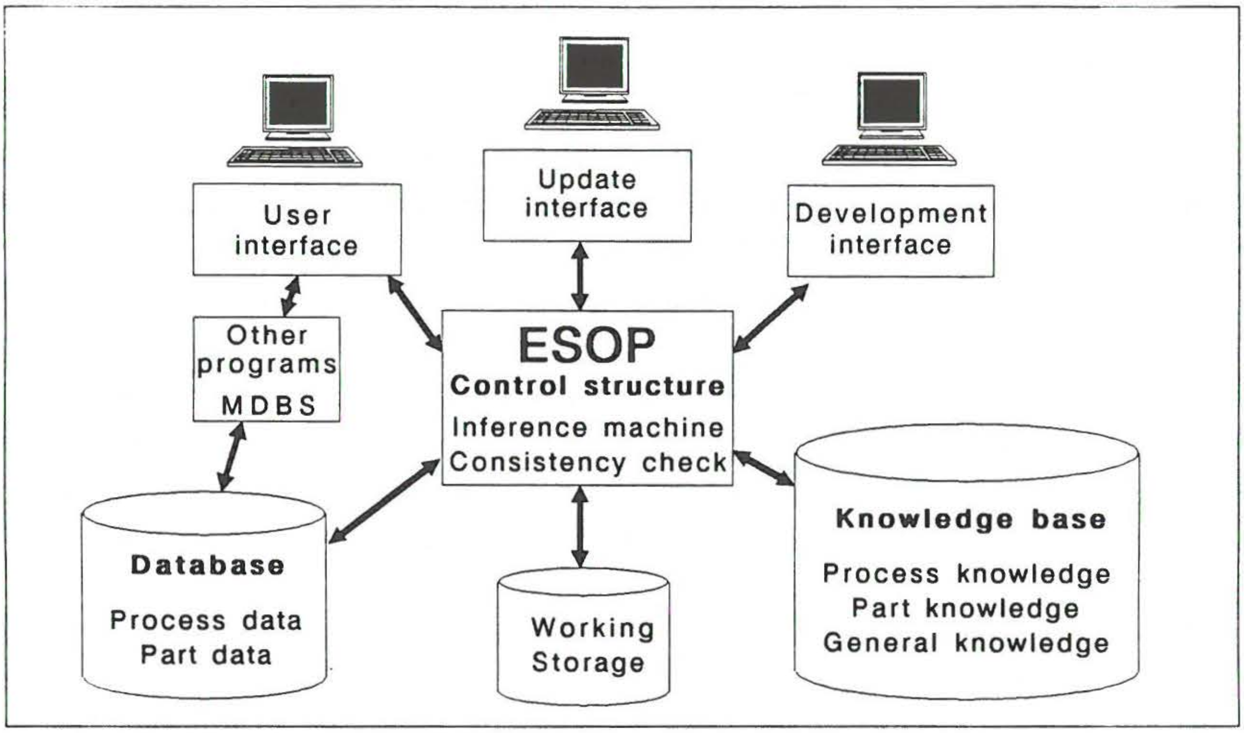


Figure 8.5 The structure of the ESOP system.

Knowledge base

Declarative knowledge about processes, materials and products can be represented in a hierarchy that allows representation of both general and specific knowledge. Procedural knowledge includes rules about advantages and disadvantages for a process, special relations that cannot be expressed as declarative knowledge and general knowledge about physical and chemical relations. Both rules and if-needed procedures can be used to express this procedural knowledge. Rules are suited for the general knowledge (e.g. knowledge about corrosion) and for conditional knowledge of a specific process or a group of processes (e.g. IF acid environment and sharp edges THEN poor adhesion is a risk.). The advantages of using rules are that they are relatively easy to read and that it is easy to add new or remove already existing rules. On the other hand larger amounts of rules can be overwhelming and hard to overview.

If needed procedures (demons) can be used when a function logically belongs to a specific part of the declarative knowledge. An example of an if-needed procedure is a bookkeeping function that counts and updates information about the results that are found during program execution.

Control Structure - Inference Machine

It has been a major concern that the *user interface* and the *control structure* allow both user driven and system driven control. In user driven control the user selects the requirements that will be used for process selection. When the system driven control is used, ESOP will ask a series of questions that can lead to a process selection. For this purpose there is included a facility in ESOP called "Interview". In both control structures the user can see how many processes that currently fulfil the requirements. It is possible to switch from the one control structure to the other at any time. The system control structure is especially suited for people that are unfamiliar with the system.

In order to rank the processes that to a larger or smaller degree fulfil the functional requirements, the system remembers for each process whether a requirement advocates or talk against the process. This makes it possible to find not only the best match but also to find the processes that almost fit. During the search for processes the user can specify the "almost fit" as the processes that fit all requirements with the exception of a specified number. The critical requirements can in this way be encircled and reevaluated.

The control mechanism used for rules is forward chaining. Every time the user has entered new requirements or changed the existing ones, the control structure will activate the rules and demons that the new information involves. Before any input is made by the user, ESOP assumes as a first hypothesis that all processes can be used. Every time the user changes the requirements, the hypothesis is modified and more processes are eliminated.

User interface

ESOP uses three screens, a welcome screen (an introduction screen), a working screen and a process screen (the screens are shown in Figure 8.8, 8.9 and 8.10). The welcome screen is used when ESOP starts up and contains information about version number and date. The working screen is used to set up a profile of the desired process, and it is divided into three parts: system menus, input area and result area. The system menus contain options for starting an interview (system control), for further information about the use of the system, and for quitting the program. The input area in the working screen contains the different types of requirements that can be set up. When a type of requirement is selected (e.g. basis material) a menu will pop up and show which specific requirements that currently can be selected (e.g. stainless steel or aluminium). Within the result area it is shown how many processes that currently satisfy the requirements set up. In the process screen the knowledge about a process is displayed together with information about the requirements it can satisfy. Extra information is attached to the most menus / questions and can be displayed on request. At the bottom of the screen a status area displays information about what the system is doing at the moment. This is to avoid that the user gets irritated when response times are long.

Selection of software tool

Based on the previous described analyses of the design work and consideration about how the system should work, the list of requirements to a software tool as shown in Figure 8.6 was set up.

Selection of a suitable software tool was made in two rounds, a general sorting on a few but important criteria (see appendix 1 & 2), and a more detailed examination of two selected software packages. The general sorting was necessary due to the many programming tools available on the market. The first round was based on experiences with the software tools DCLASS, KES, Personal Consultant, ESTA and Prolog, and on the literature [*Bastlund*.87], [*Schmeltz*.86]. The taxonomy described by [*Schmeltz*.86] (also described in this thesis in chapter 3) was used as a checklist when setting up the requirements.

- **Hardware:** Must run on PC and preferable on mainframe computers.
- **High level language** (short development time)
- Handling of **hierarchical** knowledge, i.e. both specific knowledge about a certain surface process and general knowledge about a group of processes.
- **Rulebased**
- **Grouping of rules**
- **Forward and backward chaining** of rules (control structure)
- As general as possible, i.e. covering as many relevant representation techniques and inference methods as possible.
- **Multiple input** (multiple choice menus), numerical and character input, preferable string input (whole sentences).
- **Dynamical knowledge base**, i.e. a knowledge base where it is easy to input new knowledge, without changing the structure of the existing knowledge.
- Handling of **multiple parallel solutions.**
- Possible to **regret/alter/change previous input.**
- Handling of **uncertain knowledge.**
- Access to user **defined programming** ("open" system). It must be possible to use user defined programming to deal with limitations in the software tool. Interface to **other programming languages** would also be a plus.
- Interface to a common used **relational database.**
- Easy understandable user interface and preferable **Danish support** (as a minimum support of the special Danish characters æ,ø and å).
- Access to add **explanations** and graphics to the questions.
- Explanation of **line of reasoning.**
- Low cost **run-time versions.**
- **Price** within budget

Figure 8.6 Requirements to software tools for ESOP.

The criteria that was used for the first sorting includes handling of parallel solutions (which also means that it must be possible to use multiple choice menus), both forward and backward chaining, interface to a relational database, the price on the development system and on run time versions, and the access to user defined programming. Especially two software systems could fulfil those requirements, namely GURU [*Bastlund*.87] and Goldworks [*Goldworks*.87], [*Levin*.88]. A limited and inexpensive demonstration version of GURU was bought and used for more detailed examination that among other things included building a small prototype of ESOP. Goldworks was tested together with a programmer from the Danish vendor. Both systems have many advantages but Goldworks was selected due to its more general nature and the more extensive access to user defined programming. The fact that Goldworks had an experienced vendor in Denmark that could offer support did also play a role.

Hardware

From the start of the project it was decided to use a PC based software development tool, primarily for economical reasons. It also played a role that the systems based on larger hardware was difficult to transfer to PCs (run-time versions). In the first part of the project IBM PC AT was used, but as the knowledge base grew bigger, it was necessary to change to more powerful equipment (386 processor based PC). For the final run-time versions a central storage of knowledge would be suitable to ensure up-to-date knowledge. This could be achieved on a mainframe computer with PC's connected as terminals. In this way the knowledge base could be distributed for local execution. Another possibility was to use graphic terminals and then run the support systems directly on the mainframe, but this solution has a number of drawbacks. Graphical terminals are not inexpensive compared with PCs and do not work as independent as workstations do.

Most of the larger software development tools require extended RAM memory on standard PCs. Goldworks require 5 - 10 MB of RAM memory. A graphical screen is required so drawings and curves can be presented. Normal PC screens are usable but it would be expedient if the screens were larger.

8.4 The program level

The development of ESOP used a general expert system development environment called Goldworks. ESOP is designed to work on personal computers with extended RAM and disk memory.

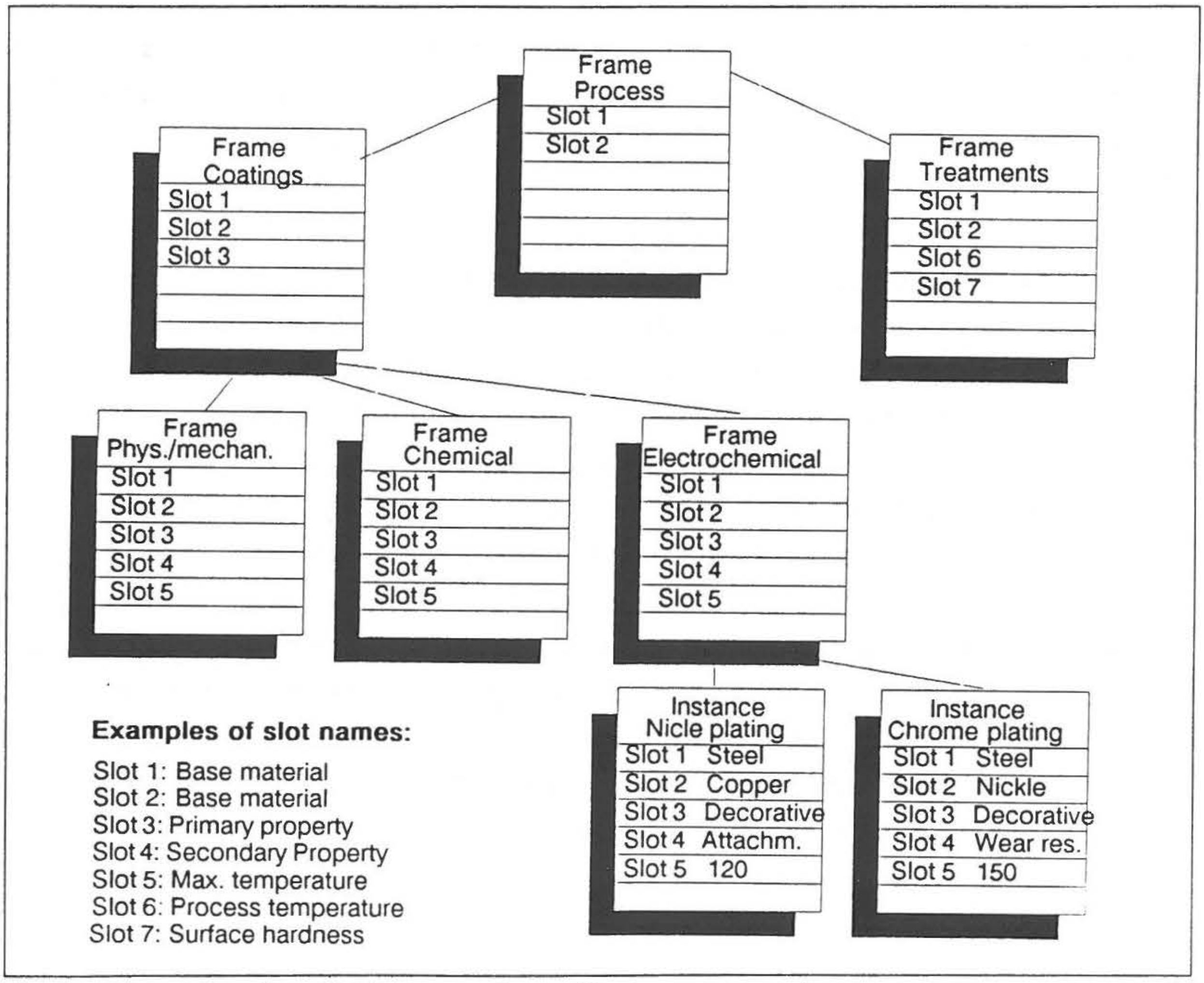


Figure 8.7 Example of the ESOP frame/instance structure.

Knowledge Base

The declarative knowledge is represented in a frame/instance hierarchy as shown in Figure 8.7. A frame can both inherit data slots and functions from other frames higher in the hierarchy. For example can a specific process inherit process specific knowledge from a process frame and material knowledge from a material frame. The frame structure offers a convenient and structured way of organizing the knowledge. The procedural knowledge is handled in rules using forward chaining and in if-needed functions called demons, i.e. functions that are activated when the value of a certain variable is changed. As much as possible of the procedural knowledge is placed in demons. Demons are more efficient concerning processing time and give a better overview when the number of rules are kept low. Procedural knowledge about surface processes is to a large extent placed in rules that are organized in groups with the Goldworks facility "Sponsors". Sponsors are used to group rules and to control at what time during execution the rules can be used.

Control Structure - Inference Machine

The control structure basically uses forward chaining, i.e. the control structure checks if the user has entered or changed the input and thereafter activates the relevant rules or functions. All input is placed in a special frame/instance where *demons* (when-modified functions) check if anything is changed. The process selection works in two steps as explained earlier. In step 1 where relevant processes are encircled, actions are carried out through the demons. When a part requirement is changed, e.g. another basis material is selected, a demon in the material slot in the input frame checks all processes to see if they match the new requirement. If they do, a register that keeps track of whether a process can be used or not is updated. The same demon also handles the situation when a part requirement is retracted.

ESOP keeps detailed accounts of the status of each process, whether they match the part requirement or not. For the designer it is not only interesting to know which processes that can be used considering the part requirement. It is relevant to know which processes that come close and what the critical part requirement are. This is handled in a register that for each process keeps track of whether it matches the different part requirements.

```
Quit
================================================================

                         E S O P

               Consulting for selection of
               surface treatment processes

                   Click here to start

                          START

          Date: 1-2-1989          Version AP-13

Status area
```

Figure 8.8 The initial or welcome screen in ESOP.

```
Process data  Interview  Information  Quit
================================================================
* User demands *
Basis materials                Carbon steel
Primary applications           Attractive appearance
                               Corrosion protection
Min.allowed hardness           300
Max.layer thickness
Desired corrosion class        1
Max. application temp.         200
Color
Largest price index            600

* # of processes that possible can be used *
0  processes match all demands
3  processes match all demands but 1 demand
Status area
```

Figure 8.9 The working screen in the ESOP system.

In step 2 a selected process is examined more thoroughly. The detailed knowledge about each process is kept in rules and the rules are grouped in sets called *sponsors*. Step 2 is started by activating the sponsor for the process in question.

```
Return
========================================================================
* Description of a selected process *
Process name                           Zinc coating
Basis material                         Carbon steel
                                       Automate steel
                                       Cast iron
Primary application                    *Corrosion protection
Hardness,               minimum        90       maximum
Layer thickness,        minimum        5        maximum 40
Max. application temp.                 250
Color                                  Gray
Price index                            300

* Arguments in favour *                * Arguments against *
# of arg. in favour 6                  # of arg. against  1
Desired properties:                    Desired properties:
Basis material  Carbon steel           Prim.application  Attractive appearan
Prim.applicat.  Corrosion resistance
Min.Hardness    300
Max.applic.temp.200
                                     >                                     >
 Status area
```

Figure 8.10 The process screen in the ESOP system.

User interface

The *user interface* is made using the *Goldworks* "screen toolkit" that includes a number of procedures for window and menu creation.

Two of the main screens in ESOP is shown in Figure 8.9 and 8.10. In the working screen the user can enter his demands concerning basis material, primary applications, hardness, corrosion class, etc. Each time the user enters, removes or changes a demand, ESOP determines the number of processes that match the input and it also tells how many that comes close. The process screen shows detailed information about a selected process, and the arguments that talk in favour or against the process.

ESOP is designed to help the designer, but since much of the knowledge can be used by other company functions (manufacturing and quality control departments), an interface to a relational database (DBASE III) is included. Much of the declarative knowledge can be stored in this database and then accessed from other programs.

8.5 Evaluation and future developments

The ESOP system described in the previous sections has been developed to a prototype level. ESOP can be used by the designers to determine which surface treatments he can use, and it works as a tool that can trigger good ideas about new and more unknown processes to use. It gives an excellent basis for future development of design support systems. Experiences drawn from the work are a realization of how important it is to have a systematic approach to the analyses of the problem domain and of the work content, and to examine the environment that the user functions in.

The tests of the ESOP prototype showed that there is a need for integration with material selection and for a further development of the user interfaces. The development work has until now been limited to the process selection, but the strong dependency between material and process selection indicates that material considerations must be integrated in the system. The further development of user interfaces partly concerns the ease of use and partly the users need for asking more generally formulated questions.

The main purpose of using an advanced software development tool in the project was to save development time by virtue of the many facilities in the tool. The experience from the project has been that the time savings depends on the type of development work. Simple prototypes that only use the easy accessible facilities in the tool can be developed relatively fast. More advanced prototypes require the use of more facilities from the tool. In our case it was necessary to add user defined programming in the Lisp programming language. If the programmer is unfamiliar with the programming language, the education time must be added to the development time. This can very easily mean that there are no time savings for the first advanced prototype. When the programmer has become familiar with the facilities and the basic programming language, time savings could be achieved due to advanced facilities like debug, trace, advanced editor, etc.

References - ESOP

[Bastlund.87] Bastlund C., et al.: Programmeringsværktøjer til ekspertsystemer, Datalogisk Institut, Københavns Universitet, DIKU rapport nr. 87/2, Februar 1987, (in Danish).

[Goldworks.87] Goldworks (TM) Goldhill Computers Inc., 163 Harvard Street, Cambridge, MA 02139, USA.

[Lenau.88.c] Lenau T., L.Alting, Artificial Intelligence for Process Selection, CIRP Int. Sem. on Manuf.Technology, Singapore, 18-19 August 1988. Published in Journal of Mechanical Working Technology, August 1988, Elsevier, p.33-51.

[Lenau.88.d] Lenau T., ESOP - Ekspertsystem til valg af overfladebehandlingsprocesser, Institute for Product Development, Technical University of Denmark, September 1988. Publ.nr. AP 88.22 (in Danish).

[Levin.88] Levin, Mogens og Klausen, Bjarne: Goldworks - et PC-baseret udviklingsmiljø EDB-nyt nr.1 1988. (in Danish).

[Schmeltz.86] Schmeltz Pedersen Gert (editor): Erfaringer med videnbaserede systemer, Dansk Datamatik Center, DDC 219/1986-10-31, Lyngby, Denmark, 1986 (in Danish), 153 pages.

CONCLUSION

The overall objective of this research work has been to investigate how advanced computer techniques (especially knowledge engineering) can be used to facilitate the information flow between design, production and production preparation in order to improve the company competitiveness.

The goals were to get an overview of the different techniques and tools in knowledge engineering and to develop prototype systems for selected areas. Those areas were manufacturing process planning and design support systems for process selection. The areas were chosen being central integrating elements in the work of reaching computer integrated manufacturing, and they are considered well suited for knowledge engineering because of their knowledge intensive, ill-structured nature.

Knowledge engineering provides very powerful methods that can be utilized to improve and integrate various manufacturing activities. It is discussed how information, decision making and planning can be modelled in knowledge engineering. Various representation techniques and control methods are described.

Process planning includes selection of manufacturing processes, selection of machines, planning of operations, and calculation of machining parameters and time/cost consumption. Different methods for computerizing process planning are described and a prototype system for generative process planning (XPLAN) have been developed. XPLAN is developed using decision tree logic that makes the knowledge easy to overview and change.

It is investigated how the design process can be modelled and where design support systems are required. Design support systems can be utilized for many different purposes and an important area is process selection. Design support systems for process selection are found to be a powerful tool for integrating design and production.

A design support system for selection of surface treatment processes (ESOP) has been developed in cooperation with major Danish companies. The system is based on the expert system development environment Goldworks (TM), and uses a frame representation and a rule based control structure. ESOP helps the user to set up the specifications which are used for selecting a process. The system is interactive and explains advantages and disadvantages in the actual case.

During the research work the need for systematic description and development methods was realized, and three different methods are therefore investigated. The IDEF method contains powerful techniques for information and function description. The cognitive engineering framework concentrates on the decision modelling, and emphasizes the importance of considering organizational aspects as well as strategies and preferences of the user. The three level development methodology concerns practical guidelines for system development. All the techniques require training to be utilized and can be used very differently by different people. But they all give a common mean that can be used as documentation and as discussion basis in cooperative development work. Furthermore the formalized techniques ensure that more factors are taken into consideration during the development work.

ENGLISH SUMMARY

The purpose of this research work has been to investigate advanced computer techniques (especially knowledge engineering), and their application to manufacturing process planning and design support systems.

Chapter 1 gives an introduction to knowledge engineering, process planning and design support systems, and describes hypothesis and objectives for the research work.

In chapter 2 the different activities in manufacturing are described and it is enlightened how manufacturing can be improved through computer integration.

Chapter 3 concerns knowledge engineering and artificial intelligence (AI), and describes the basic elements in a knowledge based system. Various representation techniques for declarative and procedural knowledge are explained including frame representation, rule based systems and object oriented programming. Perspectives for new AI techniques like neural networks are discussed.

In chapter 4 methodologies and modelling techniques for system development are discussed. The IDEF0 and IDEF1 methods are described as function and information modelling techniques. The cognitive engineering framework can be used for decision modelling and take a broad range of organizational and user specific parameters into consideration. The three level methodology supplies guidelines for the development work.

Chapter 5 concerns selection of manufacturing processes and processes planning. It is discussed how systematic process selection is carried out and a practical example illustrates the need for design support systems. The different phases and elements in process planning are described together with a process planning example.

Chapter 6 shows how process planning can be automated and computerized and a prototype process planning system (XPLAN) that has been developed is described. XPLAN is a generative process planning system that can generate process plans for mainly rotational parts. XPLAN handles part specification, process selection, selection of machines and operation planning. Experiences from a company are presented.

Chapter 7 discusses how product design can be made more efficient through the use of design support systems. It is discussed how the design process can be modelled and where process selection fits into a design model. The concept of design for manufacturing is discussed. The different elements in design are analyzed and it is investigated how they can be improved. Knowledge sources and techniques for design support are described together with the possibilities for intelligent CAD. The three level methodology described in chapter 4 is used to specify how design support systems for process selection can be made.

In chapter 8 a design support system for selection of surface treatment processes (ESOP) that has been developed in cooperation with major Danish companies is introduced. The three level methodology is used to describe the development work and how the system works. ESOP can be used by the product designer to search for and get information about surface treatment processes that are relevant in the actual case.

The conclusion of this thesis work is that there is a large potential in process planning systems and design support systems, and that knowledge engineering offers powerful techniques that can facilitate the development of such systems. It is also concluded that there is a need for systematic methodologies for system development.

DANSK RESUME

Formålet med dette forskningsarbejde har været at undersøge avancerede computerteknikker (især knowledge engineering / videnbaserede systemer), og deres anvendelse til automatisk procesplanlægning og konstruktionsstøtte systemer.

Kapitel 1 giver en introduktion til knowledge engineering, procesplanlægning og konstruktionsstøtte systemer, og beskriver forskningsarbejdets hypotese og formål.

I kapitel 2 bliver de forskellige aktiviteter i en moderne fremstillingsvirksomhed beskrevet, og det bliver belyst hvordan computer integration kan forbedre samspillet mellem disse aktiviteter.

Kapitel 3 omhandler knowledge engineering og kunstig intelligens (artificial intelligence - AI), og beskriver elementerne i et videnbaseret system. Forskellige representationsteknikker for deklarativ og procedural viden bliver forklaret, her iblandt "frames", regel baserede systemer og objekt orienteret programering. Perspektiver for nye AI teknikker såsom neurale netværk bliver diskuteret.

Kapitel 4 vedrører metoder og modelleringsteknikker for system udvikling. IDEF0 og IDEF1 teknikkerne beskrives som funktions og informations modellerings teknikker. Det cognitive rammesystem til ingeniørmæssige opgaver er beregnet til beslutningsmodellering og tager hensyn til en bred vifte af organisatoriske og bruger relaterede parametre. En tre fase metode opstiller retningslinier for udviklingsarbejdet.

Kapitel 5 behandler valg af fremstillingsprocesser og procesplanlægning. Det bliver beskrevet, hvordan systematisk procesvalg bliver udført og ved hjælp af et praktisk eksempel belyses behovet for konstruktionsstøtte systemer. De forskellige faser og elementer i procesplanlægning bliver beskrevet sammen med et eksempel.

Kapitel 6 viser hvordan procesplanlægning kan automatiseres og computer understøttes, og et prototype procesplanlægningssystem (XPLAN), som er blevet udviklet, bliver beskrevet. XPLAN er et generativt procesplanlægningssystem, som kan generere procesplaner hovedsageligt for rotationssymetriske emner. XPLAN håndterer emnespecifikation, valg af processer og maskiner og operationsplanlægning. Erfaringer fra afprøvning i en virksomhed præsenteres.

Kapitel 7 diskuterer hvordan produkt konstruktion kan udføres mere effektivt gennem brug af konstruktionsstøtte systemer. Det belyses hvordan konstruktionsprocessen kan modelleres og hvor procesvalg hører hjemme i modellen af konstruktionsarbejdet. Konceptet "konstruktion for fremstilling" diskuteres. De forskellige elementer i konstruktionsarbejdet bliver analyseret, og det undersøges hvordan de kan understøttes og forbedres. Informationskilder og teknikker til konstruktionsstøtte beskrives sammen med mulighederne for intelligent CAD. Den tidligere beskrevede tre fase metode bruges til at beskrive, hvordan konstruktionsstøttesystemer til procesvalg kan laves.

I kapitel 8 beskrives et konstruktionsstøttesystem til valg af overfladebehandlingsprocesser (ESOP), som er blevet udviklet i samarbejde med større danske virksomheder. Tre-fase-metoden bruges til at beskrive udviklingsarbejdet og systemets virkemåde. ESOP kan anvendes af konstruktører til informationssøgning omkring de overfladebehandlingsprocesser, som er relevante for det pågældende produkt.

Det konkluderes, at der et stort potentiale i procesplanlægningssystemer og konstruktionsstøttesystemer, og at knowledge engineering indeholder slagkraftige teknikker, som kan muliggøre udviklingen af sådanne systemer. Det konkluderes også, at der ved udvikling af sådanne systemer er behov for systematiske metoder til systemudvikling.

Appendix 1. Description scheme for knowledge based systems.

The form shown below can be used for description of, and comparison between expert systems shells. The form was used in the ESOP project for the first sorting of expert system shells. A few examples are included (made in autumn 1987). The author takes no responsibility for the correctness of these examples.

Name of the shell:

Vendor:

Hardware:

Price:

Run-time license:

- -

System interface:

Basic programming language used for the shell, e.g. Lisp, C or Prolog. Can the user access this programming language from the shell, and can shell function be called from the programming language.

Representation techniques:

frames, rule based, object oriented, uncertainty.

Inference machine:

Forward/backward chaining, possibilities for user defined control of inference machine (e.g. through programming).

One/more solutions:

Can the system follow several lines of reasoning and therefore reach several conclusions ? How do system handle conflicts.

Can the system handle multi-valued variables, i.e. variables that can hold several values at the same time. In ESOP for example there is a need for a variable with the name "Functional requirements" that can hold several values, e.g. "wear resistance", "decorative look", etc.

The condition and the action elements of a rule contains one or more variables.

if <condition> then <action>
condition = variable A or variable B

Remarks:

Name of the shell: **KES** (Knowledge Engineering System)

Vendor: SA&E (Software Architecture and Engineering)

Hardware: IBM PC, IBM VMS, IBM TSO, Apollo, Sun, HP, VAX, m.m.

Price: Depending of hardware, 2.500-60.000 UK pounds, Educational and quantity discount available.

Run-time license:

- -

System interface:
KES is written in "C", it is possible to call "C" routines from within the program, KES can itself be called from a "C" program.

Representation techniques:
Rule-based system, likelihood factors, meta rules.

Inference machine:
Backward chaining production rule control strategy, some kind of forward chaining should be possible, hypothesize and test control strategy.

One/more solutions:

Multiple answers to questions allowed, several answers possible, likelihood factors.

Remarks:

Name of the shell : **Personal Consultant Series** (Easy and Plus)

Vendor : Computer Resources International A/S (CRI)
Vesterbrogade 1A, 1620 København V, tlf: 01 321166
(Vendor for Texas Instruments)

Hardware: IBM PC XT/AT and compatible, Texas Instruments explorer.

Price:

Demo-system	DKR	400,-
Personal Consultant Easy	DKR	7000,-
Personal Consultant Plus	DKR	25000,-

Run-time license: DKR 2300,- per module

- -

System interface: PCS is written in the LISP dialect PC-sceme, that also can be utilized for further programming. Interface to databases: dBase II, III og III Plus. DOS commands can be executed from PCS, e.g. execution of other .com og .exe files.

Inference machine: PCS is made for backward chaining), but a kind of forward chaining is also possible. Forward chaining is obtained by defining the parameters that has to answered by the user before the backward chaining is started. The inference machine is controlled by defining goals and sub goals.

One/more solutions: Several answers on menu questions possible, several solutions possible. Likelihood factors.

Remarks:

* Spelling errors are hard to debug. Could have been avoided with a consistency check, e.g. is this parameter used in any other rule.
* Difficult to overview rule hierarchies/networks.
* After editing it is easy to make mistakes when saving th new version, and save it under an existing name and in this way destroy the existing knowledge base.
* Not particular fast (probably because of LISP).
* Do always return to the top menu instead of the ones previously used.
* Editor uses "overwrite" as default ("insert" do not destroy existing text).

Appendix 1. Description of expert system shells.

Appendix 2. Taxonomy for KBS used on two ES.

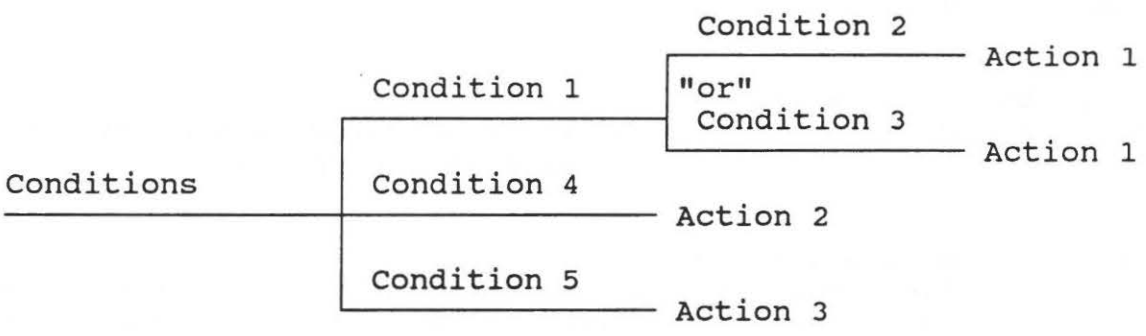
The taxonomy for knowledge based systems described in section 3.10 are here used to describe two expert system shells (autumn 1987). The author takes no responsibility for the correctness of these examples.

DCLASS

DCLASS is a decision tree logic system, and has been used for the development of XPLAN (described in section 6.10). DCLASS is developed at Brigham Young University, Provo, Utah, USA.

I.A. Rule based inference machine (control structure)

DCLASS can be used as a rule based system with forward chaining. A rule is expressed as shown in Figure A2.1, where an action is selected depending on the value of the condition. An action can activate another rule, and the rules are in this way placed in a hierarchy (a tree structure). A problem is solved by following one or more paths through the tree structure (depth first search). Rules can only be activated from the root of the tree, but it is possible to build several tree structures and in that way achieve several start situations. A tree structure can call another tree structure. DCLASS do not use likelihood factors.




```
   IF (condition 1 and condition 2) or
      (condition 1 and condition 3)
 THEN action 1

   IF condition 4 THEN action 3

   IF condition 5 then action 4
```

Figure A2.1 A rule expressed as a DCLASS tree structure.

II.A. Domain/knowledge base description

State description

A state is described by the branches in the tree structures. A branch can contain text, numbers (integer or real) and logic variables ("keys"). A sequence of states is saved as the path through the tree (called a bitstring), as the values of text variables, number variables, and logical variables (the keys).

State change
The state changes are described through the ramifications in the tree structure. One or more branches are selected by the user by answering the question at the rammification. Alternatively branches can be selected automatically through the logical variables (keys) attached to each branch. Automatic selection of branches can also be made from number variables.

Control of the inference machine
When designing the tree structure it is decided which rules should follow others. Through recursive tree structures or Fortran programming it is possible to change the control structure.

II.B. User interface
DCLASS uses a single window on the screen where questions and menus are shown. Menus are formed with a headline and numbered options. An answer is given by inputing the number of the relevant option. Neither functional keys nor mouse is used. It is possible to input commands instead of answers (the command HElp for example explains about the other commands). Danish characters can be used.

II.C. System interface
Programs written in Fortran can be executed from within DCLASS. DCLASS run on the most PC and mainframe computers.

III.A. Form of execution
The tree structures are made and changed through a special editor that include a number of format checks. It is also possible to use an external editor. The tree structure is compiled to internal representation before execution. It is possible compile "backward" from the internal representation for debugging purposes (Treedraw). Answers from a session can be saved.

III.B. Execution facilities
It is possible to use graphics and text as extra information for each question/menu.

IV.B. Editor
Dclass uses a dedicated editor.

IV.E. Implementation
The compiler check datatype and spelling errors.

ESTA is an expert system shell developed by the Danish company Prolog Development Center. The shell is based on PC/TURBO prolog.

I. Inference machine (control structure)
ESTA is rule based with rules expressed as

IF <condition> THEN <action>

Forward chaining is used. The rules are placed in a hierarchy with a root called the "start section". A section can hold one or more rules and the start section holds the rules that should be applied first. The action part of a rule denotes what rule to use next. A solution is reached by following a path through the hierarchy. ESTA have two limitations : 1) It is only possible to follow one path through the hierarchy. 2) The first possible solution is selected even though there are other (an maybe better) alternatives (depth first search). It is not possible to use likelihood factors.

II.A. Domain / knowledge base description

State description
The state is described by the "parameters". There are four types of parameters: Logical, Number (integer and real), Text and Category. Parameters of the category type can hold one of a number of predefined values.

State change
The state change is described through rules of the type "IF <condition> THEN <action>". The condition part is a logical expression, and the action part is one or more procedural expressions (Advice, Execute, Call, Stop, Exit). The rules define the relations between the parameters.

Control of the inference engine
In the rule (section) hierarchy it is defined which rules that follow others and the hierarchy starts at the root (the start section). It is possible to list and change parameter values at any time. An explanation facility can display the conditions that were fulfilled for the last fired rule ("why was advise given").

II.B. User interface
ESTA has a easy to use window and pop-up menu interface. It is possible to use a mouse, but it is actually easier to use the arrow buttons. Questions are mainly answered by selecting from a a list of displayed options. Graphics (e.g. PCPAINT) can be used as extra information. Danish characters can be used without problems.

II.C. System interface
From ESTA it is possible to call programs written in PC/Turbo prolog, C, Assembler and IBM pascal. For this the advanced ESTA PLUS version is needed. It is possible to activate the DOS

operating system from within ESTA and execute one or more DOS commands. Alternatively can DOS commands be executed directly from ESTA. ESTA only runs on personal computers (IBM PC, RC).

III.A. Execution form
The knowledge base (rules and parameters) are compiled to internal representation. The compilation is done by an compiler integrated in ESTA ("Check active knowledge base"). The knowledge base can be edited in an internal or an external editor. It is not possible to save values entered during a session.

III.B. Execution facilities.
Extra information can be attached to each parameter, i.e. question. All elements in the knowledge base (i.e. sections (=rules), parameters, answers) can be displayed at any time.

IV.B Editor
ESTA has a good internal editor. External editors can be used as well. Section (=rule) and parameter hierarchies can be displayed.

IV.E. Implementation
The compiler check datatype (e.g. if a numerical parameter is used as a textual parameter), definition of parameters (e.g. if all the parameters used in the rules are defined), and if there are rules that can not be reached.

Appendix 3. XPLAN manual.

X P L A N

Generative Process Planning System

Reference Manual

version 1.51

for IBM PC XT/AT,
compatible and
IBM PS2 series

February 1989 T.Lenau

CONTENTS:

1. INTRODUCTION

XPLAN is a generative process planning system which generates a process plan for a part on the basis of part specifications put in by the user. The system is generative which means that XPLAN creates an individual plan for an individual part. This is based on the logics in the program that can be changed easily without the need for a comprehensive programming experience. Since XPLAN contains all the logics necessary for the process planning, the user only needs to input component specifications - he does not need to be an expert on process planning.

The main objectives of XPLAN is to

* Generate process plans at expert level
* Enable automatic plan generation for alternative machines
* Allow machine availability and other priorities to be included in the process selection and the priorities to be externally updated.
* Include individual operation specifications, tooling selection and calculation of processing data and time.
* Provide a very user friendly system both concerning use and developments/changes.

XPLAN is developed on the basis of the DCLASS information handling system, which gives it the advanced possibilities of DCLASS. Information and decision rules are build into tree structures, which can be changed easily.

XPLAN is an interactive program where the user only has to answer a sequence of questions by selection from a menu of multiple choices or by entering a numeric value / a character constant. It is also possible to use the special DCLASS commands.

2. IMPLEMENTATION OF XPLAN

System configuration

The following system configuration is necessary for running XPLAN on IBM PC :

* IBM PC XT, AT, or equivalent (compatibles, IBM PS2 series)
  Min. 10 MB Winchester Disk
  1 5-1/4" Dual sided Floppy drive (360 k)
  512 k.bytes RAM
  (Parallel port for printer)
  Serial port for DCLASS Interface Device Communication box
* DCLASS version 4.2

* IBM DOS version 2.00 or higher

For mainline development a FORTRAN compiler and linker is necessary :

* MS FORTRAN version 3.31

Implementation procedure

The following procedure must be used to implement XPLAN on the computer :

* Create a new directory

```
CD\
MD XPLAN
```

* Copy The contents of the XPLAN diskettes into the new directory

```
CD\XPLAN
COPY A:*.*
```

XPLAN can now be executed by typing 'XPLAN'.

XPLAN is set up to use the C: hard disk and the COM1 communication port (for the DCLASS interface box). If another disk or another comunication port is used, the file FDCDFLTS.DAT has to be altered. An example of the contents of FDCDFLTS.DAT is shown below

```
C: IBM-XT+COM1        DISK-COMPUTER-PORT
01   OTHER FILES
C:                    FDCTBU.DCL
123456789012345678901
REM THIS IS THE DEFAULT CONDITION FOR REGULAR WINCHESTER SYSTEMS
```

XPLAN can work with either a small DCLASS system (676 sub-trees) or a large DCLASS system. The present version uses a small DCLASS system.

3. FILE ORGANIZATION

In order to save space on the Winchester disk a file structure as shown in figure A3.1 is recommended.

Place a PATH\CMD statement at the end of the AUTOEXEC.BAT file in the main directory.

When using the different DCLASS execution modules (DCPREP, DCUPDT, TREEDRAW, COMPRESS, TESTBOX) under XPLAN, they are called from the DCLASS directory through *.bat files. DCLASS object code files (*.obj) are called from the OBJECTCD directory (relevant when linking user developed mainlines).

```
                              MAIN DIRECTORY
                              --------------
                                     |
 ------------------------------------------------------------------------
 |              |                          |                        |
CMD          DCLASS                    OBJECTCD                   XPLAN
---          ------                    --------                   -----
Fortran      DCLASS files :            DCLASS files :           XPLAN files :
Editor         *.exe                     *.for                    *.exe
               *.src                     *.obj                    *.for
               fdc*.* (for DCLASS)       *.cmn                    *.obj
               other DCLASS files        compile and              *.cmn
                                         link procedures          *.src
                                                                  plan.*
                                                                  fdc*.* (for XPLAN)
                                                                  compile and
                                                                  link procedures
```

Figure A3.1. XPLAN File organization

4. SYSTEM DESCRIPTION

XPLAN works in four steps (Figure A3.2). The first step is the part specification where the user has to answer a number of questions about the part. It is questions concerning geometry, dimensions, material, batch size, etc. Second step is the process selection where the relevant manufacturing processes are selected. Third step is to setup requirements for each process and search for suitable machines. Fourth is the operation decision and time calculation for each machine. Finally, an output generation routine creates the process plan. Alternative machines found in the third step can then be examined and new process plans generated.

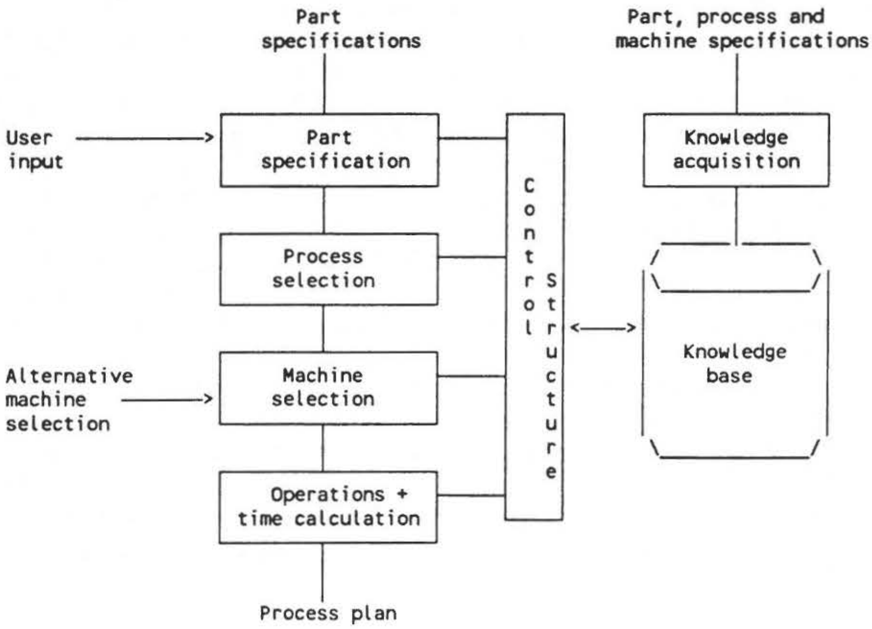


Figure A3.2. The XPLAN structure

Part specification

The part is specified through an interactive dialogue which is setup in the part specification tree. The tree contains the different questions about the part like geometry, dimensions, tolerances, etc. Each answer activates a flag called an output key, which is used to make automatic decisions (answer on questions) in the same or in other trees. The question can also be formed as an numeric / character input where the answer will be saved in a variable. The path through the tree is saved after the tree traversal as a bitstring including variables, output/input keys, product ID, etc. This information can then later be used to traverse the part specification tree or other trees automatically. Part specification trees are stored in the file XKOMP.SRC.

Process selection

Machines can be grouped into different groups depending on which kind of machining they can do. The term process means a group of machines with similar machining capabilities. Examples of processes are turning, milling and surface treatment.

Selecting proper manufacturing processes and their sequence is done by automatic traversal of the process selection tree. The output keys generated in the part specification tree are automatically compared with the input keys on the branches in the process selection tree. If there is a match the related branch is selected. A two character code is generated for each process (Figure A3.3). This code is used to select a process classification tree (for the machine selection). The process selection trees are stored in the file XPROC.SRC.

Machine selection

Machines are automatically selected for each of the chosen manufacturing processes. First a target profile is set up for each process and used to find the candidate machines, i.e. machines that can perform the actual machining. The profile consists of three sections, a process code, a bitstring and some variables. The process code identifies the process and is used to find the machines connected to the process when doing the machine search. The variables are used to drop machines not capable of producing the part. For example if a given batch size is required, all machines suitable for smaller batch sizes are not taken into consideration. The bitstring holds information about the tree traversal and tells which branches in the process tree that were selected. By comparing the bitstring from the target profile with the bitstring related to a machine, the candidate machines can be selected (Figure A3.4). Here is used a so-called DCLASS general search strategy, so only machines that fulfil the demands set up in the target profile can be used. i.e. only machines which profile holds the target bitstring as a subset will be chosen.

PROCESS SELECTION TREE :

```
                     (MATL)
                 Raw Material  %   Raw Material
              :-----------------:  "PF"
              :                 :---------------
              :          (ROT)                         Codes are marked with " "
              :  First turning %   First turning
              :-----------------:  "PC"          <------- This branch selects the machine
              :        (MULTAX) :---------------         selection tree "PC"
              :  Second
              :  Turning       %   Second
              :-----------------:  Turning "PC"
              :            (*)  :---------------
              :  Milling
              :  /((SLOT1 OR
              :  SLOT2) OR
    Processes :  SLOT3)/       %
  -----PA-----:-----------------:  Milling "PE"
              :         (HOLE)  :---------------
              :  Drilling      %
              :-----------------:  Drilling "PD"
              :            (*)  :---------------
              :  Control
              :  /(KONT OR
              :  MTEST)/       %
              :-----------------:  Control "PH"
              :                 :---------------
              :       (OVERFL)
              :  Surface           Surface
              :  Treatment     %   Treatment
              :-----------------:  "PG"
                                :---------------
```

MACHINE SELECTION TREE :

```
                                STK - Enter
                              V stock-size
                              :---------------
            stock size      %:  Variables for
          :-----------------:   refinement      %   10 STK OL1
          :                 :-----------------:F  OD0
          :                                   :-------------
          :
          :                                      (RAWROD)        The code "PC00" is used to find
          :                                   Bar feed           machines in the database
          :  Clamping                         possible
          :     "PC00"    %                   :-------------
          :-----------------:  Clamping data  :      (SEMIP)
  Turning %:                :-----------------:  Single part
  ----PC-----:                                :  loading
          :  Operations                       :  possible
          :---------------                    :-------------
          :
          :                    Maximum
          :                    part length L     OL1 - Enter
          :  Machine           (turning)     %   maximum
          :  specification  :-----------------:V length
          :  s             %:                 :-------------
          :-----------------:
                            :  Maximum           OD0 - Enter
                            :  diameter D    %   maximum
                            :-----------------:V diameter
                                              :-------------
```

Figure A3.3. Process and machine selection trees

SEARCH PROFILE :

```
                                    (RAWROD)
                                 Bar feed
               Clamping          possible
                   "PCOO"      :=============
             :=================:      (SEMIP)
             :                 :  Single part
             :                 :  loading
             :                 :  possible
             :                 :-------------
    Turning %:
  ====PC=====:
             :                    OL1 - Enter
             :  Machine           maximum
             :  specification   V length
             :  s              %:=============
             :=================:
                               :  OD0 - Enter
                               :  maximum
                               :V diameter
                               :=============
```

=== indicates that the branch has been selected

--- indicates that the branch has not been selected

MACHINE 1's PROFILE :

```
                                    (RAWROD)
                                 Bar feed
               Clamping          possible
                   "PCOO"      :=============
             :=================:      (SEMIP)
             :                 :  Single part
             :                 :  loading
             :                 :  possible
             :                 :=============
    Turning %:
  ====PC=====:
             :                    OL1 - Enter
             :  Machine           maximum
             :  specification   V length
             :  s              %:=============
             :=================:
                               :  OD0 - Enter
                               :  maximum
                               :V diameter
                               :=============
```

This machine is selected

MACHINE 2's PROFILE :

```
                                    (RAWROD)
                                 Bar feed
               Clamping          possible
                   "PCOO"      :-------------
             :=================:      (SEMIP)
             :                 :  Single part
             :                 :  loading
             :                 :  possible
             :                 :=============
    Turning %:
  ====PC=====:
             :                    OL1 - Enter
             :  Machine           maximum
             :  specification   V length
             :  s              %:=============
             :=================:
                               :  OD0 - Enter
                               :  maximum
                               :V diameter
                               :=============
```

This machine is not selected

Figure A3.4. Machine search

The candidate machines are then prioritized according to one of two strategies : a low cost strategy or a time consumption strategy. This is done by sorting the machines on the basis of the priority number attached to each machine ID. This number has a value between 0 and 9, where high numbers will be preferred for low numbers.

It is possible to list the machines in the database in three ways. In all cases enter option 5 in the XPLAN main menu. A general listing of all machines is made by selecting option 2, or by selecting option 5 (DCLASS manager). Using the DCLASS manager the main tree PB has to be selected and the DCLASS command LK (List Keywords) then entered. Listing of machines belonging to a certain process is made by selecting the according main tree for the process and entering the LK command. Machine selection trees are stored in the file XPROC.SRC.

Detailed planning (operation sequencing)

XPLANs first suggestion to a process plan is a plan consisting of the machines with the highest priority according to the chosen strategy. For each machine an operation sequence is generated. An operation is a basic transaction like rough turning of a diameter or drilling of a hole. For each different machine, all possible operations are contained in an operation tree, where it is also possible to place algorithms for selection of tooling and calculation of feeds, speeds and time consumption. For each process and for each operation, a four character code is generated. The process code has always a letter 'P' as first character and any other letters can be used for operation codes. The operation selection trees are stored in the file XOPER.SRC.

Time calculation

In the file XTIME.SRC the time calculation algorithms for two different lathes are placed . The algorithms are made to demonstrate how it is possible to include time calculation in XPLAN. The time calculation algorithms are formed as DCLASS tree structures and are called from an operation tree in XOPER.SRC. The time consumption in minutes is shown in the plan for each operation (in some cases only the total time for more than one operation is shown). For all operations on a single machine, the total time consumption is calculated and shown in hours. The total time consumption is adjusted for type of material and personal time.

Output generation

It is possible to print the plan on the screen (default), on a printer, or to save it in a text file on the disc.

The file XPLAN.INP contains all the lines used in the process plan. The codes generated during the detailed planning are used to select the right lines, and it is done in the order the codes are generated. Each code can generate from 1 to 4 lines. Each line is searched for brackets. The contents of the brackets are a DCLASS variable name. This is replaced with the value of the DCLASS variable during the output generation. The layout of the top and the bottom of the plan are also contained in the file XPLAN.INP, so if another plan layout is desired it is only necessary to make changes in this file. A number is automatically assigned to each process and each operation and printed where the @ character is placed in the XPLAN.INP file.

Plan layout compiling

If there has been made any changes in the file XPLAN.INP the plan layout compiling routine has to be executed before the changes are valid. This is done by selecting option 5.7 in XPLANs main menu. The compiled version of XPLAN.INP is stored in a file called XPLAN.XYZ.

Alternative machines

After generation of the first plan, it is possible to select alternatives among the candidate machines. This is done by selecting option 3 in the XPLAN main menu and entering the number of the process where other machines are desired (the process number is shown in the plan). All candidate machines for this process are then displayed on the screen, and the user is prompted for entering a new choice. The machines are listed in the priority order i.e. the machine with the highest priority (highest number) will be listed first. An Asterix (*) indicates the machine most recently selected. It is possible to select alternatives for more than one process at a time. Operation trees for the new machines are then traversed and a new plan generated.

DCLASS manager

By selecting option 5.5 in the XPLAN main menu, it is possible to use the DCLASS manager (same as DCLASS main menu option 10). The DCLASS manager has many functions, but in connection with XPLAN the following functions will be useful.

DF	Deletes a tree from the database.
DK	Deletes an id from the database.
EK	Gives the possibility to list an id.
LK	Lists all ID's under the selected tree, i.e. parts or machines

DF and DK can only be accessed by users with system level password.

Passwords

To ensure that only qualified personnel can make changes in the system a password facility is build in. There are three different passwords, and at delivery they have the values 'AAAA', 'BBBB' and 'CCCC'.

'AAAA' is the operator password and gives access to generate a process plan and some DCLASS manager functions (see the DCLASS manuals). The operator will only be allowed to use a subset of the XPLAN functions. For example he is not allowed to make changes in the database or in the plan layout.

'BBBB' is the technical manager password. Here it is possible to add machines to the database and make changes in the trees.

'CCCC' is the highest level password and it is used by the system manager. He is allowed also to use the delete functions (delete machines, parts or trees).

Passwords can be changed by the system manager with the DCLASS manager command MP.

Execution - running XPLAN

XPLAN is executed by entering the XPLAN directory (CD\XPLAN) and then type "XPLAN". Hereafter, XPLAN prompts for what it needs. The commands "99", "EX" or "E" can be used in most places to return to the XPLAN main menu.

5. TECHNICAL DEVELOPMENT AND UPDATE

Changes at the technical level include any changes made in the tree structures in XPLAN. This is relevant when changing part specification menus and when new processes or machines are included in XPLAN.

Part specification

Changes in the part specification are carried out by changing the part specification tree and running DCPREP. First letter in the name of new trees must be a K according to the tree name convension. Part specification trees are stored in the file XKOMP.SRC.

New manufacturing processes

Adding new processes can be necessary if new types of machines e.g. grinding machines are to be included in the process planning. To include new processes, the user has to create a new branch in the process selection trees PA and PB and to develop a process classification tree. The names of the process selection and process classification trees have to start with a P and have to be MAIN trees. Those trees also need a four letter code e.g "PC00". This code is used for the machine retrieval. Any letters can be used, but it is recommended to use the corresponding tree name e.g. "PC" followed by two zeroes "00". The new branch in PA also needs an input key (for automatic selection) and a two letter code identical to the name of the process classification tree.

Variables to be used at the refinement of the group under the machine search are included by using a user program call (son node .9 command : '10 varnam1 varnam2 etc.'). Only variables mentioned in this user program call will be used for the refinement (Figure A3.5). All process trees are stored in the XPROC.SRC file.

```
                               STK - Enter Batch
           "PC00"            V size
         Batch size        %:-----------------
       :-----------------:
       :
       :
       :                     Maximum
       :                     part length L      OL1 - Enter
       :  Machine            (turning)     %    maximum
       :  specification    :-----------------:V length
       :  s                %:                 :-------------
       :-----------------:
       :                 :   Maximum            OD0 - Enter
       :                 :   diameter D    %    maximum
       :                 :-----------------:V diameter
       :                                     :-------------
  Turning %:
  ----PC-----:
       :
       :F 10 STK OD0 <---------------------------- Through this call of a user program, the variables
       :-----------------                               STK and OD0 will be used to refine the group
                                                        of machines found
```

Figure A3.5. User program call in the machine selection tree

New machines

New machines will belong to one of the processes already defined in XPLAN. If this is not the case a new process classification tree has to be added (see above).

New machines are included in the database in the following way :

* edit the file XOPER.SRC
* develop an operation sub-tree for the new machine
* name the sub-tree using the earlier mentioned conventions.
* add code(s) to the tree (4 letter codes), one process code and one or more operation codes
* develop a time and parameter calculation tree (if needed) for the new machine (in the file XTIME.SRC)
* name the sub-tree using the earlier mentioned conventions.
* add input keys to the different branches
* compile the files using DCPREP
* edit the file XPLAN.INP
* add the code and corresponding machine and operation descriptions (1-4 lines)

* compile the file using XPREP (XPLAN main menu option 5.7)

* classify the new machine using XPLAN main option 5.6.

* Run DCUPDT

Develop an operation tree for the machine in the file XOPER.SRC. In the list with tree name conventions it is shown which tree names are recommended. For example, an operation tree for a milling machine will start with an 'F'. For each tree, there must be at least one code. This code is used to find lines for the process plan in the file PLAN.INP. The first code in the tree must be a process code (starting with a 'P'). The reason is, that the first letter in the code is checked during the plan generation. If it is a 'P' a process number is written to the line in the final plan, otherwise an operation number.

Input keys have to added to the different branches in the tree to allow automatic tree traversal (the input keys have to match output keys in the part specification tree). Having designed the tree structure, the DCPREP facility is used to compile it. The same procedure is used to develop time and parameter calculation trees (if needed).

Next step is to edit the XPLAN.INP file. For each code in the operation tree there has to be a corresponding code with 1-4 lines in the file. The format of the new lines has to match the lines already in the file. The file XPLAN.INP is compiled with XPREP (XPLAN main menu option 5.7).

Last step is to classify the machine using XPLAN main menu option 5.6. When entering the ID number, the first 2 characters must be the name of the corresponding operation tree (if the name of the operation tree was FD the first two letters in the id must also be FD). The priority criteria is used to sort the machines. A high number (e.g. 9) means high priority i.e. the machine is selected in front of a machine with a lower priority (e.g. 1). Having answered the different questions about the machine, select the 'STORE ID #' option. Finally run DCUPDT to update the database.

A machine can sometimes perform more than a single process. For example can some NC-lathes also do some milling with a milling spindle (4 axes lathe). This can also be handled by XPLAN. The machines are included in the database in the same way as described above, only is more than one process selected in the last step. It is only necessary with a single operation tree.

Plan layout

The layout of the plan is controlled in the file XPLAN.INP. Each line can be up to 79 characters long. First and second line in the file tells how many lines the top and the bottom of the plan takes (see figure 6). Maximum is 9 lines for the top and 5 lines for the bottom. Brackets indicate that the value of a DCLASS variable has to be inserted. The distance between the brackets tells how much space the value is allowed to fill (including the brackets). A number of predefined variables can be used in the plan. If any variables in the trees have the same names as the predefined variables, the value of the predefined will be used in the plan. The following predefined variables are available :

DATE	:	Current date e.g. 1986- 2-16 (10 characters)
PLANNER	:	Planner name as entered in the start of the XPLAN session (max. 20 characters)
PARTID	:	The part ID used to identify the part specification (max. 20 characters)
DESC1	:	Part description (character 1-20)
DESC2	:	Part description (character 21-40)
DESC3	:	Part description (character 41-60)
DESC4	:	Part description (character 61-80)

```
  8                                                                         Number of lines for top of plan
  2                                                                         Number of lines for top of plan
  ------------------------------------------------------------------------- 1 line in top of the plan
                              X P L A N                      (DATE      ) 2 line in top of the plan
                            PROCESS  PLAN                                   3 line in top of the plan
  Part ID:(PARTID              )             Planner:(PLANNER           ) 4 line in top of the plan
  ------------------------------------------------------------------------- 5 line in top of the plan
    P#  O#     Description                              Tool                6 line in top of the plan
  ------------------------------------------------------------------------- 7 line in top of the plan
                                                                            8 line in top of the plan
                                                                            1 line in bottom of the plan
  ------------------------------------------------------------------------- 2 line in bottom of the plan
  DH01       Drill hole d= (RD1  )                     Drill d=(RD1    )mm  a line for the plan
-> TD01      Turn outer diameter d=(OD1    )             Sidebit            another line for the plan
 |
 |
The code identifying the line for plan
```

Figure A3.6. Plan layout (File XPLAN.INP)

Debugging aids

In both DCLASS and XPLAN, a documentation level is used as a debugging aid. The documentation level has the default value '0', but can be changed in two ways. If a comma is entered after the password (e.g 'CCCC,') the system will ask about a new value for the documentation level. The 'MO' command (Modify Option) can also be used. 'MO' can be accessed either during the tree traversal or in the DCLASS manager. See the DCLASS manuals for further information on documentation levels.

Documentation level 12	:	Every action with variables is displayed on the screen.
Documentation level 20	:	Detailed output for process selection and machine search.
Documentation level 21	:	Detailed output for operation decision detailed planning).
Documentation level 22	:	Detailed output for sorting of machines.
Documentation level 23	:	Detailed output for output (plan) generation.

Guidelines for use of characters in tree name

Since XPLAN uses a lot of trees, it is convenient to have a convention for the use of characters in the tree names . The guidelines shown below are only meant as a guidance and can be changed if desired.

KA	Main tree for part specifications (main tree)	
K*	Part specification	
PA	Main process selection tree (main tree)	
PB	Main process classification tree (main tree)	
P*	Process classification trees (main trees)	
Q*	Sub trees for process classification trees	
T*	Operation tree for lathes	
D*	Operation tree for drilling machines	
L*	Operation tree for raw material handling	- One tree for each different machine (T* to C*)
F*	Operation tree for milling machines	
S*	Operation tree for surface treatment	
G*	Operation tree for grinding	
C*	Operation tree for control	
N*	Time and parm. calc. for lathes	
E*	Time and parm. calc. for drilling machines	
J*	Time and parm. calc. for storage	- One tree for each different machine (N* to U*)
H*	Time and parm. calc. for milling machines	
M*	Time and parm. calc. for surface treatment	
I*	Time and parm. calc. for control	
U*	Extra time and parm. calc. trees	

All process trees must be MAIN trees

The list from above in alphabetic order :

A*	Not used
B*	Not used
C*	Operation tree for control
D*	Operation tree for drilling machines
E*	Time and parm. calc. for drilling machines
F*	Operation tree for milling machines
G*	Operation tree for grinding
H*	Time and parm. calc. for milling machines
I*	Time and parm. calc. for control
J*	Time and parm. calc. for raw material handling
KA	Main tree for part specifications (main tree)
K*	Part specification
L*	Operation tree for storage
M*	Time and parm. calc. for surface treatment
N*	Time and parm. calc. for lathes
O*	Not used
PA	Main process tree (main tree)
PB	Main process classification tree (main tree)
P*	Process classification trees (main trees)
Q*	Sub trees for process classification trees
R*	Not used
S*	Operation tree for surface treatment
T*	Operation tree for lathes
U*	Extra time calculation trees
V*	Not used
X*	Not used
Y*	Not used
Z*	Not used

6. SYSTEM DEVELOPMENT

Mainline changes should normally be avoided for two reasons. First, it is one of the basic ideas with both XPLAN and DCLASS that development work can be handled at the tree level with debugging advantages like TREEDRAW and the documentation level features. Next, mainline changes will require programming experience in Fortran. But in some cases it is necessary to access the Fortran code, e.g. when linking XPLAN with other software programs.

To save space, only 2 byte integers are used. This is obtained either by declaring the variable as INTEGER*2 or by placing the meta-command '$STORAGE:2' in the top of the Fortran file. The meta-command '$STORAGE:2' should always be used since all DCLASS integers are implicit declared i.e. without declaring the size. If integer*4 variables are used for DCLASS communication (e.g. parameter transfer) strange and non explainable run time errors occurs.

To do mainline development (FORTRAN programming) following files are necessary :

Object code files :	XMAIN.OBJ	Main program
	XOVER.OBJ	Output generation
	XLIB.OBJ	General routines
	XLIB1.OBJ	DCLASS group selection
	XLIB2.OBJ	DCLASS group refinement
	XLIB3.OBJ	DCLASS bit comparison
	XPREP.OBJ	PLAN.INP pre processor
	DCM1.OBJ	Some screen output routines
	APLIB.OBJ	General routines
	ASMLIB.OBJ	General routines
	DCERRORS.OBJ	DCLASS error messages.
	All DCLASS objectcode files (placed in OBJECTCD directory)	
Common area files :	XHEAD.CMN	Head and bottom lines of plan
	XOP.CMN	Output generation (plan array)
	XOPT.CMN	Options (Clear screen)
	XPROC.CMN	Process and machine arrays
	XVAR.CMN	Variables for refinement at machine search
	DCLASS common areas (placed in XPLAN dir.)	
Command files :	C.BAT	Fortran compile
	JLNKX.BAT	XPLAN linker procedure
	LNKX.CMD	Files to be linked

LIST OF REFERENCES

[Alting.82] Alting Leo: Manufacturing Engineering Processes, Marcell Dekker, Inc. N. Y. 1982

[Alting.86.a] Alting Leo: Integration of Engineering Functions/Disciplines in CIM, Annals of the CIRP Vol. 35/1/1986, p.317-320.

[Alting.86.b] Alting L., Lenau T.: Integration of the Engineering Functions in the CIM Philosophy, ISATA, Flims, Switzerland, 6-10 okt. 1986.

[Asimow.62] Asimow Morris: Introduction to Design, Prentice Hall, N.Y., 1962, 135 pages.

[Bancroft.88] Bancroft Carol E.: Design for manufacturability - half speed ahead, Article in Manufacturing Engineering, September 1988.

[Bastlund.87] Bastlund C., et al.: Programmeringsværktøjer til ekspertsystemer, Datalogisk Institut, Københavns Universitet, DIKU rapport nr. 87/2, Februar 1987, (in Danish).

[Bilberg.89] Bilberg Arne: Simulation as a tool for developing FMS control programs, PhD thesis, Lab. of process and production engineering, Technical university of Denmark, Lyngby, Denmark, January 1989, 258 pages.

[Bolz.63] Bolz Roger W.: Production Processes, The Producibility Handbook, Inductrial Press Inc., New York, 1963.

[Boothroyd.87] Boothroyd G., Design for Assembly - The key to design for manufacture, International Journal of advanced Manufacturing Technology, Vol 2 (3), p.3-11.

[Bradford.88] Bradford David: Through the labyrinth of solids modeling, Mechanical Engineering, March 1988, p.30-36.

[Bringby.65] Bringby Einar: Tidsformler 4, Sveriges Verkstadsförening, 1965, 57 pages.

[Brunak.88] Brunak Søren, Lautrup Benny: Neurale netværk, Nysyn, Munksgaard, 1988, 160 pages, (in Danish).

[Bucciarelli.88] Bucciarelli Louis L.: An ethnographic perspective on engineering design, MIT, Cambridge, USA, 40 pages.

[Chang & Wysk 85] Chang Tien-Chien, Wysk Richard A.: An introduction to automatic process planning. Prentice-Hall 1985.

[Christiansen.84] Christiansen Jens Juul, Christensen Steen Colliander: PFP systemer, Master thesis, Lab. of process and production engineering, Technical University of Denmark, 1984, Vol.5. Publ.no. AP.84.14, 75 pages, (in Danish).

[Choi.82] Choi Byoung Kyu: CAD/CAM compatible, tool-oriented process planning for machining centers, PhD-thesis, Purdue Univ., Dec. 1982, 277 pages.

[Christensen.88] Christensen Steen C.: Computer Integrated Manufacturing Miniature Laboratory (CIM-ML), Overall structure, database and communication interface, Ph.D. thesis, Institute of Manufacturing Engineering, The Technical University of Denmark, Publ.no. AP.88.16, July 1988, 160 pages.

[Clocksin & Mellish.84] Clocksin W.F., Mellish C.S.: Programming in Prolog, Springer-Verlag, Berlin, 1984, 293 pages.

[Corocut.87] Corocut Handbook, Sanviken Coromant, Abildager 26-28, 2605 Brøndby, 1987.

[DeGarmo.69] Degarmo E.P.: Materials and processes in Manufacturing, 3 ed., London Macmillan, 1969, 949 pages, (also published in 1984).

[DCLASS 87] DCLASS - The expert logic processor, CAM Software inc. Provo, Utah,USA, 1987.

[Donohue.87] Donohue J.F., Excellence is standard for CAD/CAM data exchange, Mini-Micro Systems, June 1987, p.51-52.

[Evans & Sackett.85] Evans Steven & Seckett Peter J.: Financial justification of state-of-the-art investment : a study using CAPP, CAPP computer aided process planning, CASA/SME, Editor Joseph Tulkoff, 1985, p.47-50.

[Evans & Sackett.84] Evans S., Sackett P.J.: Realized performance in a novel approach to computer aided process planning, Int. Conf. on Computer aided Engineering, Univ. of Warwick, Coventry, UK, Dec.10-12 1984, p.48-52.

[Ewersheim.85] Eversheim W.: Survey of Computer Aided Process Planning Systems. CIRP annals 1985, vol 34/2/1985.

[Ferreirinha.87] Ferreirinha P., Konstruktionswissenschaft und rechnerunterstützte modellierung des konstruktionsprozesses, Int. Conf. on Engineering Design, ICED 87, Boston MA, 17-20 August 1987, p.455-462.

[Genesereth & Nilsson.87] Genesereth Michael R., Nilsson Nils J.: Logical foundations of artificial Intelligence, Morgan Kaufmann Publishers, Inc. 1987, 405 pages.

[Goldworks.87] Goldworks (TM) Goldhill Computers Inc., 163 Harvard Street, Cambridge, MA 02139, USA.

[Hayes-Roth.83] Hayes-Roth, F. et al.: Building Expert Systems, Addison-Westley Publishing Company Inc., 1983, 403 pages.

[Hubka.82] Hubka V.: Principles of Engineering Design, Butterworth Scientific, London, 1982, 113 pages.

[Ibsen.88] Ibsen Jess Julin, Karsholdt Peter, Vesterager Johan: Introduktion til IDEF0 & IDEF1, Driftteknisk Institut, DTH, 1988, Vol.1 (51 pages) + Vol.2 (75 pages).

[Intro.86] 8371 Introduktion til driftteknik, note 5, Metodestyring, Driftteknisk Institut, DTH, 1986, (in Danish).

[Jepsen.75] Jepsen Erik Lund: Principper for omkostningsberegninger og forkalkulationer, Afdelingen for Mekanisk Teknologi, Danmarks Tekniske Højskole, Publ.nr. AP.75.15, 1975, 80 pages, (in Danish).

[Jepsen.78.a] Jepsen Erik Lund: Udvikling af en hensigtsmæssig strategi for procesvalg, Lab. for Almen Procesteknik, AMT, Technical University of Denmark, Publ.no.78.10, 1978, 148 pages, (in Danish).

[Jepsen.78.b] Jepsen Erik Lund: Valg af industrielle fremstillingsmetoder, PhD thesis, Lab. for Almen Procesteknik, AMT, Technical University of Denmark, Publ.no. AP.78.04, 1978, 111 pages, (in Danish).

[Jepsen Jensen.84] Jensen Lars Jepsen: Computer aided tool and die design, PhD-thesis, Lab. for Almen Procesteknik, Danmarks Tekniske Højskole, Aug. 1984, Publ.no. AP.84.18, 146 pages.

[Jørgensen.78] Jørgensen Jørgen: Systematic tool and die design, Lab. for Almen Procesteknik, Danmarks Tekniske Højskole, Sept. 1978, Publ.no. AP.78.111, 139 pages.

[Jørgensen & Alting.81] Jørgensen Jørgen & Alting Leo: Manufacturing Process Taxonomy, Lab. of Process and Production Engineering, Technical University of Denmark, Publ. no. AP.81.07, 1981, 10 pages.

[Kim.87] Kim, S.H.: Managing the development of advanced knowledge-based systems, CIRP Seminar on Computer Aided Process Planing, Pensylvania State University, PA, USA, June 1987.

[Klein.88] Klein Arthur: A solid Groove - Feature-based programming of parts, Mechanical Engineering, March 1988, p.37-39.

[Krouse.82] Krouse J.K., What every engineer should know about Computer-aided Design and Computer-aided Manufacturing, Marcel Dekker Inc., New York, 1982, 144 pages.

[Kvist.88] Kvist - Konstruktion af videnbaserede systemer, Nyhedsbrev nr.1, 2, 3, Dansk Datamatik Center, 1988, (in Danish).

[Larsen.89] Larsen Niels Erik: Simulation - a concept for production scheduling, PhD-thesis, Lab. of process and production engineering, Technical university of Denmark, Lyngby, Denmark, January 1989, 398 pages.

[Lenau.85] Lenau T.: Datamatstøttet Procesplanlægning, Eksamensprojekt (Master thesis), Procesteknisk Institut, DTH, 1985, 99 pages, Publ.nr. PI.85.04.B/AP.85.03, (in Danish).

[Lenau.86.b] Lenau T., Alting L.: XPLAN - An expert process planning system. The Second International Expert Systems Conference, London, UK, Sept/Oct. 1986. Publ.no. AP 86.26

[Lenau.88.c] Lenau T., L.Alting, Artificial Intelligence for Process Selection, CIRP Int. Sem. on Manuf.Technology, Singapore, 18-19 August 1988. Published in Journal of Mechanical Working Technology, August 1988, Elsevier, p.33-51.

[Lenau.88.d] Lenau T., ESOP - Ekspertsystem til valg af overfladebehandlingsprocesser, Institute for Product Development, Technical University of Denmark, September 1988. Publ.nr. AP 88.22 (in Danish).

[Levin.88] Levin, Mogens og Klausen, Bjarne: Goldworks - et PC-baseret udviklingsmiljø EDB-nyt nr.1 1988. (in Danish).

[Locam.88] Locam CAPP, Brocure material from PAFEC, Kampsax data, Dagmarhus, 1553 København V.

[Lu.88] Lu S.C-Y., Subramanyam S.: A computer-based environment for simultaneous product and process design, Advances in manufacturing systems engineering, 1988, p. 35-46.

[Lyons.86] Lyons J.W.: The role of process planning in computer aided manufacturing, Knowledge engineering and computer modelling in CAD conference (CAD'86), London 2-5 Sept. 1986, p.379-384.

[Manufacturing Consortium.85] Manufacturing Consortium, Edited by Dell K. Allen & Leo Alting, Brigham Young University, Provo, Utah, USA, 1985, 17 volumes.

[Metals Handbook.69] Metals Handbook, American Society for Metals, Metals Park, Ohio, volumes 1-7, 1961, 1064, 1967, 1969 and later.

[Metcut.87] Brocure material for MetCAPP process planning software and Cutdata calculation software, Metcut Research Assocoates, 3980 Rosslyn Drive, Cincinnati, Ohio 45209.

[Myrup.88] Myrup Andreasen M., Olesen Jesper, Rosenberg Ryan, Ferreirinha Pedro: CADOBS konstruktionslogik, Institute for Engineering Design, Technical University of Denmark, 1988., 46 pages, (in Danish).

[Nilsson.82] Nilsson, Nils J.: Principles of Artificial Intelligence, Springer Verlag, NY, 1982, 427 pages.

[Negoita.87] Negoita Constantin V., Ralescu Dan: Simulation, Knowledge-based Computing, and Fuzzy Statistics, Van Nostrand Reinhold Company, New York, 1987, 158 pages.

[Pedersen.88] Pedersen Morten Als: Computer Integrated Manufacturing Miniature Laboratory (CIM-ML), Cell control, robot programming, communication, Ph.D. thesis, Institute of Manufacturing Engineering, The Technical University of Denmark, Publ.no. AP.88.17, July 1988, 116 pages.

[Peters.88] Peters J., Campenhout D.V., Manufacturing oriented and functional design, Annals of the CIRP, Vol. 37/1/1988, p.153-156.

[Rasmussen.88] Rasmussen Jens: A cognitive engineering approach to the modeling of decision making and its organization in: process control, emergency management, CAD/CAM, office systems, and library systems, Advances in Man-Machine systems research, JAI press inc., vol.4. p.165-243, 1988.

[Rasmussen.89] Rasmussen Jens: Analysis of Cooperative Work, Discussion paper for planning of field studies, Risø National Laboratory, Roskilde, Denmark, February 1989, 31 pages.

[Schmeltz.86] Schmeltz Pedersen Gert (editor): Erfaringer med videnbaserede systemer, Dansk Datamatik Center, DDC 219/1986-10-31, Lyngby, Denmark, 1986 (in Danish), 153 pages.

[Schwartz.85] Schwartz David C.: The Lisp Machine Architecture, Artificial Intelligence - Toward practical application, Elsevier Science publishers B.V. 1985, p.161-165.

[Stahl.77] Stahl Henrik, Tjalve Eskild: Konstruktionsteknisk Problemløsning, Lab. for Konstruktionsteknik, AMT, Technical University of Denmark, 1977, Publ.no. AP.78.01, 118 pages, (in Danish).

[Suh.88] Suh, Nam P.: Basic concepts in Design for Producibility, Keynote paper, Annals of CIRP vol 37/2/1988, p.559-567.

[Tengvald.84] Tengvald E.: The design of expert planning systems, Linköbing University, Linköbing, Sweden, 1984, 162 pages.

[Tong.87] Tong, Christopher: Toward an engineering science of knowledge-based design. Artificial Intelligence in Engineering, 1987, Vol.2, No.3.

[Value.63] Value Control Guide, Value Analysis Inc., N.Y. USA. 435 pages, 1963.

[Waterman.86] Waterman, Donald A.: A guide to expert systems, Addison-Westley Publishing Company Inc., 1986.

[Winston.84] Winston, Patrick Henry: Artificial Intelligence, Addison-westley Publishing Company, Reading, Massachusetts, USA, 1984, 524 pages.

[XPLAN.86] Lenau Torben: XPLAN generative process planning system, Reference Manual, Institute for Product development, The technical University of Denmark, December 1986, 39 pages.

[Zhang & Alting.89] Zhang Hongchao, Alting Leo: Computer aided process planning: the state-of-the-art survey, Int. Journal of Production Research, april 1989, Vol.27, no.4, 31 pages.

[Zucherman.87] Zucherman, M.I. A knowledge base development for producibility analysis in mechanical design, CIRP Seminar on Computer Aided Process Planing, Pensylvania State University, PA, USA, June 1987, p.301-307.

BIBLIOGRAPHY

[Ackoff.68] Ackoff Russel L.: Scientific Method - optimizing applied research decisions, John Wiley & Sons, Inc., New York, USA, 1968, 464 pages.

[Allen.86] Allen D.K., Computer aided process planning: Software Tools, Integrated and Intelligent Manufacturing, Anaheim, CA, USA, 1986.

[Alting & Zhang.88] Alting L., Zhang H.C., Lenau T.: XPLAN - an expert process planning system and its further development, Proc. of the 27 int. MATADOR conference, Manchester, UK, 20-21 april 1988. Publ.nr. AP 88.26

[Alty.84] Alty J.L., Coombs M.J.: Expert systems, concepts and examples, NCC Publications, Manchester, England, 1984, 194 pages.

[Chandrasekaran.83] Chandrasekaran B.: Towards a Taxonomy of problem solving types, The AI magazine, Winter/spring 1983, p.9-17.

[CAPP.87] Computer-aided Process Planning, 4 papers, CIM review - The Journal of Computer-Integrated manufacturing, Fall 1987, Vol.4, No.1.

[CIM.85] Int. Conf. on Manuf. Science, Technology and Systems of the Future, Ljubljana, Yoguslavia, 1985, Published in Robotics & Computer-Integrated Manufacturing, 1988, Vol 4, No. 1/2, 295 pages.

[CIRP.87] Computer Aided Process Planning, CIRP seminar at Pensylvania State University, PA, USA, June 1987.

[Gibbons.88] Gibbons Albert J.: A standards framework for the computer-integrated enterprise, CIM review, Vol 4, No. 3, p.10-17.

[Halevi.80] Halevi Gideon: The role of Computers in Manufacturing Processes, John Wiley & Sons, New York, USA, 1980, 495 pages.

[Henderson.84] Henderson P.: Functional programming, Application and implementation, Prentice-Hall International, 1984, 333 pages.

[Kinoglu.85] Kinoglu Faith, Riley Don, Donath M.: Expert System Model of the Design Process, Autofact Conference, Detroit, Mi, USA, November 4-7, 1985, p.7-1/15-27.

[Kochlar.79] Kochlar A.K., Development of computer-based production systems, Edward Arnold Ltd., UK, 1979, 250 pages.

[Lenau.86.a] Lenau T.: Knowledge Engineering indenfor det mekaniske område (Ekspert Systemer), Indlæg på FTU-temadag, DTH, Lyngby, August 1986, (in Danish).

[Lenau.86.b] Lenau T., Alting L.: XPLAN - An expert process planning system. The Second International Expert Systems Conference, London, UK, Sept/Oct. 1986. Publ.nr. AP 86.26

[Lenau.86.c] Lenau T., Alting L.: Tutorial on Computer Aided Process Planning The Autofact Conference, Detroit, Michigan, USA, Nov.1986. Publ.nr. AP 87.02.

[Lenau.87] Lenau T.: The role of Process Planning within the factory, Lab. for Almen Procesteknik, PI, DTH, May 1987, Publ.nr. AP 88.06.

[Lenau.88.a] Lenau T.: Datamatstøttet Procesplanlægning, ATV-SEMAPP Seminar on Computer aided Process Planning, The Technical University of Denmark, 7 April 1988.

[Lenau.88.b] Lenau T.: Computer Aided Process Planning, 50 min. Video for the European PACE educationprogram, June 1988.

[Lenau.88.c] Lenau T., Alting L., Artificial Intelligence for Process Selection, CIRP Int. Sem. on Manuf.Technology, Singapore, 18-19 August 1988. Published in Journal of Mechanical Working Technology, August 1988, Elsevier, p.33-51.

[Lim.87] Lim Beng Siong: An IKBS for integrating component design to tool engineering, Expert Systems, 1987, Vol.4, No.4, p.252-267.

[Major.87] Major F., Grottke W.: Knowledge engineering within integrated process planning systems, Robotics & Computer-Integrated Manufacturing, 1987, Vol.3, No.2, p.209-213.

[Marks.87] Marks Peter C.: Expert System-Based Generative Process Planning, Autofact Conference, Detroit, USA, November 1987.

[Meguid.87] Meguid S.A.: Integrated Computer-aided Design of Mechanical Systems, Elsevier Applied Science, London, UK, 1987, 193 pages.

[Olson.87] Olson Judith R., Rueter Henry H.: Extracting expertise from experts: Methods for knowledge acquisition, Expert Systems, August 1987, Vol.4, No.3, p.152-168.

[Ray.87] Ray Steven R.: Process reasoning, IFIP WG5.7 Working Conference on Information Flow in Automated Manufacturing Systems, NBS, Gaithersburg, MD, August 6-7, 1987, 12 pages.

[Solberg.85] Solberg J.J., Anderson D.C., Barash M.M., Paul R.P.: Factories of the future: defining the target, CIDMAC, Purdue University, In. USA, 66 pages.

[Tengvald.84] Tengvald E.: The design of expert planning systems, Linköbing University, Linköbing, Sweden, 1984, 162 pages.

[Vedel.88] Vedel M., Siegumfeldt J.: Procesplanlægning for prismatiske emner, Procesteknisk Institut, DTH, marts 1988, Publ.nr. AP.88.05 / PI 88.06, (in Danish).

[Viden.87] Videnbaserede systemer i Danmark, Teknologistyrelsen, Formidlingsrådet, September 1987, (in Danish).

[Warnecke.88] Warnecke H.J., Bässler R., Design for assembly - Part of the design process, Annals of the CIRP, Vol. 37/1/1988, p.1-4.

[Weill 85] Weill R.D.: Present tendencies in computer aided process planning. PROLAMAT 85, Paris, France, June 1985.

[Yeomans.87] Yeomans R.W., Choudry A., Hagen P.J.W.ten: Design rules for a CIM system, North Holland, Amstredam, 1987, 454 pages.

[Zucherman.87] Zucherman, M.I. A knowledge base development for producibility analysis in mechanical design, CIRP Seminar on Computer Aided Process Planing, Pen. State Univ., USA, June 1987, p.301-307.

List of Figures

INDEX