PART GROUPING FOR EFFICIENT PROCESS PLANNING

By

KHALIL AHMAD

A Doctoral Thesis

Submitted in partial fulfilment of the requirements

for the award of

Doctor of Philosophy

of the Loughborough University of Technology

Department of Manufacturing Engineering, LUT

April 1994

To my parents who taught me the merits of èducation. To my wife and children, without whose moral support this work would not have been completed.

ACKNOWLEDGEMENTS

I wish to express my thanks to my supervisor Dr. N. Z. Gindy for his direction in this research. I would like to thank the people who have in some way assisted me to reach my objective. Especially, my Director of Research, Mr. J. E. Middle for the invaluable support he gave me throughout this research and particularly in the preparation of this thesis. The Head of Manufacturing Engineering Department, Professor D. J. William for his invaluable moral support and partly financial support, he arranged for me towards the end of this research. Mr. Paul Humphrey for his helpful advice in the finalising of this thesis.

I would also like to thank my friends Syed Amjad Hussain Shah, Mr. Joseph Emmanuel Ogendo (who is not in this world any more), Mr. Shahraiz Iqbal and Mr. Faisel Hussan for their moral support, friendship, and inspiration throughout the period of my work.

I would like to thank Mr. Trevor Downham, Mr. Jaspal Singh and Mr. Richard Price for their technical advice on machining practice and also my colleagues in the department.

I also thank GEC Large Machines of Rugby for providing the data for this project.

I would especially like to thank my wife Rubina Yasmin, children Sarmad Khalil and Fahid Khalil, and the rest of my family for the patience and support they have shown through all the disruptions which producing this thesis has brought to their lives.

Finally, special thanks to the Ministry of Science and Technology, the Government of the Islamic Republic of Pakistan for providing financial support during this research.

Part Grouping for Efficient Process Planning

Synopsis

A framework to provide automated part grouping has been investigated in order to overcome the limitations found in existing part grouping techniques. The work is targeted at:

- Exploration of criteria for feature-based part grouping to make the process planning activity efficient.
- Determination of the optimal number of part families in the part grouping process.
- Development of an experimental hybrid process planning system (HYCAPP).
- Investigation of the effects of improved part grouping on manufacturing cell design.

The research work has explored the creation of a feature-based component data model and manufacturing system capability data model, and checked the limitations inherent in existing part grouping techniques i.e. part grouping: around mediods; based on part geometry; based on machining processes; and based on machines. The has led to the definition of a set of classifying attributes which can improve the demerits found in the above mentioned grouping methods. The MCU (Machine Capability Unit) concept which is the most important of the set of classifying attributes, has been introduced. The MCU provides better links between part features processing requirements and production system capabilities, thus making the parts processing solution independent of specific machine tools.

Based on the classifying attributes a software called CAFBG (computer-aided feature-based grouping) System has been written to accomplish part grouping automatically. A procedure has also been established which can identify the optimal number of part groups. A hybrid computer-aided process planning system, HYCAPP, is developed for a family of components. Furthermore the potential benefits to cell design of utilising composite components for part families generated by the proposed criteria have been demonstrated.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Manufacturing industry today operates in an increasingly turbulent environment. Shorter product life cycles, increase in the variety of product types, introduction of small batch sizes, rapid technological changes, increasing market competition, short delivery periods, high demand for quality products, and fluctuating demand all contribute to this turbulence. These types of constraints have led to the aim of fully automating the manufacturing system. In the previous decades several "Computer Aided" methods were introduced in order to automate the whole manufacturing function. Well known terminologies are CAD, CAM, CAE, and CAPP. However these methods have proved inadequate to solve the whole problem of manufacture. Automation in the manufacturing process is not optimising one production step but passing the existing information from one process to another. Computer Integrated Manufacturing (CIM) is the future trend of manufacturing industry. CIM is an integration of all aspects of the manufacturing function from the design stage to the packaging of finished products including business aspects. Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM) are two important components of CIM.

There is a need to look at CAD and CAM together so that comprehensive systems may be structured which make it possible to design parts efficiently, which can then be manufactured easily and economically. Design and manufacturing can no longer be separate islands, they must be unified. Group Technology (GT) is considered as the means of integrating design and manufacturing. GT is a way of identifying and bringing together related parts so that design and manufacturing can take advantage of their similarities. Parts with design and manufacturing similarities are grouped into families because GT is the realisation that many problems are similar, and by grouping similar problems, a single solution can be found. GT can be considered as the underlying organisational principle of CIM [Ham *et al.* 1985]. Over the past few years the GT concept has broadened to encompass other manufacturing activities such as product design and Process Planning using computers and has become an integral part of CIM. Benefits from GT are realised mainly due to more efficient use of set-ups, reduced material handling, and the use of manufacturing resources that are common to a group of components.

The concept of using component features for design and manufacturing applications is of considerable significance and is a recent approach to link CAD with CAM. The feature approach restricts the designer/process planner to working with a set of features which have importance for both design and manufacturing. The designer is currently using a set of features (e.g. holes, slots) from which a manufacturing method can be identified. In the past he used a model consisting of graphics primitives (e.g. lines, vertices, arcs) for the description of part geometry that was of no use in the reasoning process. In other words features are considered to be a higher level of abstraction containing both the geometrical information and the technological information contrary to the lower level of abstraction containing purely the geometrical level of information, a description using lines, arcs and dimensional curves.

Process Planning has always been a part of manufacturing. Many industrial parts are now made in small lots which used to be produced in mass production in the past. To cope with this frequent change of batch-types, it becomes important to make the Process Planning function efficient. The Process Planning function is also important because it bridges design and manufacturing. The philosophy of GT can be used to make the Process Planning function efficient thereby increasing its productivity.

Computer Aided Process Planning (CAPP) is a systematic determination of methods by which a component is to be manufactured economically and competitively with the aid of computers. Generally, the CAPP system is considered as a bridge to connect CAD and CAM. There are basically, two approaches used to automate the Process Planning function: (1) Variant, and (2) Generative. In the Variant approach, parts are grouped using the GT approach into part families and standard plans are stored for each family. Planning for new parts involves classification and retrieval of existing plans for the part family, and modification of the plan for the new part. This approach is not adaptable for complete automation. In the Generative approach, the system synthesises process information to create a process plan for a new component automatically. The Generative method seeks to emulate the decision making by the process planner and creates process plans automatically, which is a step towards complete automation.

The implementation of a Generative Process Planning system is still a long way from matching the industrial expectations. The potentials of both Generative and Variant planning systems are utilised in the approach called the Hybrid Process Planning system. The Hybrid system is based on the Variant approach to exploit the benefits of GT. Developing a Generative Process Planning system becomes easy and manageable if a domain of similar components or boundary of the problem is explicitly defined. GT provides this basis by dividing the similar components into part families.

The work of this thesis addresses part grouping which can make the Process Planning activity efficient and productive. In this context research was concentrated on the selection of classifying attributes which can provide the basis for identifying clusters of similar components from the manufacturing point of view. As Process Planning logic compares the component machining requirements with the processing capabilities to select feasible solutions therefore existing manufacturing system capabilities have also been incorporated in the part grouping function.

In order to address the research work, there was a need to concentrate the research to address the issues listed below:

- Exploration of feature-based part geometry input to the part grouping as features are considered a means to integrate CAD and CAM.
- Exploration and structure of a feature-based data model that can be used for the grouping of components, CAPP function and manufacturing cell design.
- Exploration and structure of the manufacturing knowledge that can be used in the part grouping, the Process Planning activity, and manufacturing cell design.
- Exploration and contribution to the selection of classifying attributes for the part family formation so that realistic part families can be achieved which can make the Process Planning function efficient.
- Exploration of the feature-based grouping of machined components.

- Exploration and contribution to finding the natural clusters of groups in the grouping process.
- The investigation of the issues regarding the development of a Hybrid Process Planning system.
- The investigation of the issues related to the manufacturing cell design.

This introductory chapter of the thesis has attempted to set the work into context and whet the reader's appetite for the chapters that follow. Chapter 2 covers a survey of relevant literature, in relation to the concepts of the elements that will be dealt with related to this research. Chapter 3 highlights the role of the Feature Technology in the integration of CADCAM, the form feature taxonomy which is used in this work, concept of feature connectivity, feature-based part data model and its implementation in the knowledge based system.

Chapter 4 details the author's contribution work in the area of part grouping. Moreover it discusses an approach to come up with the optimum number of groups in the grouping process. Comparison of the proposed part grouping CAFBG System with other traditional part grouping techniques has also been made in this chapter. Chapter 5 reports on the software implementation and case studies in relation to the part grouping discussed in the previous chapter. Results are critically evaluated and compared in this chapter as well. Chapter 6 explains the work performed by the author in developing HYCAPP—a Hybrid Process Planning system while chapter 7 describes the software implementation side regarding HYCAPP. Chapter 8 includes the work in the area of cell design. Concluding discussion is made in chapter 9. Finally research conclusions are drawn, and recommendations made for future work in chapter 10.

CHAPTER 2

LITERATURE SURVEY

2.1 INTRODUCTION

The aim of this literature survey is to give a background assessment of the research work carried out in the fields of Group Technology, Process Planning and Cellular Manufacturing. The identification of the problems and the approaches adapted by other researcher in these areas are covered.

These topics include the CAD/CAM integration, Feature Technology, concept and role of Group Technology, part grouping techniques, Artificial Intelligence and Expert Systems, knowledge representation, Process Planning and Cellular Manufacturing.

Investigations and research pertaining to live issues in the above mentioned areas have been carried out internationally by prominent researchers in research institutes, universities and manufacturing industry. These findings are mainly reviewed in this literature survey.

2.2 CAD/CAM INTEGRATION

The terminology of CAD, CAM, and CIM is by now well known in industry. The terms are closely related. Computer Integrated Manufacturing (CIM) possesses slightly broader meaning than CAD/CAM. CIM includes all the engineering functions of CAD/CAM, but it also includes the business functions as well. Detailed interpretations of these terms and their relationship can be found in the publications Horrington [1973], Barash [1980], Sadowski [1984], Vernadat [1984], Groover and Zimmer [1984], Broaden [1985], Baxter [1985], Broaden and Dale [1986a, 1986b], Besant and Lui [1986], Ranky [1986, 1990], Groover [1987], Lim and Knight [1987], Kanumury *et al.* [1988], Haywood [1990], Bedworth *et al.* [1991].

The eye is our most efficient data channel. Over 70% of all the information received by the brain is extracted by visual input [Bo 1989]. Many complex problems are simplified when translated into graphic images, CAD systems provide the ability to manipulate such images easily and accurately which enormously enhances problem solving abilities, and hence raises the creative productivity of designers and engineers [Bo 1989, Schlechtendahl 1988].

CAD/CAM is a term that means computer-aided design and computer-aided manufacturing. It is the technology concerned with the use of digital computers to perform certain functions in design and production. CAD can be termed as computer based systems to assist in the creation, modification, analysis, or optimisation of the design, while CAM can be termed as the use of computer systems to plan, manage and control the operations of a manufacturing plant [Besant and Lui 1986].

According to Schlechtendahl [1988] CAD can be defined as a discipline that provides the required know-how in computer hardware and software, in systems analysis and engineering methodology for specifying, designing, implementing, and using computer based systems for the design process. Allen and Smith [1984] defines CAM as the effective utilisation of computer technology in the management, control, and operation of a manufacturing facility through direct or indirect computer interface with the physical and human resources of the company.

CAD/CAM methods cannot solve all the problems in manufacturing separately. Different CAD/CAM functions are shown in figure 2.1. Clearly the main problem of automation is not optimising one production step but passing the existing information from one process to another. If each application is dealt with separately, a separate solution is made. Each will have its own data model with a specific data representation and stored in several places. This is waste of time and effort and proliferation of data as well. So, to harvest the fruit of true automation, there is a need to integrate CAD and CAM. Integrating between design and manufacturing functions means that the part description, generated during the design phase, can be used as input for the manufacturing.

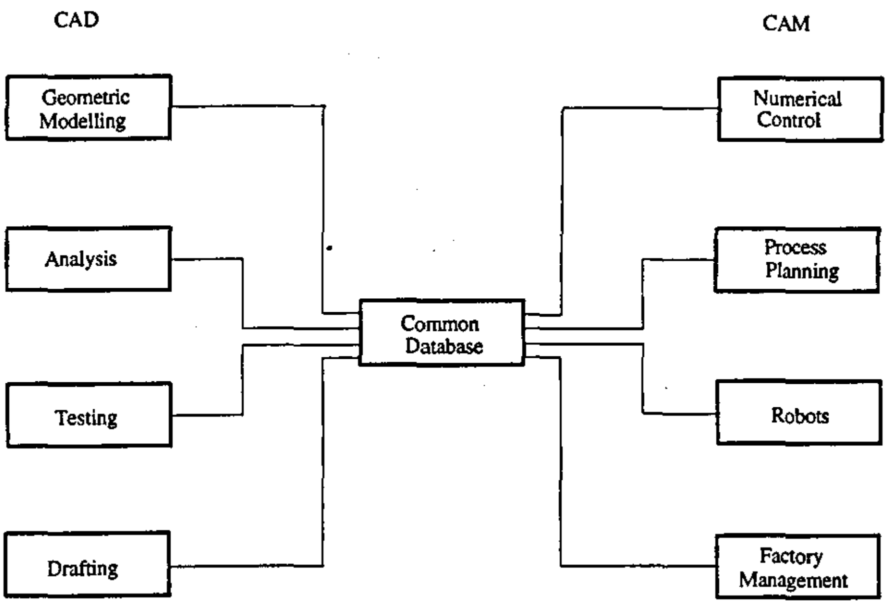


Figure 2.1 CAD/CAM Functions

[Adapted from Hordeski 1986]

The integration of design and manufacturing promises the manufacturer higher productivity, greater efficiency, better quality at lower cost, ability to give better customer service and flexibility to meet the demand for an increasing array of products and options that have shorter life cycles than ever before. Ham and Lu [1988] in their keynote paper emphasise the need for an integrated planning approach to manufacturing problems, highlighting this as an important future direction of planning research.

The Computer Aided Process Planning (CAPP) system is to integrate the CAD and CAM systems. In order to realise true integration of the both systems, there is a need to develop the CAPP system that can generate the process plan based on the part geometry designed by the CAD system and then down load for CAM activity. A common database is needed to integrate CAD/CAM functions based on which decisions can be made, at any stage in the product's life cycle. Software programs can be written to deal with different functions of manufacturing engineering which will be using the common database. Product Modelling is based on the common database concept. Product Modelling is currently an active area of research. Product Modelling allows the acquisition and availability of required information (data/knowledge) related to a product during the entire cycle of design, planning and manufacturing. It covers the whole product life cycle. Further discussion on Product Modelling can be found in the articles [Kimura *et al.* 1984, Shaw *et al.* 1989, Spur *et al.* 1986, Tilley 1992, Young 1991, Imamura *et al.* 1988]. Graphics Standards like PDES and STEP are based on this Product Modelling concept. The Concurrent Engineering approach is also a step towards integrating CAD/CAM systems. Ham and Lu [1988] emphasise the need for an integrated planning approach to manufacturing problems, highlighting this as an important future direction of planning research as said earlier. The Concurrent Engineering concept aims at evaluating the manufacturing activities while designing the product. Cutkosky *et al* [1988] argue that a product should be designed in terms of manufacturing plans that can produce them.

A feature is defined as 'a region of interest on the surface of part'. This definition is deliberately very broad in order to encompass the myriad of items that engineers consider to be features of a part at various stages in the design and manufacturing processes [Pratt and Wilson 1985]. They are in agreement that form features provide one of the essential links between design and manufacturing in a fully automated CAD/CAM environment.

Features correspond to familiar terms for describing product characteristics. Features provide a basis for communication and understanding between engineering and manufacturing. The application of Feature Technology can therefore be expected to yield benefits throughout the product development and production cycle [Brimson and Downey 1986].

Features provide significant advantages to design engineering, manufacturing engineering and production planning because the features data can be shared among these activities. Butterfield *et al.* [1985] also, in their CAM-I report argue that features are considered to be the communications medium between design and manufacturing. Keeping this in view, it can be argued that features and CAPP can result in the true integration of CAD and CAM.

2.3 PART DESCRIPTION FORMATS

Manufacturing is a means to realise the design [Chang 1990]. The geometry of the part can be represented in a number of ways. The traditional methods include physical models (clay models, wooden prototypes), symbolic representations and engineering drawing.

As a next step, GT Classification and Coding systems [refer to section] have been used to describe the part. Part description via GT codes has been reported in various Variant Process Planning systems. One of the example of Variant system system based on GT codes is MIPLAN [Schaffer 1980]. Some Generative systems such as APPAS [Wysk 1977] and GENPLAN [Tulkoff 1981] also used GT codes as input. Interpretation of part data has to be performed manually and exact size information is lost; therefore, GT codes are also not suitable for the reasoning process. Description languages have also been used for the part description. Systems using this approach are AUTAP [Eversheim *et al.* 1980], CIMS/PRO [Iwata *et al.* 1980]. This approach could also not get much popularity because it could not provide the base for the reasoning.

With the advent of computers and plotters, the age of CAD came. Systems like, CADCAM [Chang 1980], TIPPS [Chang and Wysk 1984] used the CAD models for the part description in the CAPP functions. The problem with these models is that they still require the development of reasoning or logic to identify the manufacturing features. Computer Technology has also resulted in a new part geometry representation scheme called AI technique which is most suitable for the reasoning process needed in the planning function. The CAD systems and AI based techniques will be taken for discussion in the following:

2.3.1 THE CAD SYSTEM

The method by which the CAD modellers hold information and the way information is structured to produce a geometric model differs for each of the corresponding models listed below [Bedworth *et al.* 1991]:

1. Wireframe modeller,
2. Surface modeller,
3. Solid modeller.

In 1953, the first wireframe modeller CAD system was developed at the Massachusetts Institute of Technology (MIT) by Sutherland and enabled designers to draw 2D graphics [Sutherland 1963]. A wireframe modeller CAD system holds information about the model in terms of edges and vertices alone [Medland 1988]. The component cannot be represented unambiguously by using this system. For example, it is very difficult to distinguish between the interior and exterior image of the object in this scheme. As a result, with this system it is very difficult to recognise the machined surfaces without the interpretation of the users.

In the surface modelling system, objects are represented by their bounding faces where faces consist of edges and vertices. There is the same problem with the recognition of interior and exterior images of the object in this type of system as well.

A solid model is the recent approach used in geometric modelling. This is a superior system for creating the 3D models. Solid modelling has the potential to support the automation of many technical tasks in industry, from detailed strength analysis to planning of machining and assembly [Walker and West 1990]. Requicha and Voelcker [1982] define the term solid modelling as encompassing the emerging body of theory, techniques, and systems focused on "informationally complete" representation of solids-representations that permit (at least in principle) any well-defined geometrical property of any represented object to be calculated automatically.

Current solid modellers lack tolerancing and surface finish information which is necessary for downstream Process Planning. They also need some method to incorporate form features such as thread, surfaces, etc. The concept of a datum plan and reference surface for dimensioning needs to be incorporated. Further research in these areas is needed before a complete effective CAD interface can be developed [Joshi *et al.* 1988a].

Perhaps the most remarkable instance of growth in productivity as a result of information technology is in the design of the parts and production processes. CAD programs can carry out geometric transformations so fast that the designer is no longer limited to the top, side and front views of a part that were characteristics of manually prepared drawings. He can observe the rotation of the part about any axis on the screen, zoom in close to see details or take up a distant point of view to visualise an object as a whole. Any cross section of the part can be displayed. If the part is to be mated with other parts during assembly, the designer can move the parts about on the screen to check for fit. Hence many prototypes and engineering models can be eliminated.

There are basically two approaches used in this model as (1) Boundary Representations or B-Reps and (2) Constructive Solid Geometry or CSG. In a CSG model an object is represented by a binary tree consisting of geometrical primitives, transformations, and symbols representing boolean operators [Chang 1990]. B-Reps are represented as a union of faces, with each face represented in terms of its boundary of edges. A further discussion on the solid modelling can be found in the publications [Requicha and Voelcker 1982,1983, Mullineux 1986, Mantyla 1988]. Recently, a comprehensive discussion on the geometry representation has been made by Case and Gao [1993].

There are also four other schemes which are frequently used in conjunction with CSG or B-Rep for certain kinds of applications [Besant and Lui 1986]:

1. Spatial Enumeration
2. Cell Decomposition
3. Sweeping
4. Primitive Instancing

2.3.2 THE ARTIFICIAL INTELLIGENCE (AI) BASED TECHNIQUES

The input format of any system involving a reasoning process affects the ease with which the system can be used, and the capability of the system in terms of the degree of automation. Recently Expert Systems are being used to convey the design information of the part. According to Joshi *et al.* [1988a] Expert Systems provide an excellent framework for incorporating the decision-making process of the planner and making it suitable for automation. Many planning systems like, GARI [Descotte and Latombe 1981, 1985], TOM [Matsushima *et al.* 1982], PROPLAN [Mouleeswaran 1984], EXCAP [Davies and Darbyshire 1984], SAPT [Milacic 1985], SIPP [Nau and Chang 1985] use this approach for the description of a part.

This research uses AI based technique for a part description, development of CAFBG and HYCAPP Systems and Cell design. Detailed description of AI will be taken in latter.

2.4 PRODUCT MODELLING

The need to integrate different islands of automation (CAD, CAM etc.) in the CIM paradigm has resulted in the necessity to develop a scheme which can be used for a wider range of engineering information to be represented in a systematic and structured way. The scheme has been named as 'Product Modelling'. Product Modelling allows the acquisition and availability of required information (data/knowledge) related to a product during the entire cycle of design, planning and manufacturing. In short, it covers the whole product life cycle. Further information about the topic is given by Kimura *et al.* [1984], Spur *et al.* [1986], Imamura *et al.* [1988], Shaw *et al.* [1989]. Young [1991], and Tilley [1992].

Information in the product model can be divided into three categories: (1) Design information (functional) which is related to the function and geometry of the component, (2) Manufacturing information (technological) related to the component, and (3) Production information (managerial). Product model research has expanded basically from the realisation that solid modelling does not bring the necessary information to allow an integration between CAD/CAM [Spur *et al.* 1986].

2.5 FEATURE TECHNOLOGY

Recently the concept of using component features for design and manufacturing applications has received much attention and research effort. Features give a higher conceptual meaning to component characteristics by dissecting component geometry into recognisable and meaningful forms. Features are meant to represent the details of engineering drawing by higher order forms rather than individual lines, arcs and vertices (lower order forms). Features describe the collection of entities, the way engineers think. The handling, management, and use of these groups of geometry (lines, vertices etc.) can be seen as a practical way of converting design into manufacturable products. Thus features are considered to be the communications medium between design and manufacturing. Features include geometric shapes, engineering notes, specification data, record management data, and other characteristics considered [Butterfield *et al.* 1985].

Feature Technology is a method of identifying, based on part features, the unique manufacturing requirements for each part. Part feature-standard descriptions of a part's geometric characteristics should be independent of company or function and readily transformable into manufacturing operations and sequences of manufacturing operations. The components of material to be removed in the machining operation are called delta volumes. The process of machining can be thought of as a series of removals of the material. Each distinct step is represented by a delta volume until the final product is completed. Features can be correlated to each distinct manufacturing operation and thus to each delta volume. By associating specific machining techniques for removing material with these features, the combination of features, delta volumes, and sequences provides a complete representation of the manufacturing process [Brimson and Downey 1986].

The Automated Manufacturing Research Facility (AMRF) Process Planning team proposes the definition of a feature as below [Unger and Ray 1988]:

"A feature is a higher level of grouping of geometrical, topological, and functional primitives into an entity more suitable for use in design, analysis, or manufacture".

Each researcher working in the features arena has his own definition of a feature and the definitions differ. The definitions of a feature, as given by some of the major researchers, are presented here:

... Geometry that corresponds to primary machining operations [Grayer 1976].

... geometric and topological patterns of interest in a part model, which represent high level entities useful in part analysis [Henderson *et al.* 1990].

... a bounded (boxed-in) volume which consists of an entry boundary, an exit boundary and a depth boundary [Gindy 1989].

... a set of geometric entities (faces, edges, etc.) which together define a topology and geometry [Joneja and Chang 1991].

... a set of connected faces related to a specific manufacturing process [Henderson 1984a].

... a set of information related to a part's description. The description could be for design purposes, or manufacturing and inspection or even for administrative purposes [Shah and Rogers 1988c].

... 'a region of interest on the surface of part'. This broad definition is given by Pratt and Wilson [1985] to encompass all the different design and manufacturing functions.

... a geometric shape defined by parameter set, which has the special meaning to a designer or manufacturing engineer [Clark and South 1987].

... a portion of the workpiece generated by a certain mode of metal cutting thus relating features to the manufacturing methods [Choi *et al.* 1984].

... any geometric form or entity whose presence or dimensions in a domain are germane to manufacturing evaluation or planning, or to automation of functional analyses [Dixon 1986].

... Any geometric form or entity that is used in the reasoning of one or more design or activities, feature can be geometric, topological, or attributes [Cunningham and Dixon 1988].

... the regions of the part that have some degree of manufacturing significance. Put another way, features form reoccurring geometric and technologic patterns for which the process engineer has acquired years of manufacturing experience [Hummel and Brooks 1986].

... a feature is a geometric form or entity, whose presence or dimensions are required to perform at least one CIM function (i.e. graphics, analysis, Process Planning), and whose availability as a primitive permits the design process to occur [Luby *et al.* 1986].

All of the definitions of features share the idea of a 'geometric entity'. All of the definitions also assume that features are combined to form parts or other objects of importance. All of the definitions also imply that features provide a higher level model of the object than does a traditional CAD geometric model, and thus may be easier to reason about or use [Unger and Ray 1988].

The definitions of features relate to the machining processes recognised from a design drawing by a Process Planner. A feature can usually be directly linked to a specific set of machine tools, viz., through slots are machined using a mill, possibly with a form tool. Examples are holes, slots, pockets, faces, threads or fillets. The faces may be on the surface of the actual part, or alternatively could be defined on the volume or volumes of material which are removed from the stock material. The data includes geometry and topology of the feature together with manufacturing parameters such as surface finish, tolerances and material specifications [Henderson 1984].

2.5.1 FEATURE RELATED RESEARCH

A feature is considered as a building block for an engineering part carrying special meaning. The representation of a part in terms of features is a recent approach by most of the researchers. Feature related research falls into three categories: (1) human-assisted feature recognition, (2) feature recognition and extraction, and (3) designing by feature.

2.5.1.1 HUMAN-ASSISTED FEATURE RECOGNITION

This method was designed basically for Process Planning and NC tool path generation. Human-assisted feature recognition implies interpretation of a part model and translating it into a set of machinable features, but today this approach is neither convenient nor efficient, requiring too great investment in time and manpower. This approach is presented in several works Chang and Wysk [1984], Hummel and Brooks [1986], Nau and Gray [1987].

2.5.1.2 FEATURE RECOGNITION AND EXTRACTION

Feature recognition and extraction approach is to identify the form features by reasoning the geometric model. The solid modellers, in the design activity, allow the storage of data related to a component's definition in terms of low level geometric entities (vertices, lines, etc.) and containing Boolean operators. Automatic feature recognition and extraction programs try to identify and extract generally pre-defined form features.

Because of the importance of solid modelling in CAD/CAM integration, the extraction of geometric features from a CAD database has received considerable attention. Woo [1977] proposed a methodology for extracting geometric information from a CSG model for 2&1/2-D workpieces. Chen [1982] took a different approach in which the solid model ROMULUS and NC programming system APT were linked together so that the geometric definitions of APT can be obtained interactively from the faces of the part created by ROMULUS. Chang and Wysk [1984] explored the hole-making processes using a B-Reps model. Choi and Barash [1984] utilised a simplified B-Reps format to identify machined surfaces by recursively calling a tool-profile/FMS elementary machined surface matching routine.

2.5.1.3 DESIGNING BY FEATURES

Designing by feature approach emphasises the use of predefined features to construct the geometric model. In this approach the designer creates the models directly from the features. Therefore, manufacturing features are incorporated in the product description right from the beginning [Dixon 1988, Shah and Roger 1990, Chang 1990]. The designer uses a features library, similar to the primitives of a CSG system. Different operators such as add, delete and modify are used to define an engineering object.

The feature recognition process is complex, the number of features that can be recognised is limited and the designer's intent is lost. The design by feature approach allows the designer to model a part in terms of features and this eliminates the need for feature recognition whilst maintaining design intent. The main problem with the design by features approach is a limited available library of features [Goe and Case 1993].

Some of the advantages of using a feature-based design system for engineering design has been mentioned by Chang [1990]. Many design features and manufacturing features are similar. Sometimes there is a one-to-one correspondence of the two. A part designed using features may be planned automatically.

2.5.2 FORM FEATURE TAXONOMY

Features are classified into different categories based on their geometric, topological attributes. The features classification helps the engineers to characterise the groups of features which have common attributes in different functions. Usually, features are classified in the hierarchical form. The hierarchical structure enables objects to inherit data slots and attributes from other objects located above in the hierarchy; a class can receive some information from its super-classes. The hierarchical structure provides a large amount of information in a small number of nodes as the interpretation of each succeeding node is dependent on the information given on the preceding node. This classification structure is called a feature taxonomy. Different taxonomy schemes are found in the literature.

A significant study on form features was conducted by the Deere & Company, Moline Illinois for the CAM-I. According to this study, features are categorised into three main types, namely; (1) sheet features (2) non-rotational (or prismatic) features, and (3) rotational features. Sheet features are further classified as flat pattern features and formed shape features. Prismatic features are further categories as depressions, protrusions, and surfaces (depressions and protrusions, both, are again divided into internal, external, and special functions features). Rotational features are classified as concentric and non-concentric. This scheme also classifies material features and heat treatments. According to Butterfield *et al.* [1985] though the report was intended only for the Process Planning function, yet this is quite general and broad.

According to Pratt and Wilson [1985] it is convenient to divide representations of features into two types as follows:

1. EXPLICIT, where all the geometric details of the feature are fully defined, and
2. IMPLICIT, where sufficient information is supplied to define the feature but the full geometric details have to be calculated when required.

Simple explicit features were further divided into four main types: through holes, protrusions, depressions and areas, with possible sub-division in terms of their cross-sectional shapes such as rotational and prismatic. A similar taxonomy was adapted by Hummel and Brooks [1986] in their expert Process Planning system called XCUT.

Gandhi and Myklebust [1989] take parametric approach to the definition of features. Their feature taxonomy is based on topology of feature primitives, i.e. features having the same topology are grouped together so that they could be defined using the same number of parameters.

Gindy's [1989] feature taxonomy provides a good base for defining a feature completely. According to this taxonomy, component form features are treated as volumes enveloped by entry/exit and depth boundaries. Feature geometry is described by deciding on its external access directions (0,1,2,3,4,5,6), i.e. the number of imaginary faces included in the feature definition, its boundary type (open, closed), its exit boundary status (through, not through), and its form variation with respect to its depth axis. Based on their geometric attributes, features are uniquely classified into categories, classes and subclasses which may be followed by secondary forms to fully describe compound features.

Butterfield's taxonomy is general purpose and can be used in the Process Planning but Gindy's taxonomy scheme can be regarded as specifically for the Process Planning. Whereas Pratt and Wilson's scheme is more oriented towards the solid modelling environment.

2.5.3 FEATURES BENEFITS

Potential benefits reported by Chung [1988] and Shah *et al.* [1988] are that;

- users can express easily the design intent by manipulating features directly, eliminating tedious intermediate steps.
- features data flow allow reasoning systems to perform tasks such as heuristics optimisation, manufacturing analysis.
- features can contain knowledge to facilitate NC machine programming, Process Planning and automatic finite element (FE) meshing.

It would be important to add that features are the perfect elements to be used in the modelling of a product in a product model environment, therefore allowing a perfect integrated manufacturing system.

2.6 GROUP TECHNOLOGY

Group Technology (GT) is drawing increasing interest from manufacturers because of its many applications for boosting productivity. GT is an approach to manufacturing that seeks to maximise production efficiencies by grouping similar and recurring problems and tasks [Hyer 1984a]. GT technique/philosophy is not new and it has been used or presented in different ways, in different countries, at different times. In 1937 Sokolovsky postulated the real premise base of GT where he suggested that parts of similar configuration and features should be manufactured by a standard process. His ideas were implemented and spread by work carried out by Mitrofanov, S. P. considered by most as the father of GT. Opitz gave a big impetus to GT by the development of a code system for the classification and codification of machined components. Burbidge pursued the use of GT as a philosophy by starting with a rationalisation of the shop floor, the so called Production Flow Analysis (PFA) [Snead 1989].

Mitrofanov, S. P., a Russian engineer, published his book, *Scientific Principles of Group Technology.* It was translated into English in 1966. Mitrofanov [1966] defined GT as:

"GT is the method of technological process development, equipment planning and efficient setting of the machine tool, so as to ensure the most profitable technical planning of production in the shortest time".

In the editors forward to the English translation of Mitrofanov's book, T. J. Grayson describes Group Technology as follows:

. . . a method of manufacturing piece parts by classification of these parts into groups and consequently applying to each group similar technological operations. The major result of this method of manufacture is to obtain economies which are normally associated with the large-scale production in the small scale situation and it is therefore of fundamental importance in the batch production and jobbing sections of industry.

Some other definitions of GT encountered in the literature are given below:

Burbidge [1979] defines the GT as an approach to the organisation of work in which the organisational units are relatively independent groups, each responsible for the production of a given family of products.

Gallagher and Knight [1986] describe GT as being a manufacturing philosophy or principle with the basic concept of identifying and bringing together related or similar parts and processes, to take advantage of the similarities which exist during all stages of design and manufacture.

According to Hyer and Wemmerlöv [1984], GT is an approach to manufacturing that seeks to maximise production efficiencies by grouping similar and recurring problems or tasks.

GT is a concept to increase production efficiency by grouping various parts and products with similar design and/or production process. The application of GT results in the mass production effect to multi-product, small-lot-sized production [Ham *et al.* 1985].

GT is the realisation that many problems are similar and by grouping similar problems, a single solution can be found to a set of problems, thus saving time and effort [Solaja and Vrosevic 1973].

GT is a manufacturing philosophy in which similar parts are identified and grouped together to take advantages of the similarities in manufacturing and design [Groover and Zimmer 1984].

GT is a manufacturing philosophy based on a fundamental organisational principle: to identify and bring together related or similar parts and process (and other pertinent data entities) to take advantage of their similarities in design and manufacturing [Millar 1984].

G T is the identification of subsets or families of similar product within the population at large for the purpose of design and manufacturing efficiencies through the consistent application of "best practice" technology to the characteristic attributes of the family [Nolen 1989].

As seen above, the main essence of GT is simply to identify and bring together related or similar parts, elements or/and activities to take advantage of their similarities. The GT philosophy has a major impact on the information handling and manufacturing organisation providing the right platform for the integration of CAD/CAM and implementation of CIM. Similar parts are grouped into part families. Parts classified and grouped into families produce a much more tractable database for management. A new design can be created by modifying an existing component design from the same family. Components in a family require similar processes for their manufacturing. Using this concept, a composite component for each family can be identified. A composite component is a hypothetical part that embodies all the features (design or manufacturing) for any part family. A manufacturing cell can be built for processing this composite component which will eventually fulfil the processing requirements of all the family members.

Apart from the economic benefits gained by using GT, the strategic benefit of GT is of paramount importance in design and manufacturing [Houtzeel 1981]. The strategic benefits of Group Technology are those that have an impact on quality, flexibility, timing and position relative to the company goals and the market's demands [Allen and Smith 1984]. The detailed benefits which result in using GT can be found in Ivanov [1968], Edwards [1971], Thorley [1971], Ranson [1972], Arn [1975], Ham *et al.* [1985], Gallagher and Knight [1973, 1986], Groover [1987], Hyer [1987], Allen

[1988], Burbidge [1975, 1979, 1988], Hyer and Wemmerlöv [1984b, 1989], Snead [1989].

2.6.1 ROLE OF GROUP TECHNOLOGY

The role played by Group Technology in the different areas of design and manufacturing is discussed briefly in the following:

2.6.1.1 PRODUCT DESIGN.

It is not unusual for a company to find several versions of basically the same part during a preliminary investigation of the part population. The part can serve the same function but differ in terms of tolerances, radii, and so on. Design proliferation of this kind occurs because of difficulties with design retrieval. The most obvious application of GT in design engineering is efficient retrieval of previous designs. If the exact part design cannot be found, perhaps a small alteration of existing design will satisfy the function. These features help speed up the design process and curb design proliferation.

Another benefit of GT is that it promotes design standardisation. The aim of design standardisation is to reduce variations, to make the parts efficiently. Standardisation does not mean that all parts with the same function must be identical. It does mean, however, that norms are established for tolerances, dimensions, angles, and other specifications. Setting these norms would be done with both manufacturing and design considerations in mind, bridging the gap between two areas and making design engineers more aware of manufacturing costs and restrictions.

2.6.1.2 CELLULAR MANUFACTURING

The most advanced GT application is through the creation of manufacturing cells. A cell is a collection of machine tools and materials handling equipment grouped to process one or several part families. Preferably, parts are completed within one cell.

The advantages of Cellular Manufacturing are many, especially when the cells are designed with one dominant materials flow and with a fixed conveyor system connecting the work stations. A cell represents a hybrid production system, a mixture of a job shop producing a large variety of parts and a flow shop dedicated to mass production of one product.

The allocation of equipment to a subset of parts or part families will reduce interference, improve quality, make materials handling more efficient, cut set-up and run times, and therefore trim inventories and shorten lead times. Shortening parts manufacturing lead times can reduce the response time to customer orders and thus lead to smaller finished goods inventories as well. These benefits are likely to be greater with a physical arrangement of machinery into cells.

2.6.1.3 PROCESS PLANNING

Some of the largest productivity gains have been reported in the creation of Process Plans that determine how a part should be produced. With computer-aided Process Planning (CAPP) and GT it is possible to standardise such plans, reduce the number of new ones, and store, retrieve, edit, and print them out very efficiently.

Equipment selection, and plant layout are directly affected by standardising the routings through GT application. As Process Plan proliferation is reduced, increased traffic will occur along certain production flows within the manufacturing facility. The material handling benefits because of locating the two machines in closer proximity become evident.

2.6.1.4 SYSTEM INTEGRATION

To achieve integration in a system, it is necessary to integrate the information system, integrate the control system and integrate the material flow system. GT provides the right platform for this total integration by the application of family related concepts based on geometrical function and processing similarities among parts and activities.

A database related to predefined and identified attributes (features) is the foundation of an efficient and rationalised integrated system. Part classification systems can be used as a tool for integration into a computer database structure that links all activities bringing to reality CIM based on GT and component features.

As the development of CAD/CAM goes more towards a Generative approach for Process Planning, concurrent design for manufacturing and the appearance of Product Modelling systems, GT should emerge as an important element in the implementations of such techniques. A path to this evolution is by the definition of component related features that are not related only to specific activities (design, manufacturing, Process Planning) but carry associated information (data, knowledge) related to the whole spectrum of CIM activities.

2.6.1.5 OTHER AREAS

GT can also be applied in purchasing. Relying on the GT coding of purchased components and raw materials and on information from the production planning system, a purchasing manager can obtain statistics not directly available with a traditional parts numbering system. GT can help reduce proliferation of purchases of different kinds of parts.

Another interesting application is in sales. The same company received a request for immediate delivery of any particular component that was not a stock item. A search of the GT database, however, turned up a substitute part that fits the customer's need and could be delivered right away. GT can also be used for cost estimation. Several companies have found that GT generated cost estimates can be constructed more quickly and with greater accuracy than those made by traditional methods. The approach is also helpful during the design process to help select components that will lower the total cost of the proposed product.

2.7 PART GROUPING

The end aim of grouping parts is to produce them on equipment dedicated to specific product groups. Several approaches have been developed to form the part families. These approaches can be broadly categorised into two main groups:

1. Part characteristics based approaches, and
2. Production methods based approaches.

2.7.1 PART CHARACTERISTICS BASED APPROACHES

Part characteristics based approaches usually employ classification and coding (C&C) schemes and parts are mainly classified based on their similarities in design features and functionalities.

2.7.1.1 CLASSIFICATION AND CODING

Classification is the process in which parts are categorised into groups or families based on existence or absence of characteristic attributes already established in any classification scheme. The objectives are to group together similar parts and to differentiate among dissimilar parts. Coding is a process of establishing symbols to be used for meaningful communication. Coding can be used for classification purposes. These symbols should have meanings that reflect the attributes of the part, thereby facilitating analysis. Before a coding scheme can be constructed, a survey of all component characteristics/features must be completed, then coding values can be assigned to the features. The selection of relevant features is dependent on the application of the coding scheme. Several C&C systems have been developed, but none of the systems have yet received universal acceptance because of the various needs of different companies.

Classification refers to the assignment of parts into predefined groups or classes, while coding is the allocation of identities to these groups. The type and amount of information contained in the code depends on the potential uses of the system. A designer may wish to retrieve designs to obtain relevant information and to use existing parts in new items, while retrieval is also necessary in connection with costing, planning, variety reduction, etc. For this reason the design of a classification and coding system is normally a compromise that attempts to satisfy as many potential demands as possible [Wild 1985].

Hyer and Wemmerlöv [1989] report the merits gained by introducing Group Technology in a variety of areas in 53 different types of industry. They claim that in the majority of cases, firms use classification and coding systems as tools in applying GT. Recently, an Expert System for part C&C has been reported by Zeng and Yang [1992].

CODING STRUCTURES

There are three types of coding structures in GT coding systems: (1) Hierarchical (2) Chain or Polycode and, (3) Hybrid. In hierarchical code structure, the interpretation of each succeeding symbol depends on the value of the preceding symbols. Such a coding system can be depicted using a tree structure. A hierarchical code provides a large amount of information in a relatively small number of digits. Design departments frequently use hierarchical coding systems for part retrieval because this type of system is very effective for capturing shape, material, and size information.

In the polycode type of code structure, the interpretation of each symbol in the sequence is fixed. It does not depend on the value of the preceding symbol; thus, each attribute of a part can be assigned a specific position in an attribute code. This code system is popular with the manufacturing organisations because it makes it easy to identify parts that have similar features that require similar processing. The disadvantage of this code is that a position in the code must be reserved for each different part attribute; therefore, the resulting code may become very long.

Most codes that are used in industry are neither hierarchical nor polycode, but are hybrid or a mixture of the both. A common form of hybrid coding is to divide the population into small groups using one or two monocode digits and then place a polycode series in each branch where the polycode series of questions have significance to the group in the branch.

2.7.1.2 CLASSIFICATION AND CODING SYSTEMS

There are more than a hundred GT classification and coding systems used in industry today. Some of them are given in the following section to provide insight into the types of the systems available and to indicate the direction in which commercial classification and coding systems developers are heading.

THE OPITZ SYSTEM

This parts classification and coding system was developed by Opitz of Aachen in Germany. It is one of the pioneering efforts in the Group Technology (GT) area and probably the best known system [Groover 1987].

The Opitz system is a hybrid type consisting of five basic digits for geometry and four supplementary digits. The first five digits, called the "form code" describe the primary design attributes of the part. The "form code" represent component class, basic shape, rotational surface machining, plan surface machining, auxiliary holes, gear teeth and forming. The next four digits indicate the attributes that would be of use to manufacturing (dimensions, work material, starting raw workpiece shape and accuracy). The Opitz system of classification is concise and easy to use. It has been adapted by many companies as their coding system and several computer-aided Process Planning systems (CAPP) based on Opitz system have been reported in the literature.

THE CODE SYSTEM

CODE is a classification and coding system provided by Manufacturing Data Systems, Incorporated (MDSI) [Haan 1977]. It is an eight-digit hybrid code, similar to Opitz system, used primarily to classify and code mechanical piece parts. Each digit of the code is represented by a hexadecimal value. Using hexadecimal numbers allows more information to be represented with the same number of digits. CODE contains form and dimensional information, but does not have accuracy or material information.

The CODE system encompasses both rotational and prismatic components. The CODE system can better represent the size information as more digits in this system have been assigned to auxiliary shapes. CODE directly classifies major dimensions, instead of using the ratio of dimensions as in Opitz system. Two digits have been reserved for the dimensions.

MDSI has developed their software to handle the code of the part, as well as other relevant part information such as tooling, routing, tolerance etc. This data can be stored by CODE into database or the CODE software can be adapted to retrieve this information from other established databases in the company [Snead 1989].

THE DCLASS SYSTEM

The DCLASS system was developed in the Computer-Aided Manufacturing Laboratory of Brigham Young University [Allen 1979]. DCLASS is an acronym for Decision and Classification Information system. DCLASS is a tree-structured system which can generate codes for components, materials, processes, machines, and tools. For components, an eight-digit code is used:

Digits 1-3	Basic shape
Digits 4	Form feature
Digits 5	Size
Digits 6	Precision
Digits 7-8	Material

It is not a fixed code classification and coding system but is a computer software system designed to rapidly and efficiently traverse decision tree logic. The software is able to process the various decision tree types. As it traverses the decision trees, the program sets a string of binary digits that is a code to the computer but is transparent to the user. This binary set code can then be used to compare the information with databases of other coded information [Snead 1989].

A Generative Process Planning system using DCLASS has been reported by Allen and Smith [1979, 1980].

THE MICLASS SYSTEM

The MICLASS classification and coding system owned by OIR (Organisation for Industrial Research, Inc.) was developed out of the work done by the Organisation for Applied Scientific Research in the Netherlands (TNO) to develop a system for both design and manufacturing needs [Houtzeel 1981]. MICLASS is an expandable hybrid code system that has the first twelve digits standardised. These digits relate to shape, form, dimensions, tolerances, and materials. The system can be enlarged to thirty digits to cover any classification attribute desired by the user.

OIR has made two significant technological changes in their coding system since the introduction of MICLASS. First one is the creation of MULTICLASS. MULTICLASS is flexible and traverses only the decision tree branch required for the item being coded instead of a fixed decision tree format as MICLASS. OIR has complementary software systems to MULTICLASS that are used for Process Planning (MULTICAPP) and Group Technology applications (MULTIGROUP) The second major change is a new generation of application known as the MULTI-II family. MULTI-II is a set of integrated systems operating with the use of a relational database that addresses engineering, production, and business applications [Snead 1989].

THE BRISCH BIRN SYSTEM

The Brisch system of classification and coding was first developed in England. The present system has carried forth the early Brisch system basic theory of providing a classification and coding system to introduce order, system, and control into the total manufacturing complex. The Brisch Birn code is all-numeric and is not fixed, but is tailored by Brisch to meet the needs of the company implementing it. Each application becomes a fixed number of digits that provides a unique place for each part.

The Brisch code is well suited for design retrieval. The company has developed software to provide for the integration and retrievability of data necessary to provide a company's product [Snead 1989].

Description of other classification and coding systems can be found in [Snead 1989], and [Wang and Li 1991].

2.7.2 PRODUCTION METHODS BASED APPROACHES

Production methods based approaches involve the manufacturing data such as production methods, Process Plans and process route sheet information. The algorithms under this category can be divided into the following major groups.

2.7.2.1 PRODUCTION FLOW ANALYSIS (PFA)

Production Flow Analysis, devised by Professor J. L. Burbidge is a systematic and flexible method for developing a conceptual model of part flows that facilitates conversion of job shop or operations to a Cellular Manufacturing approach thus grouping the similar parts in the part families.

By a progressive analysis of the information contained in the route cards for the components and assemblies produced in a factory, Production Flow Analysis looks for the natural divisions of groups and families into which the components will fall on the basis of similar routes in terms of machines used. It also identifies any exceptional components which do not fit the solution suitable for the majority.

PFA is concerned only with the methods, plant and tooling which are currently being used in the factory, and does not attempt to change these established processing methods, or try to achieve a technological revolution. According to this method, the best way to introduce Group Technology is to change first to group layout with existing methods, which require the least possible investment in new plant and tooling. It does not use part drawings to identify families. Instead, PFA is used to analyse the operation sequence and machine routing for the parts produced in the given shop. It groups parts with identical or similar routing together. These groups can then be used to form logical Machine Cells in a GT layout.

Groover [1987] argues that the disadvantage of using PFA is that it provides no mechanism for rationalising the manufacturing routings. It takes route sheets the way they are, with no consideration being given to whether the routings are optimal or consistent or even logical.

PFA is a technique for pre-planning a whole factory into groups, so as to achieve a more ordered layout resulting in improved work flow. PFA is a hierarchical method having the following five steps of analysis [Burbidge 1971].

- Company Flow Analysis (CFA)
- Factory Flow Analysis (FFA)
- Group Analysis (GA)
- Line Analysis (LA)
- Tooling Analysis (TA)

CFA is necessary only when a company has multiple plants and production is being divided between these plants. FFA divides products and machines in a particular plant into major departments. GA is further division within the department to develop product-oriented cells where part families will be machined. LA is to determine the ultimate layout of each machine within each cell for efficient processing of components, and TA is to develop specifications for the assignment of tooling and other resources in each cell. Further discussion on the topic can be found in Burbidge and Zelenovic [1983] and Burbidge [1963, 1964, 1971, 1974, 1975, 1977, 1979, 1988, 1989, 1991].

The stage of group analysis is probably the most difficult and critical in the application of PFA. The basic steps include the formation of a machine component matrix constituting the data required, and the derivation of machine groups and component families by a quantitative analysis of this data. GA is considered to be the "backbone" analysis required for developing a Cellular Manufacturing plan and results in the generation of part families and allocation of machines to cells simultaneously.

The main objective of GA is [Burbidge 1989] to form cells which include:

- Complete all the parts they make,
- Contain the facilities they need to make these parts,
- Use existing plant without the need to purchase new equipment,
- Use existing processing specifications with only minor changes to eliminate exceptions.

El-Essawy and Torrance [1972] introduce part flow analysis. This is a similar technique to PFA [Burbidge 1971]. Because the characteristics of operation and handling are both considered, not only can this method be used in the mass production of parts, but also in plant layout and process flow analysis.

2.7.2.2 ARRAY BASED CLUSTERING APPROACHES

The array-based clustering approach has recently been the subject of extensive research. This technique rearranges rows and columns of an input matrix to produce a block diagonal solution matrix but does not always give good grouping solutions to some problems and must rely on the user to identify exceptional elements and bottleneck machines and prevent the formation of a block structure in a solution matrix.

King [1980] introduced the rank order clustering (ROC) algorithm to obtain the close relationship matrix of machines versus components by using binary values as the weight to represent the rank of the machine-component matrix. King and Nakornchai [1982] present the ROC-2 method to shorten the total arrangement procedure of the matrix with the translations of column and row rank. Chandrashakharan and Rajagopalan [1986] present the modified rank order clustering (MODROC) algorithm to avoid the disadvantages of ROC algorithm caused by the initial machine-component matrix. Though it improves the ROC algorithm, no effective solution is offered in dealing with bottlenecks or exceptional operations.

Chan and Milner [1982] develop Direct Clustering Analysis (DCA) algorithm for forming component families and machine groups for Cellular Manufacture by progressively restructuring the machine component matrix. The method allows interaction from the user when exceptions and overlap between groups cause the iterative algorithm to prematurely stop. Lee and Garcia-Diaz [1993] use a part-machine matrix for the part family formation problem using the cluster analysis approach.

McCormick *et al.* [1972] also contributes by optimising the bond energy between the adjoining row and column elements in a matrix and forming groups. Bond energy analysis has been also reported by Gongaware and Ham [1981].

2.7.2.3 SIMILARITY COEFFICIENT METHOD

Similarity coefficient methods involve the computing of the similarity coefficient between pairs of machines/parts. Those machines/parts with higher level of similarity coefficients are arranged into the same groups following a particular algorithm. These methods are more flexible in incorporating manufacturing data in the Machine Cells formation process than the array-based clustering methods.

A methodology consisting of progressive aggregation of machines into cells based on similarity coefficients, called single linkage, was first proposed by McAuley [1972], and latter improved by Carrie [1973], Rajagopalan and Batra [1982], Waghodekar and Sahu [1984], Seifoddini and Wolfe [1986, 1987]. Carrie [1973] represents the numerical taxonomy method which uses the component similarity coefficients to form Machine Cells. Waghodekar and Sahu [1984] devised an algorithm called MACE which is based on similarity coefficients, in order to minimise the number of exceptional parts. DeWitte [1980] has also used the similarity coefficient approach in his work.

The purpose of cluster analysis is to group objects together into several distinct, mutually exclusive subsets known as clusters, in such a way that all elements in the same cluster exhibit a high degree of association among themselves. Anderberg [1973]. Hartigan [1975], Kennedy [1974], Kusiak [1990] and Kusiak *et al.* [1985] present extensive bibliographical surveys of cluster analysis and related GT applications.

A model for the duplication of bottleneck machines based on inter-cellular moves has been devised by Seifoddini and Wolfe [1986]. In their next work, Seifoddini and Wolfe [1987] devise a technique to determine the number and size of machine groups instead of just choosing the threshold values. This work is based on considering both the inter-cellular and intra-cellular material handling costs. Costs involved in the case of material handling in the inter-cellular process and duplication of machines have been analysed by Seifoddini [1989a]. He proposes the Average Linkage Clustering (ALC) method to overcome the chaining problem of SLCA [Seifoddini 1989b]. Seifoddini [1989c] in his next paper argues that ALC reduces the chance of improper machine assignment. A probabilistic model considering the probability of different product mix is given by Seifoddini [1990].

The approaches discussed so far consider only similarity based on the production requirements. The similarity between the parts in terms of their operation sequences rather than between machines has been discussed by Vakharia and Wemmerlöv [1990]. They consider the intra-cellular machine sequence and machine loads in the machine-part family formation process. The similarity coefficient based part family formation problem has also been reported by Kusiak and Cho [1992]. A detailed overview of the various similarity coefficient based clustering techniques has been given by Mosier [1989].

2.7.2.4 GRAPH THEORETICAL METHODS

In graph theoretical methods machines and components are considered as nodes and the machining of components as arcs connecting these nodes. The objective is to obtain disconnected sub-graphs from the machine-component graph, identifying component families and machining cells [Batra and Rajagopalan 1975].

After the introduction of the concept of graph theory by Batra and Rajagopalan [1975], Vannelli and Kumar [1986] used network decomposition heuristics. There is another non-heuristic network approach proposed by Vohra *et al.* [1990] to form part families. According to this approach, the machine-part matrix containing machining times is represented as a network which is subsequently partitioned by using a modified Gomory-Hu algorithm to minimise intercellular interactions.

Chandrasekharan and Rajagopalan [1986] divide the machines and parts into two mutually exclusive sets in a bipartite graph. Then the part family formation problem becomes that of finding a disconnected bipartite graph in which only the vertices of a group are connected to each other. In the extension work, their approach eliminates the arbitrariness of choosing ideal seeds which play the role of the centres of initial, imaginary and perfect graphs [Chandrasekharan and Rajagopalan 1987].

2.7.2.5 MATHEMATICAL PROGRAMMING METHODS

In this approach, a heuristic method derived from the mathematical programming methods can be included. Purcheck [1975] tried to solve the GT problems by constructing a combinatorial programming model. One of the mathematical techniques called p-median formulation is proposed by Kusiak [1985] to solve the part family formation problem. This technique minimises the total sum of distances between two parts, then the same technique is extended to deal with different Process Plans [Kusiak 1987]. Srinivasan *et al.* [1990] propose a model which overcomes some of the shortcomings of the p-median method. Kusiak and Heragu [1987] introduced another approach called Quadratic Programming Formulation to solve the problem of part family formation. Another algorithm using the subcontracting costs is proposed by Kusiak and Chow [1987]. The combinatorial method 'hosts combinations and guest combinations' and other mathematical and heuristic approaches were developed by Purcheck [1975a, 1985]. Choobineh [1988] also, approached the part grouping problem by using mathematical programming.

Xu and Wang [1990] and Li *et al.* [1986] have reported a part family formation method based on fuzzy mathematics in which the uncertainty inherent in the similarity measurement is emphasised in the family formation process. Chu and Hayya [1991] claim that parts belong to different part families with different degree of membership and they adapt a fuzzy c-means clustering algorithm to formulate the cell formation problem.

2.7.2.6 LINGUISTIC TECHNIQUES

A syntactic pattern recognition approach has been used for part family formation by Liu and Fu [1985]. With syntactic methods, a pattern is represented by a sentence (a string or a tree) in a language. The basis of this approach is to decompose a pattern into primitives according to certain "production rules". The use of a local grammar requires a description of the part families to be formed. This means that a priori knowledge of the number and structure of the part families is required.

The synthetic index (SI) algorithm introduced by Wu and Chang [1990] integrates machine similarity coefficients, component similarity coefficients, and the density indices of the machine-component matrix when forming Machine Cells for production processes. This algorithm also deals with bottlenecks and exceptional operations.

A formal language theory based syntactic pattern recognition technique used in part family formation has been reported by Wu *et al.* [1986]. Information on machine sequences is used in this approach. There are four steps involved in this approach:

(1) Primitive Selection
(2) Cluster Analysis
(3) Grammatic Inference
(4) Syntactic Recognition

2.8 ARTIFICIAL INTELLIGENCE AND EXPERT SYSTEMS

Artificial Intelligence (AI) is the discipline that aims to understand the nature of human intelligence through the construction of computer programs that imitate intelligent behaviour [Bonnet 1985]. Expert System technology has been referred to as a specific branch of Artificial Intelligence [Waterman 1986]. An Expert System is a knowledge based system that emulates expert thought to solve significant problems in a particular domain of expertise [Jackson 1986]. Its main emphasis is the construction of a computer program that can mimic the problem solving process of a domain expert in response to a situation that one is experienced with. Reviews of AI and Expert Systems in the context of manufacturing were conducted by Rayson [1985], Shaffer [1986], Beigel [1986], Roth *et al.* [1988], Chu and Wang [1988]. An interesting discussion of AI in the context of Process Planning and design has been taken by Cheung and Dowd [1988] and Dixon [1986] respectively.

There are two major reasons for using Expert Systems - firstly the knowledge of an expert who is going to retire soon can be stored so that this knowledge is not lost to the company forever. This can also apply if there is a strong possibility of experts changing jobs, particularly if they take a long time to train. A second reason for implementing Expert Systems is to lighten the load on the specialists - if an Expert

System can be created to solve the easier problems in a particular domain then the specialist will be left with the more demanding and rewarding problems to solve.

This research uses an Expert System shell. An Expert System shell is a built-in Expert System without knowledge. A user can develop his application using the shell without writing the system. The structure of an Expert System shell will be discussed in the following.

2.8.1 ARCHITECTURE OF AN EXPERT SYSTEM SHELL

An Expert System shell is a tool for building Expert Systems. Figure 2.2 illustrates a generic Expert System shell. The shell is made up of a number of components [Beynon 1993]:

1. **The knowledge base:** The repository of facts and rules that represent the domain-specific knowledge.
2. **The inference engine:** The driver of the system in the sense of making inferences from the knowledge base.
3. **The working memory:** A data area for storing intermediate results generated by problem -solving.
4. **Development tools:** Designed for use by knowledge engineers, these are tools for building and testing the knowledge base.
5. **User interface:** Allows end-users to run the Expert System and interact with it. One of the most important interactions is with the system's explanation facility. This enables the user to ask questions of the system, about how, for instance, the system came to a particular conclusion.
6. **Knowledge acquisition facility:** This is a set of facilities designed primarily to enable a domain expert to impart his expertise to the system directly, i.e. without the intervention of a knowledge engineer.

Further comprehensive details on the topic can be found in Simon [1969], Nilsson [1980], Barr and Feigenbaum [1981, 1982], Feigenbaum [1982], Forsyth [1984], Coombs [1984], Buchanan and Shortliffe [1984], Charniak and McDermott [1985], Harmon and King [1985], Jackson [1986, 1990], Milacic [1986], Beigel [1986], and Schutzer [1987].

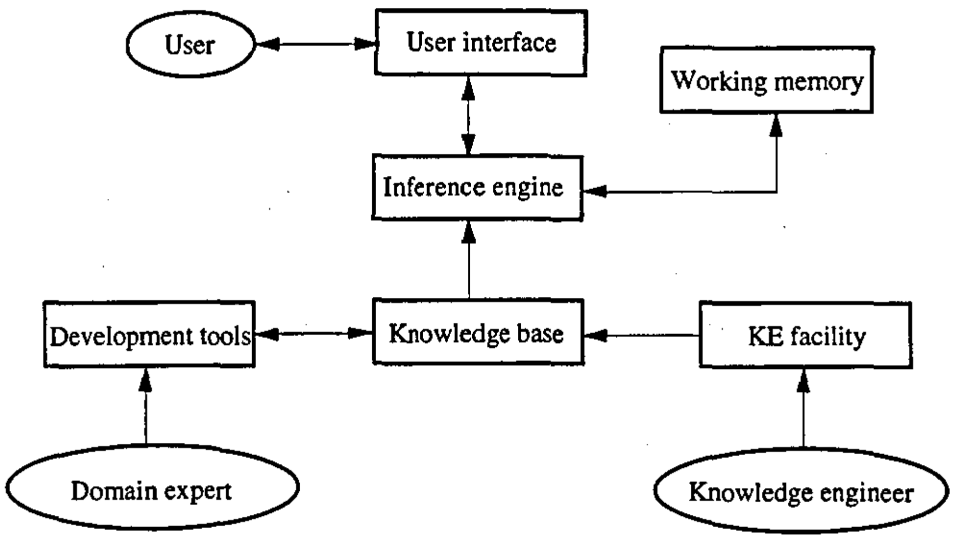


Figure 2.2 Architecture of a generic expert system shell.
Adapted from Beynon [1993]

The knowledge base and inference mechanisms constitute the kernel or core of the Expert System and thus are the essential parts of it.

A distinction can be made between a knowledge base and a database, the essential difference being that heuristic items often included in the former are usually absent or present only to a very limited extent in the latter. The knowledge base is a combination of rule base and database. The software tradition of:

Data + Algorithm = Program
is replaced by:
Knowledge + Inference = Expert System

2.8.1.1 THE INFERENCE ENGINE AND FORWARD/BACKWARD CHAINING

The inference engine is that part of the Expert System that performs the reasoning activities on the current knowledge base. There are two methods of reasoning which are most commonly used in Expert Systems with production rules - forward chaining and backward chaining [refer to section 2.9.5 for production rules].

Forward chaining is sometimes called data driven or top-down reasoning because the inferencing starts from the known data and reasons forward as far as possible with that data. To do this, one begins by testing each rule and firing every rule whose antecedent can be shown to be true. The system performs this process until no more rules can fire.

Backward chaining is also called goal-directed or bottom-up reasoning. As the name applies, we assume a hypothetical solution (the goal) which we will try to find evidence to prove. First, the knowledge base is searched for the rules, which might give us the desired conclusion - the goal appears in the conclusion of the rule. We search the rule antecedents to see what we need to fire the rule. If we can fire the rule, we have found our goal. If not, we backchain the rule we are working on and set-up a sub-goal of proving the antecedent of the rule. This process is continued until no more rules can be stacked.

2.9 KNOWLEDGE REPRESENTATION

Knowledge can be represented by using AI (Artificial Intelligence) programming languages like LISP and PROLOG etc. The following sections explain main knowledge representation techniques briefly [Harmon and King 1985, Doukidis and Whitley 1988].

2.9.1 LOGIC

Logic was one of the first knowledge representation schemes used in AI. The most fundamental notion is that of 'truth'. Predicate calculus is a commonly used procedure to represent basic knowledge.

2.9.2 PROCEDURAL REPRESENTATION

In a procedural representation, knowledge about the world is contained in procedures programs that know how to do specific things, and how to proceed in well-specified situations. The problem associated with this scheme is the difficulty in verifying and changing procedural representations if modification is necessary.

2.9.3 SEMANTIC NETWORKS REPRESENTATION

This representation consists of two parts, namely, objects (or concepts) and associations (or relations). It consists of points (circles) called nodes joined by arcs.

2.9.4 REPRESENTATION BY FRAMES

Frames (also called schemas or objects) is a type of knowledge representation that combines declarative and procedural knowledge representation.

Knowledge representation by a frames scheme uses a relational database model which resembles tabular arrays where a column within the table roughly corresponds to a data field and a row in the table roughly corresponds to a logical record. A frame is a record-like data structure used for representing a class of objects, a general concept, or a specific instance of any of these. A frame has a number of sub-structures called "slots" which contain a set of attributes and associated values: the name of a slot describes the purpose of the slot, and the value of the slot gives its state at a specific time; frames are organised into a hierarchy, with those at the upper levels representing more generic classes, and those at lower levels, being specialisation or instances of these classes.

The basic idea of a frame is rather like that of record in data processing. All the attributes of an object are collected together in a single composite structure. However, it is more than a passive data structure. Procedures can be invoked automatically if a value for an object is required and none is explicitly stored. The value of the frame can also be linked to other frames, rather like the element and subset links in a semantics network.

The major advantage of frames lies in the ability to represent structured data objects and relationships, augmented by procedures for default and inheritance reasoning. Most applications using frames have used them to complement other forms of knowledge representation, particularly production rules rather than to supplant them. Each frame contains many sub-entities (slots), and each sub-entity is assigned a value (filler).

2.9.5 REPRESENTATION BY PRODUCTION SYSTEM

A production system consists of three parts; namely, (1) a rule base, (2) a database, and (3) the inference engine as shown in figure 4.3.

The database contains symbols (e.g. facts, assertions). The contents of the database (or working memory) represent the state of the production system. The interpreter scans the condition parts of each rule until one is found that can be fired. The firing of rules changes the state of the database by adding or replacing symbols. Then the next cycle starts and the inference engine tries to find another rule that can be fired. The execution ends if no rules are applicable.

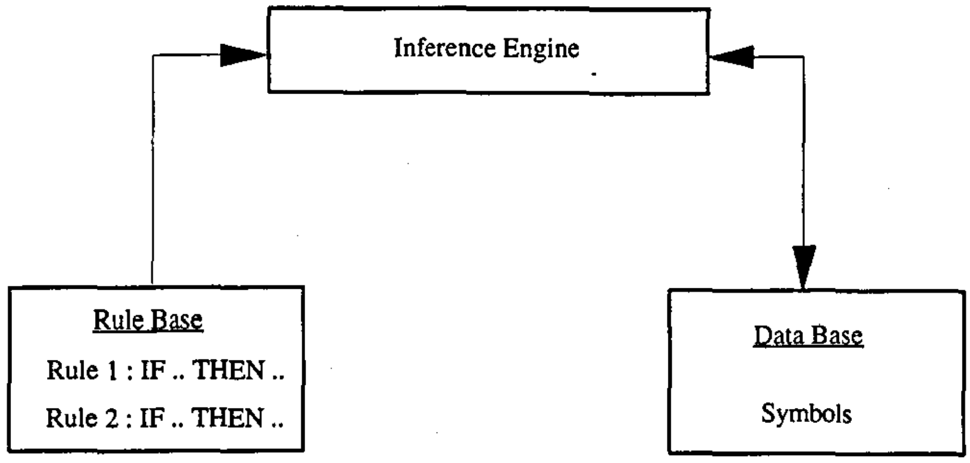


Figure 2.3 Knowledge Representation by Production System

2.10 DATABASES

Databases are used to store the data about the problem domain as the facts. Databases can be divided into three types: hierarchical, network, and relational. In hierarchical databases, data is structured in a layered organisation which proceeds from the general to the specific. Hierarchical structures resemble organisational charts with pyramid like structures containing many workers at the base level and progressing to the pinnacle occupied by the chief executive officer.

Networks have a branch-and-node organisation and resemble the pattern of interstate highways that connect major cities.

Relational databases resemble tabular arrays in which a column within the table roughly corresponds to a data field and a row in the table roughly corresponds to a logical record.

Figure 2.4 graphically illustrates the differences between these three types of database structures.

The advent of relational database technology and development has had a great impact on manufacturing technology. Relational databases are being used in nearly every field of manufacturing technology because of their efficient retrieval of information from the database. Such retrievals are accomplished by using higher-level, English-like, fourth-generation structured query language (SQL) in which the user need specify only the data and the results he or she desires [Nolen 1989].

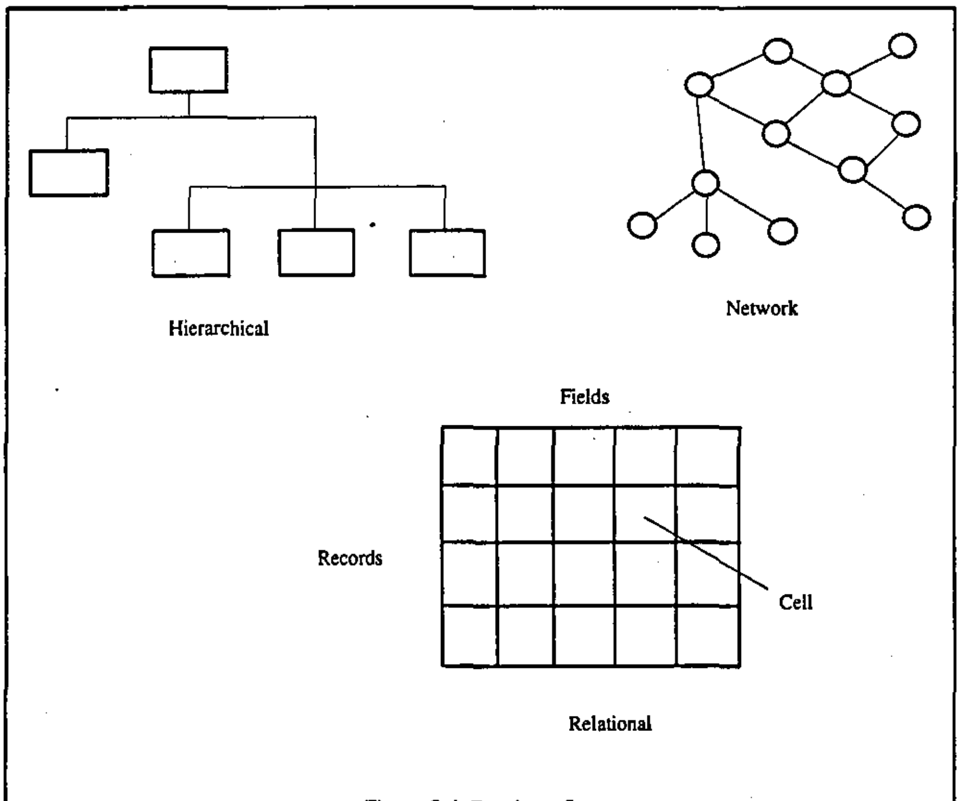


Figure 2.4 Database Structure

[Adapted from Nolen 1989]

2.11 PROCESS PLANNING

Process Planning is a critical activity in the link between design and manufacture [Wolfe 1985, Wang and Wysk 1987, Chang and Wysk 1985, Weill 1988, Hillyard 1978, Ham and Lu 1988, Li and Zhang 1989]. Process Planning is the act of preparing detailed work instructions to produce a part or in other terms, Process Planning involves the preparation of the detailed work instructions to produce a part. Process Planning includes [Ham and Lu 1988]:

- Selection of machine tools
- Selection of cutting tools
- Selection of set-ups
- Selection of machining operations and their sequences
- Selection of jigs and fixtures
- Calculations of cutting parameters
- Calculations of operation times

Other synonyms for the Process Planning are machine routing, material processing and manufacturing planning. The input to the Process Planning function is design information of the part to be manufactured, batch sizes or volume of production, and available equipment and tooling for production. Part design is the most important and provides the technical basis for the Process Planning. Therefore, part design information should be elaborate and unambiguous such that the planner can take right and effective technological decisions after analysing it. In the flow of information from design to manufacturing, Process Planning forms a 'bridge' between the two engineering functions.

At present, in part manufacturing three trends can be distinguished:

1. an increase in product mix,
2. an increase in the complexity of parts, due to an ongoing integration of physical functions in one part,
3. an increase in accuracy, caused by miniaturisation and the increased requirements for (automated) assembly.

These factors require faster, more accurate and more flexible Process Planning [Boerma and Kals 1989].

2.11.1 COMPUTER-AIDED PROCESS PLANNING

The number of required Process Planning personnel available today do not match the actual demand, hence many people in the industry have begun to use the computer to generate the plan [Chang and Wysk 1985]. Computer Aided Process Planning (CAPP) is a systematic determination of methods by which a component is to be manufactured economically and competitively with the aid of computers. Researches have reached the conclusion that CAPP is one of the cornerstones of Computer Integrated Manufacturing (CIM) initiative [Weill 1988, Li and Zhang 1989, Iwata and Sugimura 1987, Graves *et al.* 1988]. For these reasons, CAPP attracts a substantial amount of academic and industrial research. In the following section, we will be reviewing briefly the trends and directions that are emerging. Detailed information on the topic can be found in several publications [Ham and Lu 1988, Alting and Zhang 1989, Chu and Wang 1988].

The use of computers in Process Planning has brought a best process rationalisation by applying and producing consistent rules and plans resulting in increased productivity [Groover and Zimmer 1984]. The detailed advantages and disadvantages of planning systems i.e. traditional, Variant and Generative are given by Allen and Smith [1984]. As already said that CAPP systems fall largely into two categories namely, Variant and Generative. As Generative Process Planners are not yet a commercial reality, a third class of CAPP system is beginning to emerge that combines the concept of both Generative and Variant planners called a Hybrid system [Alting and Zhang 1989, Ham and Lu 1988]. A detailed discription of these systems will be taken in the following.

2.11.1.1 THE VARIANT APPROACH

The Variant method uses Group Technology (GT) techniques to create Process Plans by retrieving and modifying existing plans for similar parts which are grouped into families [Wang and Wysk 1987]. For each part family, a standard Process Plan, which includes all possible operations for the family is, stored in the system. Through classifications and coding, a code is built up by answering a number of predefined questions. These codes are often used to identify the part family and associated standard plan. The standard plan is retrieved and edited for the new part. The Variant approach is widely used.

The Variant system has two operational stages: a preparatory and production stage. The preparatory stage is when the parts are coded, classified and grouped into part families. The associated standard plans are stored in the database and indexed by family matrices. The production stage is when the system is ready to search for a family of parts and retrieve the standard plan. Further details about the above mentioned two stages can be found in Chang and Wysk [1985].

In comparison with manual performed Process Planning, the Variant approach is highly advantageous in increasing the information management capabilities. Consequently, complicated activities and decisions require less time and labour. Also, procedures can be standardised by incorporating a planner's manufacturing knowledge and structuring it to a company's specific needs. Therefore Variant systems can organise and store completed plans and manufacturing knowledge from which Process Plans can be quickly evaluated. Alting and Zhang [1988] identifies the problems in the sense that the quality of the Process Plan still depends on the knowledge and experience of both the engineer who created the master plan and any other personnel who made subsequent modifications, and this can lead to inconsistency problems.

THE VARIANT CAPP SYSTEMS

Many CAPP systems are reported but only three of the representatives systems have been selected for brief discussion.

The CAM-I CAPP automated Process Planning system (CAPP) is a Variant system, developed by McDonnell Douglas Automation Company (McAuto) under a contract from CAM-I in 1976. It is perhaps the most widely used Variant system. CAPP is a database management system written in ANSI standard FORTRAN. The code scheme for part classification and output format are added by the user. PI-CAPP which is an extension of CAPP has its own (built-in) coding and classification system.

MIPLAN and MULTICAPP are both Variant systems developed in conjunction with OIR (Organisation for Industrial Research, INC.) [Shaffer 1980]. Both use the MICLASS coding system for part description.

2.11.1.2 THE GENERATIVE APPROACH

Generative Process Planning is a second type of CAPP. The operating principal of Generative systems is based on the use of rules and algorithms to analyse the component geometry in order to determine its method of manufacture automatically. According to Chang and Wysk [1985] Generative Process Planning system can be defined as a system that synthesises process information in order to create a Process Plan for a new component automatically. In a Generative approach, Process Plans are generated by means of decision logics, formulae, technology algorithms, and geometry based data to perform uniquely the many processing decisions for converting a part from raw material to a finished state. The rules of manufacturing and the equipment capabilities are stored in a computer system. When using the system, a specific Process Plan for a specific part can be generated without any involvement of a Process Planner.

Form feature is the current approach being used in these systems as being the building blocks of the parts. Despite the early progress made by various research teams, Generative systems are still at the experimental stage of their life-cycle.

Research into Generative CAPP systems has proceeded on a number of broadly parallel fronts. Examples of AI based Process Planning systems are Descotte and Latombe [1981], Davies and Darbyshire [1884], Mouleeswaran [1984], Milacic [1985], Joshi *et al.* [1986], Hummel and Brooks [1986], Brooks *et al.* [1987], Marks [1987], Hinde and Alton [1987], Wang and Wysk [1986, 1987], Muthsam and Mayer [1990].

Chang and Wysk [1981], Matsushima *et al.* [1982], Choi *et al.* [1985], Iwata and Sugimura [1987], Wang and Wysk [1987], Graves *et al.* [1988], Joshi and Chang [1988], Joshi *et al.* [1988a] interface solid modelling with CAPP.

Reseach on the feature extraction/recognition front has been undertaken by Graves *et al.* [1988], Joshi and Chang [1988]. Work on process selection has been conducted by Nau and Chang [1980, 1983], Chang and Wysk [1981], Chang *et al.* [1988], Joshi and Chang [1988]. Process sequencing has been reported by Iwata and Sugimura [1987], Chang *et al.* [1988], Roy and Liu [1988] whereas front of operation sequencing was dealt with by the Matsushima *et al.* [1982], Iwata and Sugimura [1987], Detollenaere *et al.* [1988]. A comparison of statistical and AI approaches to the selection of process parameters has been made by Chryssolouris and Guillot [1990].

THE GENERATIVE CAPP SYSTEMS

Many CAPP systems are reported but only a few of the Generative systems representatives have been selected for brief discussion.

TIPPS (a new generation of APPAS and CADCAM) is an acronym for "Totally Integrated Process Planning System" developed by Chang and Wysk at Virginia Polytechnic Institute and State University in 1982. It is probably the first system that integrates CAD and Generative Process Planning into a unified system utilising the AI and decision tree approaches. A special language called PKI (Process Knowledge Information) is used to describe the procedural knowledge (process capabilities). A CAD boundary representation as data input is used by the system.

AUTAP is an acronym for 'Automatisch Arbeits Planerstellung' developed at Aachen Technical University West Germany. It is one of the most complete Generative Process Planning systems in use today. The system can handle different kinds of rotational parts like shafts, disks, rings, gear wheels, bearing caps, as well as sheet metal parts.

GARI is probably the first AI based CAPP system to be reported in the literature, developed by Descotte and Latcombe [1981] at the University of Grenoble France. Knowledge is represented by production rules dealing with conditions for the part being manufactured and advice representing technological and economical preferences. GARI is implemented in MACLISP language. GARI cannot create detailed operation scheme as needed in the manufacturing shop floor. A recent Expert System, PROPEL [Tsang 1987], however, improves upon some of the limitations of GARI.

TOM is an acronym for 'Technostructure of Machining' developed at the University of Tokyo in Japan in 1982. It is a rule-based CAPP system written in PASCAL.

TURBO-CAPP is one of the most complex intelligent Generative Process Planning systems up to date developed by Wang and Wysk [1987] at Penn. state university. A combination of both frame and rule-based methodology is used to build up the knowledge base of the system.

PART [Evre and Houten 1988] is a Generative computer aided process and operation planning system which is developed at the Laboratory of Production Engineering of the University of Twente, Netherlands. The PART system covers all the major process and operation planning functions for operations to be carried out on CNC lathes and machining centres.

2.11.1.3 THE HYBRID APPROACH

In the Hybrid approach, potentials of both the Generative and Variant systems are utilised. The Hybrid system is based on a Variant approach to get the benefits of Group Technology (GT). Group Technology is the realisation that many problems are similar, and by grouping similar problems, a single solution can be found. Based on this GT concept, components are first grouped based on their similarities in their manufacturing processes. The set of similar components can be called a production family. A composite component can be thought of as a representative component of a family which contains a solution for every feature. Keeping in view, the processing requirements of the composite component, a manufacturing cell can be designed where the entire family of the components can be processed.

When similar groups of the components are identified, the Generative Process Plans can be written for each family. Hybrid systems are most suitable for those applications in which there are few part families, but each family member has many variations. An example of a Generative system written for each part family is CPPP which requires user-supplied decision logic for each family [Kotler 1980a, 1980b].

THE HYBRID SYSTEMS

Many CAPP systems are reported but only a few of the Hybrid system representatives have been selected for brief discussion.

CPPP was developed by the united Technologies Research Centre. It was designed for planning cylindrical parts. The principle behind CPPP is a composite concept. A composite component consists of all the features of the components in one part family. By building a process model that contains the solution for every feature, components in the entire family can be planned. CPPP incorporates a special language, COPPL to describe the process model.

GENPLAN is an acronym for 'GENerative process PLANning system' developed at Lockheed-Georgia in 1981. It uses a Group Technology-based coding scheme and covers both part geometry and process variables to generate comprehensive operations.

XPS-1 is an acronym for 'eXperimental Planning System' developed by the United Technologies Research Centre (UTRC). Its first version was a Hybrid system. It is a prototype advanced Generative Process Planning system written in FORTRAN 77.

2.12 CELLULAR MANUFACTURING

A new type of manufacturing system has emerged in recent years to improve the productivity and efficiency in the manufacturing industry. This is achieved by replacing the traditional functional layout of machines by manufacturing cells, where machining requirements of part families will be fulfilled. A part family is a group of parts with similar manufacturing attributes achieved by applying any Group Technology (GT) approach. A GT analysis develops the families of parts which can be manufactured by a flexible, Cellular grouping of machines. GT is a manufacturing management philosophy which identifies and utilises the similarities of manufacturing characteristics which exist among the parts. According to Wemmerlöv and Hyer [1987] Cellular Manufacturing is a philosophy and innovation to improve manufacturing productivity and flexibility.

A term used by Burbidge for a manufacturing cell is 'group'. According to Burbidge [1979] a group is a combination of a set of workers and a set of machines and/or other facilities laid out in one reserved area, which is designed to complete a specified set of products. The workers in a group share a series of common output targets in terms of lists of products to be completed by a series of common due-dates.

Other definitions encountered in the literature for the topic are given in the following:

Black [1983] takes the manufacturing cell as a cluster or collection of machines to treat a specific group of parts.

To Pullen [1976] a manufacturing cell is a group of machines or processes of functionally dissimilar types that are physically placed together and dedicated to the manufacture of a specific range of parts.

The definition given by Tanner [1985] to Cellular Manufacturing is as a cluster or collection of machines designed and arranged to produce a specific group of component parts.

Nolen [1989] defines a manufacturing cell as a set of dedicated production resources designed to efficiently produce a product or family of products.

Warren and Moodie [1993] define the Cellular Manufacturing (CM) as the pursuit of small-batch production of discrete parts (part families) using machines grouped into cells.

Groups (Cellular systems) were developed by the engineers who were mainly interested [Burbidge 1979]:

1) in finding the methods which would reduce stocks and work-in-progress, reduce throughput times and reduce set-up times, thereby increasing capacity.
2) to increase workers motivation and job satisfaction.

Cellular Manufacturing is an organisational approach for producing parts with similar processing requirements ('part families') in machine groups or 'cells'. Each cell consists of dissimilar types of machines located close to one another and possesses specific manufacturing capabilities to process one or more part families. Cellular Manufacturing provides an opportunity to reduce set-up times, thus allowing manufacturers to reduce lot sizes, trim work-in-progress inventories, and shorten manufacturing lead times. In effect, Cellular Manufacturing provides a basis for Just-In-Time production. Ideally, a cell is a group of dissimilar machines physically located in close proximity such that a part is processed from start to finish in a single, continuous flow (i.e. without backtracking) [Ballakur and Steudel 1987].

A cell represents a hybrid production system, a mixture of a job shop producing a large variety of parts and a flow shop dedicated to mass production of one product.

Three approaches in cell design have been adapted by the researchers:

1) Identify machine groups and then assign parts to machines [Burbidge 1977, De Beer *et al.* 1976, De Witte 1980, McAuley 1972, Rajagopalan and Batra 1975, Boe and Cheng 1991].
2) Identify part families and then assign machines to part families [Opitz and Wiendahl 1971, Gettelman 1971, Carrie 1973, Hyer and Wemmerlöv 1985, Wemmerlöv and Hyer 1987, Choobineh 1988].
3) Identify part families and machine groups simultaneously [Burbidge 1963, 1971, 1973, 1975, 1977, McCormick *et al.* 1972, El-Essawy and Torrance 1972, King 1980, Gongaware and Ham 1981, King and Nakaornchai 1982, Chan and Milner 1982, Waghodekar and Sahu 1984].

Some of the reasons most often cited as motivations for Cellular Manufacturing are [Warren and Moodie 1993]:

- To improve competitiveness by reduction of lead times and throughput time (primarily reduction of times associated with movement of parts and material).
- To use modern production techniques and respond to pressure from customers to introduce productivity programs that are compatible with their own CM programs.
- To reduce high cost of work-in-progress (WIP) under an existing system, and desire to reduce WIP, achieved primarily by the introduction of reduced batch size policies.
- To increase plant capacity by reducing the time required for set-ups.

- To counter erratic delivery performance (primarily by simplifying operational management issues so that more delivery performance visibility is obtained) via improved delegation of responsibility and assignment of accountability.
- To become more cost-effective (via improved product quantity), and increase worker productivity.
- To reduce of material and part transportation distances (and consequently product damage from handling).
- To prepare for automation.

Burbidge [1980] discusses the other benefits of cell design. Black [1983] quotes that set-up times reduction is possible in these cells and smaller lot sizes are achievable. Schonberger [1983] stresses that cell design being product oriented, makes just-in-time production possible. Several case studies describe the costs and benefits of cellular design are described in Hyer and King [1984], Hyer and Wemmerlöv [1984], Dumolin and Santen [1983].

Thorough reviews of the cell (GT) formation problem can be found in Chu [1989], King and Nakornchai [1982], Kusiak [1987], Waghodekar and Sahu [1984], Bedworth *et al.* [1993], Tanner [1985], Warren and Moodie [1993], Groover [1987], Nolen [1989], Edwards and Koenigsberger [1973], Hitomi [1979], Ham *et al.* [1985], Arn [1975], Burbidge [1963, 1964, 1971, 1975, 1979, 1988, 1989], Gallagher and Knight [1973, 1986], Hitomi [1979], Hyer [1987], Kusiak [1988], Ranson [1972], Snead [1989], and Wemmerlöv and Hyer [1986].

2.12.1 TYPES OF CELL

Investigators like Kerr and Mitrofanov began to develop the technique of grouping machines (in the beginning) to manufacture a family of parts that have common manufacturing characteristics. Their idea was to use common manufacturing characteristics to develop tooling and set-ups which would permit the removal of one part type from a machine and the loading of a new part type with little or no change in the tooling and set-up [Snead 1989].

A 'virtual cell' approach was used in the beginning. A virtual cell is not a physically constructed grouping of machine resources, rather is an identified path to select machines that part families will flow through. The virtual cell equipment usually, is characterised by differing colours of paint. The approach could not gain popularity because of complexity in scheduling and greater material handling as long distance was to be travelled by the parts same as in job-shops [Snead 1989].

Tanner [1985] categorises the cells into two groups:

1) manned cells, and
2) unmanned cells.

2.12.1.1 MANNED CELLS

Manned cells are operated by versatile operators who are trained and are skilled in the operation of more than one machine resource in a cell. These types of cells prove to be effective as the number of operators can be adjusted and minimised to meet the requirements. All the machines conventional, numerical controlled (NC), computer numerical controlled (CNC), machining centres, can be accommodated in these cells. Burbidge [1975] states that manned cells should contain between 6 and 15 machines.

2.12.1.2 UNMANNED CELLS

These types of manufacturing cells are run by automated equipment with few workers needed in the cells. By mechanising and automating the materials handling and the manufacturing processes, unmanned cells can be created. A robotic work cell designed to process a small set of part families, for example, consists of computer controlled machine tools located around one or more materials handling robots. Since there is no fixed machine sequence, this type of cell can be quite flexible. Burbidge [1975] quotes that as compared to the manned cells, automated cells may have smaller number of machines due to both hardware and software costs and the need for sophisticated control and material handling systems. These types of cells can further be classified by type of arrangements.

FIXED AUTOMATED

Such cells exhibit the automatic transfer line mechanism in which the volume of production is large and production runs long. These systems usually contain automated conveyor system which locates the parts and transports them from different machine resources as well. The line, in this case, is balanced so that the part spends the same amount of time at each production station.

FLEXIBLE AUTOMATED

These types of cells work on the concept of flexible manufacturing system (FMS) and the robotics cell. These cells are equipped with direct numerical control (DNC) and computer numerical control (CNC) machines, robots, automated guided vehicles (AGVs) and computer-controlled conveyor systems. Computer-controlled conveyor systems are to transport the parts to any machine, in any order. DNC and CNC machines can change tools and programs to deal with variety of components. This type of system can handle a part family with medium to large lot sizes.

Different types of cells has also been discussed by Black [1983] and Vakharia [1986].

CHAPTER 3

FEATURE-BASED COMPONENT DATA MODEL

3.1 INTRODUCTION

Following the literature survey presented in chapter 2, this chapter sets the context of this research in the area of feature-based part family formation problems for the Process Planning function.

Section 3.2 describes the importance of features and their role in the integration of CAD/CAM. According to recent approaches, features are considered as the constituent elements of the parts which are considered more suitable for reasoning for the whole community working in the whole spectrum of Computer Integrated Manufacturing (CIM). Definition of a feature by different researchers has been given in the previous chapter under the heading 'Feature Technology'.

After discussing different form feature taxonomies in chapter 2, section 3.3 of this chapter deals with Gindy's form feature taxonomy in detail as this taxonomy is being used in this research for the definition and classification of features.

In section 3.4, feature connectivity has been detailed which is how the features (constituent entities of the part) are connected together to give structure to the part. This structural aspect of the part is an important factor in determining the machining resources, machining set-ups and sequences of the operations. Two approaches for the determination of feature connectivity in the component have been described here.

Section 3.5 details the contribution of this research towards the development of a feature-based component data model. Knowledge representation has been described in section 3.6 and finally software implementation in the "GENERIS" (Expert System) has been presented in section 3.7.

3.2 ROLE OF FEATURE TECHNOLOGY IN THE INTEGRATION OF CAD/CAM

The use of 'features' is a recent approach to integrate CAD with CAM. According to Brimson and Downey [1986], describing components by the use of features is seen by many as the key to genuine integration of the many aspects of design and planning of manufacture, particularly in a modern computer-controlled environment incorporating Computer Aided Design (CAD) and Computer Aided Process Planning (CAPP). On the design side, features are being used for the functional requirements of the parts, geometric modelling and design analysis. In the downstream planning activities like Process Planning, Inspection Planning, Assembly Planning, manufacturing Operations Planning and NC Part Programming, the description of component can be feature-based. This is not all, a concept of features is beneficial in the whole spectrum of Computer Integrated Manufacturing (CIM).

Design dictates functional objectives of a component. Design comprises not only the overall shape but also the details of the component - regions of interest such as fixing holes, cooling slots, bearing holes, etc. These regions of interest are termed as 'functional features'. The words component and part are used interchangeably in this work. In the modern CAD environment a designer is able to add geometric details to a drawing by simply obtaining the geometry of commonly used shapes from a library of features. In other words, with such types of CAD facilities, the part geometry can be modelled or described by adding or deleting from predefined set of features. The next stage of this feature-based geometric model will be a representation of the part to be manufactured.

The job of the Process Planner is to specify how the part is to be manufactured, which is normally based on the 'manufacturing features' of the part. In most cases, the manufacturing features are the same as the design features, an example being a 'fixing hole' for the designer and a 'through drilled hole' or 'bored hole' for the manufacturer.

With the advent of feature approach Designer/Process Planners are constrained to work with a set of features which have importance or significance for both the communities working in the areas of design and manufacturing. Instead of using a model consisting of graphics primitives (e.g. lines, points), the designer is using a set of features (e.g. holes, slots) from which manufacturing operation can be specified. Recently, the feature-based modelling concept is gaining popularity among designers for the purpose of integration between CAD and CAM. Butterfield *et al.* [1985] recognised the importance of features as means of integration of CAD/CAM.

Features represent a higher conceptual level than the lines, arcs, and text used by current CAD/CAM systems because they inherently contain more information. That higher level information is represented by functional and manufacturing descriptions which have unique meaning and includes the basic geometry. By organising geometry information into a hierarchy of functional or manufacturing objects, engineers can utilise the stored parameters for a variety of design and manufacturing activities. Features are considered to be the communications medium between design and manufacturing [Butterfield *et al.* 1985].

Pratt and Wilson [1985] in their work which was jointly funded by the CAM-I Advanced Numerical Control, Geometric Modelling and Process Planning Programs, argue that form features provide one of the essential links between design and manufacturing in a fully automated CAD/CAM environment.

Research into the use of features is now a mature body of work, but there are still considerable divisions as to the particular features approach to be adopted and their use in particular fields of application. Attempts to define the precise nature of features are fraught with difficulty because of the wide interpretations placed upon the term by different researchers. If features are to be the bridge between the design and manufacturing planning then a unifying 'Feature Representation' is required that meets the needs of every body without compromising their objectives [Case and Gao 1993].

3.3 FORM FEATURE TAXONOMY

Features are classified into different categories based on their geometric and topological attributes. The features classification helps engineers to characterise the groups of features which have common attributes depending on applications. Usually, features are classified in a hierarchical form. The hierarchical structure enables objects to inherit data slots and attributes from other objects located above in the hierarchy; a class can receive some information from its super-classes. The hierarchical structure provides a large amount of information in a small number of nodes as the interpretation of each succeeding node is dependent on the information given on the preceding node. This classification structure is called a feature taxonomy.

The taxonomy of features is a descriptive generalisation of a class, and associated subclasses of features. It also specifies how to represent the feature and what parameters are to be used.

The purpose of feature taxonomy is to group the features into a finite number of classes such that a set of rules can be more easily developed for manipulating these features in a computer environment. Feature taxonomy schemes help to structure information which make the planning task of design and manufacturing easier. Brimson and Downey [1986] feeling the need for the feature taxonomy argue that a hierarchy of features must be defined to obtain a description of relationships between separate features.

The taxonomy should be independent of the 'intent' of the user and considered to correspond ideally, to both the designer's geometric view of features and to the manufacturing engineer to write the Process Plans.

A major advantage of a feature taxonomy is the structured way in which it can be used to classify features, and the reduction in execution time for feature recognition. Another advantage arises from the notion of inheritance. Since the instance of a particular subclass is also an instance of its super-class, the properties of the super-class can be inherited by the subclass without explicitly being repeated. Thus a more compact representation of knowledge is achieved. [Joshi and Chang 1988].

There should be a system which permits features to be identified and the data about them catalogued. This must be done independent of the processes which are currently used to produce them. A predetermined, standard data structure that is based on geometric types and parameters is of prime importance to the organisation of features [Butterfield *et al.* 1985].

A significant study on form features was conducted by Gindy [1989]. Gindy's Feature Taxonomy is being used in this research for the features representation and their classification. This taxonomy has been selected as it is considered more suitable for Process Planning. A brief discussion of the scheme has already been undertaken in section 2.5.2 and a detailed description of the taxonomy is given in the following.

3.3.1 ENTRY, EXIT AND DEPTH BOUNDARIES

A feature is taken as a bounded volume having an entry boundary, an exit boundary and a depth boundary. Entry and exit boundaries are to define the perimeters of real or imaginary surfaces. Real surfaces are actual faces of the solid, while the imaginary surfaces are to complete the faces of any volume under consideration. Perimeters of a boundary are either closed or open. A feature boundary is taken to be closed if its perimeter geometry represents a continuous closed loop and when this is not the case, the boundary is considered to be open.

In figure 3.1, the feature present on the solid part is a rectangular notch. A possible entry boundary for the notch is an imaginary surface, defined as enveloped by the lines joining the vertices 1, 2, 3 and 4. Similarly, a possible exit boundary is a real surface, enveloped by the lines joining the vertices 5, 6, 7 and 8. There are other possible pairs of entry/exit boundaries, the surfaces enveloped by the lines joining the vertices (1, 4, 8, 5 and 2, 3, 7, 6) and (4, 3, 7, 8 and 1, 2, 6, 5) respectively.

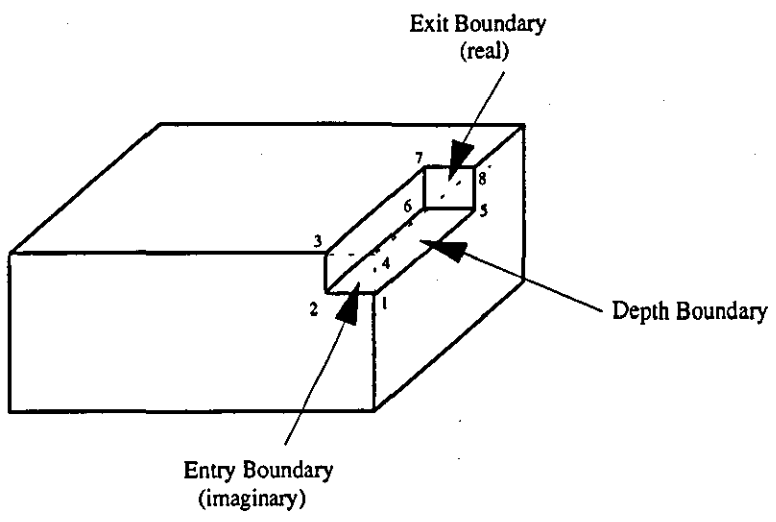


Figure 3.1. Form feature geometry.

A depth boundary represents the progress of form between the entry and exit boundaries along a depth axis. A depth axis is any arbitrary line between any chosen pair of entry and exit boundaries of the feature. A depth boundary of a feature can be described by considering the following four parameters:

(a) Type of depth axis.
(b) Symmetry with respect to depth axis.
(c) The relationship between entry and exit perimeter geometry.
(d) The type of form variation along the depth axis.

Figure 3.2. further illustrates the above mentioned four parameters.

3.3.2 EXTERNAL ACCESS DIRECTIONS (EAD)

External Access Directions (EAD) are the possible directions which can be used to machine the feature. An EAD is an imaginary surface of the feature through which a cutting tool can pass to machine it. The number of EAD's depends on the type of the feature. A "Real Surface" has five EAD's as a tool can approach it from five directions to cut it. While a Boss has no EAD'S as it does not have any imaginary surface. A

Boss is machined by removing its surrounding volume. Figure 3.3 shows the definition of some common features along with their EAD's.

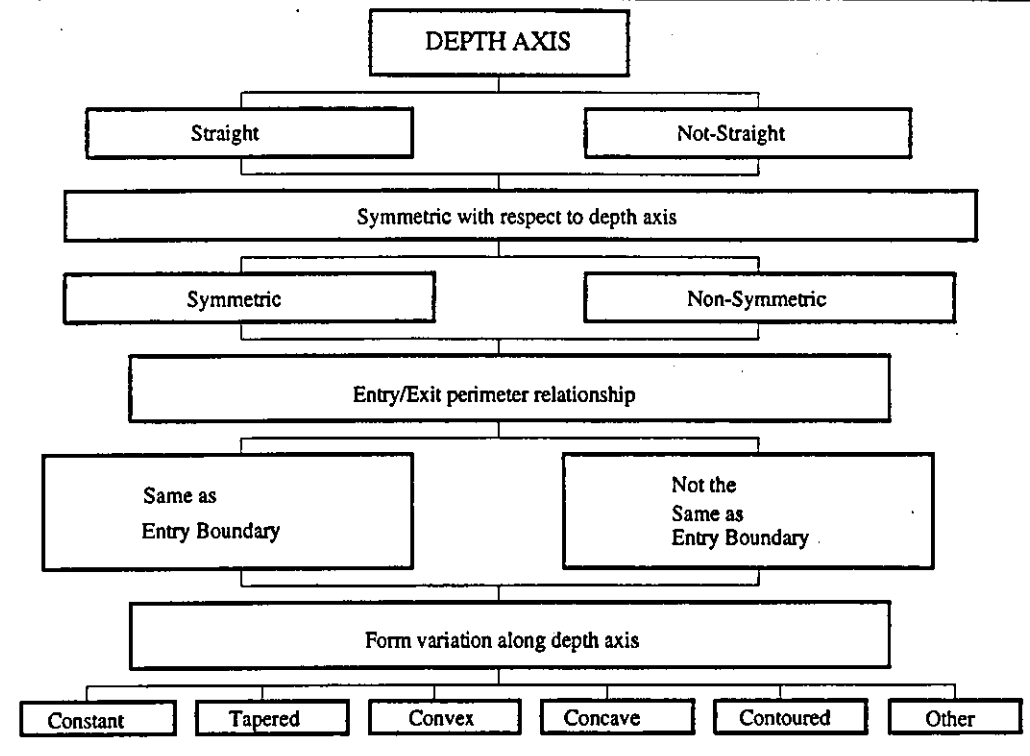

DEPTH AXIS
Straight
Not-Straight
Symmetric with respect to depth axis
Symmetric
Non-Symmetric
Entry/Exit perimeter relationship
Same as Entry Boundary
Not the Same as Entry Boundary
Form variation along depth axis
Constant
Tapered
Convex
Concave
Contoured
Other

Figure 3.2. Depth Boundary variation.

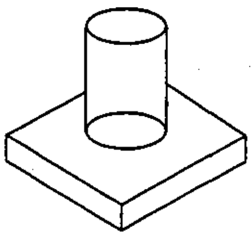
3.3.3 EXIT BOUNDARY STATUS

The exit boundary status determines whether the feature is 'through' or 'not through'. A feature is considered to be 'through' if on the opposite side, there is also an imaginary surface (EAD) while moving along its depth axis.

A feature is 'not through' when it does not have two opposing external access directions (EAD's) along its depth axis. The exit boundary status in the case of some primary features is given in figure 3.3.

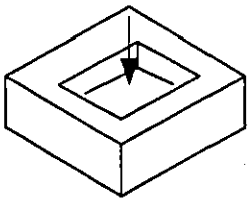
BOSS

Category	: Protrusion
Entry Surface	: Real
Exit Surface	: Real
Boundary	: Closed
Exit Status	: Not-Through
EAD's	: None



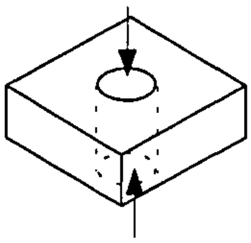
POCKET

Category	: Depression
Entry Surface	: Imaginary
Exit Surface	: Real
Boundary	: Closed
Exit Status	: Not Through
EAD's	: One



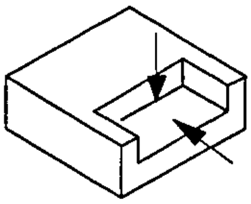
HOLE

Category	: Depression
Entry Surface	: Imaginary
Exit Surface	: Imaginary
Boundary	: Closed
Exit Status	: Through
EAD's	: Two



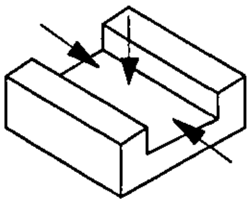
SLOT (NOT THROUGH)

Category	: Depression
Entry Surface	: Imaginary
Exit Surface	: Real
Boundary	: Open
Exit Status	: Not Through
EAD's	: Two



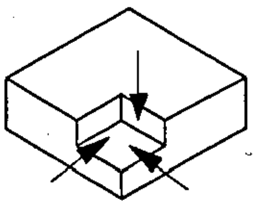
SLOT (THROUGH)

Category	: Depression
Entry Surface	: Imaginary
Exit Surface	: Imaginary
Boundary	: Open
Exit Status	: Through
EAD's	: Three



NOTCH

Category	: Depression
Entry Surface	: Imaginary
Exit Surface	: Real
Boundary	: Open
Exit Status	: Not Through
EAD's	: Three



STEP

Category	: Depression
Entry Surface	: Imaginary
Exit Surface	: Imaginary
Boundary	: Open
Exit Status	: Through
EAD's	: Four

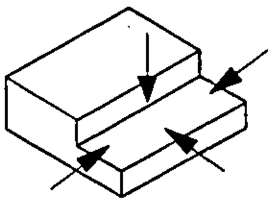


Figure 3.3. Definition of some primary features.

3.3.4 FORM FEATURES CATEGORIES

According to this taxonomy, form features are first divided into three categories, namely protrusions, depressions, and surfaces.

Protrusions are termed positive in solid modelling terms. Protrusions are always external features of the solid and always have closed boundaries.

Depressions are taken as negative in solid modelling terms. In the case of depressions material is to be removed to generate the features. Depressions can have both boundary types i.e. closed or open.

Surfaces are without depth. In other words configuration of an entry and exit to a volume coincides in the surfaces. Surfaces are taken as real when the inside boundary is solid, and imaginary, when the boundary is enveloping an empty area.

3.3.5 FORM FEATURES CLASSES AND SUB-CLASSES

After dividing the features into categories, features are further classified into classes. This division into classes is decided upon by their:

(a) external access directions
(b) boundary type (open, closed)
(c) exit boundary status

Features at this stage are called 'primary features'. Primary features are Boss, Pocket, Hole, Not through Slot, Through Slot, Notch, Step, Real and Imaginary Surfaces as shown in figure 3.4.

A feature subclass is characterised by identifying its perimeter geometry. A feature at this level is a recognisable entity such as Round Hole, Square Boss, etc. For example, let us consider the notch shown in figure 3.1. By answering the questions relating to: external access directions, the type of boundary and exit boundary status and perimeter geometry, the feature can be characterised as a 'Rectangular Notch' as shown below:

Category:	Depression
Class:	Notch
Sub-class:	Rectangular Notch
Depth boundary:	Defaults: straight, symmetric, same, constant

Some common geometric perimeters which are used to define the feature subclass are given in figure 3.5.

3.3.6 SECONDARY FORM CLASSES AND SUB-CLASSES

Secondary form is a local geometry which normally does not exist in its own right but is superimposed on the body of the basic feature. Threads, Keyways are examples of secondary forms which cannot exist on their own but are superimposed on the basic features. Secondary forms are also divided into classes and subclasses. Classes represent the major divisions between the various forms while subclasses are further divisions within secondary classes. Examples of secondary forms classes and subclasses are shown in figure 3.6. An overview of form feature classification is given in figure 3.7.

3.4 FEATURE CONNECTIVITY

The workpiece geometry not only comprises feature information, but also, geometric and technological information relating to different features, sometimes called "feature relationships". Geometric relationships model the interference between features. They comprise parent-child relationships and relationships generated through the geometric tolerances between different features. Another geometric relationship could relate to feature patterns. Feature relationships are important while determining machining resources, component set-ups as well as operation sequences in the Process Planning function. Feature relationships can also influence the selection of processing resources needed to machine the component.

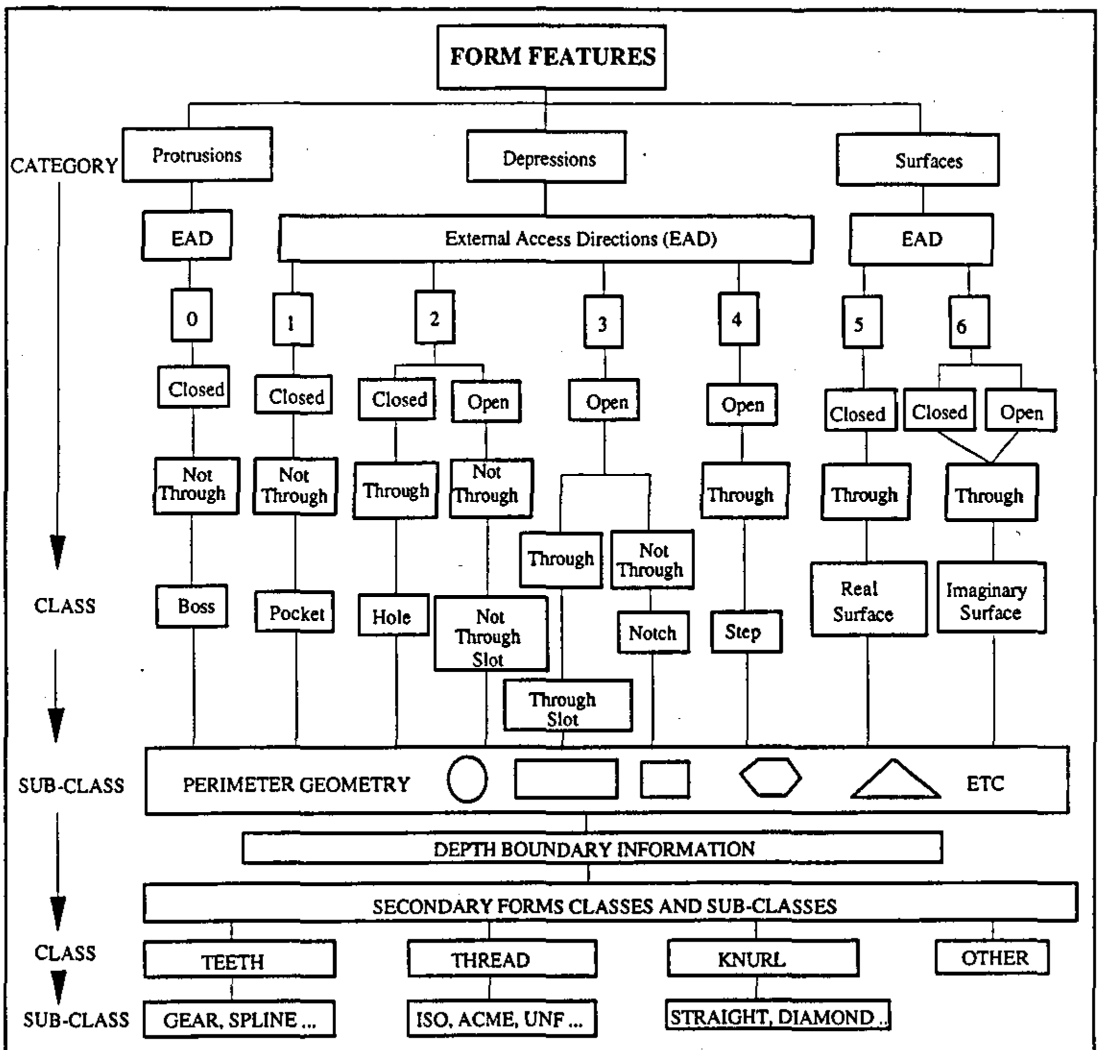


Figure 3.4. Form feature classification.

The terminology used by Gindy *et al.* [1993] for the structural aspect - how the part features are connected together to form the component, is termed as 'feature connectivity'. Feature connectivity has more significance in the determination of processing requirements, component set-ups and operation sequences.

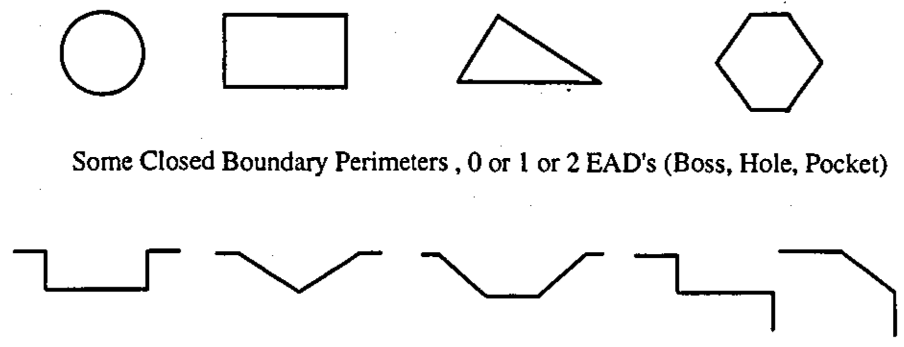

Some Closed Boundary Perimeters , 0 or 1 or 2 EAD's (Boss, Hole, Pocket)
Some Open Boundary Perimeters , 2 or 3 or 4 EAD's (Slots, Notch, Step)

Figure 3.5. Feature perimeter geometry.

Feature connectivity determines how the component is constructed from its constituent features. This information is necessary for determining the component processing needs and the downstream function of Process Planning. Features which have geometric relations in common and have the same manufacturing directions (Approach Directions) are grouped together as a set of features for the purpose of machining them together in the Process Planning function.

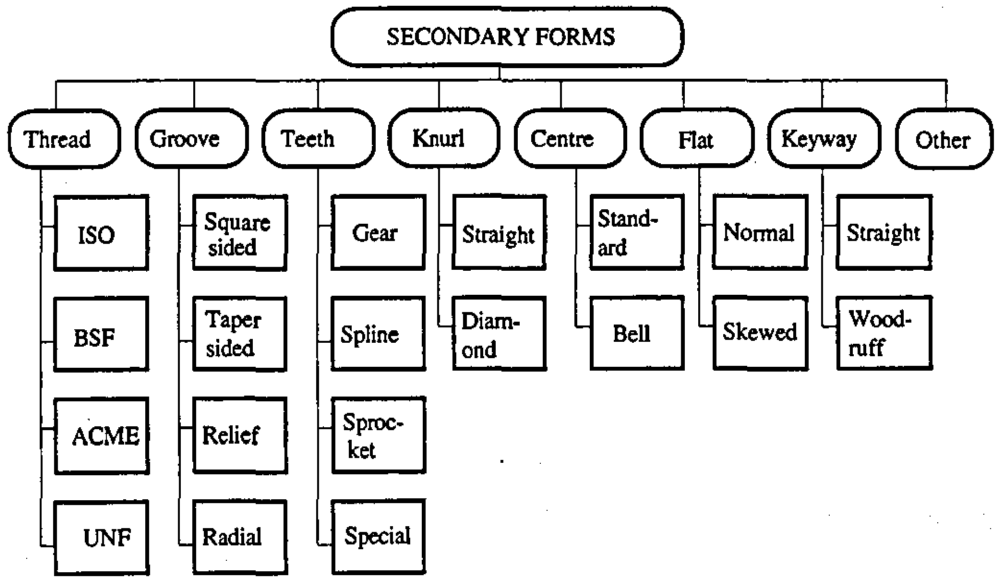


Figure 3.6. Secondary forms classes and subclasses.

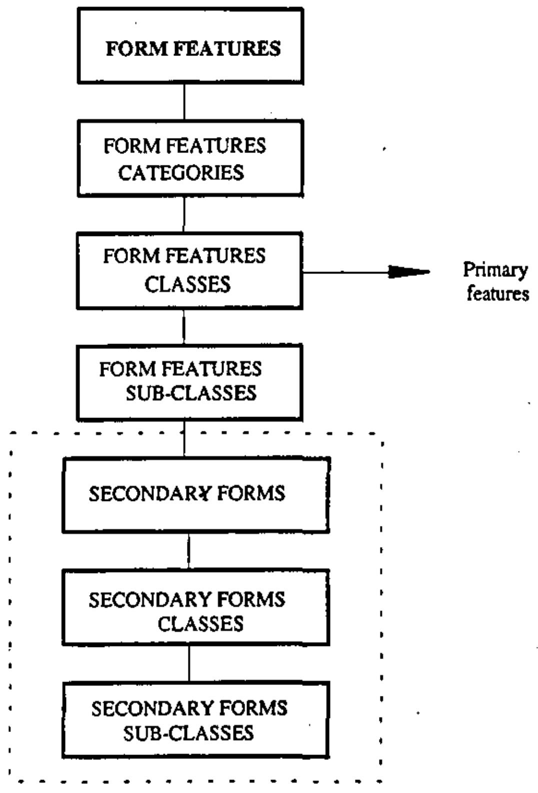


Figure 3.7. An overview of form feature classification.

Feature connectivity identifies the parent-child relationship which exists among component features. Feature connectivity is determined with the help of EAD's. Each feature has a fixed number of EAD's fixed at its primary class level. An EAD of a feature as described elsewhere in the thesis, is an imaginary surface through which a feature can be accessed by the tool to machine the feature. The first feature through which a second feature is accessed by the tool is considered to be the parent feature while the second feature is termed as the child feature. The parent feature will be machined before the child feature thus giving the sequence information. Connectivity can be made to imply the global information for machining the features as well as the number of sets-up for the machining of the component.

The feature connectivity aspects are represented by two types of link: external access direction links for relating individual features to the basic component directions, and inheritance links that relate adjacent features, with some features becoming parents to other features. The inheritance tree at the component level has its origin in the component free surfaces. A free surface is the parent of all the features that have imaginary surfaces within its boundary. Any child feature becomes a parent feature of the adjacent features which has imaginary surfaces contained within the boundaries of any of its real surfaces [Gindy and Ratchev 1991].

Potential Access Directions (PAD's) are common for features with EAD normals which are co-incident. It is worth noting that not every EAD of the feature is a feasible access direction for the feature. Interactions among the features also need to be taken into account. It is, therefore, inevitably necessary to reason about the features which EAD's are feasible to be considered as PAD's.

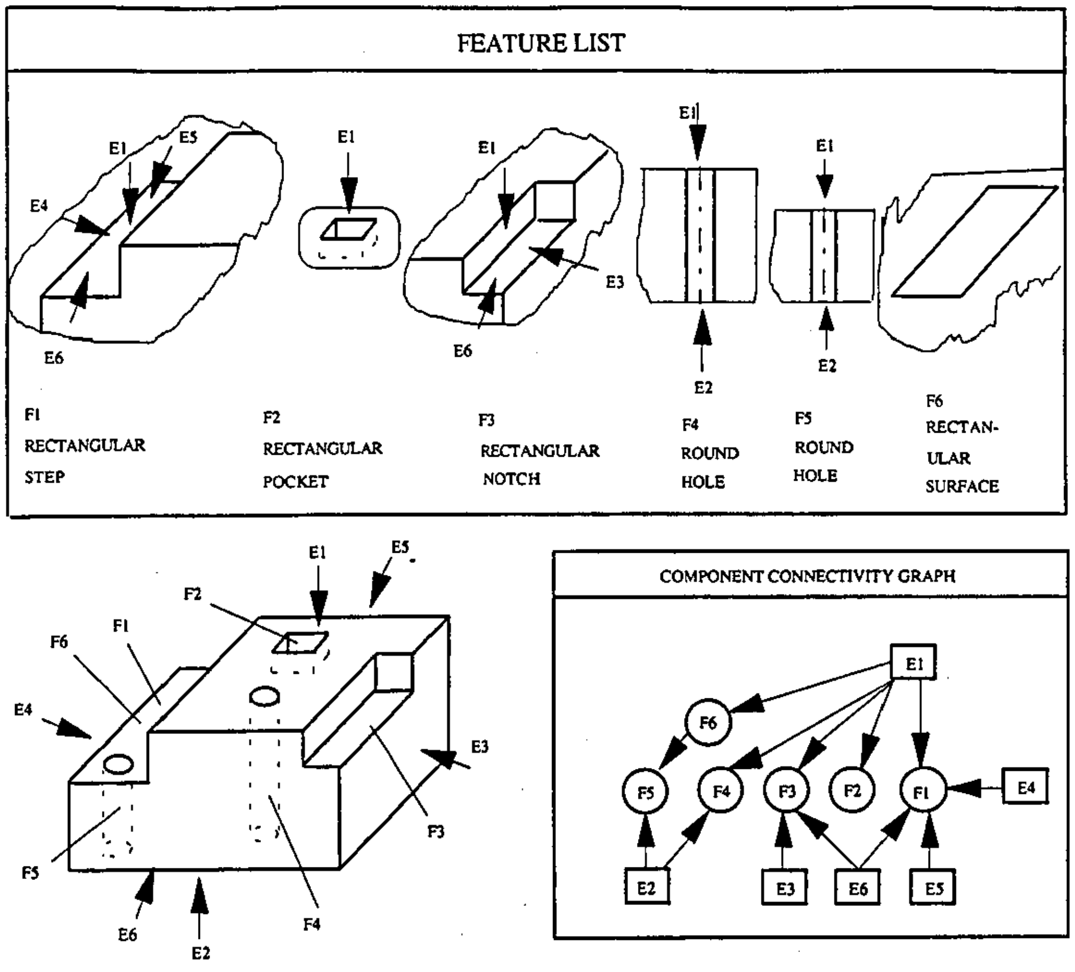
3.4.1 FEATURE CONNECTIVITY REPRESENTATION

Feature connectivity can be represented by a connectivity graph. The technique is described below.

3.4.1.1 CONNECTIVITY GRAPH

Component connectivity information can be represented as a directed graph as shown in figure 3.8. The arrows are from parent to child. Not necessarily every connection is a parent-child relationship but gives the potential access directions (PAD's) through which a feature can be accessed. After reasoning out the feasibility of all the EAD's as well as considering the interference among the features, the component PAD can be optimised and the Approach Directions (AD's) determined.

Six features on the component have been coded and represented each with connectivity information as shown in figure 3.8. Component connectivity as a directed graph is shown. Furthermore, the features connected with each AD are shown in the figure. In the example the AD for all six features is E1.



FEATURES AND POTENTIAL APPROACH DIRECTIONS (PAD's)

F1, F2, F3 F4, F5, F6	APPROACH DIRECTION E1
F4, F5	APPROACH DIRECTION E2
F3	APPROACH DIRECTION E3
F1	APPROACH DIRECTION E4
F1	APPROACH DIRECTION E5
F1, F3	APPROACH DIRECTION E6

Figure 3.8. Component connectivity and potantial approach directions (PAD's)

3.5 FEATURE-BASED COMPONENT DATA MODEL

Feature-based component data modelling is the current trend to represent a part under consideration, in the areas of design, manufacturing and production planning. Features are considered as a key to Process Planning for two reasons. Firstly, features are a natural form of communication (Process Planners think in terms of holes and pockets etc.). Secondly, there are only a finite number of ways to manufacture any feature. The knowledge of a feature-based component data model represented in a knowledge based system can be used for generating a plan by applying the logic used by Process Planners, as well as extracting the required attributes, after reasoning out for part family formation purposes.

Artificial Intelligence (AI) based techniques are designed for capturing, representing, organising, and utilising knowledge on computers, and hence will be the key technology for intelligent and integrated planning in the future [Ham and Lu 1988]. An Expert System is a branch of AI. Currently, Expert Systems are being used to convey the design information of the component. Joshi *et al.* [1988a] argue that Expert Systems provide an excellent framework for incorporating the decision-making process of the planner and making it suitable for automation.

The following section describes some of the information generally found on the component drawing.

3.5.1 COMPONENT DESIGN INFORMATION

The component design information generally consists of the following:

- geometry, topology and dimensions
- general tolerances and surface finish
- information about component features
- reference surfaces, datum faces etc.
- feature relationships (geometric tolerances, patterns, compound features, parent-child relationships)
- connectivity information

- EAD's information (from which direction the feature can possibly be machined)
- general specifications (material, hardness etc.)
- some special requirements (heat treatment, batch sizes, special requirements, other notes etc.) and
- management information (component identity, component description, drawn by, revision, etc.)

3.5.2 COMPONENT DATA MODEL

A feature-based component data model which conveys the design information of the component has been implemented in the knowledge base of the Expert System. The data model is used for the decision-making process for extraction of classifying attributes for grouping the similar components, cell design, and generating the Process Planning system.

A feature is considered to be a local geometric entity. For the Process Planning activity, information at local level (features) as well as the global level (component) must be available. Therefore component information can be divided into two levels: (1) component level information, and (2) feature level information.

Component level information consists of the following:

- management information (component identity, component description, drawn by, revision, etc.)
- general specifications (component block size, material, hardness, roughness etc.)
- Shape (proportions, complexity etc.)
- feature list
- feature relationships (features pattern, interference, geometrical tolerances e.g. parallelism, perpendicularity etc.)
- connectivity information (EAD's information leading to parent/child relationships, how the features are connected to the component PAD's and tool approach directions).

and feature level information can be described as below:

- feature type
- feature parameters
- linear tolerances, surface finish or accuracy of the feature
- face list, face identity and type (real or imaginary)
- form tolerances (at feature level e.g. straightness, flatness, cylindricity, etc.)
- geometric tolerances (at feature level i.e. parallelism, perpendicularity, etc.)
- location of the feature
- secondary features (superimposed on the body of feature) along with their parameters.

The feature-based component data model used in this work is shown in figure 3.9.

3.6 KNOWLEDGE REPRESENTATION

Knowledge representation for the components should be comprehensive and unambiguous such that the complete information is available for the downstream functions like, Process Planning, part family formation and cell design etc. AI-based techniques are designed for capturing, representing, organising, and utilising knowledge on computers, and hence will be the key technology for intelligent and integrated planning in the future [Ham and Lu 1988].

Performing tasks which require specialist knowledge can only be performed successfully by appropriate experts. Hence software programs which perform these tasks have become known as Expert Systems. The most widely used technique for knowledge representation in Expert Systems is the use of a 'production rules based system' [Wang and Wysk 1988b]. A production rulebased system consists of:

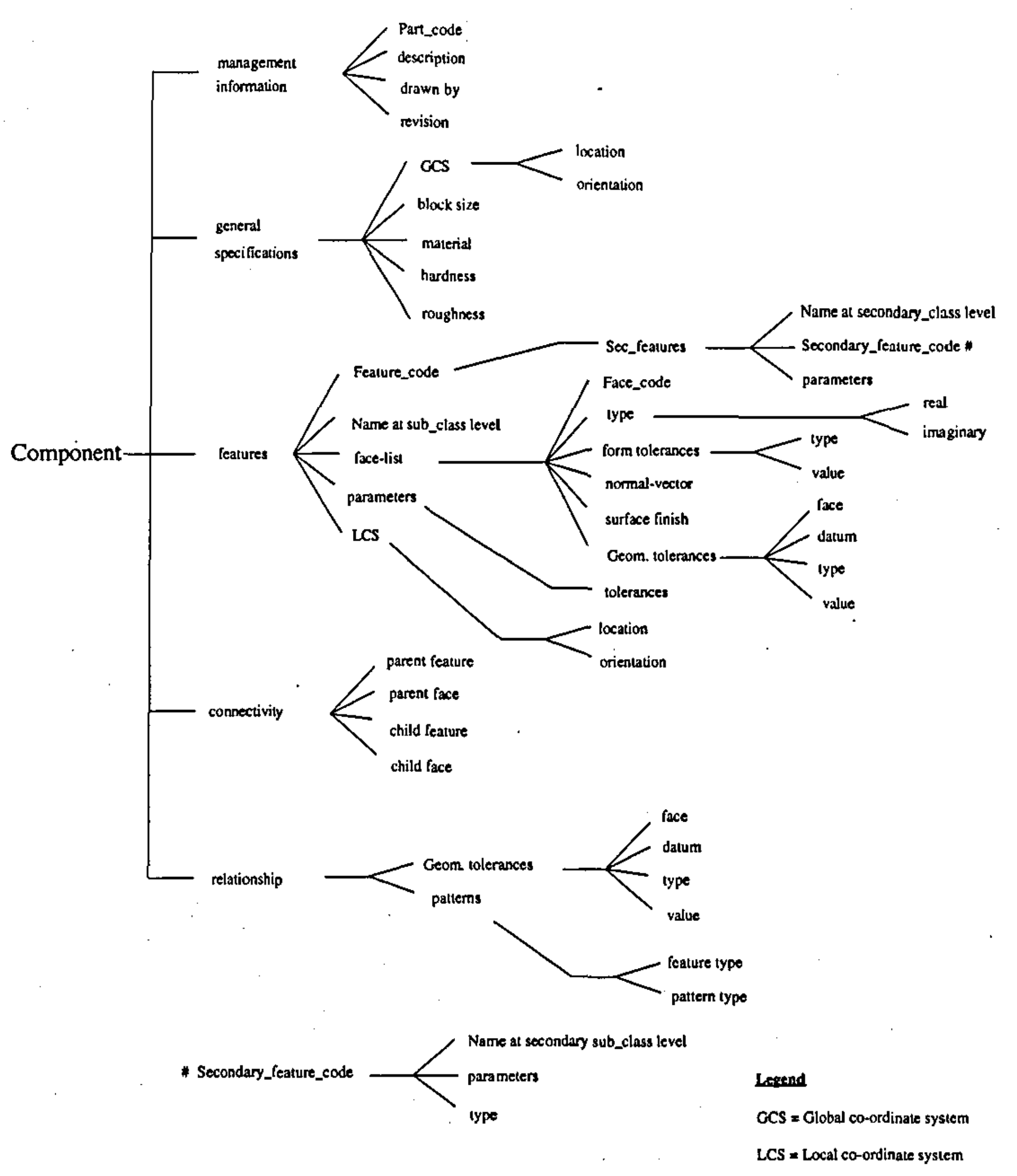


Figure 3.9. Feature-based component data model.

- a database
- a rulebase

The database is used to store the data about the problem in hand as facts. A fact can be considered as a rule without an antecedent. The 'rulebase' contains rules which represent general knowledge about the problem domain. A Production Rule takes the form of " if < condition(s) > then < action(s) >". Representing complex knowledge can be aided by the use of Frames. These are used to describe a collection of attributes that a specific object would normally possess.

Knowledge representation by a frames scheme uses a relational database model which resembles tabular arrays where a column within the table roughly corresponds to a data field and a row in the table roughly corresponds to a logical record. A frame is a record-like data structure used for representing a class of objects, a general concept, or a specific instance of any of these. A frame has a number of sub-structures called "slots" which contain a set of attributes and associated values: the name of a slot describes the purpose of the slot, and the value of the slot gives its state at a specific time; frames are organised into a hierarchy, with those at the upper levels representing more generic classes, and those at lower levels, being specialisation or instances of these classes.

The basic idea of a frame is rather like that of a record in data processing. All the attributes of an object are collected together in a single composite structure. However, it is more than a passive data structure. Procedures can be invoked automatically if a value for an object is required and none is explicitly stored. The value of the frame can also be linked to other frames, rather like the element and subset links in a semantics network.

The major advantage of frames lies in the ability to represent structured data objects and relationships, augmented by procedures for default and inheritance reasoning. Most applications using frames have used them to complement other forms of knowledge representation, particularly production rules rather than to supplant them. Each frame contains many sub-entities (slots), and each sub-entity is assigned a value (filler).

A frame is an effective knowledge representation scheme for the feature-based component data model and this form of knowledge representation is related to it to represent objects as a groups of slots which store information associated with the objects. These slots may also contain default values, pointers, set of rules or even procedures by which values may be obtained. The frame describes a setting and contains an arbitrary number of slots which may be filled with information germane to the frame's environment [Henderson 1984a].

The major advantages of the frame-object representation are:

- The frame-object is dynamic representation, since the state changes at any instance can be recorded, updating the corresponding slot about the results that are found during manipulation of the calculations at any point of time.
- All the frames representing an object are available in the system, and can be accessed and used at any point in the program run.
- It is possible to perform easily the inheritance process, in which the objects of the class can inherit slots, methods and default initialisation arguments from their super-class, inheritance can be single or multiple, depending on the number of parents that the frame has.
- The modularity of this methodology allows easy implementation of new objects and attached procedures.

3.7 SOFTWARE IMPLEMENTATION

3.7.1 KNOWLEDGE REPRESENTATION IN GENERIS

The software tool used to represent the knowledge about a feature-based component data model and reasoning in the downstream applications, in our case the characteristics/parameters extraction for the part family formation problem, and developing Generative Process Planning system is a knowledge based system called "GENERIS". This system has been selected as it provides all the facilities required in this research.

In the GENERIS system, both the representation schemes; (1) frames and (2) production rules have been used for knowledge representation. Joshi *et al.* [1988a] argue that it is found that use of both schemes can represent an intelligent reasoning process. The system facilitates a knowledge base (database + rulebase), a user interface and explanation output function. GENERIS can be accessed for intelligent queries from the application concerned either directly from the related table/tables or through inferences made by firing the rules. Comprehensive arithmetic and data processing facilities are supported in the GENERIS system. User defined windows and menus can be set-up in the system, which allows users to select the commands. A C-language interface is provided to access the system from outside to exchange the data between GENERIS and other systems.

Other capabilities of GENERIS are listed below:

1) In contrast to the numeric manipulation methods as used by the conventional languages such as FORTRAN, PASCAL, etc., one of the basic requirements for building knowledge based systems is that the language or system should be able to do symbol-manipulations based programming to allow for more comprehensive representation of human knowledge. This is well satisfied by the system.
2) It provides a rich structural language for describing the objects in tabular form. It significantly helps with rule management by providing a means of modularising, organising, indexing, scheduling, and invoking rules according to their intended use.
3) It provides a powerful programming environment for creating, updating, and debugging the knowledge.
4) It has high resolution graphics facilities. Multiple scrollable windows allow many different activities to be active at the same time. Menus allow highly interactive software to be developed which enables friendly communication between the user and the system. On the whole, a user-friendly interface can be developed by using all of these facilities.

Knowledge in GENERIS is represented in the form of *facts* and *rules*. Facts are of the form:

<subject> <relationship> <property>

Where 'subject' can be the name of any entity. An entity is any object or class of objects. 'Property' can be the name of any other entity, or an 'attribute'. 'Relationship' is a named relationship between the subject and property.

An example of a fact would be:

Part has material Aluminium

Where 'Part' is the subject, 'has material' is the relationship, and 'Aluminium' is the property. A property, in some cases, might be the subject of another. For example, another fact might be:

Aluminium has colour white

Here property 'Aluminium' is subject of another fact.

A property may either be an entity or an attribute. In the above example, the property Aluminium is also an entity. An example of a fact with an *attribute* as its property would be:

Part has length 120.0

Unlike entities, which have only names, attributes have a name, a type, and a value. In this example the name of the attribute is 'length', its value is of decimal type. Five types of attribute are provided in GENERIS - integer, decimal, date, time, and text. Status can be either 'optional' or 'mandatory' depending on the validity of attribute. For example, the length attribute will be mandatory but diameter attribute will be optional as only rotational components would have diameter. Index is the keyword used in the case of 'External Objects'. External Object is a large quantity text stored in a file outside the software which can be linked to the subject.

Groups of facts relating to a single class of object are held together in a table. The records, or rows, in each table relate to a single subject, while each column in the record relates to a particular relationship and property of that subject. A table would have a particular name and whether it is single valued table or multiple valued table. It will be multiple valued, if a particular entity has multiple records associated with it i.e. a table containing records about a part and its features. Further detail can be found in the GENERIS manual [GENERIS 1991].

Valuedness means whether a table is a single valued or multiple valued. Generic on/off shows whether the system is in mode of checking validity of records input or not. An example of a table structure without records is given below for the illustration.

Table parts

Main Subject part
Valuedness Single Valued Generic On

Relationship	Property	Data type	Status	Index
has drawing_number	drawing_number	Text	Mandatory	None
has description	description	Text	Mandatory	None
drawn by	person	entity	Optional	None
revision	date	date	Optional	None
required	heat_treatment	text	Optional	None
has mass	mass	Decimal	Optional	None
has material	material	Entity	Optional	None

Rules are of the form:

<fact> if <condition>

Where 'fact' can be any fact and 'condition' is any combination of facts which, if true, results in the initial fact in the rule being accepted as true. An example of a rule would be:

drilling is required if feature is round hole and diameter is ≤ 100.

The fact which is 'drilling is required' will be true if both the conditions, feature is round Hole and its diameter is less or equal to 100 are true. Results of rules (deductions) can also be written in the slots of the tables.

Another example for using the rules can be cited here. We know that if feature i has real face k and another feature j has same face k as its imaginary face then feature i will be the parent feature of feature j. A rule can be written to find out the parent-child relationship between the features as below:

feature.j 'has parent feature' feature.i if feature.i 'has real face' face.k and feature.j 'has imaginary face' face.k.

Figure 3.10 shows the example of the pattern of tables in the system. A line connecting two frames shows the key connection of an entity. The figure also shows the representation of subject, relationship and property. After creating the appropriate tables, records are entered in the relevant slots. A detailed representation of tables designed for storing the feature-based component data model is given in appendix B.

A user interface (a module) has been written to create the records about the feature-based component data model which interactively asks to enter the information about the component step by step. A copy of a component description module is given in appendix B. A flow diagram for the information flow is given in figure 3.11.

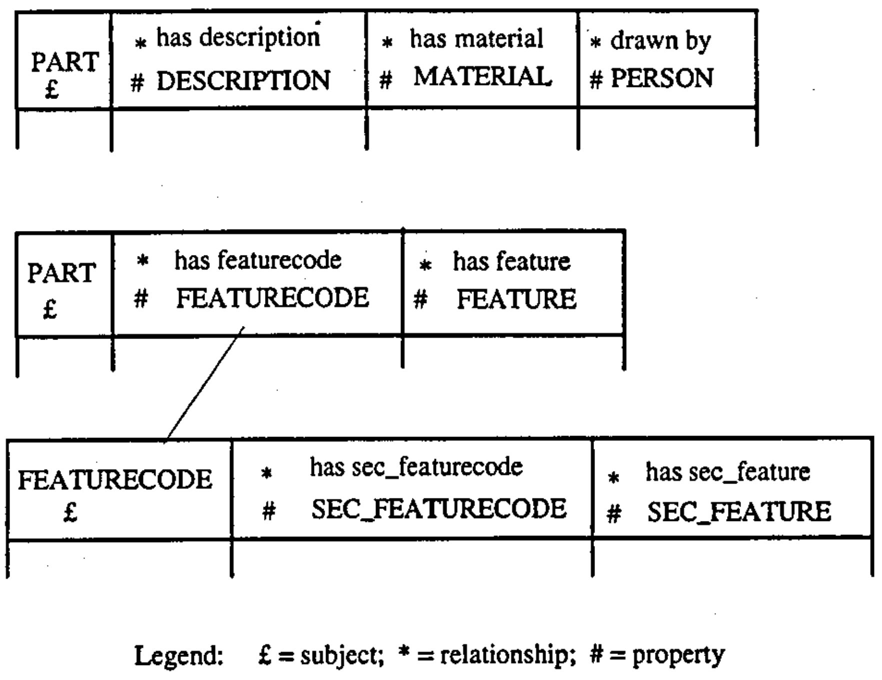

PART £
* has description # DESCRIPTION
* has material # MATERIAL
* drawn by # PERSON
PART £
* has featurecode # FEATURECODE
* has feature # FEATURE
FEATURECODE £
* has sec_featurecode # SEC_FEATURECODE
* has sec_feature # SEC_FEATURE
Legend: £ = subject; * = relationship; # = property

Figure 3.10. Example of part tables in GENERIS

The hardware used for building up the system is 'SUN4' workstation. The operating system used by the workstation is 'UNIX'. The workstation is on a network and there is a facility of data exchange with the other systems.

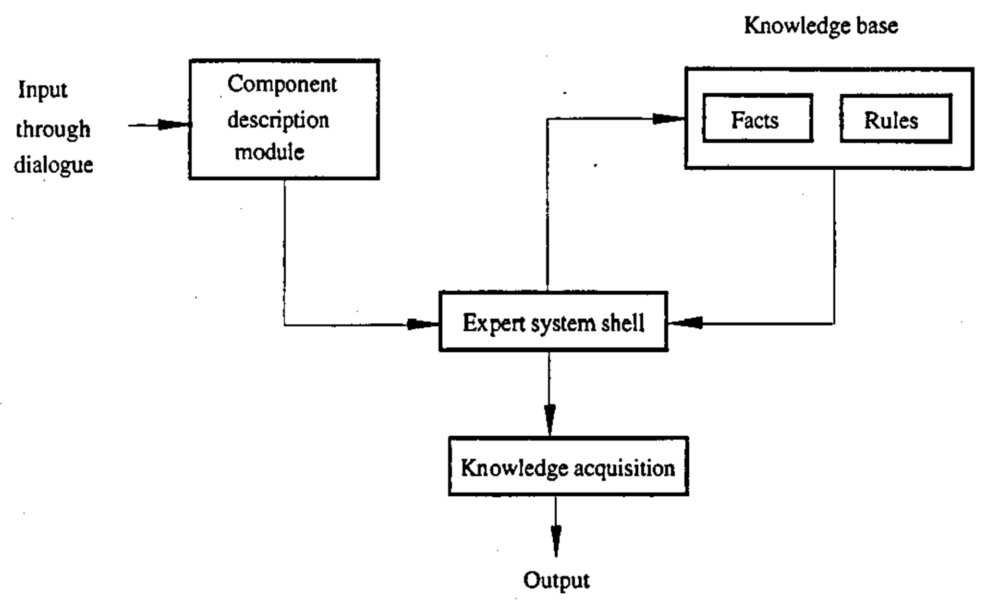
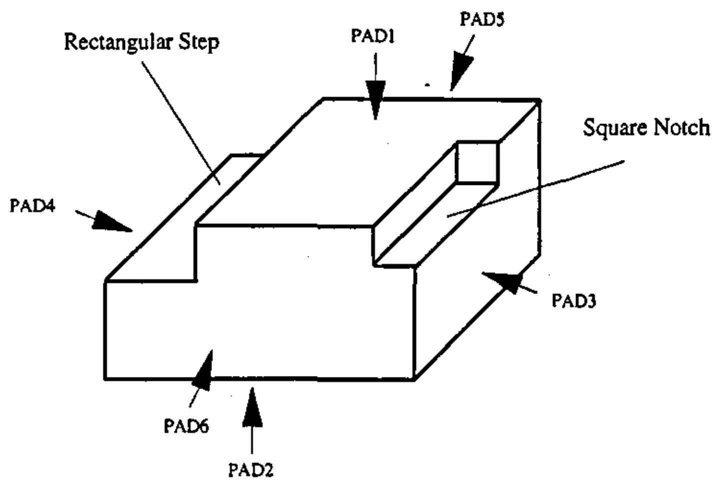


Figure 3.11 Knowledge representation and extraction of part classifying attributes.

Features are parameterised and procedure files are written for different feature types. Only relevant information to a feature will be asked to input depending on the feature type. A particular procedure file will be called for the input of a feature depending on the feature type from the main part input procedure file called 'input_part'. Examples of feature parametrised file are 'do_round_boss', 'do_round_hole', 'do_slot' etc. These procedure files are given in appendix B.



PAD = Potential Access Direction

Other information.

Specifications: block size, roughness, material, hardness

Relationships: Connectivity, Geometrical tolerances, Patterns etc.

Figure 3.12 Component level information

Figure 3.9 shows the feature-based component data model in detail. Based on this entity relationship diagram, data about the component has been structured in the GENERIS Expert System shell which uses relational database to store the component design facts. Figure 3.10 shows the example of the table structure. The complete information about the part description is organised in the table given in the appendix B. Component level information is shown in figure 3.12 and data model at feature level is shown in figure 3.13.

Feature: Square Step

Feaure identity: Feat_1

Local co-ordinate frame: i,j,k

Position: Xi, Yj, Zk

Orientation w.r.t. component datum: (Ai, Bj, Ck)

Geometric information:

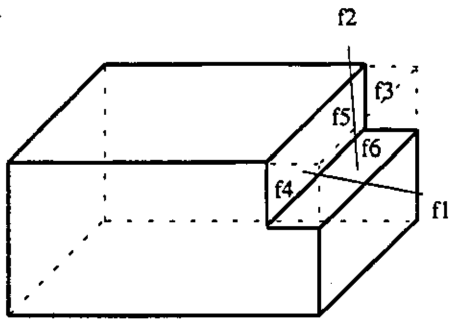
Faces: f1, f2, f3, f4, f5, f6

Real faces: f1, f2

Imaginary faces: f3, f4, f5, f6

External access directions:

EAD3, EAD4, EAD5, EAD6



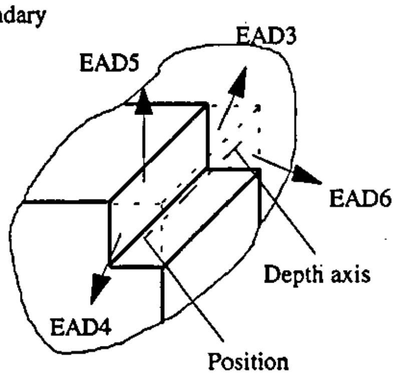
Depth boundary variation:

Depth axis: straight

Symmetric w.r.t depth axis: symmetric

Entry/exit perimeter relationship: same as entry boundary

Form variation along depth axis: constant



Depth boundary variation and EAD's

Technological information:

Parameters:

Length: value_x

Width: value_y

Depth: value_z

Dimentional tolerances:

Tolerances on value_x

Tolerances on value_y

Tolerances on value_z

Form tolerances:

Flatness, straightness etc.

Geom. tolerances at feature level

Perpendicularity, parallelism etc.

Surface finish: surf_value

Figure 3.13 Component data model at feature level

CHAPTER 4

FEATURE-BASED PART GROUPING

4.1 INTRODUCTION

The following chapter discusses the main theme of this research which is feature based part grouping for efficient Process Planning. Section 4.2 sets the pace of the chapter by introducing GT principles in more detail upon which present chapter is based.

Factors affecting the Process Planning function are discussed in section 4.3 while section 4.4 describes the part grouping based on the composite components, features, processes, and machines.

Section 4.5 details the grouping based on the Computer-Aided Feature-Based Grouping (CAFBG) System which is the proposed criteria for part grouping to make the Process Planning activity more efficient. The section also contains the details of grouping algorithm developed, stopping criteria for the clustering process to find out the optimal number of groups and pattern recognition for the new parts. The concluding section 4.6 discusses the grouping methodology used in CAFBG System in relation to Process Planning.

4.2 GROUP TECHNOLOGY

Over the last two decades, Group Technology (GT) has emerged as an important scientific principle for improving the productivity of manufacturing systems [Kusiak 1987]. GT is a simple philosophy which exploits similarities and achieves benefits by grouping like problems.

The criteria for defining similarities depend on the particular application for which GT philosophy is being applied. A simple example relating to cutting tools can be quoted here, similar cutting tools can be grouped together so that one can easily locate the required cutting tool. Which can lead to a tool management system. Criteria can be established to group the similar cutting tools according to the requirements. Several criteria might be grouping the cutting tools by the function, shape, or by machine types.

GT is being utilised in the manufacturing engineering to reduce the time and effort spent deciding on how the part will be machined if processing information is available for a similar part. GT will be dealt with only from the manufacturing point of view, in this work as grouping is performed in order to make the Process Planning function efficient.

Production activity can be divided into three types on the basis of the volumes of the products to be produced, namely (1) Job shop production, (2) Batch production, and (3) mass production. Job shop production is adapted for the small lot size manufacturing, batch production category is for medium-sized lots, and mass production is for very high production rates.

Groover [1987] argues that it has been estimated that perhaps as much as 75% of all parts manufactured are in lots of 50 pieces or less which is termed medium sized or batch production. According to Chevalier [1983] batch manufacturing accounts for sixty to eighty percent of all manufacturing activities. Batch production is an important portion of total manufacturing activity. The purpose of batch manufacturing is often to satisfy continuous customer demand for the items at irregular intervals. This trend towards more product variation has increased the emphasis on manufacturing flexibility. The outcome of all this has resulted in high cost per unit of item produced. High cost occurs because of the proliferation of manufacturing Process Plans, high tool cost, high scrap rates and high quality control. Figure 4.1 taken from the book by Chang and Wysk [1985], highlights the system selection versus production volume and variety of workpieces.

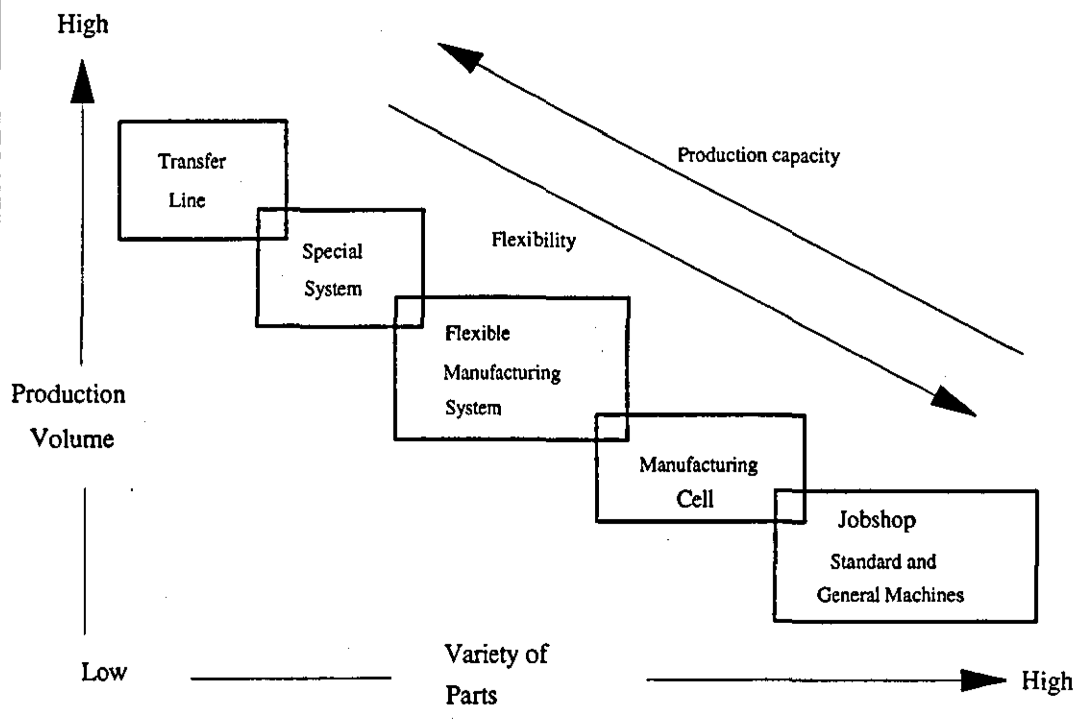


Figure 4.1. System selection vs. volume and variety of parts

According to Hyer *et al.* [1984b], GT is an approach to manufacturing that seeks to maximise production efficiencies by grouping similar and recurring problems or tasks. Similar parts are grouped together into part families. Parts classified and grouped into part families provide a much more reasonable solution for production and management. Components in a family require similar production methods for their manufacture. Manufacturing cells can be built which will fulfil the processing requirements of a part family or possibly more than one family.

One of the biggest problems faced in designing Cellular Manufacturing systems, called, cell formation, is to group parts with similar geometry, material and process into part families, and the corresponding machines into machine cells. This grouping philosophy, which can simplify the manufacturing planning and control tasks, has been widely used in flexible manufacturing systems (FMS) or flexible manufacturing cells (FMC) [Kumar *et al.* 1986, Kusiak 1985, Stecke 1983].

By exploiting the similarities which exist among the components in the families, GT promises to provide the following merits:

- reduction in the lead times and throughput times (because of reduction of times associated with movement of parts and material)
- reduction in work-in-progress (WIP) and finished inventory levels.
- reduction in material handling.
- better quality produced and reduction in time required for set-ups.
- better production planning and control.
- better labour and resources utilisation.
- less diversity in tools, jigs and fixtures.
- better space utilisation.
- simplified estimating, accounting and work measurement.
- improved job satisfaction, moral and communication.

The effects of GT on the various aspects of enterprise has been depicted pictorially in figure 4.2. Flexible Manufacturing Systems (FMS) and Cellular Manufacturing (CM) came into being to address the needs of batch manufacturing. Both FMS and CM exploit the benefits inherent in GT approach.

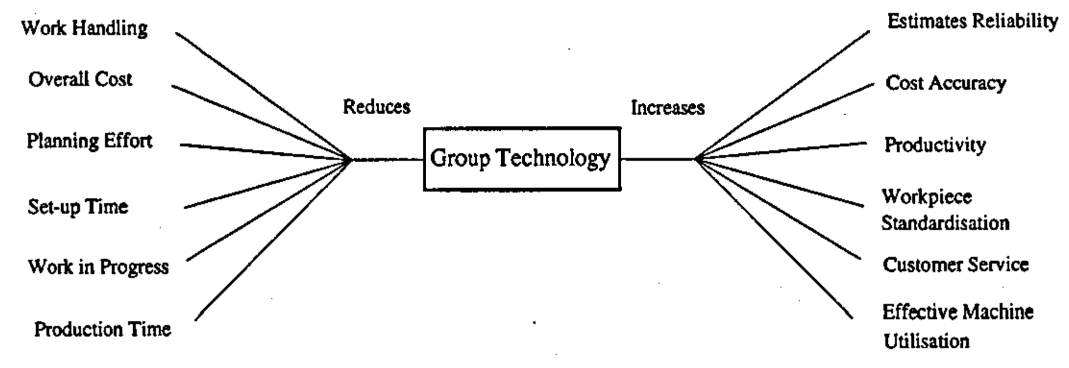


Figure 4.2. Characteristics of Group Technology

GT is an important element of CAD and CAM. An essential aspect of the integration of CAD and CAM is the integration of information used by engineering, manufacturing, and all the other departments in a firm. Group Technology provides a means to structure and save information about parts, such as design and manufacturing attributes, processes, and manufacturing capabilities, that is amenable to computerisation and analysis. It provides a common language for the users. Integration of many types of part-related information would be virtually impossible without GT; consequently, GT is an important element of CAD/CAM integration [Bedford *et al.* 1991].

4.3 MAJOR FACTORS AFFECTING THE PROCESS PLANNING FUNCTION

Selection of features/characteristics is one of the most important problems in part family formation. Attributes/characteristics for the part family formation problem have been selected which are directly relevant to the manufacture of machined parts.

4.3.1 FACTORS AT COMPONENT LEVEL

4.3.1.1 SHAPE

The basic shape of the component does affect the selection of machine tools for their processing. Cylinder type shapes can be machined on lathes, boring mills, cylindrical grinders, or any other machine tool that supplies a rotary primary motion to the workpiece. The box-type category of components includes all shapes other than cylinder type shapes, the motion of the workpieces in this case is linear one. The classification of components based on their basic geometrical shapes and dimensional ratios are important factors for the purposes of analysis of production systems required for their machining. Opitz classification system provides a good base to classify the components based on their geometrical dimensional ratios. These ratios do help in characterising the components for the machine selection, candidate processes and sometimes work holding information.

Wang [1991] advocating this idea, argues that the basic structure dictates the machining method. Note that even a basic surface may be composed of different classes of part features in different combinations. Wang is also of the view that parts with distinctly different overall shapes are machined using different machining and locating methods, and different tools. Knowing the basic shape of the part, the Process Planner will have some general ideas about the basic machining processes required for its processing. Divisions used by Opitz [1970] are discussed in the following.

Cylindrical shape components [(L/D) ≤ 0.5]

Cylindrical shape components where the length-to-diameter ratio is less than or equal to 0.5 can be classified as *discs*. For diameters up to approximately 300 mm, the component would generally be gripped in a lathe chuck, for larger diameters the workpiece would be clamped on the table of a vertical borer. Features involved in that case are Round Bosses, Round Holes, Round Pockets (sometimes called blind holes) and secondary features present on these primary features. The candidate operations might be facing, turning, drilling, boring, threading, chamfering etc. The above mentioned operations are usually performed on the lathe without changing the set-up of the workpiece unless the opposite face is to be machined. In that case regripping or change of set-up is required. In some cases, secondary features of Round Boss and Round Holes, for example, Keyways exist on these categories of components which require operations on a shaper (or slotter) or milling machine. Finally auxiliary Round Holes or patterns of Round Holes/Round Pockets (not concentric with the component axis) and secondary feature Gear Teeth may be required. A drill press is needed for the machining of these auxiliary Round Holes. Gear Teeth would be generated on the gear cutting machines.

Cylindrical shape components [0.5 < (L/D) < 3]

The class of components having length-to-diameter ratio of between 0.5 and 3.0 are *short cylindrical components*. The workpieces from which these components are produced would often be in the form of bar stock, and the component would be separated from the workpiece by parting. Features involved in that case are again Round Bosses, Round Holes, Round Pockets and secondary features present on these primary features. The candidate operations involved are facing, turning, drilling, boring, threading, chamfering etc. Which can be machined on the lathe. Strictly speaking, the features involved in this category and their solutions and disc type components are similar in processing requirements, the only difference being in the method of gripping on the machine.

Cylindrical shape components [(L/D) ≥ 3]

The components having a length-to-diameter ratio greater than or equal to 3.0 are considered as long cylindrical components (usually shafts). These components would be supported between centres on the lathe or gripped at the headstock end by a chuck and supported by the tailstock centre at the other end. Features involved in that case are: Round Bosses, Round Holes, Round Pockets and secondary features present on these primary features. A common secondary feature in this class might be Keyways which require milling or shaping processes. Auxiliary Round Holes or pattern of Round Holes (not concentric with the component axis) might exist on the components but if the shaft is too long, a horizontal drilling head is required for the processing of these features.

L/D ≤ 2 with deviation

The components in this category might be partially prismatic and partially rotational. Examples include stator of the motor, partially hexagonal bar, partially polygonal sectional component or rotational components with curved axis. These components deviating from complete rotational components are divided into two categories i.e. 1) L/D ≤ 2 With Deviation which are short components and 2) L/D > 2 With Deviation which are long components.

These components have very complex structure and usually need a lot of set-ups for machining. Most of the machining features on these components are the same as on the rotational components but other features also exist which require milling machines as well. Short components would generally be gripped in a lathe chuck, for larger diameters the workpiece would be clamped on the table of a vertical borer for machining the rotational features.

L/D > 2 with deviation

The components in this category having a length-to-diameter ratio greater than 2.0 are long components with deviation. These components would be supported between centres on the lathe machine or gripped at the headstock end by a chuck and supported by the tailstock centre at the other end. As far as the machining features are concerned, they are similar as mentioned in the category L/D $\leq$ 2 With Deviation.

Prismatic components [(A/B) ≤ 3, (A/C) ≥ 4]

A, B, and C stand for component's length, width, and thickness respectively. Box type components having proportions such that the length-to-width ratio is less than or equal to 3.0 and length-to-thickness ratio is greater than or equal to 4.0 are categorised as *flat* components. Many flat components would be machined from plate or sheet stock. Almost all features are involved in this category of components. Most common among the primary features are Surface, Step, Notch, Through Slots, Non-through Slots, Round Holes and Round Pockets. Features like Surface, Step, Notch, Through Slots, Non-through Slots might be machined on milling machines, shapers or in the case of very large workpieces on planing machines. Holes and Pocket features are also usually machined on milling machines. Sometimes features like large Round Holes might be present on the flat components which require a boring process. That can be done on a lathe or vertical borer depending on the size and weight of the component. For small parts, however, where high accuracy is required, it would be machined on a jig borer. Auxiliary Round Holes or Round Pockets might be present on the flat components, as in the case of long shafts or disc-shaped cylindrical components; and these auxiliary Round Holes or Round Pockets would generally be machined on the drill press.

The machining features on flat components will generally be normal to the two large surfaces. Also, flat components required in reasonably large batch sizes can be machined in stacks of workpieces. A shaper might be used economically for the machining of a stack of workpieces. Also stack components that are required in large quantities, can be processed on milling machines.

Prismatic components [(A/B) > 3]

Box type components having proportions such that the length-to-width ratio is greater than 4.0 are categorised as long components. Almost all features are again candidates in this category of components, most common among the primary features are Surface, Step, Notch, Through Slots, Non-through Slots, Round Holes and Round Pockets. Features like Surface, Step, Notch, Through Slots, Non-through Slots might be machined on milling machines, shapers or in the case of very large workpieces on the planing machines. Round Holes and Round Pocket features can also be machined on the milling machines. Features like large Round Holes are avoided in this class of components because generally the workpiece cannot be rotated about the required axis. For small parts, however, where high accuracy is required, it would be machined on a jig borer. Auxiliary Round Holes or Round Pockets might be present on the flat components, as in the case of long shafts or disc-shaped cylindrical components; and these auxiliary Round Holes or Round Pockets would generally be machined on the drill press.

Prismatic components [(A/B) < 3, (A/C) < 4]

Prismatic components having proportions such that the length-to-width ratio is less than 3.0 and length-to-thickness ratio is less than 4.0 are termed as *cubic components*; they are of quite complicated shape. Main bores in this category of components will often be machined on horizontal boring machines. Many cubic components would be machined from block stock. Almost all features are involved in this category of components, most common among the primary features are Surface, Step, Notch, Through Slots, Non-through Slots, Round Holes and Round Pockets. Features like Surface, Step, Notch, Through Slots, Non-through Slots, might be machined on milling machines, shapers or in the case of very large workpieces on the planing machines.

4.3.1.2 FEATURES

The use of features on the design side (design features) could relate to the fulfilment of functional requirements, the building of a geometric model, or as preparation for design analysis activities such as finite element analysis. On the manufacturing planning side, activities such as process planning, assembly planning, machine operations planning and part programming for numerically controlled machines could potentially be based upon a feature representation of a component [Shah and Rogers 1988b].

Features originate in the reasoning processes used in various design, analysis and manufacturing activities [Cunningham and Dixon 1988]. For the manufacturing engineer, part features are associated with the needs of machining. According to Choi *et al.* [1984], a feature is a portion of the workpiece generated by a certain mode of metal cutting thus relating the features to manufacturing methods. Part features associated with the needs of machining determine which way the part is to be manufactured.

4.3.1.3 FEATURE CONNECTIVITY AND RELATIONSHIPS

For the Process Planning activity, information at local level (features) as well as global level must be available. Willis [1988] argues that the features in isolation are insignificant for the synthesis of a machining plan for the whole component.

Feature connectivity decides on how the features relate to the component PAD's and feature relationships provide the constraints that may exist on the formation of feature clusters that can be machined from component PAD's.

Sometimes features are related to each other by means of geometry and/or technological types of relationship. Feature relationships play an important role while determining the process selection and sequencing, machine and tool selection, set-ups in the Process Planning function. There are four categories of relationship among the features - compound features or intersection, Geometrical tolerances among the features, parent/child relationship and pattern relationships [Butterfield *et al.* 1985].

CONNECTIVITY

For the machining of a feature, the cutter must access the feature from some direction. The possible directions which can be used are termed the potential approach directions (PAD's). Based on feature connectivity, it is possible to determine all the component PAD's, the feature can be machined from each direction. By considering, also geometrical tolerance constraints (if any) these PAD's can be optimised. A possible optimisation criterion is the minimum number of tool approach directions. The result will be the minimum number of approach directions (AD's) through which each feature can be processed. AD is the one of the parameters selected for part grouping problem.

Sometimes feature precedence relationship occurs. Feature precedence results when a feature sits on top of another feature and therefore must be machined first. Feature precedence was used by Joshi [1988], for sequencing the machining of features.

Feature connectivity has already been discussed in chapter 3 in detail. In figure 3.8, given in chapter 3 (assuming there is not any geometrical tolerance applicable in the case of this component), E1 will be AD for this component. Therefore only one AD is needed for this component.

COMPOUND FEATURES

A compound feature can be considered as a collection of related part features. The relationship is normally decided by the commonalty of the manufacturing methods used for their processing. Examples are counter bored hole, some stepped holes, etc. A compound feature might be combined because of certain objectives in mind. An objective might be to machine the compound feature on any particular machine (group of simple Round Holes, simple Round Pockets and stepped Round Pockets could be machined together for example). Similarly, a compound feature may need to be divided it into its constituents feature because it may not be possible to machine it in its compound form. Compound features tend to be company specific and it is difficult to establish general rules for their definition.

FEATURE PATTERNS

Feature Patterns can have implications for tooling and resource allocations. For example, a pattern of Round Hole features in a particular pattern will usually result in special toolings, processing together and perhaps dedicated multi-spindle drills. Patterns are, therefore responsible for special consideration in tooling and sequencing operations. Though other compound features also, have implications in Process Plan development.

TOLERANCES

The quality of the product is a matter of prime importance in machining. It receives highest priority in the Process Planning function. Dimension, shape and surface finish are the geometrical parameters of the machined part's quality. The designer is responsible for defining these parameters (i.e. the conditions under which the product is to function). A part is manufactured with its parameters defined by the part design. The closeness of geometrical parameters achieved by the machining as compared to that defined in the part drawing is termed as the manufacturing quality of the product.

The degree of closeness of part's geometrical parameters which a machine can generate, compared with the real parameters specified in the part drawing is called the machining accuracy. The degree of coincidence between the real geometrical parameters (surface finish) obtained after machining a part with those specified in the part drawing is termed as surface quality. On the component there are working surfaces as well as nonworking surfaces. Working surfaces require control of the surface characteristics for their performance such as bearings and pistons. This is called surface finish. Surface finish includes roughness height, roughness width, waviness height, waviness width etc. The roughness associated with a particular surface has a direct influence on its function and/or cost of generation. Surface finish is directly related to the features. Surface finish influences the selection of processes and calculation of cutting parameters. It is of great importance that an unnecessary tight specification should be avoided because it involves extra cost and manufacturing operations.

Machining error is taken to be the difference between the parameters of a machined part and that specified in the part drawing. Machining error always exists. A part is considered acceptable, if the machining errors are within the tolerance limits specified in the part drawing. The tolerances aim to ensure that certain characteristics related to function requirements can be achieved.

An engineering drawing conveys the geometric information of the part as well as dimension. Tolerance is associated with the dimension as it is impossible to produce the exact dimension specified. Tolerance is allowable or acceptable variation in a dimension. Tolerances can be characterised into two types, namely 1) Internal tolerances and, 2) External tolerances [Butterfield *et al.* 1985].

Internal tolerances apply within the feature boundary and can be considered as feature level attributes. These define the size and form of a feature. Internal tolerances will be discussed while discussing the factors at feature level.

EXTERNAL TOLERANCES

External or geometrical tolerances represent the tolerances related to the geometric characteristics of the part. These are the relationships among the features. Features are generally constrained by their orientation or position relative to other features or datums. These relationships capture the design intent to fulfil part functionality. Some specific examples are:

- The concentricity between two Round Holes must be within a certain value e.g. 0.05 mm.
- This surface must be perpendicular to that surface within a certain value e.g. 0.06 mm.
- The profile of this surface should not vary more than any particular value e.g. 0.12 mm relative to that datum.
- This Round Hole is positioned relative to these two faces.

Different types of external geometrical tolerances are listed below:

Parallelism	Symmetry
Perpendicularity	Runout
Angularity	True Position
Concentricity	

The geometric dimensioning and tolerancing of a workpiece sets important constraints on the selection and sequencing of operations. Four geometric tolerancing characteristics—flatness, circularity, cylindricity, and straightness are related to operation selection while the majority of other geometric tolerances influence operation sequence [Wang and Wysk 1988].

4.3.2 FACTORS AT FEATURE LEVEL

4.3.2.1 FEATURE GEOMETRY AND TECHNOLOGICAL ATTRIBUTES

Machined parts typically exhibit local surface configuration such as holes, pockets, grooves and slots, all of which are examples of form features. For manufacturing planning purposes, each feature type has associated with it a comparatively small set of possible manufacturing options. The choice between these can be made on the basis of associated technological information (the surface finish, or other tolerances required, for example) and of available production resources. A knowledge of manufacturing features of a part therefore provides the basis for the automatic generation of process plan [Pratt 1993].

Features are the main factors for selecting the processing methods, equipment and toolings in the Process Planning function. Features directly, are not feasible to be selected as classifying attributes because of the number of reasons mentioned in section 4.4.2. Features can be transformed into their machining requirements. Feature processing requirements can be decided based on its type, parameters, tolerances, surface finish, and connectivity aspect while matching them to the facilities available in the processing cell/shop. Tolerances are one of the technological constraints that determine machine tool used and machining parameters. These will be discussed in the following in more detail.

INTERNAL TOLERANCES

Internal tolerances apply within the feature boundary and can be considered as feature attributes. These define the size and form of a feature, irrespective of its relationship to the rest of the part. They are specified for each nominal dimension of the particular feature. For example, a Round Hole contains nominal diameter and depth dimensions as well as diameter tolerance and depth tolerance information. In DMIS [1982] (CAM-I Dimensional Measuring Interface Specification) size and form tolerances are given below:

1. Size i.e. angular tolerances, diametric tolerances, etc.
2 Form tolerances as given below:
   Straightness
   Flatness
   Roundness or circularity
   Cylindricity
3 Profile
   Profile of a line
   Profile of a surface

Size internal tolerances again have two types which are 1) Unilateral tolerances and, 2) Bilateral tolerances.

A unilateral tolerance denotes a decrease (or increase) in one direction in relationship to the concerned dimension, such as 4.00 + 0.03 equals 4.00 - 4.03 while bilateral tolerances denote dimensional variation from the basic size on both sides. For example 4.00 ± 0.03 equals 3.97 - 4.03.

Profile can be used with or without a datum, so it can be listed in either local or global sections. Following entities are associated with all above local tolerances except flatness.

- MMC - denotes that maximum material condition is applied to the feature for this tolerance.
- LMC - denotes that least material condition is applied to the feature for this tolerance.
- RFS - denotes regardless of feature size.

4.3.3 OTHER FACTORS

Production information other than part geometry and its technological constraints consists of initial form, weight, batch size and frequency. Other factors not related to the components include management policy, special requirements, availability of labour and machines and heat treatment.

Above mentioned factors cannot be taken as parameters because they are not a part of design information. Part grouping is considered at discrete part level.

4.4 PART GROUPING

The main target of component grouping is to achieve a stable partitioning of a set of components into a number of meaningful groups to serve a specific application. A part family may be defined as a group of components having similar attributes for the purpose of manufacturing. The assessment of similarity among the components is decided on the basis of characteristics or parameters chosen. Parameters chosen are dependent on a particular application for which grouping is being considered. There are many potential parameters based on which part families are formed. These might include the manufacturing processes, the manufacturing features present on the component, the material, the machine tools used, the weight or even the functionality of the components.

For example, if a decision is made to group the components by their material, the result will be groups of components made of aluminium, stainless steel, cast iron etc. The parameter selected in the above example was only material. The number of parameters could be more than one and usually are. In the example cited above, another parameter for grouping might be the functionality of the components. Larger number of groups may potentially be formed as the number of parameters is increased.

It becomes necessary to calculate similarity among the components mathematically in the case when the number of parameters selected for grouping are more than one. Another problem encountered in the part family formation is at what similarity level components should be grouped. At the 100% similarity level, the number of groups might be as many as the number of parts in the sample. The number of groups will go on decreasing as the similarity level decreases. At the 0% similarity level, one will be left with only one group of components.

Before determining which parameters are to be selected for grouping the components, the purpose of grouping needs to be examined in detail. It is also important to study the population of items to be grouped.

Hyde [1981] points out that classification should exhibit the following for sound results:

- Bring like things together by virtue of their similarities.
- Separate them by their essential differences.
- The fewest possible number of parameters should be used to get well balanced families.

Also, procedures necessary for the classification function have been mentioned by Hyde [1981].

1. Define the needs of the principal users of the classification.
2. Define the population of data that is to be embraced by the classification.
3. Capture the raw data and connect it to a uniform media format for handling.
4. Identify the raw data to enable it to be classified.

Cellular Manufacturing (CM) is sometimes termed as "product-oriented" production in the sense that the layout is specifically structured to cater for the production requirements of products. Traditionally, manufacturing organisations have one of two layouts of production equipment, namely the job shop (or process-based layout) - where machines of similar functional types are grouped together and the transfer line layout - characterised by dedicated manufacturing lines. The job shop or process-based approach exhibit complex production management problems such as complex scheduling and resource management issues, and typically requires substantial movement of material and parts resulting in high WIP, longer lead times etc. Transfer lines are generally for very high volume productions and involve high capital investment while providing little production flexibility. These lines typically require expensive re-tooling whenever product changes are implemented. Comparison between process-oriented and product-oriented layout is shown in figure 4.3.

Figure 4.3 shows representation of part flows in both the process-based shop and product-oriented shop. Both layouts in the figure have the same processing capabilities, yet much improvement in part flow in the case of product-oriented layout is obvious.

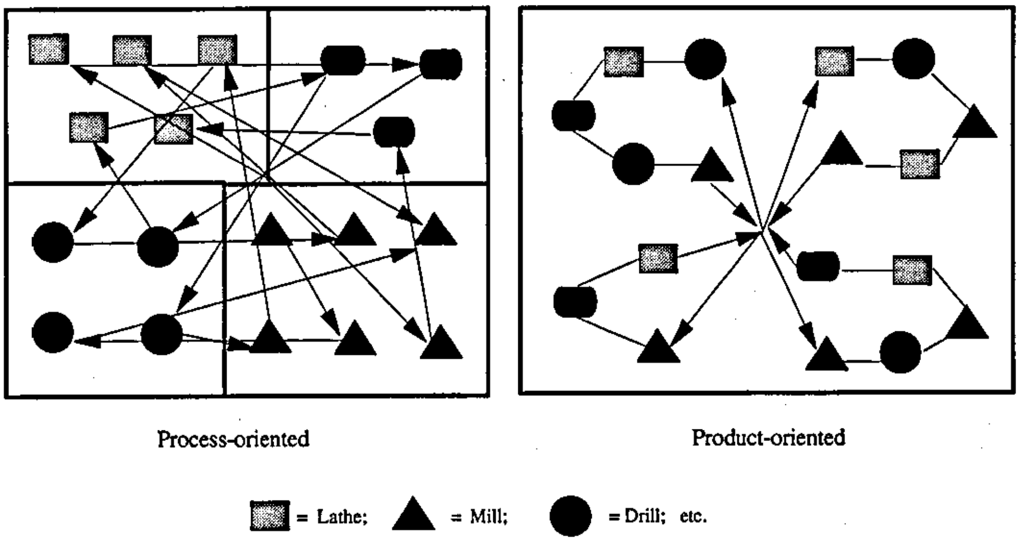


Figure 4.3 Product-based vs. process-based layout.
[Adapted from Warren and Moodie 1993]

4.4.1 PARTITIONING AROUND MEDIODS

This clustering approach is grouping the parts around already defined k composite components i.e. grouping centres. In the cluster analysis terms the k representative objects are called mediods or centrotypes [Kaufman 1990]. Two approaches are used; 1) grouping around key machine, and 2) around composite component/cell capabilities are discussed in the following.

4.4.1.1 GROUPING AROUND KEY MACHINES

Many researchers have aimed at 'seed machines' based cell formation approaches. Burbidge [1975] used approach called 'nuclear synthesis' in which each cell is formed by initially selecting the machine for which (F-Σf) is minimised, where F is the total number of parts that require the machine, and Σf is the cumulative number of parts that require the machine and that has previously been assigned to each cells. Vannelli and Kumar (1986, 1987), Wei (1987), Ballakur and Steudel (1987), Al-Qattan [1990] and Frazier and Gaither (1991) have proposed similar concept of key machines in their cell formation research. Groover [1987] also supports the grouping around the key machine idea.

As far as possible an attempt is made to spread the workload evenly among the machines in the cell. There may be some machines in the machine cell (perhaps two or three) which are more demanding and expensive than the other machines.

Under this concept the most expensive and important machines need to have a reasonably high utilisation, while the less important and less expensive can be allowed a relatively low utilisation. An expensive and/or the machine in more demand is referred to as a "key machine" while the other machines are referred to as "supporting machines". As the key machine is required to operate at maximum output, then it must be fully operational at all the times, so that the key machine becomes the bottleneck machine in the cell. With regard to supporting machines, however the aim should be to ensure maximum utilisation of labour rather than equipment. A flexible labour force is therefore needed which can alternate among the various parts of supporting equipment or key machines as needed, in order to keep the key machine or plant fully operational.

Key machines can be CNC lathes, CNC milling machines or machining centres. Components can be grouped around them taking them as centres of grouping. The features of this type of cell are that key machines are always fully loaded. This is achieved by ensuring that the capability of supporting plant which precedes and follows the key plant will have surplus capacity.

4.4.1.2 GROUPING AROUND COMPOSITE COMPONENT/CELL CAPABILITIES

A composite component embraces all of the features of the individual components in the group or family. It is used to represent the centre of grouping based on already established parameters or criteria. The composite component concept is used either to represent the processing requirements of a family of components or capabilities of manufacturing cells. Component grouping can be performed around these centres. After determining the processing needs of a composite component, it is then possible, by cancelling not required operations and tools, to manufacture any one member of the family. The composite component may be real or hypothetical, although usually the latter. A manufacturing cell can be designed to machine the composite component and therefore, with minor adjustments, the whole family. According to Gallagher and Knight [1973], a set of components grouped around the composite component will exhibit two basic characteristics. First, they will have the same holding or chucking requirements, and second, they need the same or very similar surfaces to be machined.

The world of machine tools (capable of providing a number of processes) can be thought of occupying regions in the space. Machine capabilities overlap and hence, they have shared boundaries. A process is a region in the space capable of providing geometric and technological outputs. A composite component can be thought of as a centre of region consisting of small regions of those processes in the space responsible for fulfilling the geometric and technological requirements of any part family, or any manufacturing cell.

Grouping of the components can be performed around components made of certain types of features already defined as being the centres of groups. Attraction in this idea is that features are of functional importance from the manufacturing point of view. In order to define the boundaries of the grouping, constraints like component dimensional ratios, component major dimensional limitations, component material or component initial form (cast, welded etc.) may be applied. These types of constraints along with feature types constraints will result in well defined grouping centres around which components can be grouped.

Part families can also be formed around the processes, the cell can offer to meet the requirements of components in a family or those required by the composite component. This is another viable way of grouping the components around already defined mediods as the manufacturing features on the component body can easily be transformed into their processing requirements. Similar types of boundaries such as mentioned above can be applied.

4.4.2 PART GEOMETRY AS CLASSIFYING ATTRIBUTES FOR THE PART GROUPING

A feature contains geometric and technological information. Technological information comprises surface roughness, form tolerances and a set of manufacturing directions (EAD's).

Manufacturing features can be the potential criteria to differentiate between dissimilar workpieces and similar ones in a group. According to the findings of this research, which is grouping for the efficient Process Planning function, components cannot be differentiated based simply on features. A number of reasons can be mentioned as below:

1) There is a large variety of feature types. It is very difficult to select them as classifying attributes.
2) Different features have similar processing methods.
3) The processing method of a feature is not simply linked to the feature type but also other parameters like size, accuracy etc. (technological constraints).

There is a large variety of features which might potentially exist on the components. It is, therefore very difficult to characterise the components based on the large number of feature types. For instance, let us consider the category of a Boss feature, common primary-class features in this category are listed below.

Square Boss	Elliptical Boss
Rectangular Boss	Triangle-shaped Boss
Round Boss	Diamond-shaped Boss
Partial Round Boss	Pentagon-shaped Boss
Obround Boss	Hexagonal-shaped Boss
Partial Obround Boss	Octagon-shaped Boss
Double 'D' Boss	Rosette-shaped Boss
Single 'D' Boss	Other Bosses
Spline-shaped Boss	

This is a list of features having symmetric depth boundaries. Some features among them may also exist which do not have symmetric depth boundaries. Again, each of them can further be characterised by its form variation along the depth boundary. i.e. constant, tapered, concave, convex, contoured and others. The list of feature types will go on increasing like that. It is therefore, not possible to use the feature type as a criterion for grouping.

The same processing methods can be used for a number of different feature types. Features like, Rectangular Step, Square Step, Triangular Step, Rectangular Through Slot, Square Through Slot, Triangular Through Slot, and many others can be processed by milling process or shaping process. They are similar from the manufacturing point of view as they can be processed by similar manufacturing methods. But, on the contrary, grouping based on feature types will place the components having these features in different groups as the types of features involved are different. This is another point which disproves the grouping based on feature types.

Thirdly, a feature with different technological requirements (different size, accuracy etc.) might lead to different machining methods, different tools and different machine types. A Round Hole with the requirements of average accuracy, for example, can be machined on a drilling machine but the same feature with high accuracy requirements might need to be processed on the jig borer to meet its technological requirements. Furthermore, honing or lapping facility is needed for the same Round Hole feature which has got extremely high accuracy requirements. The same feature with different geometric dimensions can lead to selection of different operations, different tools and even different machines. An example of a Round Hole feature with different geometric dimensions can be quoted here. Feature Round Hole having a small diameter may be processed by simply a drilling process but the same feature having a large diameter requiring a boring process as well as drilling and for boring process another machine tool might be needed. Another example of planer and shaper can be cited where selection of these machine tools can be characterised based on geometric dimensions of the features. This is a third vigorous point disproving grouping based on feature types.

As already discussed components have a large variety of machined features types on their bodies which discourage the grouping of components based on feature types. The number of classifying attributes/parameters should be reasonable in the classification process. One of the guidelines for classification pointed out by Hyde [1981], already mentioned elsewhere in the thesis needs to be described here again:

The fewest possible number of parameters should have been used to get well balanced families.

Besides that, there are also component attributes like feature connectivity, overall dimensions etc. which are needed for the good results of part family formation in the context of manufacture. Therefore, the discussion highlights that just characterising the components based on feature types will not result in proper grouping for the Process Planning.

With this in view, it is preferable to transform the features into their processing requirements first and then group them for the part family formation. Another point in support of this idea is that the number of manufacturing processes are limited and this makes the problem manageable.

Feature based part family formation for the limited rotational parts has been reported in the article by Perotti et al. [1991]. The parameters/characteristics chosen for the part grouping are morphological and technological attributes of the part features.

4.4.3 PROCESSES AS CLASSIFYING ATTRIBUTES FOR THE PART GROUPING

A lot of research has concentrated on the grouping of the components based on the processes used for machining the components (termed as process-based grouping). These processes are usually, taken from their Process Plans. This is a good starting point. It is possible to perform process based grouping as features can be transformed into processes.

There are a number of reasons why grouping based on the processes involved in component machining will not lead to realistic results.

A feature may have more than one valid solutions and this method does not take into account such a factor. In the process of grouping one needs to compare the components based on any standard criteria. If comparison between the components is not made on the basis of standard criteria then the grouping will not be a realistic one. Secondly, processes can also be carried out on several machines.

Thirdly, much information is still missing which is required for the selection of processing facilities where the component will be processed. There is one other dominant factor for the selection of these facilities, that is feature patterns. The decision for the selection of machine types is also based on the single feature or multiple similar features or features patterns which exist on the component. A single feature can be machined on any ordinary or conventional machine type but in the case of feature patterns special purpose machine may need to be assigned for that particular job. Process based grouping would not differentiate between the components based on such types of factors. Similarly there is another factor which needs to be addressed in the grouping process which is the structural aspect or features connectivity of the component. The part features connectivity aspect plays an important role in the selection of machine resources but process based grouping ignores all these factors.

Therefore, it can be argued that process based grouping will not lead to realistic grouping results.

4.4.4 MACHINES AS CLASSIFYING ATTRIBUTES FOR THE PART GROUPING

Burbidge [1963, 1964, 1971, 1974, 1975, 1977, 1979, 1988, 1989 and 199] has been active in promoting a technique called 'Production Flow Analysis' (PFA) during the last several decades. This technique has been very popular in the industry for converting the traditional 'process-oriented layout' to 'product-oriented layout' or Cellular Manufacturing. PFA method groups the parts with identical or similar machine routes together. The parts grouped together are called 'part families'. These groups can then be used to form machine cells in a Group Technology layout. While reducing material handling in Cellular Manufacturing, PFA technique results in bringing economic benefits for the enterprise. Simplification of part flows significantly reduces work in progress (WIP) and production lead times, thereby increasing the utility of equipment paving the way for higher return.

A lot of research has been aimed at grouping the parts based on the machines used for their processing which can be termed as machine based grouping. Machine based grouping will also not lead to grouping results which can gather similar parts in the groups. A number of reasons can be cited for the deficiencies inherent in machine based grouping.

First of all, the route sheets are usually used for this purpose. The route sheets are written by different planners or even the same planner can write different machine routes for two similar parts. There is no mechanism for rationalising the manufacturing routings. For example, two similar components assigned to different machines will result in different groups based on this grouping technique. Similar discussion has been made in detail in the previous section on the same topic, that different processing methods can be assigned to two similar components for their machining. This will obviously, allocate different machines to them thus proving them dissimilar. Even for the same processes, different machines can be selected. This will again put them in different groups.

A second important reason which can be cited for disproving this technique is that components visiting the same machines are taken as similar, no matter whether they are using the machines partially or completely. Though it can be said that components are using the same machine routes they are not necessarily similar. For example, one component using 20% of the processing facilities of any particular machine and another component using 100% of the processing facilities of the same machine can obviously not be similar to each other but machine based grouping will count them 100% similar.

Most of the techniques developed up to now and applied in the Group Technology field are mainly aimed at finding similarities between parts on the basis of which machines are required to produce them. Clustering operations based on part-machine matrices are very well known, but there have been a few problems connected with them. In fact, the same operation can be carried out on several machines and the market at the moment tends to demand a greater flexibility in machinery. This all leads to the conclusion that it is not right to form families on the basis of which machine is required but rather to keep separate the two issues, that of forming part groups and that of assigning them to machine cells. It is therefore important to cluster on the basis of the features (attributes) of the parts themselves and later assign the groups to the relevant machine cells [Perotti *et al.* 1991].

Keeping this in view, it can be argued that machine based grouping cannot result in part families having similar components.

The CAFBG System has been proposed to overcome the deficiencies inherent in the above mentioned part grouping approaches. The system is discussed in detail in the following section.

4.5 THE CAFBG SYSTEM

Following the discussion that grouping based on features or processes alone do not result in favourable part grouping which can make Process Planning function efficient. This research was aimed at part grouping based on criteria which make the grouping more efficient for the above mentioned function. Part grouping based on the machines (already discussed), one of the traditional methods used by the researchers, is also not suitable because of the availability of much more flexible machines i.e. machining centres, etc. these days than were available in the past. Secondly, similar types of operations can be performed on several machines. Therefore, it looks unreasonable to form the part families of similar components based on the machines which are needed for their processing but rather two issues should be kept separate i.e. forming the part families and designing of the machining cells. Similar types of comments are reported by Perotti *et al.* [1991].

As already said, manufacturing features are the communication medium between design and manufacturing. Therefore, part features should be input for the manufacturing in order to bridge the design and manufacture. This was also one of the main thrusts of this research that the part grouping should be feature based. The research was also aimed at fulfilling the overlooked deficiencies by the researchers in the part grouping. Part grouping is favourable if the manufacturing features are transformed into the manufacturing operations while considering the connectivity aspect and also other important design informations as grouping criteria.

4.5.1 PROCESS SELECTION

The machining process is used to remove material from the workpiece in order to shape the feature. Other than the shape producing capability, each process also has its own dimension, tolerance and shape producing capabilities. The dimension capability is dependent on the tool and/or machine tool work envelope. The accuracy of the process is a function of several parameters, like sharpness of tool, tool vibration, thermal effect during cutting, tool deflection, fixture error, control inaccuracy, etc.

Both the tool and machine are used to execute the process There are a set of constraints that need to be satisfied before the process operator can be applied. These can be geometric constraints, such as accessibility, the relative position of the surface feature, etc. There can also be technological constraints, such as the power consumption, the workpiece deflection, etc. These constraints are imposed by the tool or the machine [Chang 1990].

Alder *et al.* [1986] discuss typical machining process characteristics as below:

1. Material Properties for which process is viable - Physical, Chemical, Mechanical.
2. Surface integrity produced by process - Heat effected zones, Residual stresses, Micro cracks, Burrs.
3. Surface texture produced by process - Roughness, Waviness, Lay.
4. Material removal rate.
5. Geometric features machinable by process
   - Type of feature (plane surface, round hole, etc.)
   - Physical limitations on dimensions
   - Achievable tolerances (dimensional, straightness, etc.)
   - Finishing process only.

Process information can be divided into three levels as given below [Chang 1990]:

Universal level: This is general information regarding the processes available in the handbooks. For instance, drilling process can generate Round Holes with accuracy up to 1.6 μm.

Shop level: This is the information collected at particular shop level showing that a particular process in the shop can give this much accuracy. This data may vary from shop to shop.

Machine tool level: This is the data collected at a machine tool level showing the technological output/accuracy by a particular machine. Machine level capabilities vary due to age, holding methods etc.

The computer aided feature based grouping (CAFBG) system has been proposed in order to fulfil the deficiencies in the procedures which have been adopted by the researchers in the traditional grouping methods. In order to set the pace for the system, the author would like to introduce two terms FTD and TSF in the following (Gindy and Ratchev 1991). A TSF (technological solution at feature level) is an ordered set of operations performed sequentially to fulfil the geometric and technological requirements (size, tolerances, surface finish etc.) of a feature. Each operation in the set contributes towards fulfilling the geometric and technological requirements. In other words, a TSF is a feature level solution to satisfy its geometric and technological needs. A TSF may be decided upon by matching the requirements (both geometrical and technological) of a feature with the capabilities of operations on the available resources.

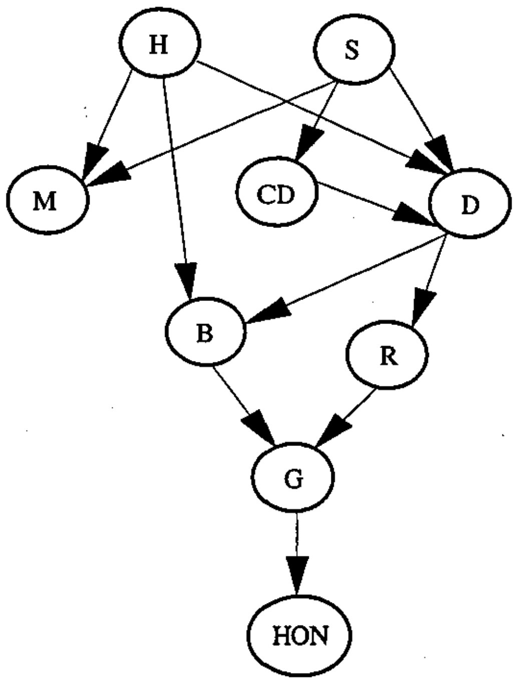
FTD (Feature Transition Diagram) is the representation of all possible solutions at feature level. FTD represents all the possible sequential operations in a directed fashion according to its needs at various levels of geometric and technological constraints. In figure 4.4, a FTD for a straight Round Hole is given. A feature usually has more than one TSF based on its geometric and technological requirements. For example, in figure 4.4, the feature Round Hole has many TSF's, some of them are mentioned below:

1) drilling
2) centre drilling, drilling
3) centre drilling, drilling, reaming
4) centre drilling, drilling, boring
5) centre drilling, drilling, boring, grinding and so on

Each machining feature usually, has got more than one chain of solutions (TSF's) for its processing requirements which can be picked up from the Feature Transition Diagram (FTD). For instance, a Rectangular Step can be produced by milling process or shaping process. The characteristics selected should be permanent ones in the part family formation problem, such that the components can be characterised based upon the fixed characteristics/parameters criteria. Therefore, a single solution or TSF for each feature is selected in order to make the comparison between the parts for their processing requirements. In other words, a chain of manufacturing processes can be allocated to each individual feature according to the process information available at universal level. A chain of processes in a TSF will be decided upon according to the geometric and technological requirements of the feature.

As already mentioned, there is a need to assign a unique solution to each feature. This unique processing solution for each feature can be identified by any optimisation procedure or based on already available resources in the factory.

Feature Transition Diagram (FTD)



H	- Initial State 'Hole'
S	- Initial State 'Solid'
CD	- Centre Drilling
D	- Drilling
M	- Milling
B	- Boring
R	- Reaming
HON	- Honing

Figure 4.4 Technical solution at feature level (TSF)

4.5.2 FEATURES MACHINING SOLUTION WITH THE CONNECTIVITY INFORMATION

For reasoning about component geometry during Process Planning, it is important to capture the structural aspects of component geometry, i.e. to describe the relationships (connectivity) that determine how the component is structured from its constituent features [Gindy *et al.* 1993].

It is not always possible to select manufacturing facilities (machines) to machine features based on their local information (feature level). Features connectivity information must be given due consideration while selecting the machining requirements of individual features. Connectivity analysis plays an important role in assigning the processing facilities to the features, as well as downstream activities of the Process Planning function like set-up determination, operation sequencing, etc. Machine resources attached without the connectivity information would not be realistic ones. Connectivity analysis can be divided into two levels or layers. First level analysis is for the identification of machining facilities required for the processing of the component and second level analysis is the detailed one, required for determining the number of set-ups and sequencing of the operations in the Process Planning function. Set-ups and sequencing of the operations can be achieved by detailed feature connectivity and feature relationships analysis of the constituent features which exist on the body of the component. Second level analysis includes feature relationships like parent/child, intersection, closeness and how the features are connected to the external free surfaces of the part in order to do reasoning for determining the set-ups and sequence of operations. Further discussion about the second level connectivity will be taken in chapter 6.

Just translating the features into processes does not help much in the selection of the resources capabilities required for machining the component features. The same features, sometimes, located on different positions of the component body require different machining capabilities. Connectivity information of the component is required for this purpose, as discussed above. In order to make the point clear, an example is being cited in the following. Consider the example of a long rotational component as given in figure 4.5. The same Round Pocket (blind hole) features at different positions of the component are shown. F1 represents radial Round Pocket, F2 is co-axial Round Pocket and other features represent the PCD pattern of axial Round Pockets. Only the co-axial Round Pocket feature is feasible to machine on conventional lathe. A radial Round Pocket might be processed on a general drilling machine while a pattern of Round Pockets may need horizontal drilling machining facility for their processing because the component is too long. The same is true even with the long prismatic component if it has got features on the extreme ends, having PAD's only on the extreme ends. Components might need different machining facilities if the features are located on different positions. Machining facilities selected at feature level without the first level connectivity reasoning of the component would not be realistic ones. Therefore, not simply feature type and its technological requirements but also other information like connectivity aspects mentioned above provides the basis for reasoning about the selection of the realistic machining facilities.

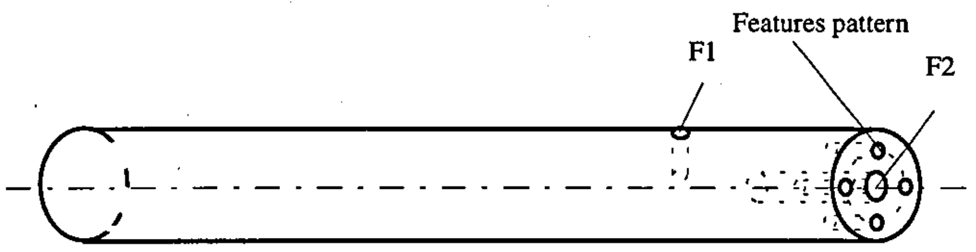


Figure 4.5 A machined component

Bearing this in mind, the author holds the view that Machine Capability Units should be assigned to the features while matching the processing needs of features to the processing capabilities available in the machine shop. It helps in differentiating the parts based on their processing requirements while grouping the similar parts for the Process Planning function. Machine resources can be divided into Machine Capability Units and can be matched to processing needs of the features and components.

A machining process is made up of a number of operations through which a raw material is transformed into parts [Wang and Li 1991]. Operations are the basic components of machining processes. There can be a number of operations in one process i.e. in turning process, operations can be rough turning, semi-finish turning and finish turning. The planning at this level can be termed as 'operation planning' which will be cutting tools selection, cutting parameters selection and sequence of operations. It might not always be possible to carry out the sequence of operations using one process while sequencing the operations in one set-up. The operation is carried out by the unchanged machine tool, unchanged workpiece, and unchanged cutting tool. Though to achieve higher efficiency, more than one tool can be engaged to perform machining on the workpiece in a single setup (e.g. lathe equipped with turret), an operation will be characterised by a single tool. Exception can be given to the straddle milling where two milling cutters are used to perform the operation. In our terminology, an operation is volume to be machined by the single cutting tool. As we have not gone to downstream activity of operation planning, therefore both the words process and operation will be used interchangeably in the rest of the thesis. Sometimes people take different views about the operations. One of the examples is being given in the following paragraph.

The operation is characterised by the unchanged equipment, unchanged workpiece, and continuity. For example, if a batch of the workpieces are machined consecutively by drilling, boring holes, and boring recesses with three different cutting tools at the same workplace (e.g., a lathe), it is single operation. If these holes are drilled on the lathe and then bored on another, the machining of the workpieces will consist of two operations instead of one [Wang and Li 1991].

4.5.3 MACHINE CAPABILITY UNIT

A new term called Machine Capability Unit (MCU) is introduced in this section to divide machine tool capabilities into one or more units to compare the processing needs of parts in the GT paradigm, based mainly on less than a whole-machine basis. This is a division of the machine tool capabilities at the lower level. This is in between the machine tool level and operation level. A MCU is a set of (perhaps similar) operations that can be performed on the least capable machine tool in any machining facility and are tagged. Each MCU can be obtained from one or more machine tools. A machine capability can be considered as consisting of one or a number of MCU's.

The operations involved in a MCU are taken as Machine Capability Unit Elements (MCUE's). After explaining the general logic development for defining the MCU's in a machining facility as well as its utility, the author would come towards the extension of the idea.

The domain of machines in a machining facility can be considered as a universal set of machines. Each machine is a member of the universal set and consists of a set of operations that the machine has.

For example, a machining facility consists of only three machine tools; 1) a drilling machine tool, 2) a boring machine tool, and 3) a lathe. The least capable machine tool in this case is the boring machine providing mainly operations like boring and recessing operations. This Machine Capability Unit can be tagged as MCU#1. The second least capable machine tool is the drilling machine, consisting of operations such as centre drilling, drilling, reaming, tapping, counterboring, countersinking, spotfacing and the capabilities of the MCU#1. The above mentioned operations can be tagged as MCU#2. The drilling machine can be considered as consisting of MCU#1 + MCU#2. The remaining operations i.e. turning, facing, screw-cutting etc. can be tagged as MCU#3. In this case a simple lathe machine will have three MCU's i.e. MCU#1, MCU#2, and MCU#3.

A similar type of concept has been given by Gindy *et al.* [1993]. They use the resource element concept to represent processing capabilities of a machining facility by a set of resource elements in the process planning function. The basis for defining the resource elements is not given in the publication.

The breakdown of machine tools into resource elements allows Process Plans generation to be based on utilising parts of machine tool capabilities. It therefore provides better links with the machine shop scheduling system and minimises the *ad hoc* re-planning of component processing on the shop floor. It also allows greater flexibility to take into account any company specific machining strategy (flow line, machining centre, conventional machine tools, etc.) that may be required for organising manufacturing activities [Gindy *et al.* 1993].

Definitions of MCU's really depend upon the available machine tools in any machining facility as they are defined based on the set of operations which any least machine will consist of. A general method or algorithm can be devised for defining the MCU's.

4.5.3.1 THE ALGORITHM FOR DEFINING THE MCU'S

The algorithm for defining the MCU's which is based on the set theory concept, can be summarised in the follows:

Step 1: initialise a variable m = 0 (variable m is used for coding the MCU's i.e. MCU#m),
Mmax = number of maximum machines present in any machining facility,
M = Mmax (current number of machines).

Step 2: From all combinations of M machines, record those combinations, N, with a non-empty intersection set of operations.

Step 3: check if N > 0 then code the new MCU#m (N consisting of ordered set of operations) by giving an increment to the variable m by 1, decrement to variable N by 1 and delete those shared operations from the M machines. Repeat the step until N > 0 is true.

else
decrease the number M by 1.

Step 4: check if number of machines i.e. M > 1, then repeat from step 2.

else
go to step 5.

Step 5: initialise M1 = number of machines left with non-empty set of operations.

Step 6: increment variable m by 1 in order to code new MCU for the undeleted set of operations for a machine and delete machine i.e. M1 = M1 - 1

Step 7: if M > 1, repeat step 6 until the number of M1 is 0.

else
stop.

The flow chart for the algorithm is shown in figure 4.6. After defining the MCU's at this level, the MCU's needs to be divided at lower level as discussed latter.

4.5.3.2 AN EXAMPLE

To further elaborate on the algorithm, figure 4.7 has been drawn to define the MCU's. The machine resources shown in the figure are the members of the universal set of machines in a machining facility. Each operation in the universal set occupies the space in the universal set, therefore machines are shown occupying the space their operations occupy. In the figure, some machines are sharing the space as those operations are common between the machines and some machines occupy the space in the universal set exclusively as those machines do not share the operations with other machines. By working out the algorithm as given above, the MCU's defined in the figure are explained in the following:

Explanation

In the example given, a universal set consists of six machines as shown in the figure. All the machines except machine 5 have shared machining capabilities in terms of operations. By working out algorithm, explanation is given for each iteration in the following:

Iteration 1

Step 1: $M = 6$ and $m = 0$.
Step 2: For 6 machines, N (number of non-empty intersection set of operations) is 0.
Step 3: As $N > 0$ is not true therefore $M = 6 - 1 = 5$ (no MCU is defined in this iteration).
Step 4: As $M > 1$, repeat from step 2.

Iteration 2

Step 2: For 5 machines, N (number of non-empty intersection set of operations) is 0.
Step 3: As $N > 0$ is not true therefore $M = 5 - 1 = 4$ (no MCU is defined in this iteration).
Step 4: As $M > 1$, repeat from step 2.

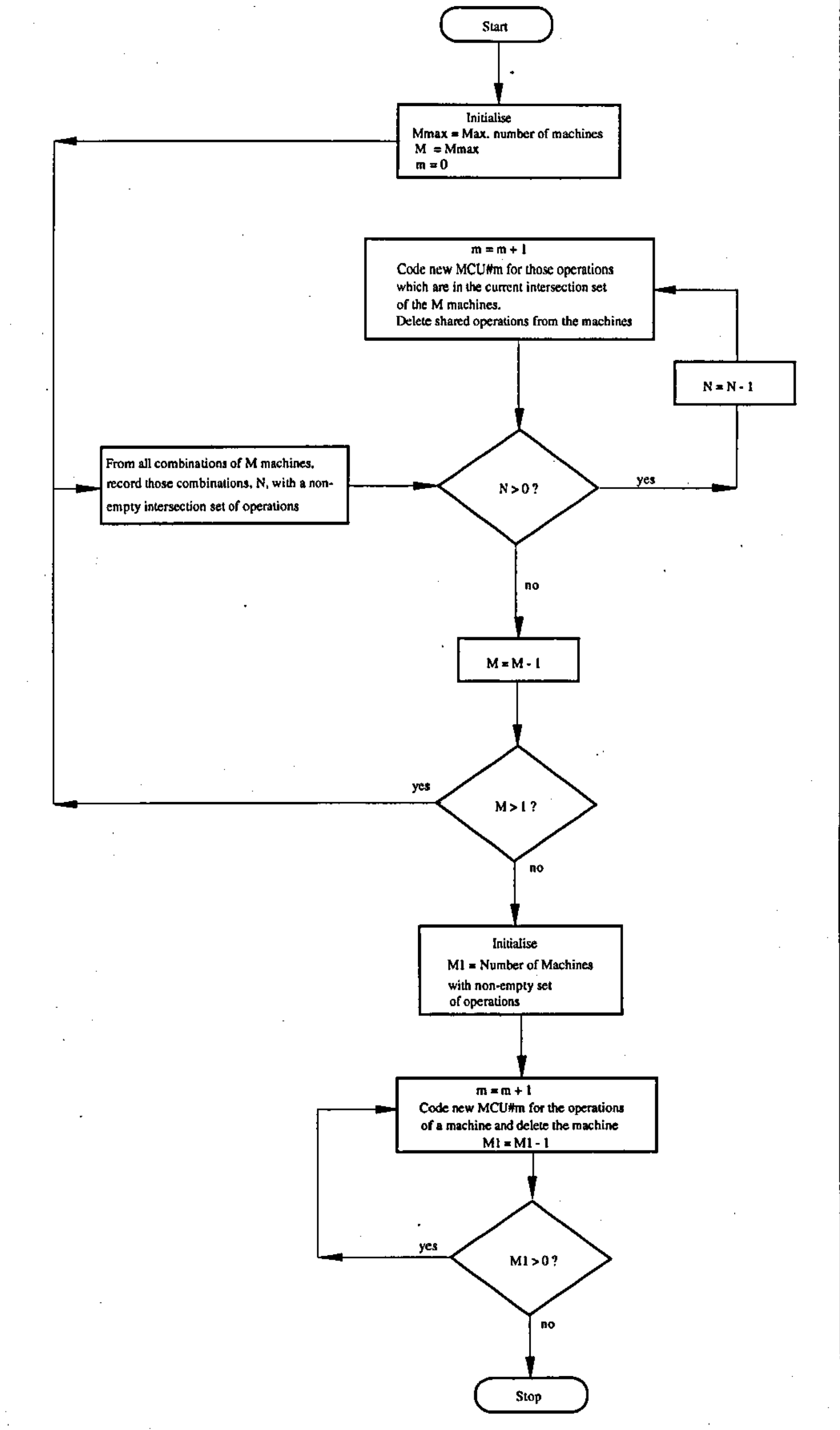

Start
Initialise
Mmax = Max. number of machines
M = Mmax
m = 0
m = m + 1
Code new MCU#m for those operations which are in the current intersection set of the M machines.
Delete shared operations from the machines
N = N - 1
From all combinations of M machines, record those combinations, N, with a non-empty intersection set of operations
N > 0 ?
yes
no
M = M - 1
M > 1 ?
yes
no
Initialise
M1 = Number of Machines with non-empty set of operations
m = m + 1
Code new MCU#m for the operations of a machine and delete the machine
M1 = M1 - 1
M1 > 0 ?
yes
no
Stop

Figure 4.6 A flow chart for defining the MCU'S

Iteration 3

Step 2: For 4 machines, N (number of non-empty intersection set of operations) is 0.
Step 3: As $N > 0$ is not true therefore $M = 4 - 1 = 3$ (no MCU is defined in this iteration).
Step 4: As $M > 1$, repeat from step 2.

Iteration 4

Step 2: For 3 machines, N (number of non-empty intersection set of operations) is 1 and $m = 1$.
MCU#1 is coded (delete shared operations).
By decreasing N by 1, it is 0 now.
Step 3: As $N > 0$ is not true therefore $M = 3 - 1 = 2$
Step 4: As $M > 1$, repeat from step 2.

Iteration 5

Step 2: For 2 machines, N (number of non-empty intersection set of operations) is 3 (i.e. intersections sets between machine 1 and 3, machine 2 and 3, machine 4 and 6 are non-empty).
Define MCU#2, MCU#3, MCU#4 respectively by decreasing N by 1, and increasing m by 1 during the course of definition of these MCU's and delete shared operations.
At the end $N = 0$ and $m = 4$.
Step 3: As $N > 0$ is not true therefore $M = 2 - 1 = 1$.
Step 4: As $M > 1$ is not true, go to step 5.
Step 5: Initialise M1 (number of machines left with non-empty set of operations) = 5 (i.e. machine 1, machine 3, machine 4, machine 5, machine 6).
Step 6: By increasing m by 1 and decreasing M1 by 1 each time, MCU's defined are MCU#5, MCU#6, MCU#7, MCU#8, MCU#9.
Step 7: Now $M1 = 0$ therefore stop.

4.5.3.3 EXTENSION IN THE DEFINITION OF MCU'S

The author uses MCU concept in the part grouping process. The idea behind this is to differentiate between the components based on the realistic processing capabilities, they need for their processing. As already explained in the previous section that simply translating the features into processes does not help much in the selection of different machine tool capabilities required for machining the component features (i.e. connectivity aspect of the features also plays an important role in this connection). The same feature types, sometimes, located on different positions of the component body require different machining capabilities. Therefore the structural aspect of a machine along with the processes it provides, is important for matching the true processing needs of the parts.

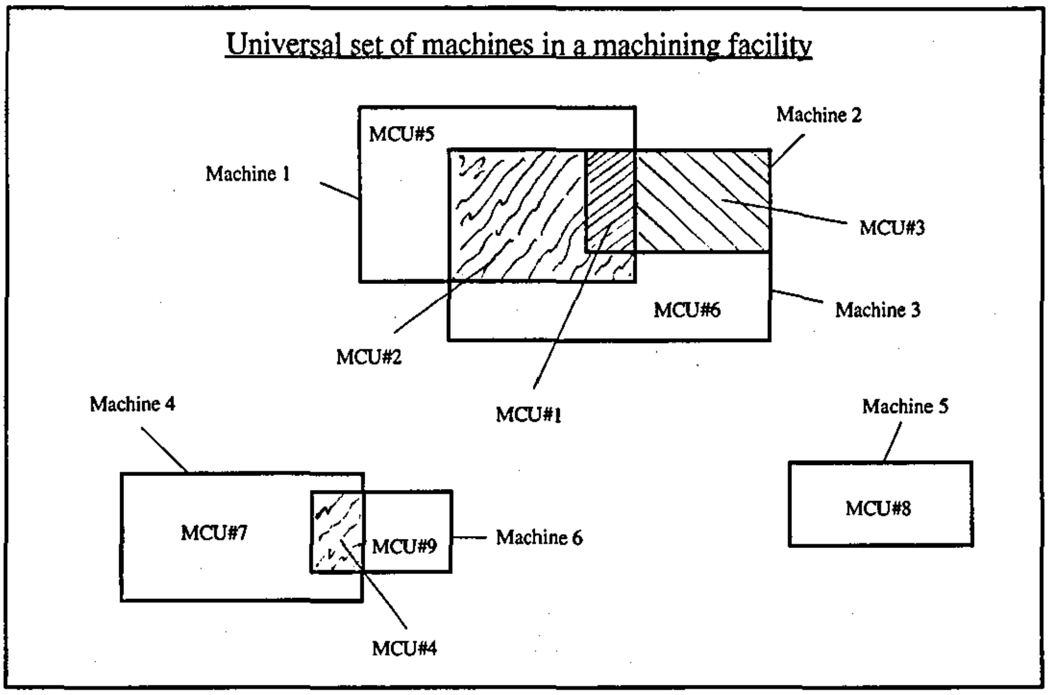


Figure 4.7 Definition of MCU's

The point is that two components should not be taken as similar if for the same set of operations, they require machining at different machine tools. The criteria for calculating the similarity between the parts should be based on the MCU's they require for their machining, and not the machines. This is to make part processing needs machine independent. An example of a set of drilling MCU i.e. MCU#2 (as already tagged in the previous example) consists of operations or MCUE's (i.e. centre drilling, drilling, reaming, tapping, counterboring, countersinking, spotfacing) can again be cited here. This MCU is available on different drilling machine tools (differentiated based on the structural aspect), for example, 1) capability available on drill press or upright drilling machine i.e. for common drilling, 2) capability available on the lathe machine i.e. for co-axial drilling, and 3) capability available on the horizontal drilling machine i.e. for drilling the features at the extreme ends of a long component. The allocation of any machine resource among the mentioned above for the processing of the component feature requiring a set of drilling operations is decided based on the position of the feature on the component. Therefore, it becomes necessary to code different MCU's even for the same set of operations where machines from same family with different structures involve. This can be considered as definition of MCU's at lower level which are available at a family of similar machines (in terms of operations) but different structure wise. Therefore, the drilling capability unit, tagged as MCU#2 can further be divided at lower level and tagged as MCU#2_a, MCU#2_b, and MCU#2_c respectively for above mentioned three cases, so that the realistic processing capabilities can be identified for the components. This is similar to hierarchical coding structure which inherits information about its upper classes. In this case, MCU#2 describes a particular set of operations and coding attached i.e. a, b, c with MCU#2 mention the set of operations available on the family of machines that are different in structures.

Another example in the case of components requiring the milling process can be given here. For example, a very narrow deep 'Through Slot' requires a horizontal milling machine facility (characterised based on the feature geometrical parameters), whereas square/rectangular pockets require a vertical milling facility (decided based on the feature type). In the case of the above mentioned machining facilities, the milling oriented MCU, for instance, MCU#m can be characterised at lower level i.e. a horizontal milling machine and a vertical milling machine capabilities can be characterised and tagged as, for instance, MCU#m_a and MCU#m_b.

Similarly, the heavy disk type rotational components are turned on vertical lathe. A turning oriented MCU can further be classified i.e. MCU#n, for example, as MCU#n_a for common lathe and MCU#n_b for vertical lathe.

The MCU concept can again be extended to further lower level which can classify the MCU's based on the envelops of the machine tools. For example, if two machines exist in the firm which can provide the MCU#2_c and they are exclusive dimension wise. In that case, the capabilities of the machines can be classified further and tagged, for instance, as MCU#2_c_x for small one and MCU#2_c_y for big one. But the classification at this level would not help if the machines are sharing the dimensional boundaries. It is hard to find the machines from a family in a firm or shop which do not have shared dimensional boundaries with other machines. Keeping this in view, the author is not going to this level in the implementation of part grouping process.

It is therefore, necessary for different types of machines even from the same family, to be differentiated based on their structural aspect and different MCU's tagged for them. Components should then be characterised for the grouping function based on these real processing requirements i.e. MCU's. The MCU concept helps in comparing the capabilities of the machine tools in the case of determining the similar components for the Process Planning function, scheduling/loading and re-planning of the components.

The idea of extending the definition of MCU's to include structural aspects and the working envelopes of the machine tools is needed. But as we go too deep in dividing MCU's, there is a danger of increasing the number of MCU's defined for a machining facility. This issue of increasing the MCU's will, in turn, increase the number of grouping parameters, thus affecting the grouping results. Keeping this in view, there is a need that definition of MCU's is general enough but representative of different machine capabilities available in a machining facility. The author keeps the view that this is a difficult balance to keep in the definition of MCU's.

A conceptual manufacturing capability model for a machining facility in an organisation/shop can be depicted as shown in figure 4.8.

4.5.3.4 MACHINE CAPABILITY UNITS (MCU'S) DEFINITION IMPLEMENTATION

MCU's defined in practical terms in the part grouping process are detailed below. Real data about machine tools from GEC Alsthom Large Machines Limited, Rugby, England, is used in case studies carried out in this work. Twenty machine tools have been taken as a universal set to define MCU's. The steps involved are given in detail as below:

MCU's definition implementation is shown pictorially in figure 4.9.

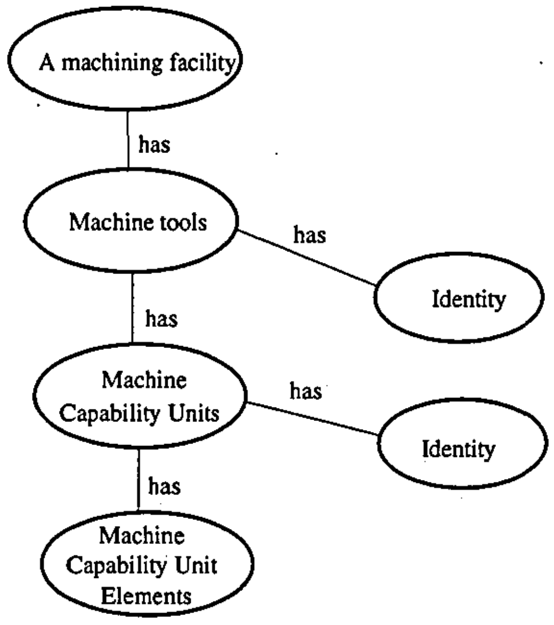
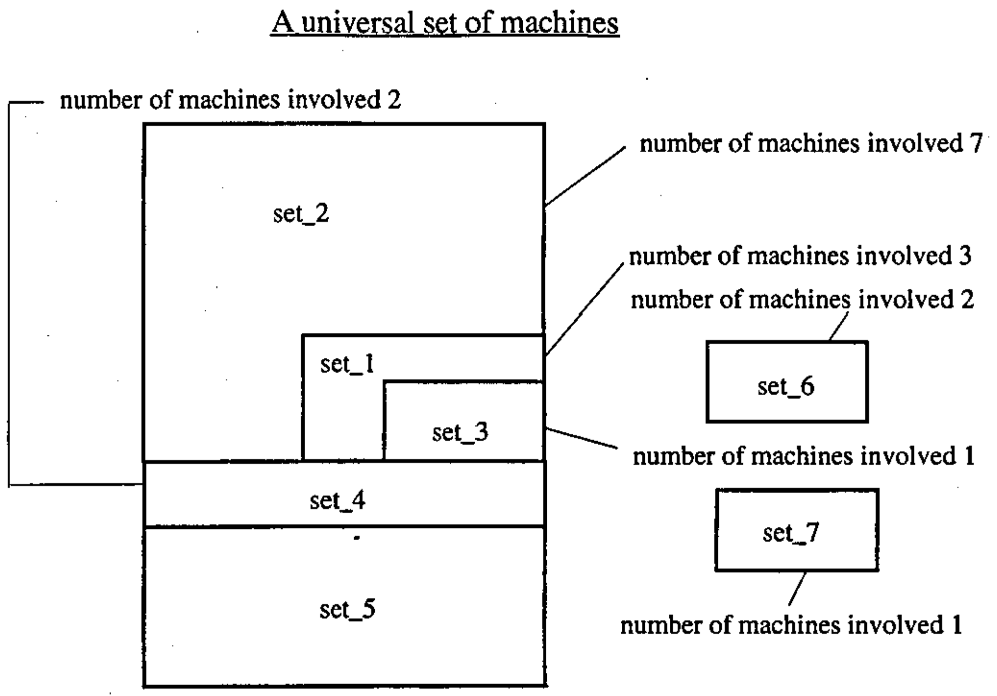


Figure 4.8 Conceptual machining capability model for a machining facility.


A universal set of machines
number of machines involved 2
number of machines involved 7
set_2
number of machines involved 3
number of machines involved 2
set_1
set_6
set_3
number of machines involved 1
set_4
set_7
set_5
number of machines involved 1

number of machine consisting of set_4 + set_5 = 2 (i.e. HOR, VER)

number of machine consisting of set_1 + set_2 + set_3 + set_4 = 1 (i.e. SHI)

number of machines consisting of set_1 + set_3 + set_4 + set_5 = 1 (i.e. MAK)

Figure 4.9 MCU definition implementation

MACHINE IDENTIFICATION CODING

The same machine identification codes are used, as used by GEC, Rugby.

MACHINE
HOR
VER
KTW
BOR
WTD
MAK
SMU
UNV
EDG
END
MGR
KEY
SHI
MRD

HEI
BUL
LAN
SLT
QKO
CAP

THE SETS OF OPERATIONS

Sets of (perhaps similar) operations usually performed on the same machines are as follows:

set_1 = {drilling, centring, taping, reaming, boring, grooving, chamfering, spotfacing, counterboring, counteringsinking}

set_2 = {turning, facing, screw_cutting}

set_3 = {boring, grooving, chamfering}

set_4 = {wibbling (keyway cutting)}

set_5 = {milling}

set_6 = {grinding}

set_7 = {slotting}

THE SET OF MACHINES WITH THE RELEVANT SET OF OPERATIONS

Drilling machines consisting of operations in set_1 + set_3 = {WTD, MRD, KTW} → number of machines involved 3

Boring machine consisting of operations in set_3 = {BOR} → machine involved 1

Lathe machines consisting of operations in set_1 + set_2 + set_3 = {SMU, END, HEI, BUL, LAN, QKO, CAP} → number of machines involved 7

Slotting machine consisting of operations in set_7 = {SLT} → number of machine involved 1

Grinding machines consisting of operations in set_6 = {UNV, MGR} → number of machine involved 2

Wibbling (keyway_cutting) machines consisting of operations in set_4 = {EDG, KEY} → number of machines involved 2

Milling machines consisting of operations in set_4 + set_5 = {HOR, VER} → number of machines involved 2

Milling centre consisting of operations in set_1 + set_3 + set_4 + set_5 = {MAK} → number of machine involved 1

Lathe centre consisting of operations in set_1 + set_2 + set_3 + set_4 = {SHI} → number of machine involved 1

MCU'S DEFINED AT FIRST LEVEL

13 machines consist of operations in set_3 → MCU_1

12 machines consist of operations in set_1 → MCU_2

8 machines consist of operations in set_2 → MCU_3

4 machines consist of operations in set_4 → MCU_4

3 machines consist of operations in set_5 → MCU_5

2 machines consist of operations in set_6 → MCU_6

1 machine consist of operations in set_7 → MCU_7

MCU'S DEFINED AT SECOND LEVEL

MCU_1 into MCU_1_a, MCU_1_b, MCU_1_c

MCU_3 into MCU_3_a, MCU_3_b, MCU_3_c

MCU_5 into MCU_5_a, MCU_5_b

FINAL MCU'S DEFINED

1. MCU_1_a consisting of operations {c_drilling, c_centring, c_taping, c_reaming, c_spotfacing, c_counterboring, c_counteringsinking}
2. MCU_1_b consisting of operations {drilling, centring, taping, reaming, spotfacing, counterboring, counteringsinking}
3. MCU_1_c consisting of operations {a_drilling, a_centring, a_taping, a_reaming, a_spotfacing, a_counterboring, a_counteringsinking}
4. MCU_2 consisting of operations {turning, facing, screw_cutting}
5. MCU_3_a consisting of operations {boring, grooving, chamfering)

6. MCU_3_b consisting of operations {c_boring, c_grooving, c_chamfering)
7. MCU_3_c consisting of operations {a_boring, a_grooving, a_chamfering)
8. MCU_4 consisting of operation {wibbling}
9. MCU_5_a consisting of operation {hor_milling}
10. MCU_5_b consisting of operation {ver_milling}
11. MCU_6 consisting of operation {grinding}
12. MCU_7 consisting of operation {slotting}

4.5.4 REFERENCE MODEL FOR THE CAFBG SYSTEM

A reference model for the CAFBG System has been shown in figure 4.10. A component model has been parsed into two levels of information i.e. component level and feature level. Component level information includes; overall size (which results in dimensional ratios and overall shape), feature patterns, feature relationships (parent/child, geometric tolerances, etc.), feature list and feature connectivity. Feature level information includes feature geometry and technological constraints. Each feature geometry is associated with a FTD. The FTD generates a number of operation sets (responsible for the machining of the geometry of the feature while fulfilling technological requirements) called TSF's (refer to section 4.5.1 for description of FTD and TSF) of the feature and TSF's are decided based on the feature technological constraints. A single TSF is assigned to a feature in our case. MCUE's from the TSF are decided with the help of complementary information like feature geometric parameters/type and connectivity and then matching the requirements with the MCU's available in the machine shop. These Machine Capability Unit Elements (MCUE's) can be termed as feature state elements. The feature state element term has been used for operations by Chang [1990].

4.5.5 CALCULATING APPROACH DIRECTIONS

After attaching the MCUE's to each feature as shown in figure 4.10, component AD's (approach directions) can be determined by clustering the MCUE's (operations) from common component PAD's. MCUE's may appear in more than one cluster in different component PAD's. The final component AD's are selected by minimising the number of clusters. This is done based on step by step selection of clusters containing the maximum number of MCUE's and then removing those MCUE's from the remaining clusters in the other component PAD's. The result would be the number of AD's from which all component features can be processed.

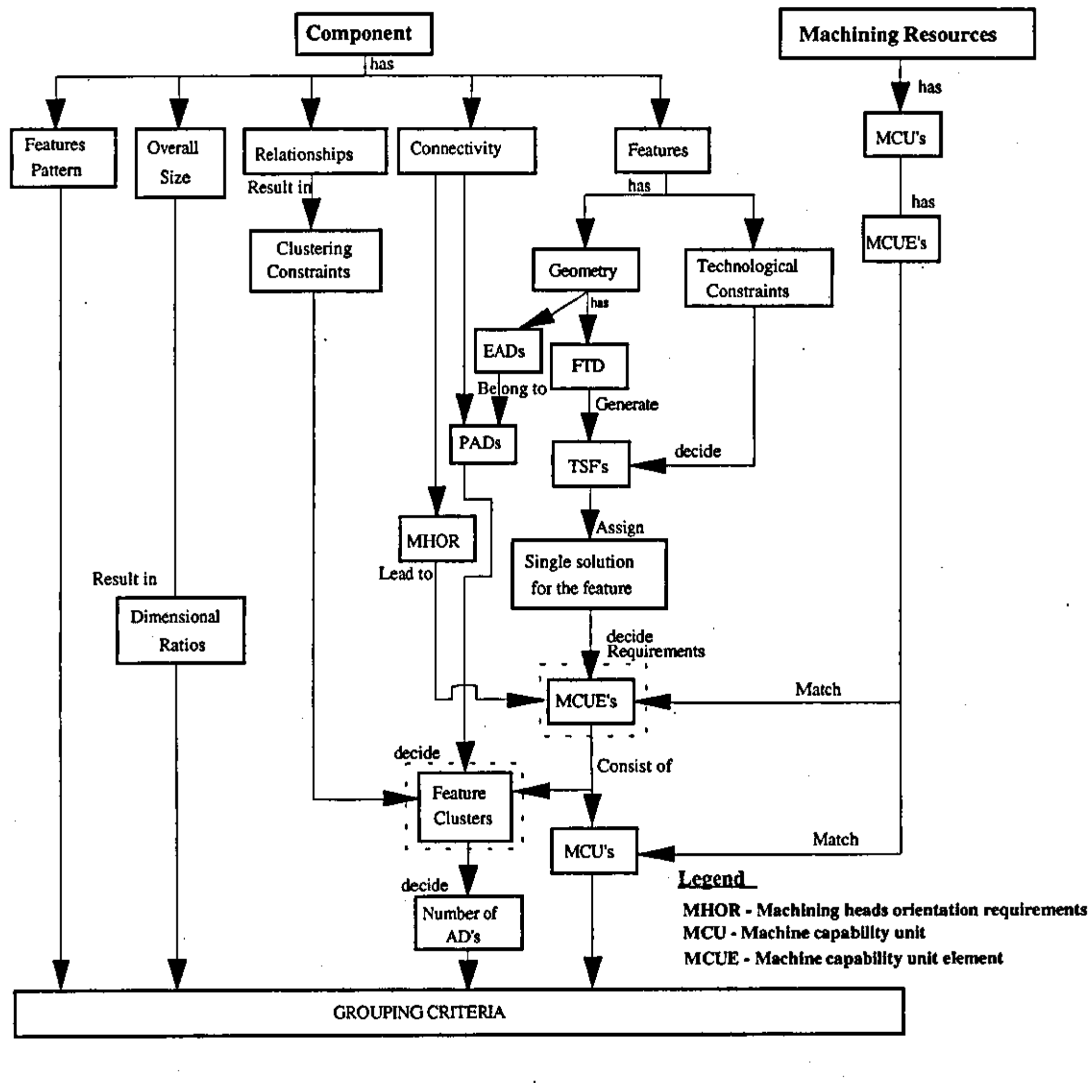


Figure 4.10 The Reference Model for the CAFBG System.

Though all other relations influence component set-ups and operation sequences it seems difficult to consider all of them at the same time in the clustering process.

It is important that clustering is performed while observing the precedence relationships that may exist between component features. It is almost impossible, however, to pre-define all the possible feature relationships (concentricity, parallelism, perpendicularity, etc.) and the implications that multi-relationships between features may have on the component Process Plan. Many of the issues involved in making these decisions are company specific and experience-based [Gindy *et al.* 1993].

Features which have geometric relations in common and have the same manufacturing directions (AD's) are grouped together as a group of features for the purpose of machining them together in the Process Planning function. The number of AD's is another proposed criterion for the part grouping in order to determine the work contents required for the component from the manufacturing point of view.

4.5.6 OTHER DESIGN INFORMATION

Dimensional ratios come from the component's overall dimensions. Design information like features pattern and dimensional ratios have also been proposed as the classifying attributes because of their importance in the manufacture of the component (already discussed in detail elsewhere in this chapter).

4.5.7 THE PROPOSED CLASSIFYING ATTRIBUTES FOR THE PART GROUPING

The selected parameters/characteristics for part grouping are given as follows:

- Dimensional ratio/ratios of the component
- Feature based processing requirements to machine the component after matching the MCU's available in the shop.
- Number of approach directions (AD's) to machine the component.
- Features pattern

A reference model for the part grouping function is given in figure 4.10.

4.5.8 GROUPING ALGORITHM DEVELOPMENT

For each particular application goal, the user can select several classifying attributes in order to use a general clustering procedure to generate the part families [Tornincasa 1991].

Cluster analysis is the name used for a variety of mathematical methods, used to find out which objects in a set are similar. Cluster analysis is one of the most frequently applied mathematical tools in Group Technology (GT). According to Kaufman [1990], cluster analysis has become known under a variety of names, such as numerical taxonomy, automatic classification, botryology, and typological analysis. Cluster analysis is rearranging the groups of data points that possess strong internal similarities. There are two main functions involved in cluster analysis. The first one is parameters extraction. Other words used for parameters are characteristics or features or classifying attributes. The purpose of parameters extraction is to reduce the data by measuring certain parameters or properties that distinguish between objects. According to Fu [1980], feature extraction is problem dependent, that is, the extraction of relevant features for classification depends upon the patterns and the number of classes under study. The second one is classification, which is basically, partitioning the data space into regions, one region for each category or class.

Consider a sample consisting of n objects and every object having p features. The n parts can be thought of as n points in a p-dimensional Euclidean Space (E-Space) and clustering is to group the points which are in close proximity as shown in figure 4.11.

According to this procedure, a part-characteristic matrix is prepared having n×p dimensions where n is the number of parts represented on the rows side and p is the number of parameters/characteristics represented as columns. Membership of each component is defined in the matrix based on already defined parameters. After the preparation of a part-characteristic matrix, the procedure will start working in a loop. The loop will run n-1 times leaving all the components in one group. At each iteration, similarity between each pair is calculated resulting in a square matrix of size equivalent to a number of clusters at that time. The two nearest clusters or groups are merged by taking the union of parameters of both in order to create the representative cluster. At the next iteration this representative will take the place of both of them.

The classification problem is based on measurement of similarity (or dissimilarity) between two samples. A clustering investigation starts from defining a suitable similarity function or distance function and computing the similarity matrix between all pairs of members. Then classification is partitioning the samples in such a way that the similarity between samples in the same cluster is significant more than the similarity between samples in different clusters.

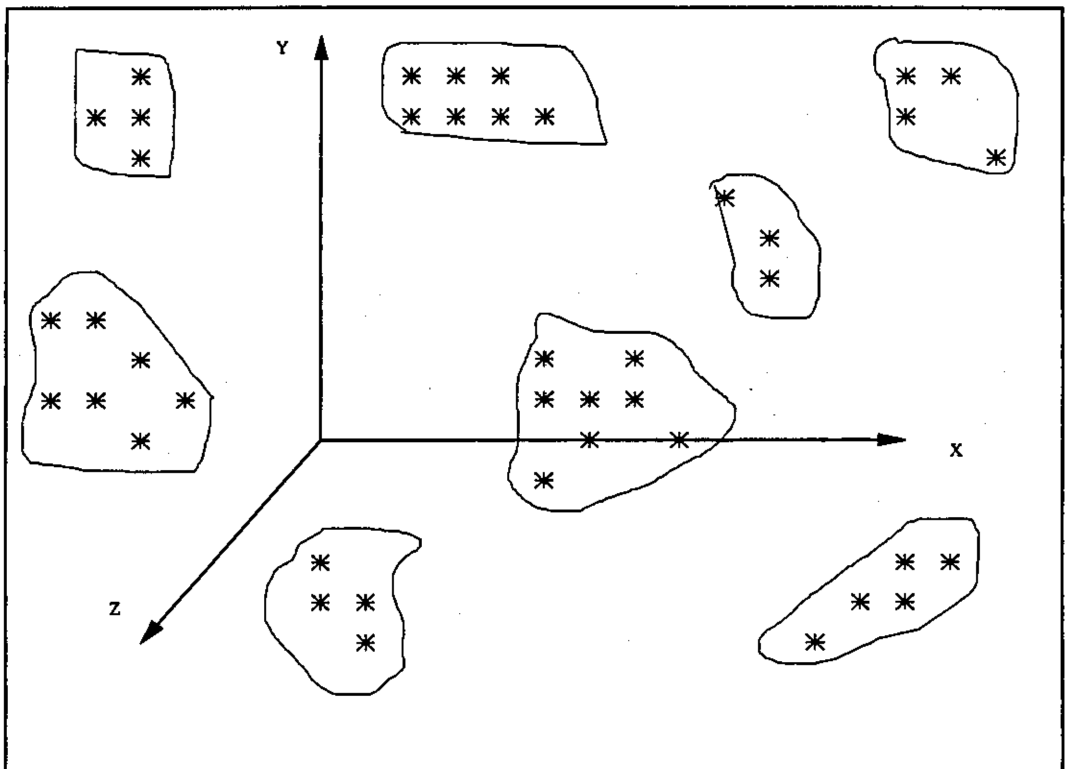


Figure 4.11 Clustering the objects in the space.

Mostly hierarchical cluster analysis methods are used. Methods of hierarchical cluster analysis follow a prescribed set of steps, the main ones being [Romesburg 1984]:

(1) collecting a data matrix whose columns stand for the classifying attributes of the objects and whose rows are for the objects
(2) standardising (optionally) the data matrix
(3) computing the values of a resemblance coefficient to measure the similarities among all pairs of objects

(4) executing the clustering method to process the values of the resemblance coefficient to form the groups of objects having similar attributes.

The cluster analysis approach provides algorithms for the study of similarities between objects in a quantitative manner, in contrast with other classification techniques, which tends to be descriptive. In other words clustering is the classification of objects based on their possession or lack of defined characteristics. The three steps involved are discussed in detail in the following:

4.5.8.1 PREPARING A PART-CHARACTERISTICS MATRIX

Suppose there are n objects to be clustered. Clustering algorithms typically operate on the two input structures. The first represents the objects by means of p measurements or parameters. These measurements can be arranged in an n×p matrix, where the rows correspond to the objects and the columns to the parameters. When the f-th measurement of the i-th object is denoted by X_{if} (where i = 1,..., n and f = 1,..., p) this matrix looks like [Kaufman 1990]:

$$\begin{array}{cc} & \text{p variables} \\ \text{n objects} & \begin{bmatrix} x_{11} & \cdot\ \cdot & x_{1f} & \cdot\ \cdot & x_{1p} \\ \cdot & & \cdot & & \cdot \\ \cdot & & \cdot & & \cdot \\ x_{i1} & \cdot\ \cdot & x_{if} & \cdot\ \cdot & x_{ip} \\ \cdot & & \cdot & & \cdot \\ \cdot & & \cdot & & \cdot \\ x_{n1} & \cdot\ \cdot & x_{nf} & \cdot\ \cdot & x_{np} \end{bmatrix} \end{array}$$

This indicates whether already established characteristics are either present or absent. Presence or absence is coded within a close interval [0,1] in the matrix. Value 0 being for absence and other than 0 for presence.

Each component can be represented by a vector $X_i = \{x_{i1}, x_{i2}, x_{i3}, \ldots\ldots, x_{ip}\}$

where x_{ip} are coefficients denoting presence or absence of characteristics within a close interval [0,1] for each component. It will represent the component membership based on defined parameters. A membership function can then be defined for each characteristic. This is a the most important step in a cluster analysis process. According to Xu and Wang [1989] this step is critical as it accommodates the experience and human judgement in the mental model of a decision maker. Because the definition of a membership function is to some extent subjective, caution should be taken to ensure that they are close to a real presentation of the characteristics of each feature in the machining process. The weightage of features can be employed here to produce more precise description of the parts. In the case of this research all the characteristics have been given full membership except the number of AD's where 0.2 weightage has been given to this classifying attribute to give this attribute less importance as compared to other attributes.

4.5.8.2 COMPUTING A SIMILARITY COEFFICIENT MATRIX

A resemblance or similarity coefficient is a measurement which measures the overall resemblance - the degree of similarity, between each pair of objects. Its value is computed by entering, for a given pair of objects, the values from their columns in the data matrix. A formula is used to calculate this that gives one value that represents how similar the pair of objects are. A resemblance coefficient is always one of two types: (1) a dissimilarity coefficient or (2) a similarity coefficient.

The difference between a similarity coefficient and a dissimilarity coefficient is a matter of 'direction'. The smaller the value of the dissimilarity coefficient, the more similar the two objects are. The larger its value, the more dissimilar they are. Conversely, the larger the value of a similarity coefficient, the more similar the two objects are, and the smaller its value, the more dissimilar they are. A clustering method is a series of steps that incrementally reduce the size of resemblance matrix.. At the same time, it develops the tree from the similarity values. At the last step, the resemblance matrix disappears completely, making the tree complete [Romesburg 1984].

From the information contained in the data matrix, the similarity between each pair of objects can be evaluated. This similarity coefficient matrix is based on the extent to which the parts share common characteristics. The similarity coefficient value varies also within the interval [0,1]. The value of 1.0 being when parts are totally identical and 0 when they have nothing in common.

One of the most famous procedures adapted by McAuley [1972] which illustrates the basic mechanism to most clustering applications is given here. He examined the measure of similarity between two machines using assignment information such as that in figure 4.12.

		Machine j	
		1	0
Machine j'	1	a	b
	0	c	d

Figure 4.12 Binary assignment data format.

Here a is the count of parts processed on machine j and j', d is the number of parts processed on neither machine j or j', and b and c are the number of parts processed on machine j' only, and machine j only, respectively. McAuley defined the measure of association between machine j and j' as

$$S_{jj'} = \frac{a}{(a+b+c)}$$

Another commonly used procedure to calculate the measure of similarity between two objects i and j is as below:

$$s(i,j) = 1 - d(i,j)$$

Where d(i,j) is the measure of dissimilarity or distance between them which can be calculated by Euclidean distance formula as below:

$$d(i,j) = \sqrt{[x_{i1}-x_{j1}]^2+[x_{i2}-x_{j2}]^2+\ldots+[x_{ip}-x_{jp}]^2}$$

Where x_{ip}, x_{jp} are the parameters of objects i and j respectively. Similarity s(i,j) typically takes on values between 0 and 1, where 0 means that i and j are not similar at all and 1 reflects maximum similarity. Values in between 0 and 1 indicate various degrees of resemblance. Following conditions are satisfied:

1) $0 \le s(i,j) \le 1$
2) $s(i,i) = 1$
3) $s(i,j) = s(j,i)$

Other approaches for similarity and distance calculations can be found in the publications by Mosier [1989] and Kaufman [1990]. Recently, a comprehensive survey on similarity and distance measures for Cellular Manufacturing has been given by Shafer and Rogers [1993].

The similarity between two parts is the closeness between two points (a vector) in the E-Space.

The similarity function between two parts is being calculated in this research by the following formula.

$$r_{ik} = \frac{\sum_{j=1}^{p} \min\left[X_{ij}, X_{kj}\right]}{\frac{1}{2}\sum_{j=1}^{p}\left[X_{ij}, X_{kj}\right]}$$

Where r_{ik} is similarity between i-th sample and k-th sample, p is number of features/characteristics and i, k = 1, 2, ..., n. The above mentioned formula has been borrowed from Fuzzy Cluster Analysis approach to deal with the different weight values [Xu and Wang 1989].

The pair wise similarities may be given in the following similarity matrix.

$$S = \begin{bmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ r_{21} & r_{22} & \cdots & r_{2n} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ r_{n1} & r_{n2} & \cdots & r_{nn} \end{bmatrix}$$

This will be a square matrix satisfying all three conditions mentioned above.

4.5.8.3 CLUSTERING

Cluster analysis examines the similarity between each pair of objects and forms groups of objects so that within each group the objects are highly similar to each other. An analogy is to think in terms of distance between objects in space. If each object can possess up to p characteristics they can be considered as occupying positions in an p-dimensional space. Objects which are highly similar to one another are close together in this space. Cluster analysis seeks to find groups of objects forming distinct clusters.

Because of their conceptual simplicity, hierarchical clustering procedures are among the best-known methods. The procedures can be divided into two distinct classes, agglomerative and divisive. Agglomerative (bottom-up, clumping) procedures start with n singleton clusters and form the sequence by successively merging clusters. Divisive (top-down, splitting) procedures start with all of the samples in one cluster and form the sequence by successively splitting clusters [Duda and Hart 1973].

CLUSTERING ALGORITHM

The clustering methodology used in this work is called "*Agglomerative hierarchical clustering*". Agglomerative clustering procedures are popular because of their simplicity in the implementation. The problem can be stated as:

Consider a sequence of partitions of n samples into c clusters. The first of these is a partition into n clusters, each cluster containing exactly one sample. The next is a partition into n - 1 clusters, the next a partition into n - 2, and so on until the nth, in which all the samples form one cluster. At any level k, we can say that we are at group $c = n - k + 1$ [Duda and Hart 1973]. Given any two samples x and x', at some level they will be grouped together in the same cluster. If the sequence has the property that whenever two samples are in the same cluster at level k they remain together at all higher levels, then the sequence is said to be a hierarchical clustering.

The algorithm for the agglomerative clustering can be summarised as follows:

Step 1 Initialise the number of groups G = number of components where each group can be represented as p-dimensional vector, given below:
$C_i = \{c_{i1}, c_{i2}, c_{i3}, \ldots, c_{ip}\}$
In the beginning each component is treated as a group.

Step 2 Find the two maximum similar groups, say C_i and C_j. These will come up by calculating the similarity value between each individual pair of clusters or groups

Step 3 Merge C_i and C_j by taking their union. C_i will be the centre or representative of the new group formed, delete C_j, and decrement G by one. Write the grouping results in a file and the number of groups and the similarity level at which two groups are merged in another file at each iteration. The second file will be later used to find out the number of groups at which the grouping process should be stopped.

Step 4 Repeat the steps 2 - 4 until the similarity level is zero or all the groups are merged into one group..

For every hierarchical clustering there is a corresponding tree, called a dendogram, that shows how the samples are grouped. Figure 4.13 shows a dendogram for a hypothetical problem involving six samples. Level 1 shows the six samples as singleton clusters. At level 2, samples x3 and x5 have been grouped to form a cluster, and they stay together at all subsequent levels. The dendogram is usually drawn to scale to show the similarity between the clusters that are grouped. In figure 4.13, for example, the similarity between two groups of samples that are merged at level six has a value of 30. The similarity values can be used to help determine whether the groupings are natural or forced.

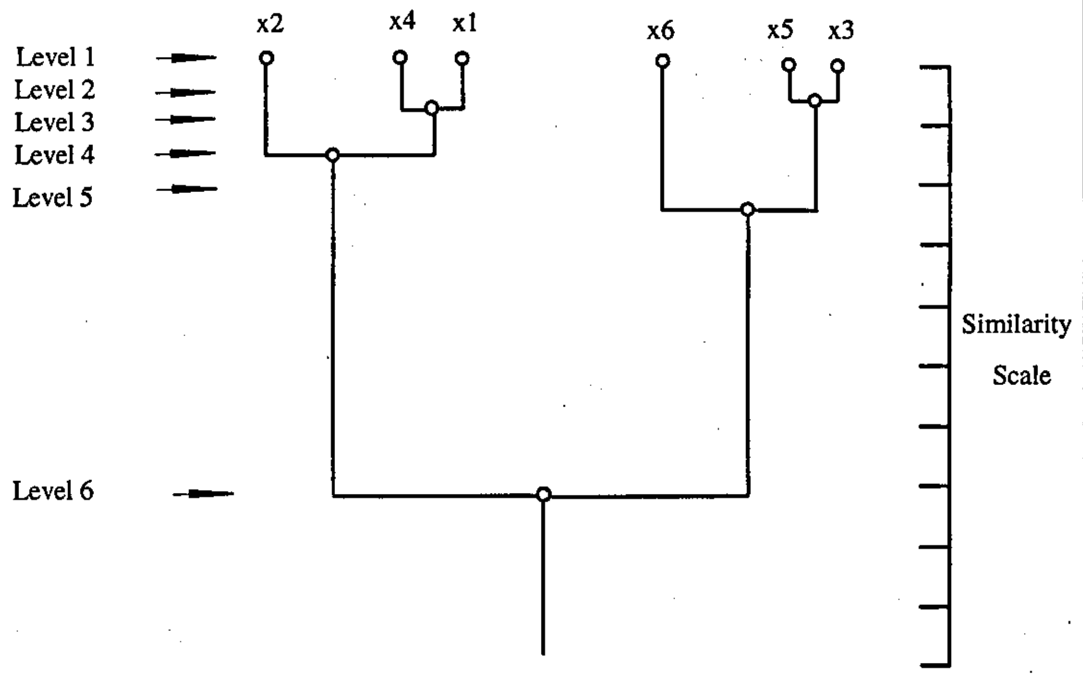


Figure 4.13 A dendogram for hierarchical clustering.

Adapted from Duda et al. [1973]

4.5.9 STOPPING THE CLUSTERING PROCESS

After selecting the classifying attributes for the part grouping process, a cluster analysis approach has been adapted to find out which components in a given domain are similar or in other words by using cluster analysis, components with similar descriptions are mathematically gathered into the same clusters.

Let us consider a sequence of partitions of the n samples into c clusters. The first of these is a partition into n clusters, each cluster containing exactly one sample. The next is a partition into n-1 clusters, the next a partition into n-2, and so on until the nth, in which all the samples form one cluster. Given any two samples x and x', at some level they will be grouped together in the same cluster. If the sequence has the property that whenever two samples are in the same cluster at level k they remain together at all higher levels, then the sequence is said to be a hierarchical clustering.

A major problem related to the application of clustering techniques is the selection of the optimum number of groups - a problem known as cluster validation [Davis and Bouldin 1979]. Most of the component clustering procedures have paid very little consideration to stop the clustering process automatically. Most of the clustering techniques are usually based on a pre-defined number of groups or fixed minimum level of similarity, at which the grouping process should be stopped. In the following, the procedure adapted to find the optimal number of groups has been explained.

In the incidence matrix for the cluster analysis, each component can be considered as a component vector as given below:

$A_i = \{a_{1i}, a_{2i}, a_{3i}, ..., a_{pi}\}$,

i.e. the classifying attributes being the co-ordinates of the component vector. Each component can be considered as a member of an n-dimensional Euclidean component space. An n-dimensional Euclidean component space will be representing n components in the space, each being represented as a point. At each clustering iteration similarity between every two components is calculated and the two most similar data points in Euclidean component space are found and merged. The level of similarity at which they merge and the number of groups at each iteration are recorded in a file say file1 while grouping results are written in another file say file2. The centre of the group is the representative component of the group. At each iteration, the centre of the group is updated as it is a representative of a new component group. The clustering process is supposed to run from n data points to one component group.

The components are then merged into one cluster, clustering results are available in file1 i.e. at each iteration, the number of groups and the level of similarity at which two most similar groups are merged. Example of the data format available in file1 after running the clustering algorithm is given below:

No of groups	Similarity lever at which the two groups are merged
n	-
n-1	0.98
n-2	0.93
n-3	0.89
n-4	0.80
.	.
.	.
.	.
1	0.30

The above mentioned data shows that if grouping goes from n groups to n-1 groups the distance (1- similarity level) involved is 0.02. From n-1 to n-2 the distance or dissimilarity level is 0.98-0.93 which is 0.05. Similarly, the maximum distance involved between two consecutive groups can be calculated at any level. This shows that if these two groups are merged, the most dissimilar groups in the whole grouping process are merged and this should be avoided and now the clusters are well separated. Another program written in C language can calculate at which level grouping process should be stopped. Grouping results in the whole grouping process i.e. from n groups to single group are available in one file. The grouping results for the number of optimal groups i.e. the components in groups at that level can be found in another file. One point worth mentioning here is that the similarity value is not always in descending order with the decrease in the number of clusters. Sometimes the case is reversed, the similarity increases as compared to the similarity level in the previous merging. This is because of updating the centre of the group i.e. by merging two groups, the centre of new group is closer in similarity to the group with which it has merged this time.

Stopping the grouping process can also be explained by the dendogram shown in figure 4.14. For every hierarchical clustering there is a corresponding tree, called a dendogram as already said, that shows how the samples are clustered. The figure shows a dendogram for a hypothetical problem involving six samples. Level 1 shows the six samples as singleton clusters. At level 2, samples x3 and x5 have been merged to form a cluster, and they stay together at all subsequent levels. The dendogram is usually drawn to scale to show the similarity or distance between the clusters that are grouped. For example, say after running the clustering procedure, following data for Cluster Number, Dissimilarity Range, and Measure of Range are obtained. The similarity values can be used to help determine whether the groups are natural or forced. This can be explained as below:

Cluster Number	Dissimilarity Range	Measure of Range
$6 \rightarrow 5$	$0.00 < \text{distance} < 0.05$	0.05
$5 \rightarrow 4$	$0.05 < \text{distance} < 0.07$	0.02
$4 \rightarrow 3$	$0.07 < \text{distance} < 0.15$	0.08
$3 \rightarrow 2$	$0.15 < \text{distance} < 0.22$	0.07
$2 \rightarrow 1$	$0.22 < \text{distance} < 0.71$	0.49

For the whole grouping process, in the case of the hypothetical problem, the distance between level 5 and level 6 is maximum. If grouping is performed between the groups at level 5 and level 6, it will lead to forced grouping or in other words the two most dissimilar groups in this domain will merge (i.e. if two groups are allowed to merge further, the distance or 'measure of range' involved is 0.49). In order to avoid this happening, one would be inclined to say that the groupings at level 5 are natural, but grouping at level 6 leads to forced grouping.

Hierarchical cluster analysis procedures group the components by measuring the similarity based on the classifying attributes selected for the application. The grouping results achieved in this case should not be viewed in the similar way as seen in the traditional rank order clustering. In the latter case, the parts are grouped against the machines they visit for their processing. Here the component is clustered in the group even though it is using only two out of eight machines, for example. Obviously, it is quite dissimilar to the component visiting all the eight machines but still it is being grouped here because it can be binary ranked in this group. In the case of this cluster analysis approach, the groups are formed on the basis of similarity level, they have with other components in the groups and the similarity level is calculated based on the chosen classifying attributes. It is possible sometimes that the components clustered in one group may be the subset of the components clustered in another group in terms of classifying attributes.

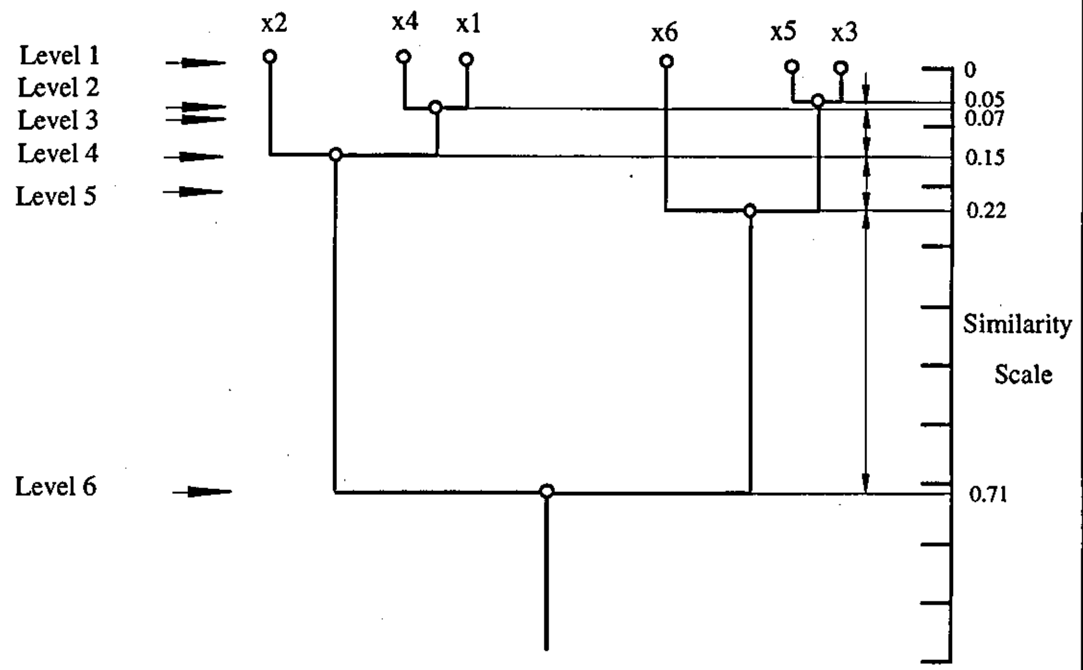


Figure 4.14 Illustration of stopping the grouping procedure.

Usually, it has been practice in the manufacturing industry to group the components by cluster analysis procedure and then make the judgement, how many groups would be optimum. This optimisation does depend really on the number of factors like, company's available resources, level of automation available i.e. material handling system available, components batch-sizes in the families. The number of families can be grouped into one family, if the number of components in these families is small and also, batch sizes in these families are smaller. When the information like the batch sizes of the components, company's available resources is not available, the decision of optimum grouping is to be made based on the information in hand i.e. the classifying attributes of the components. Near author grouping function in the cluster analysis process should be stopped at that level when clear component clusters are formed and the distance between them is maximum, proving that the distance between the groups has become now appreciable and now there is time to stop the grouping function. In other words, similar objects have been clustered into natural groups. This is shown in figure 4.11.

4.5.9.1 VALIDITY CHECK FOR STOPPING THE GROUPING

The method proposed to calculate the optimal number of groups in the cluster analysis paradigm has been tried on a set of examples given in the literature (incidence matrices) to check the validity of the method. The method used by the researchers is binary ranking. The results found are identical as calculated by other different methods. Three of the experimental implementations are discussed for the validity check of the method.

FIRST CASE STUDY

In this case study, the data set is a set of 20 machines and 35 components [Carrie 1973]. The optimal number of groups and grouping results are the same as given by Carrie. Table 4.1 shows the incidence matrix and grouping results.

SECOND CASE STUDY

Second case study data is taken from King and Nakornchai [1982] which consists of 5 machines and 7 components. The implementation results are identical as given in the paper. The data set and the grouping results can be found in table 4.2.

THIRD CASE STUDY

The data set in this case study consists of 10 machines and 20 components. The optimum number of groups for this data set are 4 as calculated by Srinivasan *et al.* [1990]. The data set and the grouping results are given in table 4.3.

```
                          1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3
        1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
______________________________________________________________________________

1       1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0
2       0 1 0 0 0 0 1 0 0 1 0 1 1 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0
3       1 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0
4       0 1 0 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0
5       0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
6       0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 1 0 0 1 0 0 1 0 0 0 1 0 0 0 0 1 0 0 1 0
7       1 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
8       1 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 0 0 1 0 1 0 0 0 1 0 0 0 0 0 0
9       0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0
10      0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 1 0 0 1 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0
11      0 0 0 1 0 1 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 0 1
12      0 0 0 1 0 1 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
13      0 1 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
14      0 1 0 0 0 0 1 0 0 1 0 1 1 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 1 0 0 0 0
15      0 0 0 1 0 1 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0
16      0 0 0 1 0 1 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0
17      1 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0
18      0 1 0 0 0 0 0 0 0 1 0 1 1 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0
19      0 0 0 1 0 0 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0
20      0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 4

                    GROUPS IS /ARE  4

          GROUP NO ==>  1 has machines  5
                  1   3   7   8   17

          GROUP NO ==>  2 has machines  5
                  2   4   13   14   18

          GROUP NO ==>  3 has machines  5
                  5   6   9   10   20

          GROUP NO ==>  4 has machines  5
                  11   12   15   16   19
```

TABLE 4.1

```
              1 2 3 4 5 6 7
  ___________________________
  1           0 1 0 1 1 1 0
  2           1 0 1 0 0 0 0
  3           1 0 1 0 0 0 1
  4           0 1 0 1 0 1 0
  5           1 0 0 0 0 0 1

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 2

GROUPS IS /ARE  2

GROUP NO ==>  1 has machines  2
              1   4

GROUP NO ==>  2 has machines  3
              2   3   5
```

TABLE 4.2

```
                                  1 1 1 1 1 1 1 1 1 1 2
              1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0
  _____________________________________________________
  1           1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
  2           0 1 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0
  3           0 1 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0
  4           1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
  5           0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 1 0 1
  6           1 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0
  7           0 0 0 0 0 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0
  8           0 0 0 0 0 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0
  9           0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 1 0 1
  10          0 0 0 0 0 1 0 0 1 0 1 1 0 0 0 1 0 0 1 0

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 4

GROUPS IS /ARE  4

GROUP NO ==>  1 has machines  3
              1   4   6

GROUP NO ==>  2 has machines  2
              2   3

GROUP NO ==>  3 has machines  2
              5   9

GROUP NO ==>  4 has machines  3
              7   8   10
```

TABLE 4.3

4.5.10 PATTERN RECOGNITION FOR A NEW PART

After the grouping process, grouping results for each iteration in terms of number of groups, their members and the characteristics of the representative components of the groups are in hand (in knowledge base). Moreover, the number of optimal groups at which the grouping process should be stopped are also available. The new part can be described to the system and classifying attributes extracted from the system in the similar fashion as already described. The level of similarity for the new part with the existing groups can be calculated. If the level of similarity is equal to or less than the threshold value then the component can be assigned to the existing groups, otherwise the grouping process will be carried out again.

An overall activity flow diagram for the CAFBG System is shown in figure 4.15.

4.6 THE CAFBG SYSTEM FOR THE EFFICIENT PROCESS PLANNING

As already discussed, the grouping results based on the CAFBG System are more suitable from the manufacturing point of view. The system is a combination of manufacturing information as well as important design information. Manufacturing requirements of the parts in a group represent the realistic processing requirements in terms of MCU's in the shop or manufacturing organisation. The centre of group or composite component of the family based on the CAFBG System represents the manufacturing requirements in terms of MCU's, the shape of the components, and the presence of features pattern. These characteristics or attributes of the composite component are quite enough for showing the processing needs of the family of the components or presenting the requirements of machining capabilities. MCU's represent the type of machining capability needed for the components. The basic shape of the component dictates the machining methods. Different components with different shapes require different machining and tooling methods. Components can be characterised on the basis of geometric shapes and dimensional ratios. These ratios help in characterising the parts for selecting the machine resources, potentially candidate operations and even sometimes work holding methods. A features pattern can have implications for tooling and sometimes machining resource allocations as well. Patterns dictate for consideration in special toolings. These things have already been discussed in detail elsewhere in the thesis.

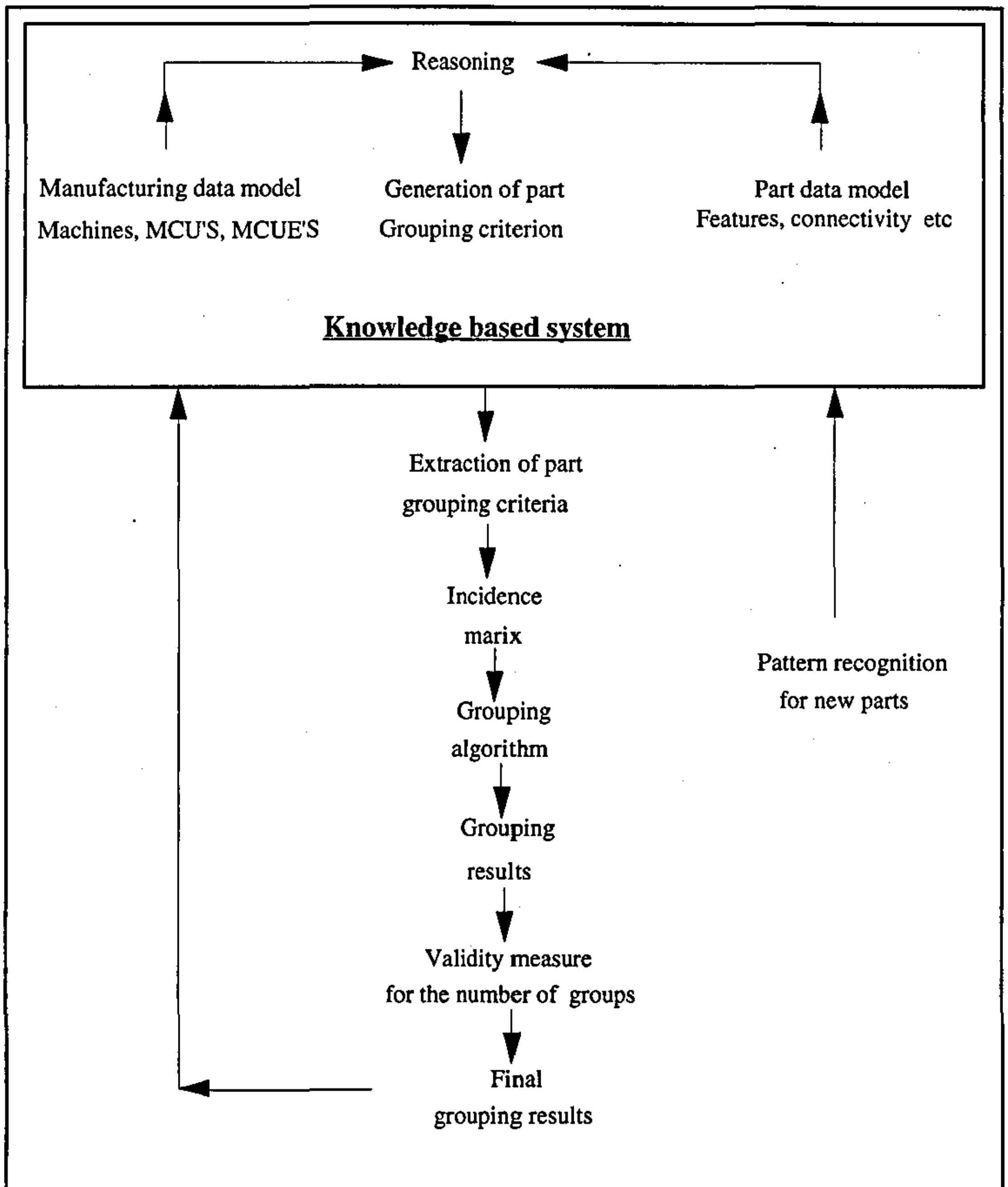


Figure 4.15 Activity flow diagram for the CAFBG System

The manufacturing planning logic compares the component processing requirements imposed by both geometric and technological constraints with the processing system capabilities to select feasible solutions. When the components in a group are similar based on their dimensional ratios i.e. shapes, realistic processing requirements, features pattern, developing the Process Planning activity becomes easy and manageable not only in Variant planning but also in a Generative planning system.

The Variant Process Planning systems, as already mentioned, represent the machining requirements (operations and machine sources) and the order of the operations in which they are performed. The composite component of the part family, in the case of CAFBG, contains the information which is closer to the master plan of the family. Closer in the sense that it contains the MCU information required for the processing of the component. MCU's can give the information about the potential machine types which can be used for the machining of the component. The thing it lacks is the sequence of operations (component level). As it also consists of component shape, sometimes, in the case of families consisting of turned or rotational components, a sequence of operations can be generated. Advocating this point, Hinduja and Huang [1989] quote that it is possible to determine the sequence of operations with a fairly large percentage of turned components in industry. The concept of MCU's facilitates a generic master plan (machine independent) that does not need to be updated when any new machining capability is added.

Therefore, it can be argued that grouping based on the CAFBG System is realistic grouping from the manufacturing point of view and helps in making the Process Planning function efficient as the representative component of the family contains the processing information in terms of MCU's which is very close to the master plan (Variant planning) of the part family.

It is very difficult to develop a Generative planning system which can encapsulate the manufacturing logic for all the domains of components in the world. The wider the component set, the more difficult the logic is. Hybrid planning systems are being developed to handle the problem by developing the manufacturing logic for the part families, thus making the problem easy and manageable. The representative component of a family does not help in developing the logic as a Generative system synthesises each component of the family individually but, it does help in defining the boundaries of the Generative Process Planning systems. Precise processing requirements are defined for the centre of the group of the family in the CAFBG System. The composite component or representative component of the family, possessing these processing requirements will be the reflection of the boundaries in terms of processing capabilities of the Generative planning system developed for that family. Moreover, the dimensional ratios or shapes and features pattern shown in the representative component exhibit the possible machining and tooling methods involved in the logic development. Dimensional ratios also define the limitations of the planning system in terms of the structural aspect of the components in the family.

In summary, part grouping achieved in CAFBG System (in the context of planning) has the following advantages:

- It provides better master plans because of the nature of the grouping parameters selected for part grouping.
- It provides generic (machine independent) master plans that does not need changing when new resources are added.
- The development of Generative planning system (logic) is simpler because the part family is formed based on manufacturing grouping parameters.

Traditionally, incidence matrix format is used for grouping similar components. The matrix $\{a_{ij}\}$ has n rows representing the components and m columns representing the machines. The elements $\{a_{ij}\}$ of the matrix is 1 if the *j*th machine is visited by the *i*th components; otherwise $\{a_{ij}\}$ is zero. An example of such a matrix is shown in figure 4.16-a. The components and machines are identified by their respective numbers. The matrix $\{a_{ij}\}$ is referred to as the components-machine incidence matrix. Concurrent grouping of parts into part families, and machines into cells, results in a rearrangement of rows and columns of the matrix. The solution expected is an arranged matrix with a block-diagonal structure in which all 1's are contained in the sub-matrices along the diagonal as shown in figure 4.16-b. The solution shows the group of components which can be processed in their corresponding manufacturing cells.

$$\begin{array}{cc|ccccc|}
 & & \multicolumn{5}{c}{\text{Machines}} \\
 & & 1 & 2 & 3 & 4 & 5 \\
\cline{3-7}
 & 1 & & 1 & 1 & & 1 \\
 & 2 & 1 & & & 1 & \\
 & 3 & & 1 & 1 & & \\
\text{Components} & 4 & 1 & & & 1 & \\
 & 5 & 1 & & & & \\
 & 6 & 1 & & & 1 & \\
 & 7 & & & 1 & & 1 \\
\cline{3-7}
\end{array}$$

(a)

$$\begin{array}{cc|ccc|cc|}
 & & \multicolumn{5}{c}{\text{Machines}} \\
 & & 3 & 2 & 5 & 1 & 4 \\
\cline{3-7}
 & 1 & 1 & 1 & 1 & & \\
 & 3 & 1 & 1 & & & \\
 & 7 & 1 & & 1 & & \\
\cline{3-7}
\text{Components} & 2 & & & & 1 & 1 \\
 & 4 & & & & 1 & 1 \\
 & 6 & & & & 1 & 1 \\
 & 5 & & & & 1 & \\
\cline{3-7}
\end{array}$$

(b)

Figure 4.16 Binary ranking

The composite component of the family does not give much information about the processing requirements of the family except the machines to be visited. In other terms, the work contents required for the part family cannot be assessed. Therefore, the grouping based on the machines is not realistic grouping and also it does not help in the Process Planning function. Sometimes processes required for machining the parts has been used instead of machines in the incidence matrices. In that case, the matrix $\{a_{ij}\}$ has n rows representing the components and m columns representing the processes. Machines can provide different machining processes and in some cases, the same processes can be achieved on different machine types. The traditional machining capability data model can be represented as shown in figure 4.17. Grouping based on the processes alone also does not result in realistic part families. The composite component of the part family in this case is exhibiting the processes required for the family of components. As already mentioned, the same type of manufacturing processes are available on different machine resources and hence, in many cases, it might be required for a component to visit different machine resources for a same type of process. In this case, the composite component is representing the processing requirements in terms simply of processes without the indication of machine resource capabilities needed. Therefore, again the composite component of the part family implied in the generation of master plans has disadvantages: 1) it is machine specific, and 2) the part family it relates to is not defined well. These advantages can be overcome by the part grouping work taken in this research. The CAFBG System provides part grouping that is better and machine independent.

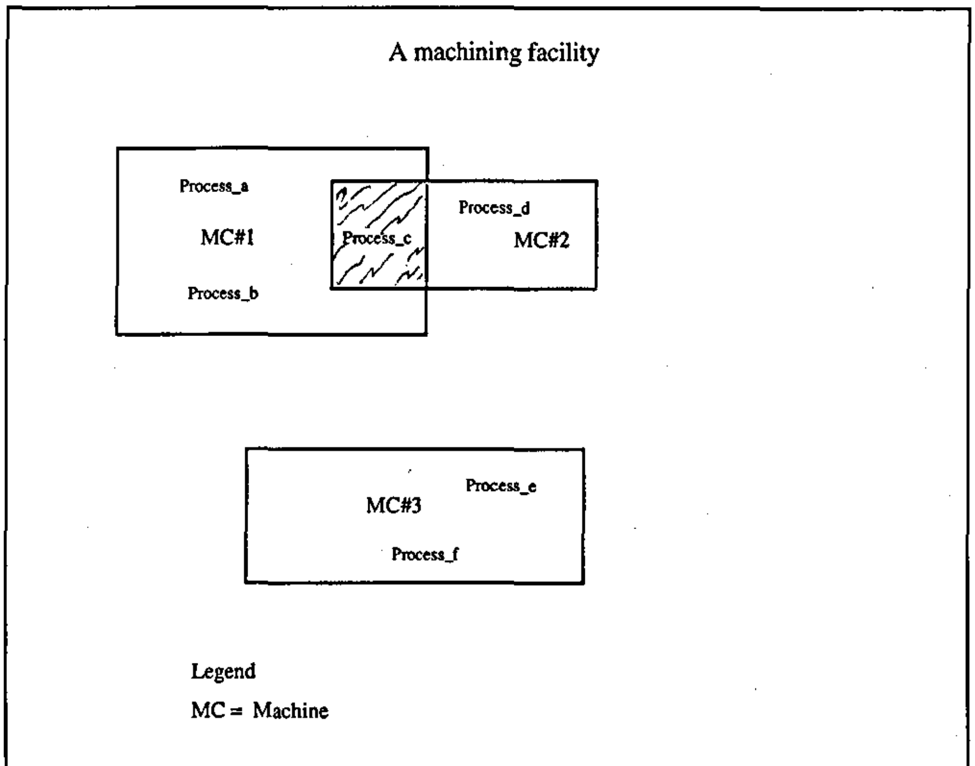


Figure 4.17 The traditional machine capability model

Mostly, classification and coding systems deal with the design information of the components for the classification purposes. The composite component of the part family generated by grouping based on any classification and coding system cannot contain all the information required based on which it can be evaluated how the part family is to be machined. First of all, it is very difficult to code completely the design information for each machining entity (feature) in any classification and coding scheme. Machining features simply, do not help in determining the precise manufacturing method to be used for the processing of the component. For example, two parts may have similar geometry and size but their materials and tolerances on size might be completely different thus leading to different machining methods. Secondly, there is a large variety of feature types which exist on the component body. It is hardly possible to expect precise grouping from the manufacturing point of view after grouping the components based on the design information. Disproving the C&C systems Wild [1985] quotes that several researchers have concluded that there is no fixed pattern of components throughout industry and that the concept of a universal classification system is at worst mistaken and at best of very limited value.

Therefore, it can be argued that the grouping based on the CAFBG System is not only closer to the master plans in the Variant planning systems but also it is helpful in defining the boundaries in terms of machining and tooling methods in the case of Generative planning systems.

CHAPTER 5

PART GROUPING — SOFTWARE IMPLEMENTATION AND CASE STUDIES

5.1 INTRODUCTION

In this chapter, section 5.2 discusses the design analysis of the components being used for the part grouping case studies. Section 5.3 contains the description of software implementation based on the composite components, manufacturing features, processes, machines involved and the CAFBG System including the pattern recognition problem for the new components for the proposed system.

Section 5.4 which is the concluding section, consists of the discussion on the grouping results based on the different criteria as mentioned above.

5.2 COMPONENT DESIGN ANALYSIS

In order to test the feasibility and viability of proposed grouping criteria, real life design data or components have been tried. These components are being manufactured by GEC Alsthom Large Machines Limited, Rugby, England. The design data for thirty components has been interpreted and this feature-based component design data has been loaded in the knowledge base of the system. The same design data was tested for the components grouping based on the features, processes, and machines and the results were compared with the grouping based on the proposed criterion.

The design data consists of a variety of components, the main types being; shafts without arms and with arms, end plates, base plates, barrel frames, stator frames, bushes, terminal boxes, terminal plates, flanges, end brackets, support blocks, support plates, bearings etc. The components vary a great deal in the geometries, sizes and other technological requirements. The components are generated by a wide range of machining processes. Most of the components are machined from raw material but some of them are cast and after that machined to size and finish. A large variety of machines from traditional types like lathes, milling machines, drilling machines, grinding machines, boring machines, honing machines to advanced machining centres are involved in machining of these components.

Design interpretation of a set of thirty components used for grouping case studies based on different criteria is given in appendix A.

5.3 SOFTWARE IMPLEMENTATION

The investigation carried out in this work has led to the implementation of experimental software work using the GENERIS Expert System shell as well as language C on the Sun Sparc Workstation. The overall task can be broken down into several logical sub-tasks to ease the implementation, as well as to provide the flexibility of enhancing and modifying tasks. Keeping this in view, the software was developed as a set of independent modules to ease maintenance and modification.

It is worthwhile to mention here that all the above mentioned modules work independently. At the time of running of any module, the banner showing the function of the module along with the request for 'wait' will be displayed. There is no need to interrupt the system during the execution. GENERIS can report the error to the user, if there is any problem. After the execution of a module, the results generated by the module are shown within the user defined window. The user needs to press <return> key twice after having a look at the results, in order either to finish the module or execute the next modules step by step.

5.3.1 PART GROUPING AROUND COMPOSITE COMPONENT/CELL CAPABILITIES

This section briefly discusses the major components of the software developed for grouping the parts around the mediods. Modules can interact with each other through the data input and output. Further discussion on the modular structure will be taken in chapter 7. The main system modules are:

1. TO CREATE THE RECORDS FOR THE PARTS
2. TO CREATE THE CAPABILITIES OF A CELL
3. TO DELETE THE CAPABILITIES OF ANY CELL
4. TO GROUP THE COMPONENTS AROUND THE CELLS
5. END

The details of the main menu are as below:

<u>*Selection 1*</u> is for inputting the design data of the components into the system.

<u>*Selection 2*</u> is to define the cell capabilities/composite component around which the components are to be grouped. By selecting this choice or selection, another sub-menu will be displayed as given below:

1. TO CREATE THE CAPABILITIES OF A CELL BASED ON PROCESSES
2. TO CREATE THE CAPABILITIES OF A CELL BASED ON FEATURES
3. END

In the above mentioned menu, *selection 1* is to create the cell capabilities based on the processes. *Selection 2* will define the composite component based on the features and *selection 3* is to end the session. Cell capabilities based on processes include:

a) The processes the cell can offer,
b) The materials the cell has the capability to machine,
c) The size or envelope of the components the cell can accommodate.

and the cell capabilities based on features can be given in the following:

a) The features the cell can machine and the extent of surface finish for each feature,
b) The materials the cell has the capability to machine,
c) The size or envelope of the components the cell can accommodate.

Selection 3 in the main menu is to delete the cell capabilities such that a new composite component can be defined. *Selection 4* in the main menu is to group the parts around the centre of groups or composite component already defined. By hitting this selection, another sub-menu will appear. The sub-menu is given in the following:

1. TO ASSIGN THE COMPONENTS TO THE CELLS BASED ON PROCESSES
2. TO ASSIGN THE COMPONENTS TO THE CELLS BASED ON FEATURES
3. END

In the above mentioned menu, *selection 1* is responsible for grouping the components around the composite components which are defined based on the processes, while *selection 2* will be grouping the components around the composite components which are defined based on the features. *Selection 3* is to end the session.

Components are assigned to the cells for which they have maximum level of similarity. In the case of a tie or in other words if for any particular component the similarity level is same for more than one cell, the component should be assigned to the cell where it can be processed with least set-up involvement. This aspect has not been implemented in the software.

The main software programs developed for grouping around composite component/cell capabilities are given in appendix C. The software programs hierarchy (the sequence in which they are called) designed for the grouping is also given in the appendix.

5.3.2 DIFFERENT PART GROUPING TECHNIQUES

The main system modules are:

1. TO INPUT THE PART GEOMETRY INFORMATION
2. PART GROUPING BASED ON THE PART GEOMETRY

3. PART GROUPING BASED ON THE PROCESSES
4. PART GROUPING BASED ON THE MACHINES
5. PART GROUPING BASED ON CAFBG SYSTEM
6. PATTERN RECOGNITION FOR A NEW PART
7. TYPE ANY COMMAND

Selection 1 is for inputting the design data of the components into the system.

Selection 2 is to group the parts based on their geometry (part features).

Selection 3 is to group the parts based on the machining processes involved in their manufacture.

Selection 4 is to do part grouping based on the machine tools involved in the machining of the parts.

Selection 5 is to carry out part grouping based on the CAFBG System.

Selection 6 can be utilised for the pattern recognition of a new part.

Selection 7 is to make any inquiry from the data-base

A set of thirty components was tried as an experiment for the part grouping based on part geometry, processes, machines and the CAFBG System. The implementation and results achieved in the case of the above techniques are attached in appendix D.

The algorithm for defining the MCU's has been discussed in detail, its flow chart shown in figure 4.6 and step by step manual implementation given in chapter 4, but the algorithm was not implemented in the software.

The main software programs developed and data files involved in the CAFBG System are given in appendix D. The software programs hierarchy (the sequence in which they are called) designed for the grouping is given in figure D.1.

Major software programs developed for the implementation of pattern recognition are also listed in appendix D. The software programs hierarchy (the sequence in which they are called) designed in the case of pattern recognition is given in figure D.2.

5.4 DISCUSSION ON THE GROUPING RESULTS

A set of 30 components from GEC Alsthom Large Machines Limited, Rugby as said before, was tried as a case study and the results of all the four cases i.e. part geometry-based grouping, process-based grouping, machine-based grouping and CAFBG System are studied and attached. Also, the design interpretation of the components are attached for reference in appendix x. The case study for all the four cases is discussed in the following:

5.4.1 COMPOSITE COMPONENT-BASED GROUPING

As the grouping has been made around the composite components, all the components having more similarity level with the composite component will be grouped around the centres of groups or composite components. It cannot be expected that the components grouped by this technique will be similar to each other. For example, the component requiring only 10% of the processing needs of the composite component and another component that might be using 100% will be in the same group. Of course, they cannot be similar. Similar results have been associated with the machine-based grouping which has been already discussed under the heading 'Machines as the Classifying Attributes for the Part Grouping' in chapter 4.

Another problem which is associated with this method of grouping is to define the composite component or design the cell before knowing the number of groups and part families i.e. on what basis the composite components/cells will be defined. Improper cells will result in inter-cell movements thus degenerating to traditional 'process-type layout'. This emphasises the need for first grouping the components and then designing the cells, It is important to cluster on the basis of the features of the parts themselves and later assign the groups to the relevant machine cells.

5.4.2 PART GEOMETRY-BASED GROUPING

- Different feature types are grouped in different groups but actually they should be in same group since the manufacturing solution for all of them is the same. This can be observed in group number 3 and 4 when the total number of groups is 10. Here different types of Step and Slot features are grouped in different groups even though their solution is a milling operation.

- Part geometry-based grouping groups the components based on the feature types but does not take into consideration the technological constraints of the features (size, surface finish etc.) and features pattern. Round Hole features, even if they require other extra operations, because of their technological requirements or special machine resources, because of the features pattern still might be in the same group as their type is the same. This can be noticed in group number 3 when the total number of groups is 10. Round Hole features with different technological constraints, in this case, are grouped in the same group.

- Sometimes rotational and prismatic components might come in the same group because of the same feature types. This can be seen in the part geometry-based grouping in group number 1 when the total number of groups is 3, and group number 2 when the total number of groups is 5. In both these cases rotational and prismatic components are in the same group because of Round Hole features on the components. Part geometry-based grouping cannot identify the different machine resource requirement because it does not use features connectivity aspect, but simply feature types.

5.4.3 PROCESS-BASED GROUPING

- Traditionally, process plans have been written by different planners. Sometimes, solutions for same features will be different by different planners. Grouping based on these plans certainly will not result in realistic grouping because different criteria are being used for the same features. The same has been observed in the process-based grouping results. Solutions for Keyway, Step and Slot features is sometimes a milling process and sometimes a slotting or shaping process.

- The features connectivity aspect has not been observed in the traditional process allocation function i.e. drilling is the solution for every Round Hole feature even if different machine resources are needed because of the position or connectivity of the feature. That is why, even in the process-based grouping, like part geometry-based grouping, rotational and prismatic components are grouped in the same group. This can be found in the results as mentioned below:

  Group number 4 when the total number of groups is 9,
  Group number 4 when the total number of groups is 7,
  Group number 5 when the total number of groups is 6,
  Group number 1 when the total number of groups is 4,
  Group number 2 when the total number of groups is 3, and so on.

5.4.4 MACHINE-BASED GROUPING

As mentioned earlier, due to the lack of rationalisation mechanism in the manufacturing routings, different machines can be allocated for the machining of similar components. This will suggest that similar components are dissimilar and will place them in different groups. The same has been noticed in the machine-based grouping results. For example, components 22 and 24 have Round Hole pattern of features, but different machines have been assigned to them. Machine 'MAK' which is a machining centre is assigned to component 22 and 'WTD' which is radial drill to component 24. This way they are 100% dissimilar. The same has been noticed in the cases of components 11, 12, and 15. All of them are of similar size and have the same Round Boss features to be machined but different machine tools such as, SMU, QKO and CAP respectively (all lathe machines) have been allocated to them for their processing. Assigning different machines to them places them in different groups whereas they should be in the same group.

The second factor which prevents the use of the machine-based grouping is that components simply visiting the same machines are considered similar, no matter what their machining work contents are. This way quite non-similar parts will come in the same groups. This can be seen in the grouping results. For example, components number 9 and 22 are taken to be similar because they are using the same machine called MAK. In the case of component 9, this machine is used for the 4 processes i.e. keyway cutting, boring, chamfering, and drilling but for the component 22, it is doing only drilling. It means dissimilar parts are grouped in the same group when they should not.

Machine-based grouping brings together the prismatic and rotational components in the same groups as it was in the cases of both part geometry-based grouping and process-based grouping. This does not happen in grouping based on the CAFBG System as will be discussed in the following section. The cases of the prismatic and rotational components gathered in the same groups can be found in the results given below:

Group number 2 when the total number of groups is 10,
Group number 3 when the total number of groups is 10,
Group number 4 when the total number of groups is 9,
Group number 4 when the total number of groups is 8,
Group number 4 when the total number of groups is 6, and so on.

5.4.5 GROUPING-BASED ON THE CAFBG SYSTEM

The inadequacies of traditional part grouping techniques, and of the part grouping approaches which have been compared with the CAFBG System (part geometry-based grouping, process-based grouping, and machine-based grouping) are largely overcome when the criteria proposed in the CAFBG System are implemented. Groups can even be identified based on dimensional ratios which dictate different processing methods for the parts. The features connectivity aspect plays an important role in identifying the different processing needs of the parts. Results have proved that processing needs allocated while observing the connectivity aspect result in realistic grouping. The features pattern as one of the proposed criteria also play a role in realising the special processing requirements if the features pattern exists. Therefore, it can be argued that the grouping based on the above mentioned system gives preferable results.

CHAPTER 6

PROCESS PLANNING IN GT

6.1 INTRODUCTION

This chapter discusses mainly the logic development for the Process Planning activity. After the parts have been grouped into part families, then how the Process Planning activity is effected in the GT environment. This has been discussed in section 6.2. Section 6.3 describes the Process Planning function in general while section 6.4 puts some light on the role of CAPP in the CAD/CAM integration.

Section 6.5 describes the decision logic, forward and backward planning, and knowledge representation in the context of a Generative planning system. Logic development for the Process Planning has been detailed in section 6.6. The section deals with the planning activities like, preparing input data, planning blank, process selection, machine tool selection, set-up determination and operation sequence function in detail in further subsections.

6.2 PART GROUPING REQUIREMENTS

Most manufacturing firms have the capability to fabricate a relatively wide variety of parts in small lots. In order to stay competitive, most firms must be able to meet special customer requirements by providing customised features on their products. This must be done within lead times that are getting shorter and at a cost that provides enough profit for the firm to remain viable. As a result, many firms are attempting to automate as many functions as possible. Computer-Aided Process Planning (CAPP) is representative of this reasoning. It should be noted that most CAPP systems are based on GT concepts. Consequently, because GT facilitates the development of CAPP, some of the benefits associated with CAPP should be credited to GT [Bedworth *et al.* 1991].

It is very difficult to develop a Generative planning system which can encapsulate the manufacturing logic for all the domains of components, as already said. Hybrid planning systems are being developed to handle the problem by developing the manufacturing logic for the part families, thus making the problem easier and more manageable. A Hybrid planning system actually utilises the potential of both the Generative and Variant systems.

In order to address both i.e. the Variant Process Planning systems and Hybrid Process Planning systems, there are requirements for part grouping as pre-requisite. Process Planning can be made efficient if true similar components from the manufacturing point of view are grouped into part families. To Ham *et al.* [1985] an automated Process Planning technique is a basis for the rational and logical approach to improve manufacturing productivity in a CIM system. Group Technology plays an essential role for development of Computer-Automated Process Planning, based on the part-family concept of Group Technology.

In the GT paradigm, a composite component is a representation of a centre of group established on the basis of defined parameters, characteristics or criteria. It is the representative of all the features of all the family members. The composite component may be real or hypothetical, although usually the latter because in most of the cases, it is not practical for a representative component to cover physically all the parameters like material, size, connectivity etc. All the parameters may differ from component to component in the family but the composite component embodies all the features of the components in one part family. By building a process model that contains the solution for every feature, components in the entire family can be planned.

The composite component concept can be used both for processing requirements of the family of components and processing capabilities to fulfil the machining requirements of the family of the components i.e. manufacturing cell capabilities.

In the context of processing capabilities, a composite component captures the processing needs of the entire family. The composite component helps in selecting the processing methods and machines which can process the entire family of components as well as set-ups and tooling requirements. After groups have been established, the centroid of the group or composite component, in our case, carry information such as the dimensional ratios in which components exist as the representation of shape; connectivity information in terms of number of AD's, from which the components will be machined; detailed processing needs in terms of MCU's required i.e. feature based model of component with precise processing requirements allocated to it; the presence of features pattern and their types. The above mentioned attributes have been used to bring similar components together which are directly relevant to the Process Planning function.

The generic nature of the composite component, defined by using MCU's, makes the master plan also generic, which helps a great deal as it does not require the changing of plans if new equipment is added to the manufacturing system. A big problem with a current Variant system is that the whole set of master plans needs to be changed when new equipment is added to the system.

The system proposed in this work represents better manufacturing requirements of part families. Thus improving the Variant planning system. Moreover the possibilities of producing a Generative planning system are much better because the boundaries are better defined.

6.3 PROCESS PLANNING

Process Plan can be considered as a recipe for transforming the designer's design intent into manufactured products. Process Planning is the activity in engineering production that links design and manufacture [Wolfe 1985, Wang and Wysk 1987, Weill 1988, Ham and Lu 1988, Li and Zhang 1989, Joneja and Chang 1991] and thus plays an important role in achieving CIM [Iwata and Sugimura 1987, Subramanyam and Lu 1988, Ham and Lu 1988, Graves *et al.* 1988, Li and Zhang 1989].

As already mentioned elsewhere, manufacturing is a means to realise the design. Process Planning function is the channel to transform the design into manufacturing or in other words, design is the input to the Process Planning and manufacturing is an output from Process Planning. The concept has been shown in the figure 6.1. The basic task in the Process Planning is the interpretation of part description as given by the designers, and the specification of instructions in sufficient detail for production.

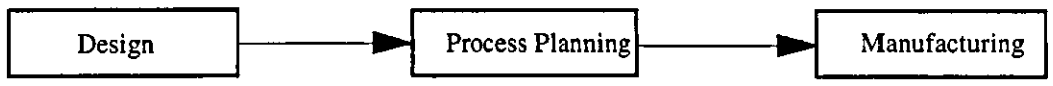


Figure 6.1 Process Planning link between design and manufacturing

In the manual approach, a skilled individual, often a former machinist, examines a part drawing to develop the necessary instructions for the Process Plan. This requires knowledge of the manufacturing capabilities of the factory: machine and process capabilities, tooling, materials, standard practices, and associated costs. Very little of this information is documented; often this information exists only in the minds of the Process Planners. This approach relies almost entirely on the knowledge of the individual planner. Consequently, Process Plans developed for the same part by different planners will usually differ unless the part is simple to make. The same planner may develop a different Process Plan for the same part if there is a long time lag between the analysis for that part, because the planner's experience may change during the time interval and/or shop conditions may change significantly.

Manual preparation involves subjective judgements that reflect the personal preferences and experiences of the planner; consequently, plans prepared by different planners for similar parts can vary. This approach is very labour-intensive, time consuming, and tedious.

In order to perform the Process Planning activities, a Process Planner must [Chang 1990]:

- be able to understand and analyse part requirements,
- have extensive knowledge of machine tools, cutting tools and their capabilities,
- understand the interactions between the part, manufacturing quality, and cost,

- possess analytical capabilities.

Design input information which sets the pace in the Process Planning function is as below [Alting and Zhang 1989]:

- batch size
- geometric configuration
- raw material properties
- dimensions and tolerances
- surface roughness constraints
- heat treatment and hardness

Factors considered in Process Planning are numerous, including material, structure and technical specifications of the part, kind of blank, production volume, machine tools, cutting tools, measuring tools, work holders, skill level of machinists, shop conditions, and so on [Wang and Li 1991]. Wang and Li emphasise that impacts of these factors on Process Planning must be comprehensively studied, and synthetically analysed in order to make optimal decisions.

Chuo and Ito [1991] emphasise the need for deep understanding of the essential features in Process Planning by the experienced engineer. A good discussion on the planning function can be found in their publication.

In the Process Planning function product quality, product efficiency, and product cost must be considered. Product quality and efficiency is of prime importance. A Process Plan will be considered efficient which achieves them at minimum overall cost.

In the research performed in this thesis cost has not been included as a planning criterion. This is to simplify the solution of the grouping problem, which is the main thrust of the research.

Spur and Opitz were among the first to write on the automation of manufacturing systems and the role that Process Planning should play in these systems. Spur was perhaps the first to define Variant and Generative methods of Process Planning and mechanisation and implementation of such planning systems [Niebel *et al.* 1989]. Generative approach is a step towards complete automation. Alting and Zhang

[1989] argue that many CAPP systems do not fit into either category and are therefore described as semi-generative. Generative Process Planning is a complex task. If one were to try to solve the entire planning problem as a whole, it would be very difficult (if not possible) [Chang and Wysk 1985]. Further details on the subject can be found in several publications Chang and Wysk [1981], Matsushima *et al.* [1982], Chang and Wysk [1985], Wolfe [1985], Wysk *et al.* [1986], Chu and Wang [1988], Chang *et al.* [1988], Ham and Lu [1988], Alting and Zhang [1989], Chang [1990], Wang and Li [1991].

The Variant approach is best suited to batch manufacturing when there is a great deal of similarity among the components. A great number of components within each part family would justify by writing of a standard plan. Much similarity among the components will result in only a minimum of alteration of the standard plan for each individual component. On the other hand, if the number of components in a part family is small and the alterations required on the standard plan are considerable, the total human effort involved in creating the standard plan for the family and carrying out the subsequent modifications for each component may be so high as to question the very justification of the Variant system. A Generative Process Planning system is more favourable in the situation where there are many families and similarity level between the components in each family is low.

Process Planning is a wide problem area and in the context of current research, it can be argued that only aspects of the total problem have been addressed by the researchers. Ham and Lu [1988] emphasise the need for an integrated framework, to enable aspects of the total Process Planning problem to be brought together. According to Wysk [1985], the issues currently being tackled by the researchers in advancing the Computer-Aided Process Planning function can be given as below:

1. The identification and capture of Process Planning logic.
2. The clear and precise definition of the part in computer compatible format.
3. The unification of captured logic and part description into a manufacturing data base.

Currently there are some promising approaches in terms of integrating CAD/CAM, for example the QTC (quick turnaround cell) system that integrates design/manufacturing/inspection for prismatic parts [Chang *et al.* 1988, Kanumury *et* *al.* 1988, Chang 1990], [Joneja and Chang 1991], the XMAPP system that integrates design/Process Planning for prismatic parts [Inui *et al.* 1987b], and the AMSI system that interfaces CAD/CAPP for rotational parts [Wang and Wysk 1988a].

6.4 THE ROLE OF CAPP IN CAD/CAM INTEGRATION

With the emergence of CAD/CAM integration as a predominant thrust in discrete parts industries, the communication between the people working in design and manufacturing has become a vital consideration. Manufacturing industry, these days is using CAD techniques extensively to design the products and similar way, using some CAM techniques, such as computerised numerical control (CNC) machines, to manufacture the parts. However, in most of the cases there is very little communication between design and manufacturing. Design engineers designs the product and then the drawing is sent for manufacturing to make the part.

Computer-Aided Process Planning emerges as a key factor in CAD/CAM integration because it is the link between CAD and CAM. After engineering designs are communicated to manufacturing, either on paper or electronic media, the Process Planning function converts the designs into instructions used to make the specified part. Automated Process Planning is an important element in CAD/CAM integration, because it provides a basis and a methodology for design and manufacturing communications and removes the 'wall' between design and manufacturing. CIM cannot be realised until this function is automated; consequently, it can be argued that automated Process Planning is the link between CAD and CAM. The idea has been advocated by many researchers [Wolfe 1985, Wang and Wysk 1987, Weill 1988, Ham and Lu 1988, Li and Zhang 1989, Joneja and Chang 1991].

6.5 HYBRID PROCESS PLANNING SYSTEM — HYCAPP

The advantages of a Generative Process Planning system over a Variant planning system are; 1) new components can be planned in a Generative planning system whereas a Variant planning system is limited to only similar components, 2) a Variant planning system cannot be used for an automated manufacturing system but on the other hand, a Generative planning system is a step towards complete automation, and 3) a Generative planning system generates consistent Process Plans without human intervention but in the case of a Variant system, experienced Process Planners are still required to modify the master plan for the planned family to suit any specific component. To create a Generative Process Planning system for the whole diversified population of components is very difficult if not impossible. Change [1990] argues that a Generative planning system is still far from being realised.

The trend in recent years is to develop a hybrid Process Planning system that combines variant and generative logic [Snead 1989]. A Hybrid Process Planning approach has been adapted in HYCAPP which utilises the potentials of both the main Process Planning systems i.e. Variant and Generative. HYCAPP is an acronym for 'HYbrid Computer-Aided Process Planning'. Hybrid system first brings together similar components into part families and then generates Process Plans for each family utilising the Generative approach. A Generative system is comparatively easy to develop when the domain and boundary of the similar components are known.

The first component of Hybrid planning system which is clustering similar components in the part families has been already detailed in chapter 4. The second component of the Hybrid planning system which is creating the Generative planning system, will be discussed in this chapter.

Wysk [1985], Chang and Wysk [1985], Chang [1990] and Zust and Taiber [1990] divide the Generative Process Planning knowledge into three components; 1) part geometry description, 2) manufacturing system knowledge description, and 3) decision making logic. This has been pictorially depicted in figure 6.2. Part description methods have already been discussed in detail in chapter 3. Manufacturing system knowledge will be discussed under the heading 'Prepare Input Data'. The main focus of this chapter will be the development of Process Planning logic in the Generative planning environment.

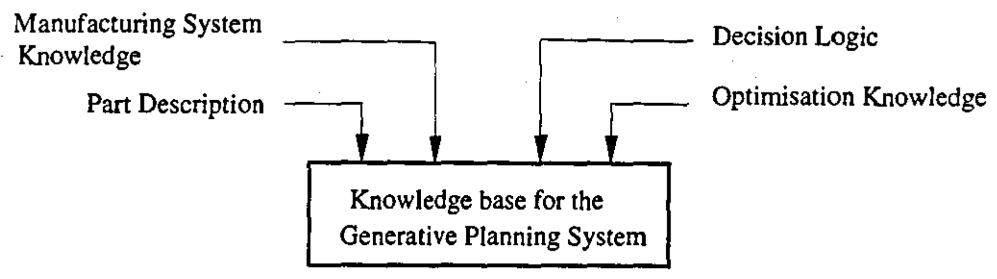

Manufacturing System Knowledge
Part Description
Decision Logic
Optimisation Knowledge
Knowledge base for the Generative Planning System

Figure 6.2 Information required for the Generative Planning System

6.5.1 DECISION LOGIC

In a Generative Process Planning system, the system decision logic is the focus of the software and directs the flow of program control. The decision logic determines how a process or processes are selected. The major function of the decision logic is to match the process capabilities with the design specifications. Process capabilities can typically be described by 'IF ... THEN ...' expressions. Information in handbooks or process boundary tables can be translated using a high-level computer language. However, such programs can be very long and inefficient. Even more disadvantageous is the inflexibility (difficulty of modification) of such software—this inflexibility leaves customised codes of this type virtually useless in Process Planning [Chang and Wysk 1985].

Knowledge-representation methods are related directly to the decision logic in these systems. The static data are the representation and the dynamic use of the data becomes the decision logic. The following decision procedures are applied to Process Planning systems [Niebel *et al.* 1989]:

1. Decision tables
2. Decision trees
3. Artificial Intelligence

The advantages and disadvantages of CAPP systems based on conventional computer languages, user developed languages, decision tables and Expert Systems have been compared and detailed comprehensively by Muthsam and Mayer [1990].

6.5.1.1 DECISION TABLES

A decision table is a tabular method for expressing the actions that should follow if certain conditions exist [Burch *et al.* 1983]. This technique permits the representation of many logical if-then expressions in a compact, structured manner. Decision tables have been used for many years as an aid to developing and documenting computer programs that involve several different actions depending on the values of some program variables. The upper half of a decision table contains the decision conditions, which are expressed as condition stubs and entries. The condition stubs are the criteria that the decision maker wants to apply in the decision process. The condition entries are the responses for each condition stub. CAPP systems have been developed based on decision table methodology; in practice, however, such a system is relatively difficult to maintain. Other approaches have proven to have more benefits. TIPPS [Chang and Wysk 1985] uses decision table methodology.

6.5.1.2 DECISION TREES

A decision table can be converted into a decision tree structure. Each decision tree is composed of a root, branches, and nodes, in a type of graph. The starting point is called a root; each tree should have only one root. Emanating from the root and the nodes are branches. The branch may represent a value or an expression; however, a branch can have one of two logical values, true or false. Relative to tables, trees have several benefits. First of all, they are easy to understand and visualise. Secondly, they are easy to maintain. APPAS [Wysk 1977] is a typical example of decision tree logic used for Process Planning.

6.5.1.3 ARTIFICIAL INTELLIGENCE

The term Artificial Intelligence (AI) was created by McCarthy at MIT at the end of the 1950s [Alting and Zhang 1989]. AI is the discipline that aims to understand the nature of human intelligence through the construction of computer programs that imitate intelligent behaviour [Bonnet 1985]. These functions include reasoning, planning, and problem solving. Applications for AI have been in natural language processing, intelligent data base retrieval, expert consulting systems, theorem proving, robotics, scheduling, and perception problems [Nilsson 1980]. The AI techniques which have been applied to CAPP development primarily Expert Systems or Knowledge Based Systems.

An Expert System is a computer system that encapsulates specialist knowledge abut a particular domain of expertise and is capable of making intelligent decisions within that domain [Jackson 1990] or in other terms, an Expert System is a knowledge based system that emulates expert thought to solve significant problems in a particular domain of expertise [Jackson 1986].

The common elements of these definitions are that:

1. An Expert System is a computer system,
2. It is designed to emulate some aspects of human expertise.

But what is expertise? Johnson [1986] usefully defines an expert as being:

. . . a person who, because of training and experience, is able to do things the rest of us cannot; experts are not only proficient but also smooth and efficient in the actions they take. Experts know a great many things and have tricks and caveats for applying what they know to problems and tasks; They are also good at ploughing through irrelevant information in order to get at basic issues, and they are good at recognising the problems they face as instances of types with which they are familiar.

The early Expert Systems were built using traditional AI languages. The knowledge in such a system was 'hard-wired', which means every time a new system was developed from scratch. There was need to borrow a great deal from already built systems. This could be done to separate out the knowledge specific to the particular domain from that part which drives the Expert System. The former part is now called the knowledge base, while the latter is known as an inference engine. The approach resulted in a new knowledge engineering tool—the Expert System shell. The Expert System shell is without the domain-specific knowledge.

Chang [1990] mentions some of the characteristics which an Expert System should have:

a) It separates knowledge itself from how to use the knowledge. In traditional computer programming, the knowledge and the control are mixed in the code.
b) The knowledge is codified symbolically, and the system is able to reason symbolically. Traditional programming converts every thing into numbers. Algorithms work on numbers to come up with an answer.
c) It can explain its reasoning process. There is no way to know how a traditional program comes up a specific answer. Although one may be able to trace all the steps a program goes through, the program cannot tell why it goes through those steps.
d) It must perform like a human expert. Even software that satisfies all the above three conditions can only be called a knowledge-based system, and not an Expert System, if it cannot perform up to the level of a human expert.

An Expert System, in the context of Process Planning, usually has three major components; 1) declarative knowledge, 2) procedural knowledge, and 3) control system.

Declarative knowledge is the representation of facts about the problem under consideration. In the Generative Process Planning domain, the declarative knowledge is design information of the parts and processing system capabilities. The representation of declarative knowledge in the system should be precise, unambiguous and compatible with the procedural knowledge.

Procedural knowledge, sometimes called domain knowledge is responsible for solving the problem in a particular domain. It consists of both facts and rules. Rules also called heuristics are the knowledge of human experts acquired over the years of experience. How well the knowledge can be used, however, depends on the way it is represented.

The control system also called the inference engine contains the general problem solving knowledge. There is no need to develop the control system as mostly Expert System shells come with built-in inference engines. It plays a role of an interpreter in the system.

There are two types of knowledge involved in Process Planning systems: component knowledge and process knowledge. The component knowledge defines the current state of the problem to be solved (it is also called declarative knowledge). On the other hand, the knowledge of processes defines how the component can be changed by processes (it is also called procedural knowledge). Applying the process knowledge to a component in a logical manner is called control knowledge [Chang and Wysk 1985].

6.5.2 FORWARD AND BACKWARD PLANNING

A direction for the planning procedure, in the case of Variant Process Planning, is not needed as plans are retrieved from the data base. On the contrary, however, in the Generative Process Planning, the system must know the initial state and the direction of proceeding (goal). The sequence of processes can be both the ways i.e. initial state → final state or final state → initial state. For instance, in the case of Process Planning, raw material can be considered as an initial state and finished component as a final state (goal). With the forward planning, one starts with the raw material, plan the sequence of operations until the finished component is accomplished. Backward planning is a reverse procedure. The initial state in the case of backward planning will be finished component and the sequence of processes will proceed until the raw material (the goal) is achieved. Each machining process will be taken as a filling process in this case.

Forward and backward planning may seem similar; however, they effect the programming of the system significantly. Planning each process can be characterised by a precondition of the surface to be machined and a post-condition of the machining (its results). For forward planning, we must know the successor surface before we select a process, because the post-condition of the first process becomes the precondition for the second process. Backward planning eliminates this conditioning problem since it begins with the final surfaces form and processes are selected to satisfy the initial requirements. The transient surface (intermediate surface) produced by a filling process is the worst precondition a machining process can accept. Any filling process that can satisfy the transient surface can be selected as a successor process. In forward planning, the objective surface must always be maintained even though several operations must be taken to guarantee the result. On the other hand, backward planning starts with the final requirements (which helps to select the predecessor process) and searches for the initial condition for something less accurate (which is easy to satisfy) [Chang and Wysk 1985].

The generation of a Process Plan in HYCAPP proceeds from raw material to the finished part i.e. forward planning.

6.5.3 KNOWLEDGE REPRESENTATION

Process Planning draws heavily on the empirical knowledge, acquired by experience. Therefore, a great deal of expertise is required to generate an efficient plan. The combination of shortage of Process Planning experts and the suitability of using computers to store vast amounts of acquired empirical knowledge has led to a proliferation of Computer-Aided Process Planning systems. The representation scheme used to store all this empirical knowledge is an important issue in Computer-Aided Process Planning systems because the efficiency in retrieving the stored knowledge and the effectiveness of generating plans are determined by the representation scheme used.

Knowledge is the symbolic representation of aspects of some named universe of discourse [Winston 1984]. Knowledge can be symbolised. That is, it can be represented in some way. Hence, AI has often been seen as being a discipline concerned with symbolic processing [Simon 1969]. Under the term knowledge representation, knowledge is stored in a knowledge base in a form most appropriate for the application concerned. Knowledge representation is an implementation issue. It concerns the means of implementing the 'intelligence' involved in some particular domain in a computational medium [Beynon 1993].

The current use of Artificial Intelligence (AI) techniques in automated Process Planning can be clearly divided into two parts [Joshi *et al.* 1988a]:

1) the use of AI for automated interpretation of the part description to perform geometric reasoning about the shape, features and relationships between features; and
2) Expert Systems for the development of the Process Plan itself.

The Process Planning system needs to have knowledge representation of part and machine capabilities. Frames and rules are considered to be most suitable for this purpose [Roy and Liu 1988]. The factual knowledge is represented by frames while the problem solving or reasoning knowledge can be described through the declarative statements as the production rules. The antecedents of a rule assert the presence of some conditions to be fulfilled for the application and the consequent parts suggest action to be taken. This action part gives some procedures which have to be applied when the condition part is satisfied. Forward reasoning is applied in building process structures. It starts with the raw material and different machining processes operate on it until it becomes the finished part.

One of the advantages of using frame representation for part description in Process Planning is that the state changes in the features after each operation can be recorded and preserved in new frames. For instance, if a hole H has its initial frame with a diameter, length, etc., it can have a 'boring—operation' frame attached as a new property showing the diameter appropriately altered to indicate the state change for the next operation 'drilling'. All these frames are available in the environment and can be accessed at any point in the program run [Joshi *et al.* 1988].

6.5.3.1 COMPONENT DATA REPRESENTATION

The decision logic in the Process Planning function is basically matching the parts machining requirements to the capabilities of the available manufacturing system. Each part feature and associated attribute such as dimensions, tolerances, and surface finish are used to infer possible manufacturing operations to convert the blank into the finished product. This procedure is fairly general. Part features are checked, considering also the technological information related to the them to attach the machining methods to the parts.

The feature based component data model has already been discussed in detail elsewhere in the thesis.

6.5.3.2 MACHINE TOOL DATA REPRESENTATION

Process knowledge about the manufacturing processes must be represented in a well defined format for the reasoning purposes in the manufacturing function. Such knowledge may be represented in different ways, namely, decision table, decision tree, and knowledge based system. The basic characteristics of a manufacturing process includes: The shape(s) the process can produce, the dimensional limitation, the dimensional and geometric tolerance constraints, surface finish constraints, and cost factors. Such characteristics can be called the capability of a manufacturing process sometimes called 'process capability'. By matching the available 'process capability' with the machined surface requirements, the necessary processes can be identified.

Process capability varies among manufacturing facilities. These capabilities are a function of the tool accuracy, size, etc. The shape capability implies the basic geometry which can be produced by a process. Other capabilities such as dimension, tolerance and surface finish can also be modelled by process boundaries. A process boundary can be termed as the limiting size, tolerance and surface finish for a process. Every machine resource has its own process boundary model.

The following sub-sections discuss the logic development in the Generative Process Planning function.

6.6 LOGIC DEVELOPMENT IN HYCAPP SYSTEM

The manufacturing planning logic compares the component requirements in terms of both geometric and technological criteria with the processing system capabilities to select feasible solutions.

6.6.1 PREPARE INPUT DATA

Process Planning is an example of an application in which the decision logic is primarily based on matching component requirements to the capabilities of the manufacturing processes and equipment to be used for their production [Gindy *et al.* 1993].

6.6.1.1 COMPONENT KNOWLEDGE

This function involves decomposing the workpiece into regions which are important from the manufacturing point of view. In other words, the workpiece geometry is divided into local interest worthy geometric entities called features. Technological information at feature level (feature parameters i.e. size, tolerance, surface finish etc.) associated with the feature will also, be covered by this function. Furthermore, the material, feature connectivity, feature relationships, and all other specifications relevant to the machining of the component will also be recorded here. The design information can be termed as component requirements.

In the initial study of the part drawing, the manufacture engineer is aided in determining what is wanted in the final product. This must be known before any processing plan can be formulated. At the termination of the manufacturing plan or sequence, the manufacturing engineer must again make reference to the part print to determine by comparison whether or not the final piece part meets all design specifications [Tanner 1985].

Component information can be divided into two types [Gindy *et al.* 1993]; absolute knowledge and constrained knowledge. Absolute knowledge consists of features on the component, feature connectivity and other relations which exist among the features. Constrained knowledge includes material, overall size, accuracy and finish required. The component absolute knowledge and constrained knowledge set the basis for the component requirements.

Machined parts may have different types of configurations and sizes. However, a common thing with the machined parts is that they all have machined surfaces or features to be machined. While analysing the part drawing or design information, it is of prime importance for the planner to find out which features a part is composed of. This is because the features are the main factors for selecting the processing methods, equipment and toolings.

The feature-based component representation approach has been adapted in this research. According to Brimson and Downey [1986], it has been understood for some years now that features play a key role in providing integration links between design and manufacturing.

The STEP organisation have addressed a structure for feature [PDES 1988] as part of their work to provide a data integration standard. This work has also concentrated on the need to provide a geometric understanding of features.

6.6.1.2 MANUFACTURING SYSTEM KNOWLEDGE

Gindy *et al.* [1993] dissect also the processing system information, which comprises machine tools and cutting tools, into two types namely; 1) absolute knowledge, and 2) constrained knowledge. Form generating functions and different tool types are considered as absolute knowledge while machine size envelop and technological output (both by machine tools and cutting tools) are taken to be as constrained knowledge.

Another term used for manufacturing system information can be processing system capabilities.

Processing system information for planning the component includes the following:

1) **Machine capability:** It refers to the processes available at the machine, spindle/spindles axis orientation information, machining head swivel information and the bed size envelop available at the machine. Machine capability can also be represented by either a set of tools and relative motions it can provide or manufacturing features it can produce.

2) **Machinibility data base:** This contains cutting speed, feed rate, depth of cut according to the workpiece material, surface roughness, tolerance and cutting type.
3) **Cutting tool data base:** Information on tool holder and inserts.
4) **Non-cutting time library:** information on set-up time and tool change time etc.

The difference between the various machine tools consist of: 1) the means by which the cutting tool and the material are held; 2) the means by which the motion of either tool or material, or both, is applied; and 3) the type of cutting tool employed. In all operations there must be a combination of work and tool movement except in some cases of surface grinding and broaching.

According to Boothroyd and Knight [1989], machine tools can be divided into three categories. Those using:

1. single-point tools
2. multi-point tools
3. abrasive wheels

Generally cutting tools may be broadly classified into four categories according to their geometric shape and basic function. They are (1) cylindrical surface-making tools; (2) flat surface-making tools; (3) hole-making tools; and (4) others. The last categories of tools include gear-making tools, thread-forming tools, etc. [Wang and Wysk 1988a].

6.6.2 PLAN BLANK

Design information depicts the part in its desired final condition. The material specifications determine how the material will be received, thus determining initial machining or fabrication operations. The first step towards the planning activity is the selection of blank or raw material. The criterion should be to choose the blank with the shape and size as close as possible to the finished part. This reduces the amount of machining and also material wasted during machining. However, to design the shape and size of blank too close to the shape and size of the finished part may be costly. The cost is justified based on the volume of the production. For example, if volume of the production is high, the blank can be cast to bring the raw material close to the part in shape and size.

The type of stock being used will influence how the workpiece can be held and, hence, the way in which set-ups can be achieved [Young 1991]. The aspect of planning the blank was not implemented in the HYCAPP System.

6.6.3 PLAN PROCESS

Under this function, the selection of processes required for the machining of the part features are undertaken. The next step is to optimise these processes for the whole component. Optimisation is based on minimising the number of processes needed to machine all the features of the part. The selection of manufacturing processes and determining of sequences of steps or operations required to accomplish the process constitute perhaps the most important function in Process Planning.

Most of the effort in the domain of Process Planning has been devoted to the extraction and normalisation of the knowledge about selecting operations [Wang and Wysk 1988a]. A process is responsible for the change of shape, dimensions, surface finish and properties of raw material, as already said. A number of variety of processes are used to convert raw materials into finished workpieces. These processes can be categorised into three main types namely 1) Forming, 2) material-Removal, and Chipless material-Removal as depicted in the figure 6.3. Most common processes involved under these classes are also shown in the figure. Among them, the machining or material-Removal process plays an important role in the manufacture of piece parts. Each part feature and associated attribute such as dimensions, tolerances, and surface finish are used to infer possible manufacturing processes to convert the blank into the finished product. The feature technological data provides the constraints on how the feature should be produced. Usually, a set of ordered sequence of processes is needed to fulfil the geometric and technological requirements of a feature.

A process is a function whose input parameters consist of a workpiece, a machine with its corresponding operating positions. A process changes a workpieces in some physical way, and a machine participates in process within given operational ranges depending on its characteristics and the types of tools it employs.

A blank undergoes a series of processes through to a finished product. Each process is responsible for giving a set of morphological outputs. Each morphological output is associated with specific technological output. For example, a drilling process can produce shapes like Round Hole and Round Pocket associated with certain levels of technological output in terms of size, dimensional accuracy and surface finish. A process is selected based on any optimisation criterion i.e. cost, time, production rate etc. or in other terms optimisation criteria helps processes to be compared and decision making based on economic grounds. Figure 6.4 shows the process function, in which the input to a process is blank and energy; and output is a component with specific type of morphological outputs at certain level of technological outputs and process is selected based on optimisation criterion. Morphological output can also be described as a set of motions along with toolings, a process can provide.

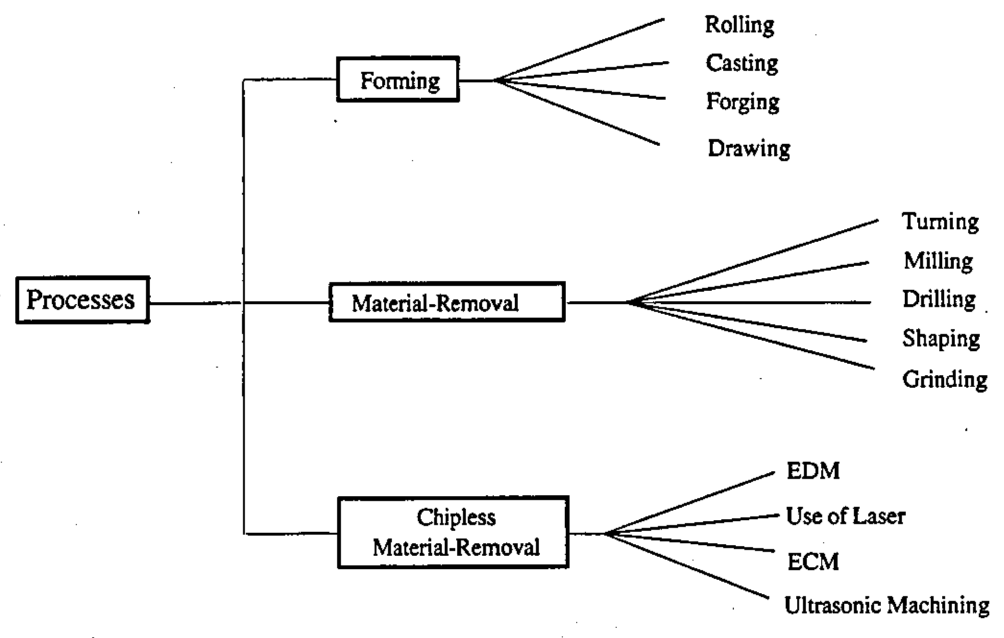


Figure 6.3 Different Processes

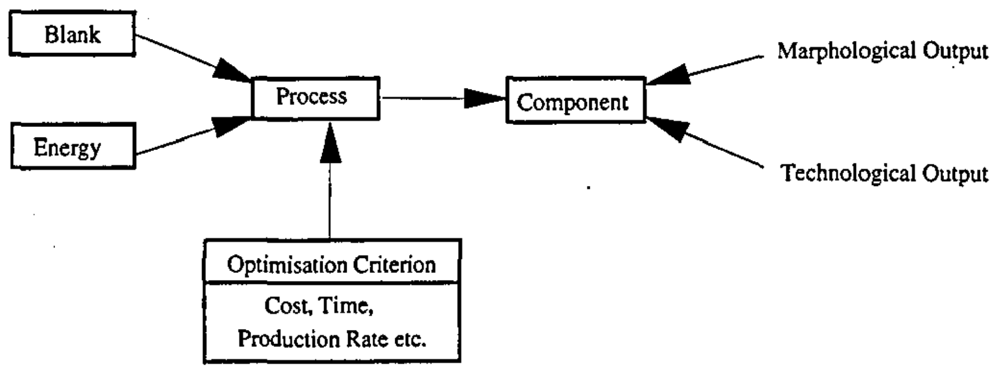


Figure 6.4 Process function

The complete modelling of the operators must take into account several factors [Joshi *et al.* 1986]:

- type of machine used,
- any preconditions to the application of the machine tool,
- type of relative motion between tool and workpiece,
- tool geometry,
- generated surface characteristics,
- range of motion.

A good discussion on modelling manufacturing process has also been given by Joshi *et al.* [1986] and Eversheim [1991].

Chang [1990] while summarising the most important capability parameters for Process Planning, quotes:

1. the shapes and size a process can produce,
2. the dimensions and geometric tolerances that can be obtained by a process,
3. the process constraints both geometric and technological constraints,
4. the economics of the process.

The process itself imposes restrictions which result from a number of factors, the most important ones being [Tanner 1985]:

1. *Material limitations.* The sequence of handling and machining a forging would differ markedly from that for machining the same part from a solid block of material.
2. *Machine flexibility limitations.* Several different operation sequences may be possible on general-purpose machines. but a special-purpose machine that is designed to machine only one part to a given configuration has one fixed operation sequence.
3. *Process capability limitations.* Machines frequently vary in their capacity to machine accurately when heavy cuts are made, requiring more or fewer operations in the process sequence.
4. *Tool limitations.* Grouping together of large-diameter cuts in the process sequence, rather than mixing them with cuts on smaller diameters which can be turned at higher speeds, reduces tool replacements and the changing of spindle speeds on turning operations.
5. *Production volume limitations.* Higher volume production can justify more efficient tooling. This may result in combining certain operations and also in shortening of the operation sequence.

Knowledge at the feature level can also be divided into two types; absolute and constrained. Geometric shape of a feature, real surfaces, imaginary surfaces and orientation it has got with respect to the component global co-ordinates, come under the category of absolute knowledge of a feature whereas constrained knowledge includes its attributes i.e. dimensions, tolerances, surface finish and auxiliary parameters such as radii etc. Every feature at absolute level is characterised by the number of its imaginary surfaces, boundary type (open or closed) and its entry/exit conditions.

After attaching the TSF's to the features, the Machine Capability Units (MCU's) are assigned to the features while matching the processing needs of the features to the processing facilities available in the machine shop or company which can machine them. It helps in determining the precise nature of machining capabilities/machine types needed for the processing of component features. Then machining requirements in terms of Machine Capability Units can be optimised in order to determine the realistic machining requirements, set-ups and AD's in the Process Planning function. Machine resources can be divided into Machine Capability Units (For detail please refer to chapter 4 section 4.5.3) and can be matched to processing needs of the features.

Approach directions can be used to assist in determining locating and fixturing faces, developing machining sequences for the faces in the part, and to obtain information such as tool length, etc. In order to determine tool approach direction both the global and local information of the part need to be considered. The procedure to determine tool approach directions is based on obtaining the possible directions for each feature and checking along that direction if any faces of the part would obstruct tool motion [Joshi *et al.* 1988].

A great deal of benefits are involved in dividing the machine resources into MCU's. Firstly, precise component processing requirements can be found and hence, it helps in identifying the similar components which are using the same MCU's. Whereas in the traditional way, the components have been taken as similar because they are visiting the same machines. In most of the cases, in spite of visiting the same machines, they are dissimilar. Secondly, machine capabilities can be easily compared in the case of scheduling and machine allocation problems. Thirdly, it allows MCU's to cope with any machining strategy i.e. flow line, machining centre, Cellular Manufacturing, at any point of time in a shop or manufacturing cell.

The author is of the view that operation optimisation of the features planned at local level for the determination of set-ups is not a good idea. Willis *et al.* [1988] argue that the features in isolation are insignificant for the synthesis of a machining plan for the whole component. Sometimes for the same operation, the component might need to visit another machine tool. Set-ups determined that way will not be realistic ones. The movement to another machine will add other set-ups in the planning. In order to make the point clear, the example of the rotational component shown in figure 4.5 in chapter 4 can be cited here again. The component has Round Pockets (co-axial, axial, radial) at different locations. Although all the Pockets require a drilling process obviously there may require more than one machine tool to process these features. If the component could have been completed on a single machine tool, two set-ups were involved and even one if the machine spindle could approach both AD's. When three different machine tools are required for machining the component features at least three set-ups are required. Therefore optimisation based on the operations attached without considering the connectivity aspect of the component features does not give the realistic set-ups picture.

The author believes that it is good idea in advancing the set-up planning if after matching the true MCU's which are responsible for machining the features are attached to the features in the beginning and then optimised. This will exhibit the realistic picture of set-ups and the processing needs of the components. The above mentioned approach has been adapted for the set-up determination in the HYCAPP System. In the beginning multiple solutions or TSF's are attached to the features based on their technological requirements at local level. Then considering the connectivity aspect of the feature, the processing requirements are matched with the available MCU's in the shop and TSF's are transformed into MCUE's. MCUE's can be taken as feature state elements or operation as each element will be responsible for the change in the feature state. Now this is the right time to optimise the feature state elements. This optimisation will result in representing the true picture of processing requirements of the component as well as leading to correct set of set-ups involved for the machining of the component. The author is of the view that components should be characterised based on their requirements in terms of machine types at generic level. This provides a realistic basis for comparing the machining capabilities of different machines in the component scheduling and machine allocation process.

After optimisation, feature solutions in terms of MCUE's are attached to the component features. Set-ups involved for the machining of the component are determined by clustering the MCUE's or feature state elements from common component PAD's. Since a feature may have several EAD's, the feature state elements may appear in more than one cluster in different component PAD's. The final component set-ups are selected by minimising the number of clusters. This is done based on step by step selection of clusters containing the maximum number of feature state elements and then removing those feature state elements from the remaining clusters in the other component PAD's. Ideally, the Feature clustering should be performed while observing the precedence relationships that may exist among component features. This aspect of observing the precedence relationships was not implemented as neither of the components in the family had this type of relationship.

Under the 'Plan Process' function, all the TSF's or feasible solutions at feature level fulfilling their geometric and technological requirements are sorted out and transformed into MCUE's while considering the connectivity aspect of the features. MCUE's fulfilling the machining requirements of the features are then optimised and assigned to the features as their solutions. The next stage would be machine tools allocation for the processing of the component.

6.6.3 PLAN MACHINE TOOL

Under this function, the machine tools available in the manufacturing cell or shop for the machining these MCUE's will be evaluated and allocated to the components.

The Process Planning function is totally dependent upon the manufacturing system. This means that the planning function is affected directly by the capability of the manufacturing system which would be responsible for producing the component. The manufacturing system, having a precision machining centre, can produce, perhaps, most of the components on a single machining centre. On the other hand, several machines may be needed for the components requiring the same precision.

At the stage when component requirements in terms of MCU's are in hand, the machine allocation problem is to find out which machine tool can provide the maximum MCU's required by the component and can accommodate it on its bed i.e. component overall size is less than machine envelop. Machine capabilities in terms of MCU's, machine head swivel constraints and bed size of the machine tool can be defined in the database.

There are mainly three types of productions which include single-piece, batch and mass. Single-piece type production is usually dealt with in the traditional process-oriented workshops. Mass production is large scale and stable type of production, for which special-purpose machine tools and tooling are employed. The machine tools are arranged based on the sequence of the manufacturing operations, some times called flow line production. According to recent trend, batch type production which is in between the other two is handled in manufacturing cell.

There are other issues which effect the machine allocation problem; the first one is related to management policy for machine allocations and the second one is the volume of production. Management strategy might be minimising the number of machine tools used to produce the component. This trend might overload the most capable or flexible machines and leave less flexible ones under-utilised. Machine flexibility is processing the capability of a machine to perform a number of processes on a component without unloading it from the machine. A flexible machine is responsible for reducing the amount of material handling as a number of processes required on a part can be performed without changing the machine. Machining centres are considered the most capable and flexible machine tools. To avoid such extremes (over-loading the flexible machines and under-utilising the less flexible ones), the policy may be adopted to sequence the parts to a number of less capable machines to give it a flow line pattern. In machine allocation the volume of production is also given due consideration. In the case of high volume of production, special purpose machines and flow line patterns are considered. In MCU terms, flow line patterns may exhibit each MCU being assigned to a separate machine and on machining centre all the MCU's requirements might be fulfilled on a single machine. In case of manufacturing cells MCU load may need to be distributed unevenly among a number of machines.

Machine allocation is a considerably complex issue in a real world situation. Sometimes some machines breakdown, in that case jobs are to be re-scheduled and yet balance the machine load in the cell or workshop at the same time. It can be, therefore, argued that the optimal solution is management specific.

The aspect of planning the machine tools was not implemented in the HYCAPP System.

6.6.4 PLAN SET-UP

After determining the MCUE's and the equipment or machine tools required along with the feature connectivity, relationships information, set-ups and fixtures required can be finalised. Chan and Voelcker [1986] propose that set-up planning should become a subset of Process Planning, with operation planning becoming a downstream activity.

Locating and clamping of a workpiece, when it is in a correct orientation relative to the axis of a machine tool is termed as a set-up. When many set-ups are needed to make a particular part, the part is turned in a number of series of orientations step by step. The number of set-ups should be minimised for the processing of a component as more set-ups involved in the machining of a workpiece cause more time, more cost and more set-up errors.

Factors which influence decisions on set-up planning can be described as follows [Young 1991]:

- the number of components in a batch, and whether different components should be produced at the same time;
- the type of stock e.g. cast, forged or solid;
- the type of machine being used;
- the available fixtures;
- suitable features of the workpiece for fixturing;
- the interaction of fixturing with machining;
- problems affecting dimensional accuracy.

As already discussed there can be a flow-line strategy or machining centre strategy or manufacturing cell strategy for the selection of machine tools which can be adapted based on the situation. The strategy adopted is a machining centre strategy which is to select the machine which can provide the maximum number of MCU's. Thus reducing the machining time, number of set-ups required and work-handling time which will result in the overall cost reduction involved in the production of a workpiece. Though single set-up time on the machining centre may be more as compared to the simple machines because of complexity of set-ups and flexible types of fixtures available on these flexible machines. The issues regarding the machines flexibility have been addressed by Nandkeolyar and Christy [1992]. They argue that more flexible machines will require longer set-up times due to their complexity.

The type of machine and orientation of spindle axis will also influence fixturing methods and set-up planning, for example vertical milling machine verses horizontal milling machine. The number of axis available on the machine also influence the fixturing methods as well as planning the set-up. The planning methods and planning set-up will be different on the three axis machine and four axis machine. A four axis machine tool having an indexing or rotating table as its forth axis can present several faces to the cutting tool. Therefore in a single set-up several AD's are available for machining. Also different structures of machine tools pose restrictions on the fixturing methods. The shape of raw material or blank will also effect the selection of set-up and the fixturing methods to be used. Geometrical tolerances influence the fixturing strategy too, for example a perpendicularity constraint on the surface can exclude those surfaces from fixturing. A considerable amount of research work has been done by Boerma and Kals [1988, 1989] in the selection of set-ups and design of fixtures. Willis *et al.* [1988] and Kanumury *et al.* [1988] have done considerable work in the use of fixture strategies. Young [1991] in his 'Machine Planning' work also discusses fixture planning.

Features are grouped by spindle axis direction. According to Joshi *et al.* [1988], the set-ups in which features can be machined relate to their spindle axis directions. Feature EAD's determine the orientation, with respect to the machine spindle, from which the feature can be accessed. After planning a set-up there is a need for a system to record the pre and post machined features such that planning can be done for the remaining features in the next set-up. If there exists overlapping of features across the set-ups, there is need to adjust the parameters of the feature to be planned in the next set-up.

Component set-ups can be considered as clustering the feature state elements or MCUE's requirements which can be done from the common PAD's on the single machine tool. All the feature MCUE's are grouped in clusters in each PAD. These MCUE's can appear in different component PAD's. The final component PAD's will be selected by minimising the number of clusters. This will be achieved by selecting the clusters with maximum number of feature MCUE's which are available on that particular machine. The selected feature MCUE's will be deleted from other component PAD's. The point worth noting here is that the clustering needs to be done while observing the precedence relationships that may exist on the component features. This will result in finding the AD's through which all the features on the component can be machined. Then with the machine swivel information, all the AD's will be grouped for a single set-up which the machining head can cover. In the case any machine has more than one spindle head which can perform cutting in parallel, the AD's covered under the swivel of multiple machining heads will be grouped. MCUE's present on these AD's will be machined in a set-up. Another point to be noted here is that fixtures may also impose restriction in selecting the set-ups. Kanumury *et al.* [1988] identifies the need to check that machining features are not overlapping the fixtures. The next step will be sequencing all the operations in this set-up.

In figure 6.5 set-up planning has been shown. Component features as well as PAD's have been tagged. Component AD's (Approach Directions) and component processing requirements in terms of MCU's have been finalised (PAD1, PAD3 Approach Directions and four MCU's required). Please notice that the solutions for Rectangular Through Slot (F3) and Rectangular Not-Through Slot (F4) are different. Feature F3 being narrow and deep can only be machined on horizontal milling machine type whilst feature F4 can only be processed on vertical milling machine type. Similarly Feature Round Hole (F2) and Round Pocket (F5), both of them require drilling process but require different machine types (different MCU's). F2 requires general drill machine whereas F5 requires horizontal drill machine as the component is too long and cannot be oriented on general drill machine. Therefore, MCU's have been characterised based on machine types and assigned to the features while considering their connectivity aspect as shown in the figure.

After optimising the feature state elements or MCUE's needed for the component, machine swivel/multiple heads information available on the selected machine/machines will be analysed. Consider, for example, a machine like plano-miller is selected where all the five MCU's are available, the next step will be to check whether this machine tool's head can approach PAD1 and PAD3 at a time by swivel, or if it has got two heads which can cover both PAD1 and PAD3 at a time. If the answer is yes then the component will be processed in a single set-up. Again, if the same machine has been selected but its head cannot approach PAD1 and PAD3 at a time, in that case another orientation of the component is required to machine the MCU's on other AD, or the machining head will be swivelled to approach the other AD and hence the component will be machined in two set-ups. If one machine cannot provide all the MCU's i.e. MCU#4 is available on another machine. In that case two set-ups are required.

Set-up planning is not a straight forward job. Sometimes back-tracking is required which can be due to parent/child relationships of the features or other geometrical constraints. Back-tracking will result in re-planning the set-ups.

Set-up planning while considering the machine allocation problem was not implemented in the software.

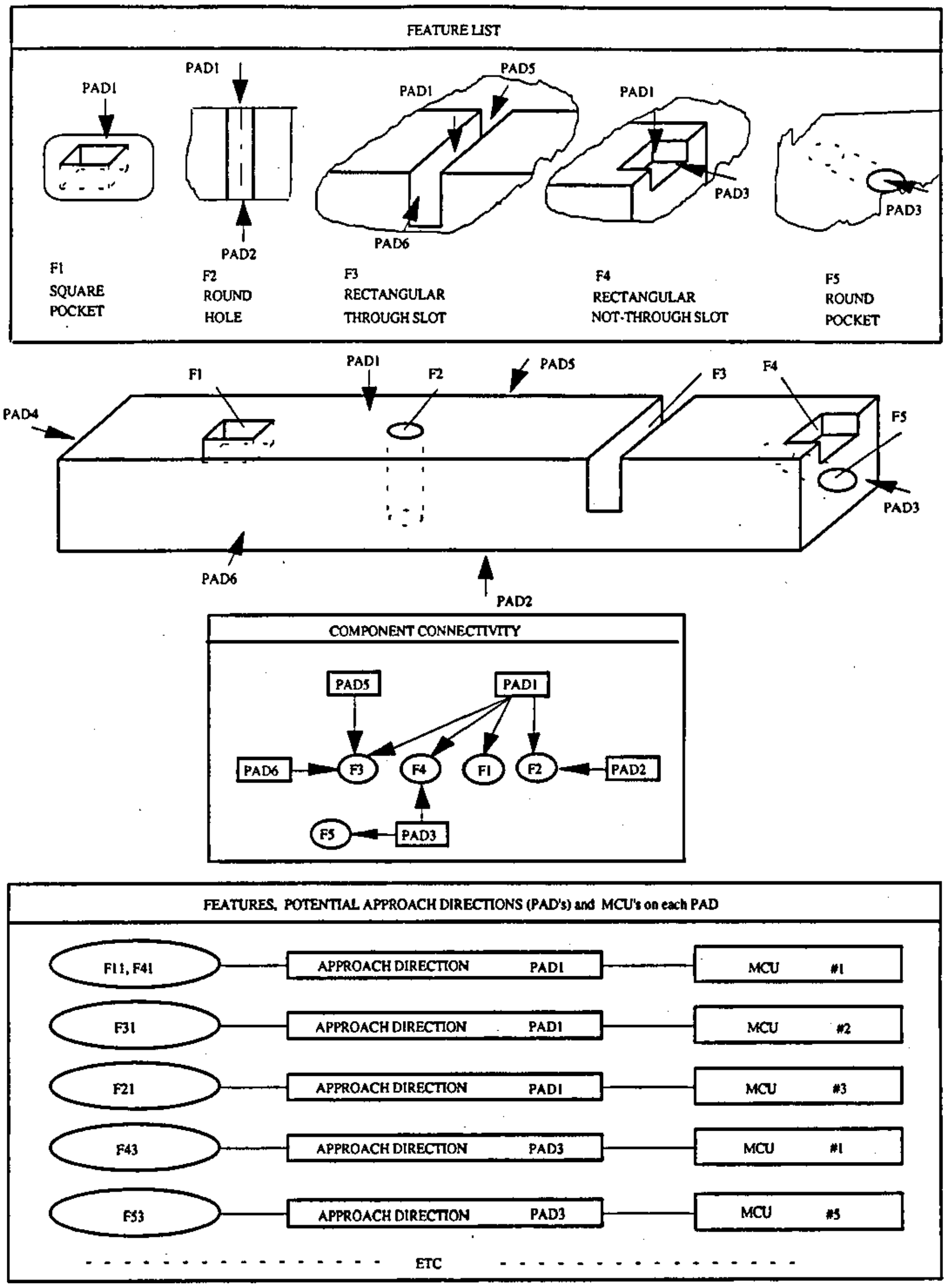


Figure 6.5 Component Connectivity and Set-up Planning

6.6.5 PLAN OPERATION SEQUENCE

At this stage, a set-up at any particular machine has been finalised where all the MCUE's available on this particular machine tool and all the AD's enabled by the machine head or swivel head can be machined.

At this level, process activities in a particular set-up can be divided into their elemental operations. After determining the operations involved in one set-up on any particular machine tool, the following can be evaluated:

- the cutting tools
- the cutting parameters
- sequence of operations

A machining process is made up of a number of operations through which raw material or blanks are transformed into parts [Wang and Li 1991]. The planning at this level can be termed as 'operation planning' which involves cutting tools selection, cutting parameters selection and sequence of operations. Further details on the operation planning can be found in the publication by CAM-I [Detollenaere *et al.* 1988]. The point worth while to be noted here is that it might not be possible always to carry out the sequence of operations in one process while sequencing the operations in one set-up. The operation is carried out by the unchanged machine tool, unchanged workpiece, and unchanged cutting tool. Though to achieve higher efficiency, more than one tool can be engaged to perform machining on the workpiece in a single set-up (e.g. lathe equipped with turret), an operation will be characterised by a single tool. Exception can be given to the straddle milling where two milling cutters are used to perform the operation. As already said in chapter 4, the words process and operation are used interchangeably since operation level planning is not addressed in this work.

6.6.5.1 TOOLING AND CUTTING PARAMETERS SELECTION

Production, process and operation planning are highly interrelated in practice [Ham and Lu 1988]. Cutting tools affect surface quality and operation cost. While planning operations, decisions have to be made on which cutting tool is to be used and what cutting parameters are to be selected. Cutting tool selection is based on the feature geometric constraints and technological constraints whilst achieving them in minimum possible time. A wide variety of cutting tools exist which can be divided into standard tooling and non-standard tooling (according to the particular needs). Commonly used standard tooling are face mills, end mills, side mills, ball-nose cutters, drills, reamers, Taps and boring bars etc. Non-standard tooling are company specific which company uses for its particular requirements.

Tool selection is based upon the type of process selected, dimensions, accuracy and material selected. Matsushima *et al.* [1982] and Chang and Wysk [1981] have studied tool selection for hole machining. Melkote and Taylor [1988] have done research work in the selection of milling cutters. Choi and Barash [1985] have studied both hole making as well as milling cutters.

Carlier and Peters [1985] point out that a specific cutting tool which is only for a particular operation should not be selected, rather a tool should be selected which can machine the maximum possible number of operations. This strategy will result in the reduction in tool change time and also the requirement of a large number of cutters in the machine's carousel.

Assigning 'good' machining parameters is important with respect to increasing productivity [Wang and Wysk 1988]. A detailed study of machining parameter selection using an Expert Systems approach is given in Wang and Wysk [1986]. The process parameters i.e. cutting speed and feed rate values depend on the particular tool, the workpiece material, operation type, feature geometry, dimensional tolerances and surface roughness. The machining time for an operation is calculated from the geometry of the feature and process parameters that have been decided. Tooling and cutting parameters selection in the AI environment has been reported by Phillips and Mouleeswaran [1985], Subramanyam and Lu [1988], and Chryssolouris and Guillot [1990].

Tooling and cutting parameters selection aspect was not implemented in the HYCAPP System software.

6.6.5.2 OPERATION SEQUENCING

The aim of sequencing the operations is to enable all the required operations on a particular machine set-up to be performed, in the minimum possible time whilst still achieving the geometric and technological requirements of the features involved. Harhalakis *et al.* [1990] argues that the sequence of operations in a part production routing has an important effect on material handling costs and times. The process of extracting and formalising rules for operation sequence is much more complicated than that of operation selection [Wang and Wysk 1988].

Tanner [1985] while discussing the process sequences sets the guidelines as given below:

1. Surface finishes in critical areas may be such that in order to prevent damage, they must be accomplished at a point in the process when the surface can best be protected from damage or mutilation.
2. The sequence of machining operations that must take place before the operation in question can be performed must be considered. As an example, a hole must be drilled before it can be tapped.
3. The degree of accuracy that must be maintained between related surfaces may require that several less critical operations be performed before the final operation on the surface can take place.
4. The introduction of auxiliary operations into the major operation sequence may determine when the work can be performed. For example, if a given surface may be hardened, it may be necessary to grind the surface to obtain the required finish.

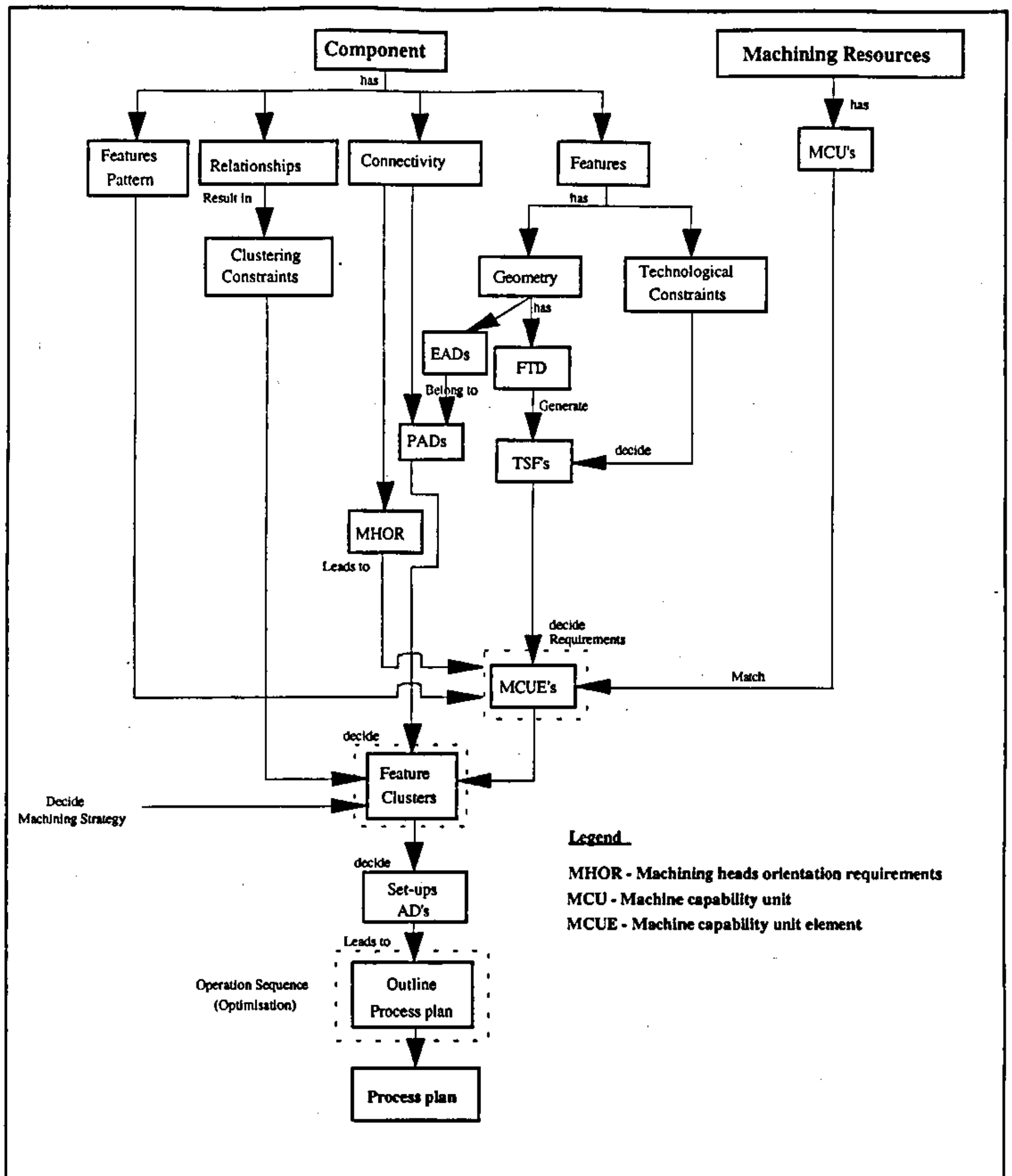


Figure 6.6 Reference Model for the HYCAPP System

Generating an efficient plan is to find a good order in which to cut the features. One of the obstacles to accomplishing this is capturing the computer reasoning logic about the problems associated with the feature interactions, feature connectivity aspect, thin walls etc.

The problem of thin walls occurs when features are too close to each other. The wall between them is feared to be distorted by the machining forces when they are machined. Such type of problem has been identified by Willis *et al.* [1988] who emphasises to take into account the thin walls problem. A lot of reasoning process is also involved about the connectivity aspect of the features which is the way features form the component. A feature interaction happens when cutting some of the features affects the way in which others can be made. Subsequent features may become totally impossible to produce, or the methods by which they can be produced are restricted.

Feature interactions have several different causes. Most commonly they result from clamping problems; producing one feature destroys the clamping surfaces needed to grip the piece while cutting another feature. An interaction can result in restrictions on either the process used to execute a step or the order of the steps [Hayes 1988].

The example chosen by the Hayes [1988] which put restrictions on the order of the steps has been given in the figure 6.7.

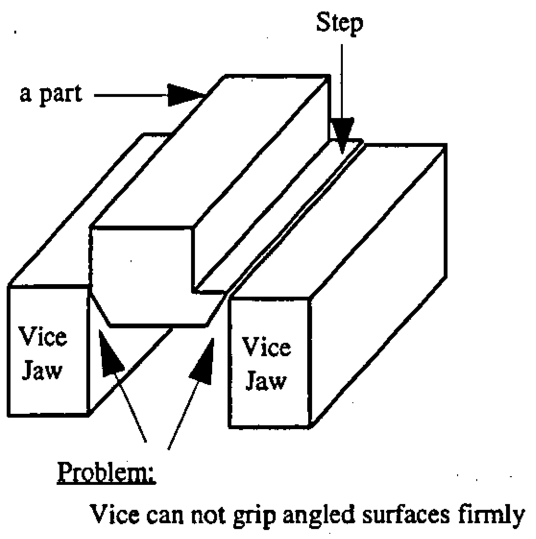


Figure 6.7 .Feature interaction: the step must be cut before the angles.

Operation sequencing can be categorised as below:

1. sequencing based on operation constraints;
2. sequencing based on geometric constraints;
3. sequencing based on tooling constraints;
4. sequencing based on geometric tolerancing constraints;

5. sequencing based on dimensional/datum relationships constraints; and
6. sequencing based on 'good manufacturing practice' constraints.

1. SEQUENCING BASED ON OPERATION CONSTRAINTS

These types of technological constraints are set in any selected TSF which is a solution at feature level. The next operation in a TSF will only be conducted if the previous one has been processed. Examples of these types of operations are drilling which is always done before the boring operation and rough turning will always be followed by the finish turning. Carlier and Peters [1985] have also identified the sequences based on operation constraints.

2. SEQUENCING BASED ON GEOMETRIC CONSTRAINTS

This type of constraint exists when one feature sits on the top of another, or the term usually used for this type of geometric relationship is parent/child relationship. Parent feature is mostly done followed by the processing of its child feature. Therefore, it is customary to machine the parent feature before the processing of its child feature. Joshi *et al.* [1987] has used geometric constraints as a means of sequencing the machining the features. The precedence information specifies the sequence in which the features are to be machined relative to each other.

The Process Plan for a part should not only include the sequence of operations for manufacturing an individual feature, but also establish a precedence among the set of features comprising the part [Joshi *et al.* 1988].

3. SEQUENCING BASED ON TOOLING CONSTRAINTS

This type of sequence constraint is set by the available tools in the workshop or cell. Similar types of features can be grouped together to machine them with the same tool, thus saving significant tool change time. For example, features like Round Holes and Round Pockets can be grouped together to machine them with the twist drill tool.

Features patterns are also grouped together in the sequencing process because they require the same tooling.

Carlier and Peters [1985] state that there may be the need to machine two features with a common tool if features are connected with a dimensional relationship.

4. SEQUENCING BASED ON GEOMETRIC TOLERANCING CONSTRAINTS

The geometric dimensioning and tolerancing of a workpiece sets important constraints on the selection and sequencing of operations. Four geometric tolerancing characteristics—flatness, circularity, cylindricity, and straightness are related to operation selection while the rest of the geometric tolerances are related to operation sequence. Two surfaces with a strict parallelism tolerance must be machined in the same set-up in order to reach the tolerance requirement. Two holes with a close concentricity tolerance should be drilled in one set-up. These type of constraints are used to lump selected operations together into groups to simplify the operation sequence problem [Wang and Wysk 1988].

5. SEQUENCING BASED ON DIMENSIONAL/DATUM RELATIONSHIP CONSTRAINTS

The critical areas on the body of the component are those areas which provide part location for subsequent machining operations, and those areas require demanding functional tolerances or shapes or finishes. In the course of planning the operation sequences, these critical areas (datum features and reference features) are planned first before any subsequent operation sequences, in order to ensure that the important dimensional relationships of the component are being maintained. Then there comes the turn of critical areas which are important for the functionality. These areas may demand control on close surface finish or tolerance.

Study of the dimensioning between part surfaces will be conductive to the determination of the machining sequence in a Process Plan [Wang and Li 1991]. In the case of milling operation, all the features whose dimensions have a reference surface milled are isolated. The milling operation is sequenced before such operations so that the milled surface can be taken as reference for these features. If features are related to a common datum then features will be machined in the same set-up and corresponding datum feature will have precedence to be machined.

6. SEQUENCING BASED ON 'GOOD MANUFACTURING PRACTICE' CONSTRAINTS

One of the factors to be considered while sequencing the operations is 'good manufacturing practices'. Applying the drilling operation after the milling is usually better practice since it is easier to remove burrs from the flat milled surface. Cross Holes are usually sequenced before the principal Hole to avoid burr opening out into the principal Holes. This is because most of the cross Holes are either for oiling or fixing purposes while the principal Holes are used for functional requirements such as the mounting of bears etc.

Though in most of the parent/child relationship cases, it may be possible to machine the child feature before machining the parent feature, it is against 'good manufacturing practice' to do that. There can be several examples for that: 1) Hole feature at the bottom of surface feature; 2) Hole feature at the bottom of Step feature; and 3) Hole feature at the bottom of another Hole feature. Obviously it is possible to machine child features before their parent features in the above cases but it will not be 'good manufacturing practice' because it involves more machining time, risk of tooling damage etc.

The first four operation sequencing methods described have been used by Wang and Wysk [1988] in their planning system.

To encapsulate all these constraints in a computer system is very difficult. According to Beigel [1986] Artificial Intelligence (AI) proves difficult to handle.

Software developed in the HYCAPP System can handle sequencing based on operation constraints, geometric constraints and 'good manufacturing practice' constraints. Sequencing based on tooling constraints was not implemented in the software. As far as sequencing based on geometric tolerancing constraints and dimensional/datum relationships constraints are concerned, no such constraints existed in the family of the components planned in the software.

CHAPTER 7

IMPLEMENTATION OF HYCAPP SYSTEM

7.1 INTRODUCTION

After discussing the Process Planning in general and Generative Process Planning, in particular, in the context of HYCAPP System, this chapter highlights the implementation of the HYCAPP System. After the introduction, section 7.2 describes the system implementation in general, need for the modular structure and details of modules developed in the system. In section 7.3, an example part from the part family has been chosen to show the results generated by the different modules of the system.

Section 7.4 discusses the composite component of the part family for which the system is developed and details the process plan generated for the composite component by the system. The concluding section describes the boundaries and limitations of the HYCAPP System.

7.2 SYSTEM IMPLEMENTATION

Developing the Generative planning system is a large and difficult problem. Most of the planning systems automate only a part of the problem. For example, the SIPS [Nau 1987b] system concentrates in finding the cost effective operations of one feature at a time. Other systems can choose cutting tools, select optimised machining parameters, plan the paths that the tool will follow as they cut, or as in computer numerical control (CNC), generate computer code for the machine tool.

However, some areas of Process Planning have not received much attention from the researchers. One such area is set-up ordering. A set-up can be taken as machining a group of features when the part is in a particular orientation with respect to the axis of any single machine. When many set-ups are required for machining the component features, the component is turned in a number of orientations. HYCAPP is capable of optimising the AD's i.e. calculating the minimum number of set-ups required to machine the component. It optimises the set-ups by selecting a minimum number of processes required for machining a component. In a case when more than one processes have equal weightage for same machining work contents on a component, the HYCAPP System will ask the user to select a process among those candidate processes in the course of process selection. The system will start the PAD from which has maximum work content in terms of MCUE's and proceeds towards other PAD's which involve the work contents in descending order. The sequencing of the operations is then performed for each set-up.

This system focuses on generating process plans for a family of parts made from prismatic blocks of metal. A number of geometric shapes or features are cut out of the block to form the part. Features are described to the software by several parameters. Each feature has a different set of parameters.

As far as geometry of the feature is concerned, feature can be considered as a combination of real and imaginary faces. EAD's of the feature are the normal vectors on the imaginary faces of the features. Each feature type has got a fixed number of EAD's that are fixed at the primary sub-class level. As already said, a feature can be approached for machining from its EAD's. All the EAD's of the feature may not be feasible to be considered as machining directions. Some feature EAD's may be considered in preference to others for technological reasons (for example, feature dimensions). Features with common EAD's are clustered together to link them with the component PAD's. A PAD is a potential access direction for a machine tool at component level through which features on the part are machined. All features are described and their faces labelled for determining their processing requirements, set-up planning and operation sequencing.

It is important to have a representation that reflects the way in which the information will be used. The system views the component as consisting of a set of features connected to six free surfaces of the component. The normal vectors specified on these free surfaces are taken as PAD's. EAD's of the component features being normal vectors to the imaginary faces or surfaces are linked to these PAD's so that features can be accessed through these PAD's.

The block that one starts with is known as the stock. The part is represented in the system as a rectangular block from which features are subtracted one by one. The problem is specified by describing the geometry of the finished part as well as six free surfaces on the component so that the connectivity of the features and parent/child relationship can be calculated. The outer envelope of the part is represented as a prismatic solid. Each side of the outer envelope is labelled as a number to the free surface of the part. These labels on the PAD's will be used later to describe the features potential approach directions and then the final optimised AD's through which all the features of the component will be machined. For instance, the feature Hole_a can be accessed from PAD_x and PAD_y for its machining as it has two EAD's.

The system does high-level planning, it groups the features into set-ups and orders the set-ups. It does the sequence of operations within set-ups, but it does not work on problems at any lower level than that. It does not dissect the features down into their components like rough cuts and finish cuts, nor does it involve in selecting the cutting tools or machine tools. It also does not plan the paths that the tools follow as they cut through the metal. The clamping of the workpieces or use of jigs and fixtures has not been considered in the program.

Programming of the system attempts to emulate the human planning process. The human planning process is described well by Hayes [1987]. Only the computer implementation will be dealt with in the following discussion. The reader is expected to bear in mind that the software developed only demonstrates the logic needed in the development of the Process Planning process and the system works only in a defined range or boundary. The system receives a feature-based description of a part which consists of a set of features which need to be cut from a block to make the part. The part is described in the system by interactive dialogues. There are eight feature types involved in this system as shown below:

round_hole	contoured_step
triangular_step	round_pocket
axial_round_pocket	rectangular_slot
surface	axial_round_hole

As already mentioned elsewhere in the thesis axial_round_pocket and axial_round_hole in the above mentioned feature list represent round_pocket and round_hole features and the word axial shows the connectivity of the features on the component body. This has been done in order to make the connectivity reasoning of the features easier in the software. The concept of defining feature type with the connectivity aspect has been also used by Detollenaere *et al.* [1988] in their expert operation planning system. TSF's for a feature are determined based on its type and technological constraints associated with it. TSF's are then transformed into MCUE's with the connectivity information of the feature. Information flow in the system, in broad terms, has been shown in figure 7.1.

The execution of rules in the software takes a lot of time, therefore, it has been the objective to avoid rules as much as possible. Mostly procedure files have been used in the design of the system, since this way has been proved to be more efficient. All the procedure files are given in the appendix E.

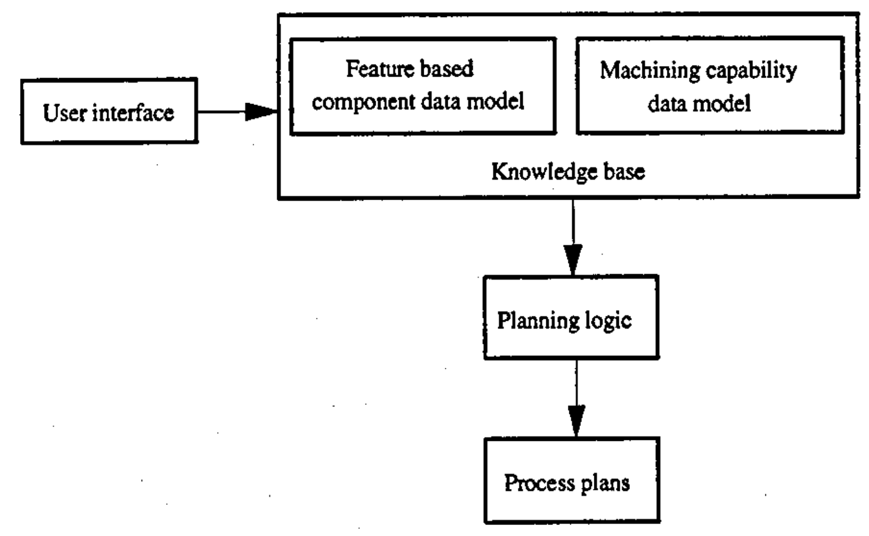

User interface
Feature based component data model
Machining capability data model
Knowledge base
Planning logic
Process plans

Figure 7.1 Information flow in the HYCAPP System

7.2.1 THE MODULAR STRUCTURE

In the CAPP system, the decision logic from an expert and the factual knowledge about the component to be manufactured and manufacturing system needs to be structured and organised into well-framed facts and rules that form the basis for inferring the new goals.

Developing Generative planning system is a lengthy and difficult job, as said before. Expansion and modification of the system would also be difficult. However, by using a modular approach, the task of developing as well as expansion and modification of the system can be made easy and manageable. Under a modular approach, the major task is decomposed into sub-tasks and modules can be developed for these individual tasks. Each module uses its input to create output and is functionally independent from other tasks. However, the modules can communicate with each other through their data input and output facility.

An interface (a module) has been written to create records about the feature-based component data model which interactively asks to enter the information about the component step by step. The module called 'input_part' can be found in the appendix B. Planning logic has been developed in the system for a family of 14 components. Design data for each component is in individual dump file outside the GENERIS application ready to be read in by the system. Only the design data for a particular part should be in the knowledge base for which process plans are to be generated.

7.2.1.1 DESCRIPTION OF THE SYSTEM MODULES

This section describes briefly the major components of the software developed in the system. After changing the home directory to directory called 'plan', the system starts by typing in 'generis' on the prompt. GENERIS Expert System opens its window for its use. The next command to be inputted here is 'do generisinit'. System opens the user defined window called 'WELCOME' and asks the password to start the application. The password is simply <return>. After entering the password, a new user defined window called 'MAIN MENU' appears, in which different system modules can be selected. GENERIS MENU FILE name is 'menu_main'. Title of the menu window is 'SELECTION' and the Menu screen shows the following modules:

1. TO CREATE THE RECORDS FOR A COMPONENT
2. TO EDIT THE RECORDS FOR ANY COMPONENT
3. TO DELETE THE RECORDS OF ANY COMPONENT
4. TO DUMP THE RECORDS FOR THE COMPONENT
5. TO LOAD THE RECORDS OF PART FOR PROCESS PLANNING
6. TO CHECK FOR A NEW PART WHETHER IT CAN BE PLANNED
7. TO GENERATE THE PROCESS PLANS
8. TYPE ANY COMMAND
9. END

The details of the main menu selections are given below:

Selection 1 is for describing the design data of the component in the system.

Selection 2 is to edit the records at feature level. Dimensional parameters and accuracy or surface finish of the feature can be changed.

Selection 3 can be used to delete the design data of any component.

Selection 4 dumps the component design data outside the GENERIS application.

Selection 5 is to load the design data of any component in the family for generating process plans after deleting the design information of already existing component in the database.

Selection 6 is for checking whether any part outside this particular family can be planned. In the beginning, the system details the boundaries of the system. Then the system displays the message after checking whether all the component features can be planned or not. A list of the features which cannot be processed will be displayed. The user will be prompted to enter the solutions for those features one by one among the processes listed. Multiple solutions can be entered for each feature. When solution for every feature would have been provided, the system gives the message that the component can now be planned.

Selection 7 generates the process plan for the loaded component. The input to this module is the design data of the component represented in the system as specified on the engineering drawing. Further details of this selection will be given latter.

Selection 8 accomplishes any command entered to the system. It might be for displaying the component records in the application, tables defined in the application, rules written in the application, entities defined, any inquiry from the system etc.

Selection 9 ends the session.

It is worthwhile to note here that all the above mentioned modules work independently. An overview of the HYCAPP System modules hierarchy is shown in figure 7.2. At the time of running of any module, the banner showing the function of the module along with the request for 'wait' will be displayed. There is no need to interrupt the system during the execution. GENERIS can report the error to the user, if there is any problem. After the execution of a module, the results generated by the module are shown within the user defined window. The user should press <return> key twice after having a look at the results, in order to either finish the module or execute the next modules step by step.

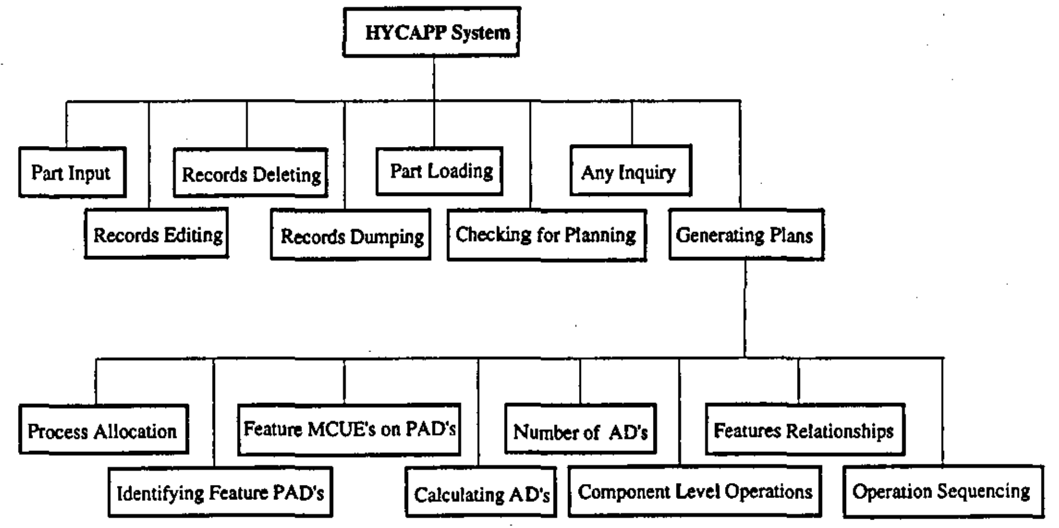


Figure 7.2 An Overview of the HYCAPP System Modules Hierarchy.

7.2.1.2 THE PROCESS PLANNING MODULE

Having loaded design information for a part and having checked whether or not it can be planned using selection 5 and 6, selection 7 is 'TO GENERATE THE PROCESS PLANS'. Eight sub-modules have been designed to accomplish the Process Planning module. In most of the cases, the modules require the input from the previous modules. User defined 'Output Forms' have been designed to show the output results from each module. The system consists of a number of modules. The details of the modules are as given below:

MODULE 1 (ASSIGNING PROCESSES TO THE FEATURES)

The input to the module is the feature types along with their technological constraints. This will result in the TSF's (all the possible solutions at feature level defined in the system), that can be used for their machining while fulfilling their geometric and technological requirements. These TSF's will then be transformed into MCUE's with the connectivity information of the features. Output form called 'potentially1_form' will be responsible for showing the output results. The results will also be written in a table called 'potentially1' for input to the next module. The output will include featurecodes, features and MCUE's assigned to the features of the component. The procedure file for this module is 'do_potentially1'. Inference rules involved in this module are as follow:

```
Inference rules in 'oper'
featurecode.1 'has feature1' feature.1 potential1 operation.1 if
        part.1 'has featurecode' featurecode.1 'has feature' feature.1 and
        feature.1 'has mc_method' operation.1 .
featurecode.1 'has feature1' feature.2 potential1 operation.1 if
        part.1 'has featurecode' featurecode.1 'has feature' feature.1 and
        featurecode.1 'has sec_featurecode' featurecode.2 'has sec_feature' feature.2 and
        feature.2 'has mc_method' operation.1 .
```

MODULE 2 (CALCULATING POTENTIAL ACCESS DIRECTIONS)

The input to the module are the real and imaginary faces of the features and component PAD's. There is need here to calculate the parent/child relationships of the features in order to calculate the PAD's for the features. Free surfaces are taken as the parent features of all the features on the component to be machined. All the faces of the features are labelled in the database. A rule named 'parent' is designed to calculate the parent/child relationship between the features. It works on the grounds that a feature_x is a parent of feature_y if feature_x has a real face_z and feature_y has same face_z as an imaginary face. As a result, usually, a child feature will come up with more than one potential parent feature. It is not always possible to machine a child feature from all of the PAD's of its parent feature.

Another rule called 'n_vector1' is responsible for calculating its parent features from which side it is possible to machine the child feature. The contents of rule 'n_vector1' are somewhat like "If normal vector to the imaginary face of the child feature is parallel to any normal vector to the imaginary face of the parent feature, then the child feature can be processed from that parent features sides or PAD's". Consequently, a child feature will inherit those PAD's of its parent features. Output form called 'results_form' will be used to show the results on the computer screen from this module. The output will be featurecodes, features, parent features and component PAD's through which the features can be machined. The output from this module will look like as in table 'results'. The procedure file responsible for this module is 'do_results'. The above mentioned rules in the real world are given in the following:

```
Inference rules in 'parent'
featurecode.1 'has parent' featurecode.2 if
        featurecode.1 'has imagin' featurecode.3 and
        featurecode.2 'has real' featurecode.3 .
featurecode.1 'has parent' featurecode.3 if
        featurecode.1 'has parent' featurecode.2 and
        featurecode.2 'has parent' featurecode.3 .

Inference rules in 'n_vector1'
featurecode.1 mach_direction featurecode.2 'has direction' nor_vector.1 if
        featurecode.1 'has parent' featurecode.2 and
        featurecode.1 'has imagin' featurecode.3 and
        featurecode.3 'has nor_vector' nor_vector.1 and
        featurecode.2 'has real' featurecode.4 and
        featurecode.4 'has nor_vector' nor_vector.1 .
```

MODULE 3 (CALCULATING MCUE's REQUIRED ON EACH PAD)

The input to this module is the results from table 'potentially1' and table 'results' i.e. module 1 and module 2. The output are the feature MCUE's attached to the PAD's through which they can potentially be machined as shown in table 'result1'. The output form called 'result1_form' will be showing the results from this module. The procedure file for this module is 'do_result1'.

MODULE 4 (CALCULATING APPROACH DIRECTIONS)

Input to the module is the results from table 'result1'. After attaching the MCUE's to each feature PAD's, component AD's (in this case set-ups) are determined by clustering the feature state elements or MCUE's from common component PAD's. Here feature state elements may appear in more than one cluster in different component PAD's. The final optimised component AD's are selected by minimising the number of clusters. This is done based on step by step selection of clusters containing the maximum number of feature state elements and then removing those feature state elements from the remaining clusters in the other component PAD's. Ideally the clustering process should be performed while observing the precedence relationships that may exist among component features. Logic for precedence relationships has not been designed in the system as there does not exist any precedence relationship among the features on the components planned for the family.

The output form called 'op_table_form' will be displaying the results from this module. The results will also be written in the table called 'op_table' for latter use. The output results from this module are shown in table 'op_table'. Table shows the featurecodes, features, parent features, PAD's (from which features are machined) and MCUE's involved. The procedure file responsible for this module is 'do_op_table'.

MODULE 5 (CALCULATING THE NUMBER OF APPROACH DIRECTIONS)

The input to this module is the results from table 'op_table'. The AD's are counted by this module and number of AD's are displayed on the screen as well as written in table 'parts'. The procedure file for this module is 'do_AD'.

MODULE 6 (CALCULATING OPERATIONS REQUIRED AT COMPONENT LEVEL)

The input to the module is the results from table 'op_table'. The output from this module is all the MCUE's required at component level (i.e. all the operations required to machine the component). The output form displaying the results from this module is 'part_process_form'. The results are written in table 'part_process' as well. The procedure file used for this module is 'do_part_process'.

MODULE 7 (CALCULATING FEATURES RELATIONSHIPS)

The input to this module is again results from the table 'op_table'. This module determines the parent/child relationship for the sequencing of the operations. The child feature will be done followed by its parent feature. The feature is the parent feature for a child feature if the child feature has an imaginary face and the same imaginary face is the real face for that parent feature. This module declares only one parent feature for a child feature, through which the child feature will be processed.

The output form designed to display the results on the screen is 'plan_table_form' for this module. The output results from this module are also written in the table 'plan_table'. Results in the table include the featurecodes, features, parent features, PAD's and the MCUE's involved. The procedure file responsible for this module is 'do_plan'. The Rule involved in this procedure file is called 'child' which is given in the following:

```
Inference rules in 'child'
featurecode.1 'has child' featurecode.2 if
        featurecode.1 'has real' featurecode.3 and
        featurecode.2 'has imagin' featurecode.3 .
```

MODULE 8 (SEQUENCING THE OPERATIONS)

The input to this module is the results from the previous module. The present module is responsible for sequencing the operations for the processing of the component. The module starts the sequencing of operations from the AD from which the maximum number of features state elements can be done. Processing will then be performed from the next AD where the next maximum features state elements to be machined are involved. Similarly, AD's are considered in descending order of the number of feature state elements involved on the respective AD's. As already mentioned, the sequencing of operations is considered at each AD separately. The procedure file for this module is 'do_plan1'.

The sequencing of operations, implemented in the software, is based upon by considering the following categories of constraints:

1. Sequencing based on operation constraints;
2. Sequencing based on geometric constraints;
3. Sequencing based on 'good manufacturing practice' constraints.

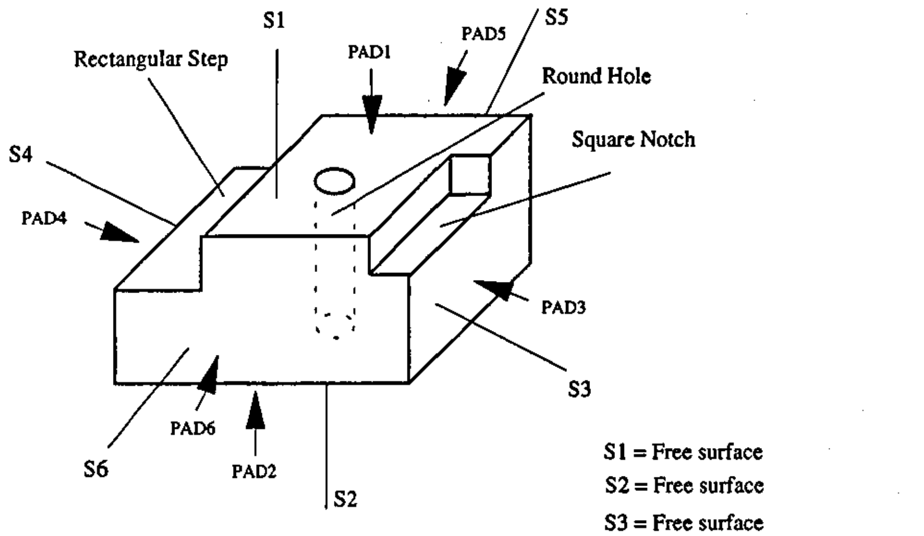
After the processing of this module, the sequencing of the operations are displayed on the screen. This screen format is shown in the example in section 7.3.2. The contents of the results are the same as from the previous module, only the operations to be carried out are displayed in order. The results are written in the table called 'plan1_table' as well as in a file called 'plan' which is outside the GENERIS application so that a hard copy of the process sheet can be taken.

The software programs developed for implementation of the HYCAPP System are listed in appendix E. The hierarchy of the designed programs is also given the appendix.

7.3 EXAMPLE PART

Part description forms a major part of the information needed for Process Planning. The way in which the part description is input to the Process Planning system has a direct effect on the degree of automation that can be achieved. Since the aim is to automate the system, the part description should be in a computer readable format [Chang 1990].

In the HYCAPP System, stock is taken as a Rectangular Boss (Block) feature. Six free surfaces and six PAD's which are normal vectors to these free surfaces are associated with the Boss in the knowledge base. Features are the geometric shapes to be cut from the stock. The concept has been shown in figure 7.3.



S1 = Free surface
S2 = Free surface
S3 = Free surface
S4 = Free surface
S5 = Free surface
S6 = Free surface
PAD = Potential Access Direction

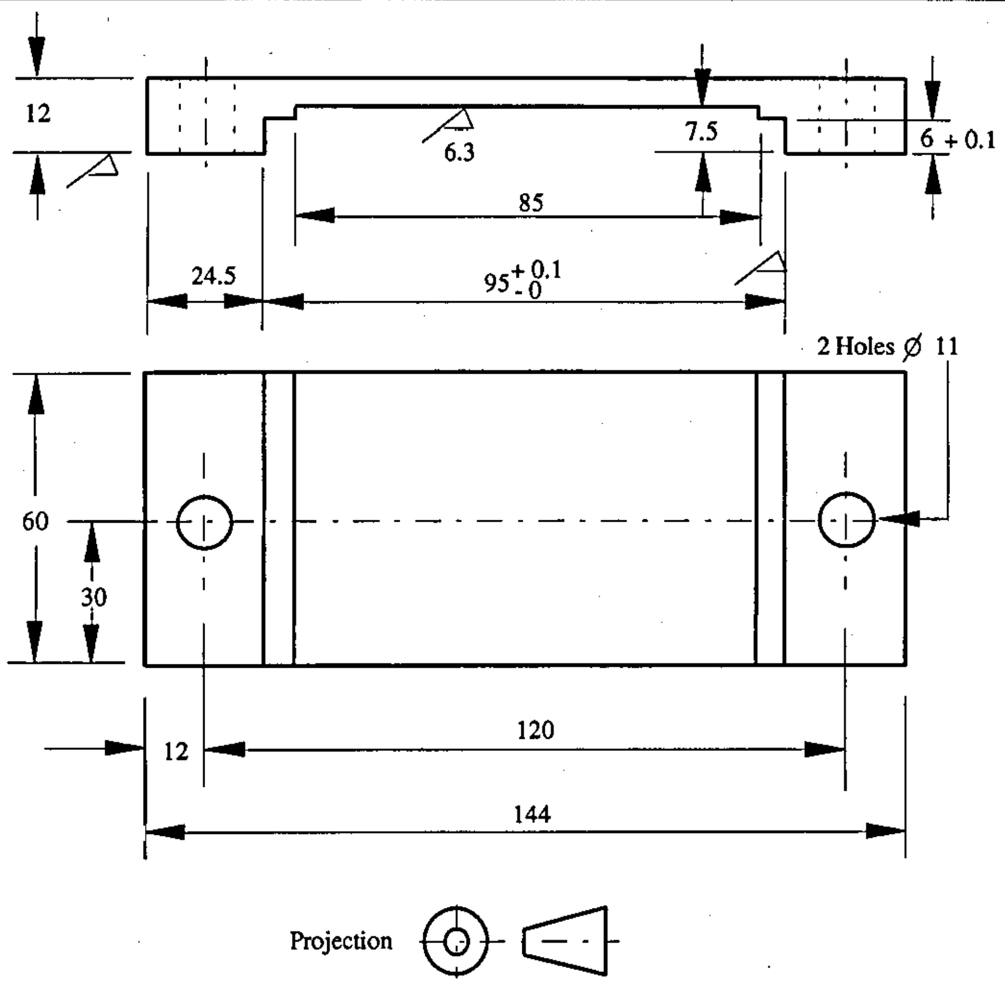
Figure 7.3 Component features, free surfaces and PAD's

Feature-base component data is represented at two levels; global or component level and feature level as already mentioned elsewhere in the thesis. Global level information includes overall dimensions of the component, tolerances on these dimensions, global co-ordinate system (GCS), material, hardness, feature list, connectivity etc. Feature level information includes feature geometry, size, position of the feature, local co-ordinate system (LCS), feature technological requirements etc.

7.3.1 THE PART DESCRIPTION

A part from the planned family has been selected for illustrating the part description in the GENERIS, results generated from different system modules during the course of Process Planning function and finally process plans generated for the part. This component has been coded as p17 and re-drawn as per its drawing as shown in figure 7.4. In figure 7.5, p17 has been shown in 3-D representation along with the features and all the faces coded. Moreover, PAD's have also been shown in the figure.

The component level information which has a one to one relationship has been represented in table 'parts'. As a part usually has more than one feature, a multi-valued table called 'fea_list' has been designed for representing the features. Notice that the free surfaces (surface features) are being treated just like other features. They will be the parent of all the features having imaginary surfaces contained within their boundaries. Feature real and imaginary faces information has been created in the table 'surfaces'. Records at feature level have been created in table 'fea_data'. Both 'surfaces' and 'fea_data' are multi-valued tables. All the features other than the free surfaces will have at least one parent. Free surfaces are used to generate parent/child relationship among the features and assigning the PAD's to the features. The PAD's information is also taken as component level information. The PAD's are defined by a feature connectivity graph as was shown in chapter 6. PAD's are used to determine the set-ups and operation sequences. The real and imaginary faces of the features are used to represent the connectivity of the component. Table 'nor_surfaces' has been designed to define the normal vectors on the faces of the features. A local co-ordinates frame has been defined for each individual feature and position of each feature with respect to the global co-ordinate system. For example, feature.2 having the imaginary surface.1 and feature.1 with the same surface.1 as its real surface will have parent/child relationship, feature.2 being the child of feature.1. Let us take feature.1 as a free surface and a normal vector to this free surface.2 is PAD2. If the normal vector to the imaginary surface of the feature.1 is parallel to the PAD2 then PAD2 can be used for its machining.



Drawing No. K4053852

Part coded as p17, Description Plate

Figure 7.4 Example part from the family

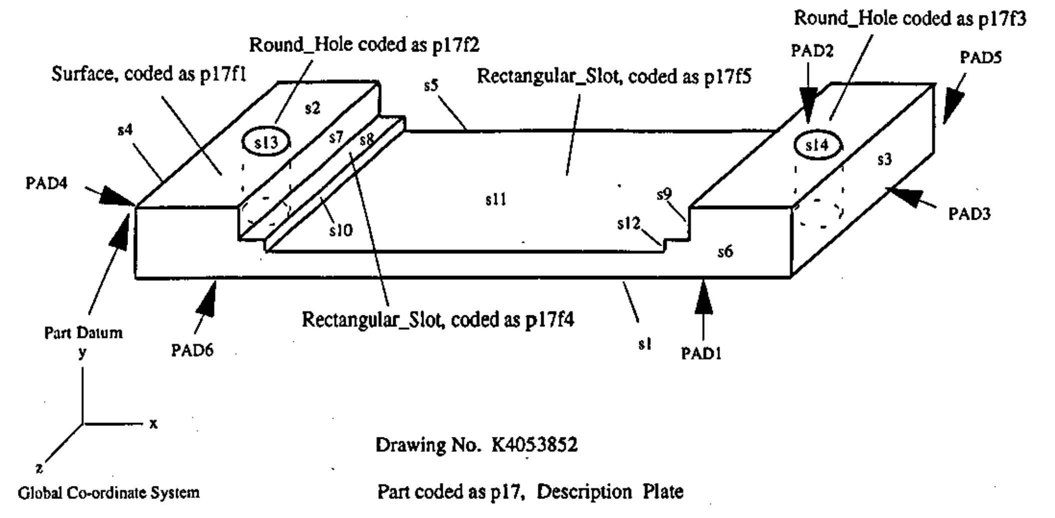

Round_Hole coded as p17f3
Round_Hole coded as p17f2
Surface, coded as p17f1
Rectangular_Slot, coded as p17f5
PAD2
PAD5
s5
s4
s2
s13
s7
s8
s14
s3
PAD4
s11
s9
PAD3
s10
s12
s6
Rectangular_Slot, coded as p17f4
Part Datum
y
PAD6
s1
PAD1
x
z
Drawing No. K4053852
Global Co-ordinate System
Part coded as p17, Description Plate

Figure 7.5 Example part in 3-D along with features and faces coded

TABLE : fea_list

PART	part has featurecode FEATURECODE	part has feature FEATURE
p17	p17f1	surface
p17	p17f2	round_hole
p17	p17f3	round_hole
p17	p17f4	rectangular_slot
p17	p17f5	rectangular_slot
p17	s1	free_surface
p17	s2	free_surface
p17	s3	free_surface
p17	s4	free_surface
p17	s5	free_surface
p17	s6	free_surface

TABLE : surfaces

FEATURECODE	featurecode has real FEATURECODE	featurecode has imagin FEATURECODE
p17f1		s2
p17f2	s13	s2
p17f2		s1
p17f3	s14	s2
p17f3		s1

p17f4	s7	s2
p17f4	s9	s5
p17f4	s8	s6
p17f5	s10	s8
p17f5	s12	s5
p17f5	s11	s6
s1	s1	
s2	s2	
s3	s3	
s4	s4	
s5	s5	
s6	s6	

TABLE : nor_surfaces

FEATURECODE	featurecode has nor_vector NOR_VECTOR
s7	PAD4
s9	PAD3
s8	PAD2
s10	PAD4
s12	PAD3
s11	PAD2
s1	PAD1
s2	PAD2
s3	PAD3
s4	PAD4
s5	PAD5
s6	PAD6

The above mentioned tables consist of either two or three slots but there are some tables in the knowledge base which include up to thirty slots. It is impractical to represent these long tables in a similar way as shown above i.e. in tabular form. These long table records have been stored in dump files outside the GENERIS application. Another point to be noted here is that the empty slots of the table are not mentioned in the 'dump files'. To make the point clear the first two records from the above table ('fea_list') are given below in the dump file format. Other long table records follow. The long tables are table 'fea_data' and 'parts'.

```
create records in 'fea_list' with 'l' and ':'
part l p17
'has featurecode' 'FEATURECODE' l p17f1
'has feature' 'FEATURE' l surface
:
part l p17
'has featurecode' 'FEATURECODE' l p17f2
'has feature' 'FEATURE' l round_hole
:
```

```
create records in 'parts' with 'l' and ':'
part l p17
'has drawing_number' 'drawing_number' l k4053852
'has description' 'description' l plate
'has material' 'material' l steel
'x_value' 'position1' l 0.000000000000000
'y_value' 'position1' l 0.000000000000000
'z_value' 'position1' l 0.000000000000000
'g_x_orientation' 'orientation' l 0.000000000000000
'g_y_orientation' 'orientation' l 0.000000000000000
'g_z_orientation' 'orientation' l 0.000000000000000
'has lengh' 'lengh' l 144.000000000000
'has width' 'width' l 60.0000000000000
'has depth' 'depth' l 12.0000000000000
'has no_of_AD' 'no_of_AD' l 1
:
```

```
create records in 'fea_data' with 'l' and ':'
featurecode l p17f1
'x_position' 'position1' l 72.0000000000000
'y_position' 'position1' l 30.0000000000000
'z_position' 'position1' l 0.000000000000000
'x_orientation' 'orientation' l 0.000000000000000
'y_orientation' 'orientation' l 0.000000000000000
'z_orientation' 'orientation' l 0.000000000000000
'fealengh' 'lengh' l 144.000000000000
'feawidth' 'width' l 60.000000000000
:
featurecode l p17f2
'x_position' 'position1' l 12.0000000000000
'y_position' 'position1' l 30.0000000000000
'z_position' 'position1' l 0.000000000000000
'x_orientation' 'orientation' l 0.000000000000000
'y_orientation' 'orientation' l 0.000000000000000
'z_orientation' 'orientation' l 0.000000000000000
'feadepth' 'depth' l 12.0000000000000
'feadiameter' 'diameter' l 11.0000000000000
'has depth_axis' 'depth_axis' l straight
'depth_symmetry' 'symmetry' l symmetric
'has ent_ext_relation' 'ent_ext_relation' l same
```

```
'has form_variation' 'form_variation' | constant
:
featurecode | p17f3
'x_position' 'position1' | 132.000000000000
'y_position' 'position1' | 30.0000000000000
'z_position' 'position1' | 0.000000000000000
'x_orientation' 'orientation' | 0.000000000000000
'y_orientation' 'orientation' | 0.000000000000000
'z_orientation' 'orientation' | 0.000000000000000
'feadepth' 'depth' | 12.0000000000000
'feadiameter' 'diameter' | 11.0000000000000
'has depth_axis' 'depth_axis' | straight
'depth_symmetry' 'symmetry' | symmetric
'has ent_ext_relation' 'ent_ext_relation' | same
'has form_variation' 'form_variation' | constant
:
featurecode | p17f4
'x_position' 'position1' | 72.0000000000000
'y_position' 'position1' | 0.000000000000000
'z_position' 'position1' | -3.00000000000000
'x_orientation' 'orientation' | 0.000000000000000
'y_orientation' 'orientation' | 90.0000000000000
'z_orientation' 'orientation' | 90.0000000000000
'fealengh' 'lengh' | 95.0000000000000
'feawidth' 'width' | 6.00000000000000
'feawidthuptol' 'widthuptol' | 0.100000000000000
'feadepth' 'depth' | 60.0000000000000
'has depth_axis' 'depth_axis' | straight
'depth_symmetry' 'symmetry' | symmetric
'has ent_ext_relation' 'ent_ext_relation' | same
'has form_variation' 'form_variation' | constant
:
featurecode | p17f5
'x_position' 'position1' | 72.0000000000000
'y_position' 'position1' | 0.000000000000000
'z_position' 'position1' | -3.75000000000000
'x_orientation' 'orientation' | 0.000000000000000
'y_orientation' 'orientation' | 90.0000000000000
'z_orientation' 'orientation' | 90.0000000000000
'fealengh' 'lengh' | 85.0000000000000
'fealengthuptol' 'lengthuptol' | 0.100000000000000
'feawidth' 'width' | 1.50000000000000
'feadepth' 'depth' | 60.0000000000000
'has depth_axis' 'depth_axis' | straight
'depth_symmetry' 'symmetry' | symmetric
'has ent_ext_relation' 'ent_ext_relation' | same
'has form_variation' 'form_variation' | constant
'has surfinish' 'surfinish' | 6.30000000000000
:
```

7.3.2 PROCESS PLANNING FOR THE PART

After the description of the design data of the example part, the results from the Process Planning modules, for the example, part follow:

Results from Module 1;

TABLE : potentially1

	featurecode	featurecode
	with_feature	with_operation
FEATURECODE	FEATURE	OPERATION
p17f1	surface	shaping
p17f1	surface	ver_milling
p17f1	surface	hor_milling
p17f4	rectangular_slot	shaping
p17f4	rectangular_slot	ver_milling
p17f4	rectangular_slot	hor_milling
p17f5	rectangular_slot	shaping
p17f5	rectangular_slot	ver_milling
p17f5	rectangular_slot	hor_milling
p17f2	round_hole	drilling
p17f3	round_hole	drilling

Results from Module 2;

TABLE : results

	featurecode	featurecode
	mach_direction	has direction
FEATURECODE	FEATURECODE	FEATURECODE
p17f1	s2	PAD2
p17f2	s1	PAD1
p17f2	s2	PAD2
p17f3	s1	PAD1
p17f3	s2	PAD2
p17f4	s2	PAD2
p17f4	s5	PAD5
p17f4	s6	PAD6
p17f5	s2	PAD2
p17f5	s5	PAD5
p17f5	s6	PAD6

Results from Module 3;

TABLE : result1

	featurecode		featurecode
	opt direction		has operation
FEATURECODE	FEATURECODE		OPERATION
p17f2	s1	PAD1	drilling
p17f2	s2	PAD2	drilling
p17f3	s1	PAD1	drilling
p17f3	s2	PAD2	drilling
p17f1	s2	PAD2	shaping
p17f4	s2	PAD2	shaping
p17f4	s2	PAD2	ver_milling
p17f4	s2	PAD2	hor_milling
p17f4	s5	PAD5	shaping
p17f4	s5	PAD5	ver_milling
p17f4	s5	PAD5	hor_milling
p17f4	s6	PAD6	shaping
p17f4	s6	PAD6	ver_milling
p17f4	s6	PAD6	hor_milling
p17f5	s2	PAD2	shaping
p17f5	s2	PAD2	ver_milling
p17f5	s2	PAD2	hor_milling
p17f5	s5	PAD5	shaping
p17f5	s5	PAD5	ver_milling
p17f5	s5	PAD5	hor_milling
p17f5	s6	PAD6	shaping
p17f5	s6	PAD6	ver_milling
p17f5	s6	PAD6	hor_milling

Results from Module 4;

TABLE : op_table

	featurecode	featurecode	featurecode
	having_feature	has appr_direction	has op_operation
FEATURECODE	FEATURE	FEATURECODE	OPERATION
p17f1	surface	s2	shaping
p17f2	round_hole	s2	drilling
p17f3	round_hole	s2	drilling
p17f4	rectangular_slot	s2	shaping
p17f5	rectangular_slot	s2	shaping

Results from Module 5;

NUMBER OF APPROACH DIRECTIONS 1

Results from Module 6;

TABLE : part_process

PART	part need operation OPERATION
p17	drilling
p17	shaping

Results from Module 7;

TABLE : plan_table

FEATURECODE	featurecode containing FEATURE	featurecode parent FEATURECODE	featurecode from direction NOR_VECTOR	featurecode with operation OPERATION
p17f1	surface	s2	PAD2	shaping
p17f2	round_hole	s2	PAD2	drilling
p17f3	round_hole	s2	PAD2	drilling
p17f4	rectangular_slot	s2	PAD2	shaping
p17f5	rectangular_slot	p17f4	PAD2	shaping

Results from Module 8;

File 'plan'

```
-----------------------------------------------------------------------------------------
PART                        DRAWING NUMBER              DESCRIPTION
|----------------------------------------------------------------------------------------
p17                            k4053852                     plate

PROCESS PLAN FOR THE COMPONENT                          p17            FOLLOW

FEATURECODE    FEATURE            PARENT     APPROACH_DIREC    OPERATION
==============================================================================
p17f1          surface            s2         PAD2              shaping
p17f4          rectangular_slot   s2         PAD2              shaping
p17f5          rectangular_slot   p17f4      PAD2              shaping
p17f2          round_hole         s2         PAD2              drilling
p17f3          round_hole         s2         PAD2              drilling
```

Interaction between different modules, part data model and manufacturing knowledge has been shown in figure 7.6.

7.4 THE COMPOSITE COMPONENT

The composite component of the planned family is given in figure 7.7 and the same component is shown in figure 7.8 after coding its features and faces. Process Planning for the composite component generated by the HYCAPP System is given in the following:

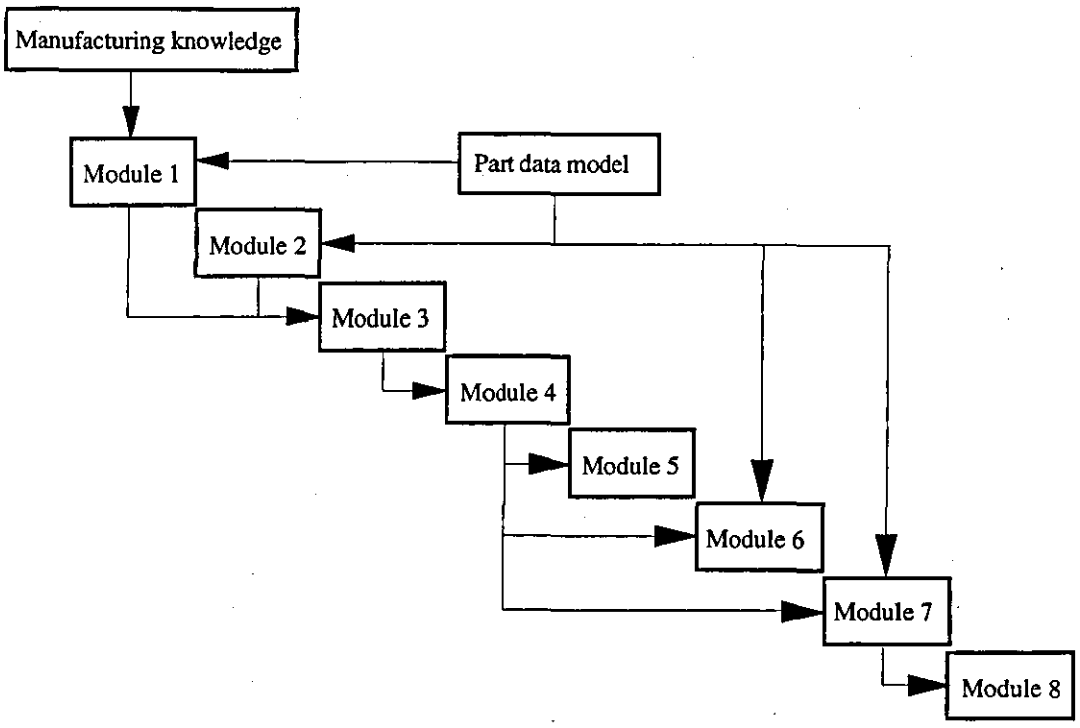


Figure 7.6 Communication between the modules, the part data model and the manufacturing knowledge

File 'plan'

```
-------------------------------------------------------------------------------------
PART                    DRAWING NUMBER          DESCRIPTION
|-------------------------------------------------------------------------------------
 p_x                         xxx                        composite_comp

PROCESS PLAN FOR THE COMPONENT                 p_x           FOLLOW

FEATURECODE   FEATURE           PARENT     APPROACH_DIREC   OPERATION
=========================================================================
p_xf2         triangular_step   s1         PAD1             shaping
p_xf3         rectangular_slot  s1         PAD1             shaping
p_xf4         rectangular_slot  p_xf3      PAD1             shaping
p_xf10        round_pocket      p_xf3      PAD1             drilling
p_xf11        round_pocket      p_xf3      PAD1             drilling
p_xf12        round_hole        p_xf4      PAD1             drilling
p_xf13        round_hole        p_xf4      PAD1             drilling
p_xf14        round_hole        p_xf4      PAD1             drilling
p_xf15        round_hole        p_xf4      PAD1             drilling
p_xf16        round_hole        p_xf4      PAD1             drilling
p_xf10        counterbore       p_xf3      PAD1             counterboring
```

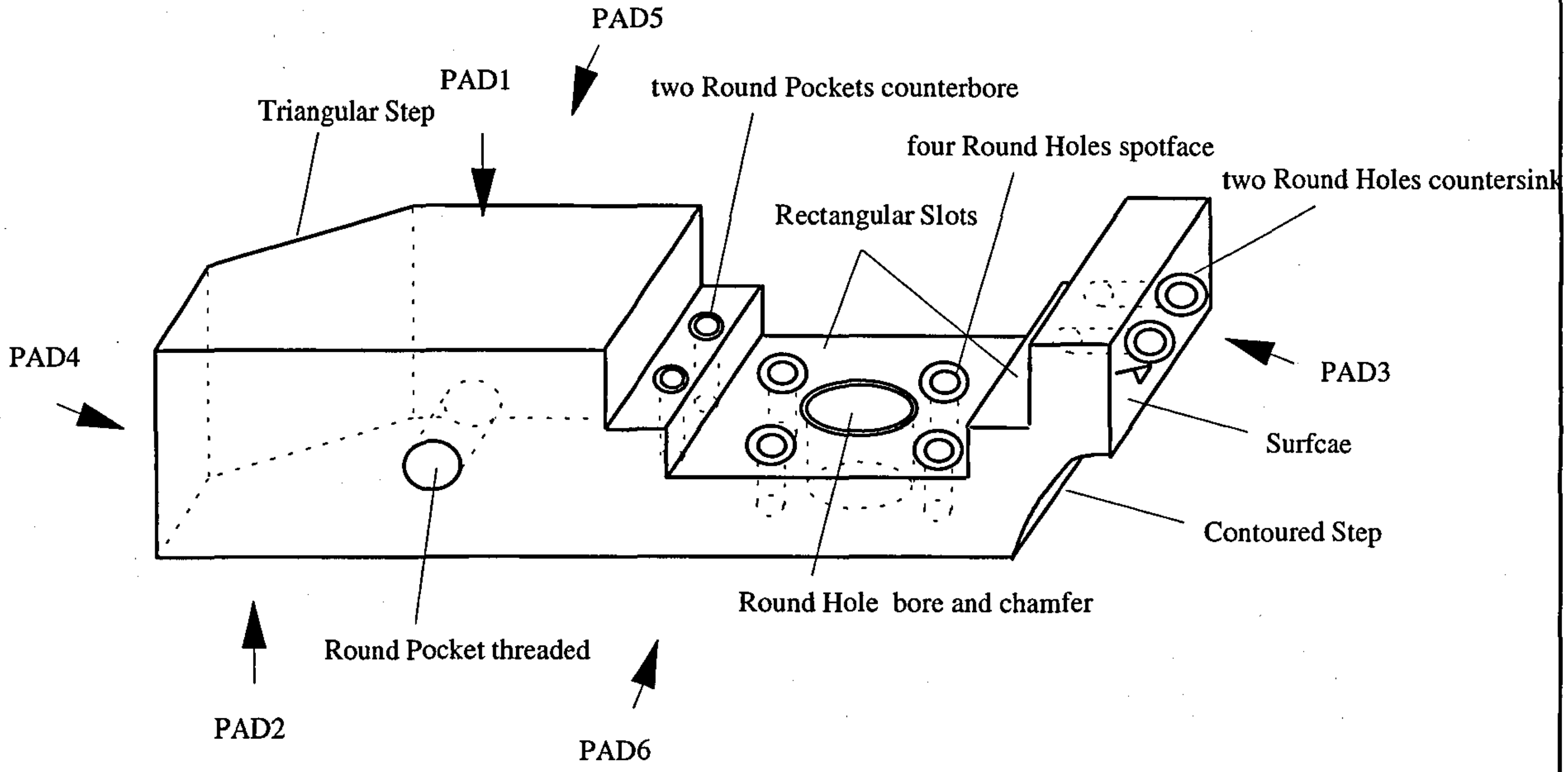

PAD5
PAD1
Triangular Step
two Round Pockets counterbore
four Round Holes spotface
two Round Holes countersink
Rectangular Slots
PAD4
PAD3
Surfcae
Contoured Step
Round Hole bore and chamfer
Round Pocket threaded
PAD2
PAD6

Figure 7.7 Composite Component for the Part Family

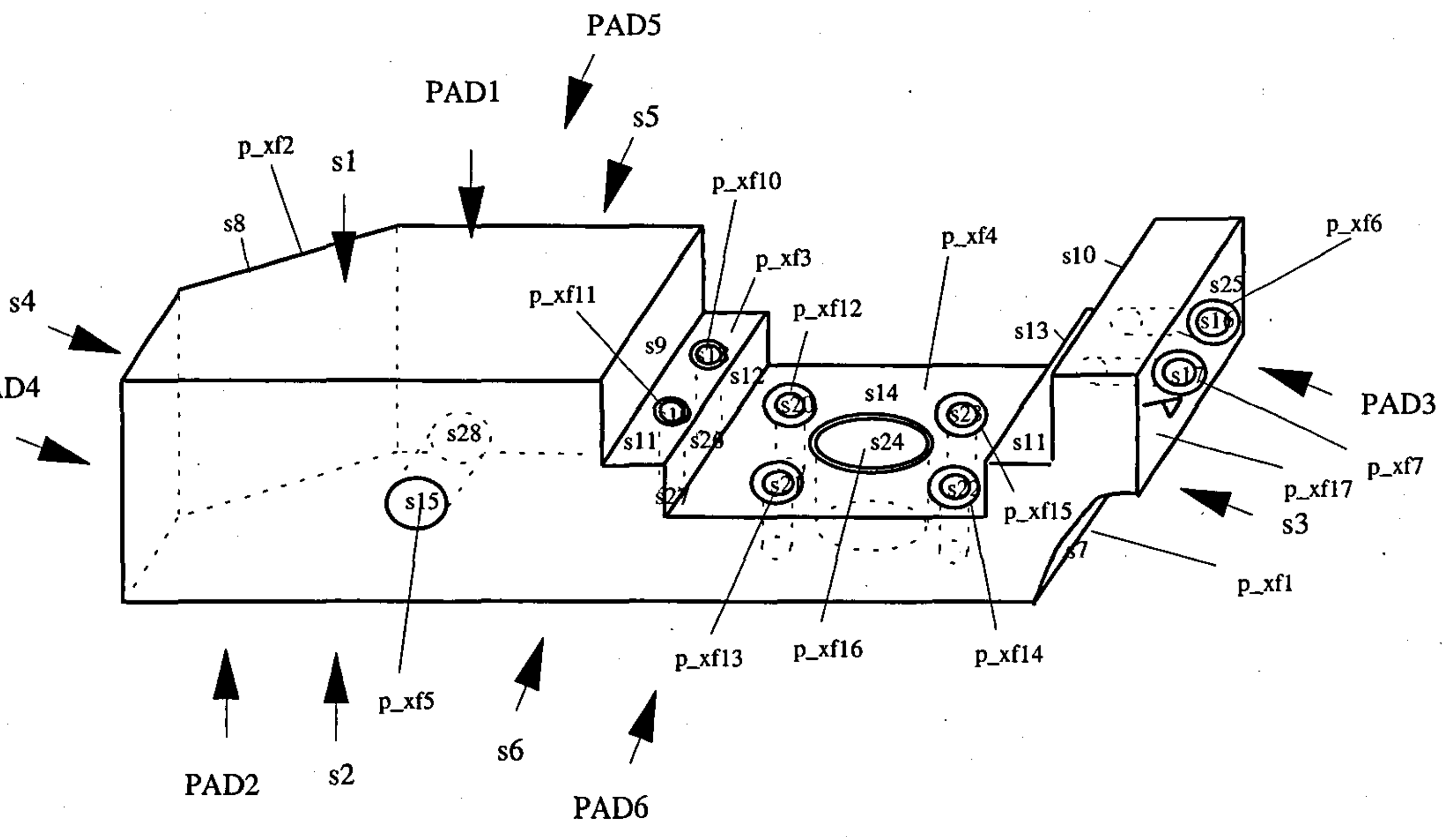


Figure 7.8 Composite Component (p_x) for the Part Family

p_xf11	counterbore	p_xf3	PAD1	counterboring
p_xf16	round_hole	p_xf4	PAD1	boring
p_xf12	spotface	p_xf4	PAD1	spotfacing
p_xf13	spotface	p_xf4	PAD1	spotfacing
p_xf14	spotface	p_xf4	PAD1	spotfacing
p_xf15	spotface	p_xf4	PAD1	spotfacing
p_xf16	screw	p_xf4	PAD1	screw_cutting
p_xf16	chamfer	p_xf4	PAD1	chamfering
p_xf17	surface	s3	PAD3	shaping
p_xf1	contoured_step	s3	PAD3	shaping
p_xf6	round_hole	s3	PAD3	drilling
p_xf7	round_hole	s3	PAD3	drilling
p_xf6	countersink	s3	PAD3	counterboring
p_xf7	countersink	s3	PAD3	counterboring
p_xf5	round_pocket	s6	PAD6	drilling
p_xf5	thread	s6	PAD6	taping

7.5 THE BOUNDARY OF THE SYSTEM

The HYCAPP System has been written for a family of 12 components resulting from the CAFBG System from a sample of 30 components obtained from GEC Alsthom, as has already been mentioned. The system can plan any of the component in the family and can plan any other component that fits the criteria of the HYCAPP System as described below. The design data for the components in the family are stored in dump files outside the system. One of the procedure files called do_load_part can load the required component data for which a process plan is to be generated.

Primary form features other than those mentioned in the feature list can be planned, if the solution of these features is present in the system. The system has the ability to check for and list those features which are outside the system boundary. The user will be asked to input the TSF required for those features from those available in the database. The user can input multiple solutions for each of the new features. The solution should be among the already defined list of processes contained in the database.

The system capabilities/boundaries are listed below:

1) A part can be planned if the part is made from a prismatic block of metal and all the free surfaces of the part are parallel to the part co-ordinate frame. Free surfaces of the part are taken to be the parent features of all the features to be machined on the part.
2) The EAD's of the features existing on the body of the part are orthogonal with respect to the part's part co-ordinate frame.
3) The primary form features, already defined in the system are listed below. Primary form features other than these will have to be defined in the system in terms of their TSF's.

Primary Form Features	Secondary Form Features
round_hole, axial_round_hole round_pocket, axial_round_pocket	thread spotface counterbore countersink chamfer
surface, contoured_step triangular_step, rectangular_slot	nil

4) The system would not be able to plan any secondary form feature other than those mentioned above.
5) The list of the MCUE's which is defined in the system is given below and the system can provide feature solutions only in the given domain.

drilling	milling
reaming	taping
hor_milling	ver_milling
spotfacing	hor_drilling
countersinking	counterboring
hor_taping	shaping
hor_reaming	grinding
honing	lapping
sur_grinding	boring
screw_cutting	chamfering

6) The system is restricted for those features whose depth boundary variation is as follows:
   a) Depth axis of the feature is straight,
   b) Depth axis of the feature is symmetric,
   c) Exit boundary of the feature is same as entry boundary,
   d) Form variation of the feature along its depth axis is constant.
7) The system is capable of responding to any accuracy or surface finish required on those features which can be planned and accordingly operations such as reaming, grinding, honing, lapping will be assigned to the feature TSF's.
8) Changes in the processes based on the tolerance values have not been defined in the system as this problem also depends upon other factors: machine accuracy is one factor, it also depends upon the cutting parameters selected e.g. cutting speed and depth of cut.
9) The system is capable of assigning any extra process required to the TSF because of the feature dimensions. For example, in the case of Round Hole feature, an extra boring process will also be attached if the diameter of the feature is 50 mm or above. In the above case if the secondary feature on the Round Hole is thread, screw_cutting operation instead of tapping will be attached.

CHAPTER 8

CELL DESIGN

8.1 INTRODUCTION

This chapter discusses how the proposed classifying attributes for the part grouping are not only used to make the Process Planning function efficient but can also help in designing manufacturing cells. Some aspects of the problems associated with Cellular Manufacturing and cell designing are discussed.

The composite component of the part family based on the proposed CAFBG System presents its processing requirements in terms of MCU's which is a direct link between the processing requirements of the family and the manufacturing system capabilities.

Section 8.2 highlights Cellular Manufacturing and potential benefits associated with it. Discussion on various stages of cell design has been taken in section 8.3. The concluding section describes software implementation.

8.2 CELLULAR MANUFACTURING

The recent trend towards the globalization of markets has resulted in an increasingly competitive environment. In response to this trend, many manufacturers have sought ways to quickly improve quality and efficiency. One approach often investigated is Cellular Manufacturing (CM) [Shafer *et al.* 1992].

The concept of Cellular Manufacturing can be taken as "a factory within a factory". CM strategy is used for the batch type production. Groover [1987] states that at least 75% of products are made in batches of less than 50 units. In the traditional functional layout manufacturing system, machine tools of different types are placed together in their respective sections. A part moves from section to section for the various machining operations to be performed on it. The disadvantage of this functional layout is that when the level of work-in-progress is high, the production planning and control function can become more difficult to manage. Moreover the lead time of parts through the shop can be unacceptably long, resulting in failure to meet delivery dates.

CM can be viewed as a middle-ground alternative to the traditional job-shop and transfer line approaches. Generally, the CM production approach is less complex as compared to job-shops but is less flexible in terms of production planning and scheduling than job-shops. On the other hand, CM has more flexibility than dedicated transfer lines, yet requires additional organisation and management compared with dedicated transfer lines that manufacture single product types [Warren and Moodie 1993].

Batch type production is mainly production-to-order. In multi-product, small-lot-sized production, the material flow for producing each of the products is dissimilar and complicated, unlike mass production. Characteristics of batch type production are [Ham *et al.* 1985]:

1) Variety of production items
2) Variety of production processes
3) Complexity of productive capacity
4) Uncertainty of outside conditions
5) Difficulty of Process Planning
6) Dynamic situation of implementation and control of production

GT has strong ties to the concept of CM. Some people take GT and CM as the same. The GT philosophy is actually the analysis that identifies and exploits the underlying similarities in product design and production processes, as said before. CM is an application of GT which groups the part families and associated machines into production cells to take advantage of machining similarities.

Cellular Manufacturing is sometimes referred to as manufacturing cell, machine cell, work cell or GT cell. The pursuit of cellular production is also termed 'product-oriented'. The concept of CM is to group processes, people, and machine resources to manufacture a specific group of parts. According to Arn [1975] the basic idea of GT cell is to split the manufacturing area into machine groups in which all the machining operations required for the manufacture of a certain parts spectrum can be accomplished. This basic form of GT layout allows a flexible operation sequence and constitutes a second or medium degree of rationalisation.

CM in the shop floor environment simplifies part flows as similar components are being handled within the cell. As all the operations are being carried out within the cell boundaries (since the transport and handling associated with a process layout is eliminated) throughput times can be greatly reduced. Because of the reduction in throughput times, work-in-progress is cut sharply. It reduces material handling costs by reducing travel, facilitating increased automation and better space utilisation. Cells also simplify scheduling by reducing the variety of parts that utilise a given group of machines. Since set-up times are usually very short between different parts in a family, CM can also result in dramatic reductions in set-up times. As set-up times approaches zero, economic batch sizes of parts can be reduced to a number approaching 1. This provides a manufacturing company with more flexibility in coping with changing schedules. Moreover, reduced through-put times will lead to shorter manufacturing lead times. Shortening parts manufacturing lead times can reduce the response time to customer orders and thus result in smaller finished goods inventories as well. A CM strategy generally gives the operators within the Cell greater responsibilities and authority and as a consequence high quality tends to be achieved. Moreover, since greater control can be exercised over operations within the Cell due dates can be readily achieved and overdue orders are reduced.

Group tooling proves another possible source of cost reduction. Once cells and associated families have been identified, it can then be possible to design group tools, and group fixtures, which can be used with most of the parts in a family. This can be another factor in the cost reduction. Managers can simplify production planning and control by considering the cell as one planning unit for which capacity planning can be performed and to which jobs can be released. Cells provide the opportunity for team work and the focusing of the production process from raw material to finished part within the cells. These advantages can add to operators job satisfaction, which can lead to better accountability, higher productivity and better quality. All these factors can provide benefits in the competitive manufacturing environment.

8.3 DESIGNING THE CELL

The cell formation problem may be defined as one of subdividing the manufacturing system into a set of subsystems or cells of machines each capable of independently producing a family of parts [Dahel and Smith 1993].

Although a vast amount of research has been carried out in the area of design of CM systems, as yet a few hard and fast rules have been established. The paramount priority in designing the cell should be to make it as flexible as possible, so that it can be expanded to include other parts or modified to accommodate additional members of the family.

In Nolen's [1989] view, a well designed cell typically exhibits:

- Unidirectional flow
- Standardised process steps that result in little or no set-up between family members
- Reduced set-ups between different families
- Flexible work force
- Statistically capable processes

According to the Gallagher and Knight [1986] in the design of a manufacturing cell, it is important:

1) To have as much work as possible completed within a machine cell without the work having to leave that cell. This may result in low utilisation figures for

some secondary machines, but provided these are inexpensive this can be tolerated and compensated for by a high degree of labour utilisation; and

2) To try and achieve a flow of work in only one direction, i.e. as the work leaves each machine the components pass along definite flow lines.

While designing the cells one needs to consider the following objectives [Francis and White 1974].

1) Each cell should be designed to handle one homogeneous family of parts.
2) Machine utilisation should be maximised.
3) Cells having a lot of moves amongst themselves should be near one another.
4) Within a cell, the machines should be laid out such that there is a unidirectional flow of parts.
5) The size of a cell should not be too large.

Wemmerlöv and Hyer [1987] divide the design phase of the Cellular Manufacturing system into five stages. The five stages are:

1) Selection of part populations and grouping of parts into families.
2) Selection of machine and process populations and grouping of these into cells.
3) Selection of tools, fixtures, and pallets.
4) Selection of material handling equipment.
5) Choice of equipment layout.

A major problem throughout the cell design process is the necessity of trading-off objectives related to structural parameters and performance variables against each other. For example, low work-in-process might be achievable only at the expense of lower output rate. Flexibility can be attained at the expense of higher equipment investment or increased control problems. Higher machine utilisation can be reached if several cells route their parts to the same machine. The drawbacks are increased queuing and control problems. Job lateness can be shortened if machine operators are mobile and can be relocated (inside and between cells) to where a capacity overload exists. This mobility, however, might adversely affect the quality and efficiency of the performed operations. The list of possible trade-offs can be quite long. It is the nature of the design process to be open-ended, and to have few initial restrictions on the solutions. This is what makes cell formation a complex problem [Wemmerlöv and Hyer 1986].

A relevant design problem is to analyse the effect of assigning more parts to a cell—that is increase the product mix. Even if the load on the cell in terms of machining hours is kept the same, we intuitively expect that an increase in the variety of parts would adversely effect flow times through the cell. This happens since more variety leads to more time lost in changeovers from one part type to the other. To minimise valuable production time lost in changeovers, batch sizes may have to be increased. Assuming variability in arrivals, if batch sizes are large, queue times will also increase [Kekre 1987].

The basic problem in the design of a cell is the identification of part families and machine groups. As already discussed in chapter 2 three approaches in the cell design have been adapted by the researchers:

1) Identify machine groups and then assign parts to machines,
2) Identify part families and then assign machines to part families,
3) Identify part families and machine groups simultaneously.

The second approach has been considered advantageous over the other two as already discussed elsewhere in the thesis.

Cell design activity can be divided into three stages:

1) Identify part families,
2) Establish manufacturing methods, and
3) Cell performance adjustment.

8.3.1 IDENTIFY PART FAMILIES

A part family is a group of parts that have some specific sameness and similarities in design features or production processes. In the case of this research a part family is a group of parts that have similarities in production methods.

As already discussed, the grouping results based on the CAFBG System are favourable from the manufacturing point of view. Manufacturing requirements of the parts in a group represent the realistic processing requirements in terms of MCU's (manufacturing capability units) in the shop or manufacturing organisation.

The centre of group or composite component of the family based on the CAFBG System represents the manufacturing requirements in terms of MCU's, the shape of the components, and the presence of features pattern. These characteristics or attributes of the composite component are quite enough for showing the processing needs of the family of the components or representing the processing requirements. MCU's represent the actual machining requirements needed for the components in a part family. The basic shape of the component dictates the machining methods. Different components with different shapes require different machining and tooling methods. Components are characterised on the basis of geometrical shapes and dimensional ratios. These ratios help in characterising the parts for selecting the machining resources. A features pattern can have implications for tooling and sometimes machining resource allocations. Patterns dictate for consideration in special toolings. These things have already been discussed in detail elsewhere in the thesis.

A composite component represents the centre of grouping or in other words, it is the representative of a set of components in a family. It embraces all the features or characteristics of the individual components in the group or family. The composite component concept is used either to represent the processing requirements of a cluster of components or the required capabilities of the manufacturing cell machining this group. By building a process model that contains the solution for every feature, components in the entire family can be process planned.

It is worth noting that at this stage, one does not know whether or not CM is warranted.

8.3.2 ESTABLISH MANUFACTURING METHODS

Under this category a best-practice method of manufacture of parts families is established which employs the manufacturing system capabilities present in today's production facility.

Cell formation means that a set of parts is identified as suitable for manufacture on a specified group of machines. To do this there must exist, or be determined, a basic relationship between a part and a set of machines [Wemmerlöv and Hyer 1986]. The relationship or link is established between the manufacturing requirements and manufacturing system in the CAFBG System which is in terms of MCU's. Part families are identified based on this link, thereby representing the true processing needs of the components.

After identifying part families, the problem is to sort out the machining resources required to machine each individual family of parts in an independent manufacturing cell.

Each cell can be represented by a vector consisting of MCU's i.e. $C_i = \{c_{i1}, c_{i2}, c_{i3}, \ldots\ldots, c_{ip}\}$ and processing requirements of n cells can be given in an n×p matrix as below:

$$\begin{array}{cc} & \text{p MCUs} \\ \text{n cells} & \begin{bmatrix} c_{11} & \cdot\ \cdot & c_{1f} & \cdot\ \cdot & c_{1p} \\ \cdot & & \cdot & & \cdot \\ \cdot & & \cdot & & \cdot \\ c_{i1} & \cdot\ \cdot & c_{if} & \cdot\ \cdot & c_{ip} \\ \cdot & & \cdot & & \cdot \\ \cdot & & \cdot & & \cdot \\ c_{n1} & \cdot\ \cdot & c_{nf} & \cdot\ \cdot & c_{np} \end{bmatrix} \end{array}$$

where c_{ip} are coefficients denoting presence or absence of MCU's coded as 1 and 0 for each cell respectively. It will represent the cells membership based on the requirements of MCU's as defined in the cell vector.

As already mentioned MCU's are the link between the processing requirements of a family of parts and the machining capabilities needed to process the family. Also there are machining capabilities in terms of MCU's and their work envelop.

Capabilities of machines can be shown in the co-ordinate system in terms of their dimensions and MCU's available on them. Machine envelopes can be described in a two dimensional co-ordinate system if those require only two dimensions to describe their work envelop i.e. in case of lathes and three dimensional co-ordinate system if three dimensions are needed for the description of their work envelop. A separate diagram is needed for each machine.

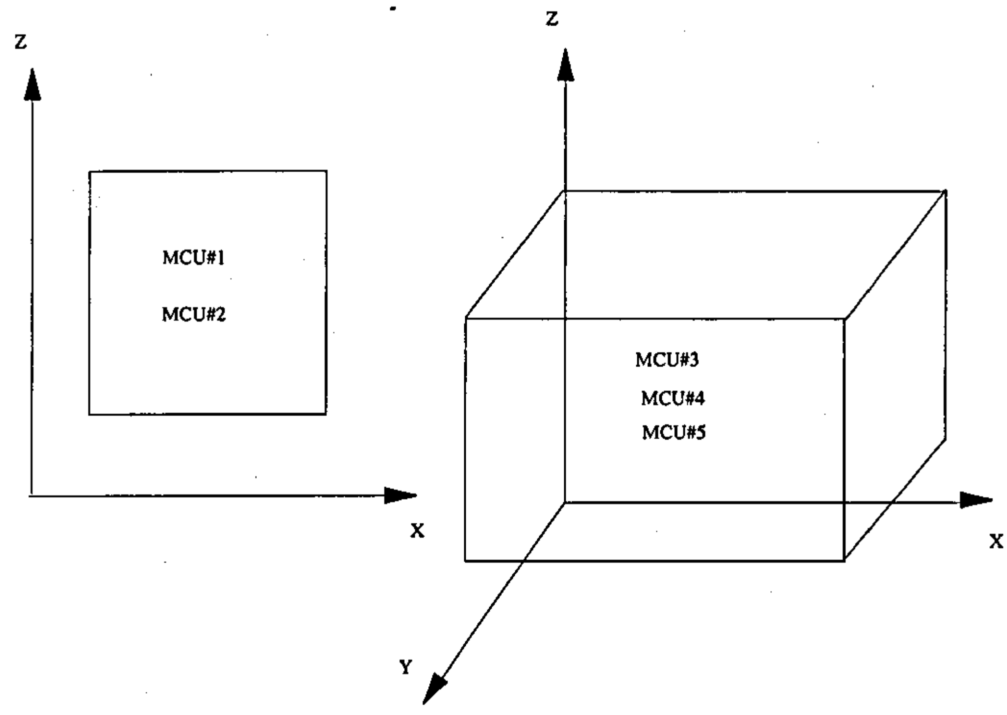


Figure 8.1 Machine Capability Model in terms of MCU's and work envelop

The processing requirements of components in each part family in terms of MCU's and their overall dimensions can be compared with the MCU's and work envelop available on the machine resources. After this analysis, the candidate machines for each manufacturing cell are established as shown below.

$$\begin{array}{cc} \begin{bmatrix} M1 \\ M3 \\ M4 \\ M6 \\ M7 \end{bmatrix} & \begin{bmatrix} M1 \\ M2 \\ M5 \\ M6 \\ M8 \end{bmatrix} \\ \text{Cell\_i} & \text{Cell\_j} \end{array}$$

Candidate machines for two cells.

At this stage, one machine might be a candidate for more than one manufacturing cell. In that case, it is needed to check in which cell this particular machine can be more utilised. The procedure is given as below:

For a particular machine M_k (machine type k), utility can be evaluated as given below:

i	=	1, n	parts in cell
j	=	1, p	MCU's of the machine type k
x_{ij}	=	1	if part i is using the MCU j

$$\sum_{i=1}^{n} \sum_{j=1}^{p} X_{ij}$$

This way a judgement can be made in allocating the machines to the cells where they have more utility. Their utility is calculated based on the total sum of parts visiting the machine for using their MCU's.

8.3.3 CELL PERFORMANCE ADJUSTMENT

At this stage the work load on the machine resources identified as necessary to accomplish the production requirements of the part families are measured.

The goal of the cell formation procedure is to identify independent machine clusters which will fulfil the needs of whole part family. However, it is rare to find realistic Cellular Manufacturing (CM) examples where perfectly independent cells are identified [Shafer *et al.* 1992].

The need to transfer the production requirements of a part family from one Cell to another is also considered if CM is not warranted for that part family in the first Cell. If one part family is initially small, and if other similar part families are not found, the advantages of CM may be offset by the excessive penalty in terms of poor utilisation of resources (people, machines etc.). In that case functional or job shop is preferred.

Lot sizes of the parts also affect the decision process in the adjustment of cell design. Findings of the impact of increasing group size on cell performance are [Kekre 1987]:

1) Deterioration of response (as measured by the queuing delays) takes place with increase in group size. This is in spite of keeping the machining load constant —i.e., part variety increase, but the production time (processing requirements) in machining hours stay the same, and the same amount of machining time is allocated equally to a larger number of parts.
2) The production lot sizes that minimise queue time increase with increase in group size.
3) The marginal rate of deterioration i.e., adverse effect on queue time and lot size, decreases with increase in group size.

In grouping part families, it is important to consider the production data such as lot size, frequency, time, annual production plan, et cetera in scheduling for optimum sequencing and machine loading [Ham *et al.* 1985].

One problem is that insufficient processing capacity within the flow creates a machine bottleneck which results in physical interference of equipment. Such problems are usually not discovered until the system is in place. With the array of simulation tools available, this does not have to happen. Simulation programs allow the system designer to construct a model that mimics the real process. The simulation results are recorded and made available to the designer, who can interpret them and reach decisions as to the capability of the system to meet the expected flow demands [Snead 1989].

The total Cellular Manufacturing design problem including product-to-cell allocations, equipment selection, layout design, control system design and staffing schedules is extremely complex, and state-of-the-art solution techniques require substantial human input. Human judgement is required to resolve many of the issues and to evaluate all phases of the design procedure for suitability. Human judgement is required to resolve many of the issues and to evaluate all phases of the design procedure for suitability [Warren and Moodie 1993].

The cell design involves usually many desirable but conflicting goals. In the design of a cell, some of the following objectives are need to be achieved:

1) Maximise the within-cell utilisation of machines [Ballakur and Steudel 1987].
2) Minimise duplication of machines in different cells [Vannelli and Kumar 1986].
3) Minimise number of exceptional parts and inter-cellular strips [King 1980, King and Nakaornchai 1982, Chan and Milner 1982, Tabucanon and Ojha 1987, Seifoddini 1989a]

Cell formation solutions often contain problems like less utility of machines and Exceptional Elements (EE). EE's (parts) create interactions between two manufacturing cells. They can be considered to be the result of a bottleneck machine i.e. machines required by two or more cells. Independent cells eliminate the need to co-ordinate work or schedule between cells which thus leads to greatly simplified shop floor control. Interaction between cells results in increased scheduling and may also result in lower product quality. The problem of EE's can be resolved by adapting one of the solutions given below:

(1) Duplicating the bottleneck machines
(2) Subcontracting the EE's
(3) Manufacture the EE's by using the machines in other cells.

Cost is involved in the above methods of dealing with the EE's.

8.3.3.1 EXCEPTIONAL ELEMENTS

As said earlier EE's do not have processing facilities in their own cell, but sometimes the situation may arise that any existing machine in the cell is so much loaded that it cannot cope with all the components which require machining on that particular machine. So, a procedure is required to find the EE's created in this way. To calculate the load on the machines assigned to the cells, we suppose that after finding the machines, the processing times for the operations on each machine are available.

The number of EE's emerged because of any bottleneck machine can be calculated as follows:

NOTATIONS:

i	= 1, n parts
j	= 1, m operations
k	= 1, k machine types
C_k	= Capacity of machine type k during the planning period
M_{ik}	= Numbers of machines of type k available to produce part i
D_i	= Demand of part i during the planning period
T_{ij}	= Time needed to complete operation j on part i
X_{ik}	= 1 if part i being processed on machine type k otherwise 0
L_{jk}	= 1 if process j being used on machine type k otherwise 0

Model can be given as below:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} T_{ij}L_{jk}X_{ik}D_i \leq C_kM_{ik}$$

If this relation does not hold, this means that machine is overloaded. So, another variable B , in place of D can be introduced to find out EE's, as given below:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} T_{ij}L_{jk}X_{ik}B_{i} \leq C_{k}M_{ik}$$

So, number of EE's can be calculated:

$$X_i = D_i - B_i$$

Where X_i is the number of EE's.

After the identification of EE's, the mathematical cost model presented by Shafer *et al.* [1992] for dealing with the EE's, which emerge in the manufacturing cells during the performance adjustment, can be utilised for the problem. This is an optimising model which provides solution to the problem by minimising the total cost involved in: 1) intercells transfers; 2) machine duplication; and 3) subcontracting.

Cell performance adjustment was not implemented in the software developed for cell design.

8.4 SOFTWARE IMPLEMENTATION

The investigation carried out in the cell design work has led to the implementation of experimental software work using 'GENERIS' Expert System shell on the Sun Sparc workstation. The task has broken down into several logical sub-tasks to ease the implementation, as well as to provide the flexibility of enhancing and modifying tasks as discussed earlier. Keeping this in view, the software development job has been divided into modules that work independently.

The main menu facilitates the selection of different system modules. The menu will be comprising of the following modules:

```
1. TO FIND OUT THE PROCESSING REQUIREMENTS OF THE PARTS IN TERMS OF MCU'S
2. TO FIND OUT THE CANDIDATE MACHINES FOR THE PARTS
3. TO FIND OUT THE MACHINE WHICH IS CANDIDATE FOR MORE THAN ONE CELL
4. TO ASSIGN THE BOTTLE-NECK MACHINES TO THE CELLS
5. TO FIND OUT THE ASSIGNED MACHINES TO THE CELLS
6. TO FIND OUT FOR WHICH PART'S MCU, THERE IS NOT ANY MACHINE AVAILABLE
7. TO FIND OUT WHETHER ANY CELL CAN BE MERGED INTO ANY OTHER CELL
8. TYPE ANY COMMAND
9. END
```

The above mentioned modules have been designed to accomplish the different objectives of cell design. In most of the cases, the modules require the input from the previous modules. User defined 'Output Forms' have been designed to show the output results from the modules. The system consists of a number of modules. The details of the modules are given below:

MODULE 1 (TO FIND OUT THE PROCESSING REQUIREMENTS OF THE PARTS IN TERMS OF MCU'S)

The processing requirements of the parts are determined based on their geometric and technological constraints. With the feature connectivity information on the parts TSF's are transformed into MCU's. The output from the module is processing requirements of the parts in terms of MCU's. The following rule is used to inference the mentioned output.

```
Rules in Inference Ruleset part_mcus

 1 part.1 need m_c_unit.1 if
        part.1 require process.1 and
        m_c_unit.1 contains process.1 .
```

The output from the module is as below:

```
          part                              need
                                          m_c_unit
--------------------------------------------------------------------------------------------
```

MODULE 2 (TO FIND OUT THE CANDIDATE MACHINES FOR THE PARTS)

The input to this module is the results from the module 1, parts overall dimensions and the envelope of the machines under consideration. The output from the module is the candidate machines that can for the processing of the parts. The decision will be made based on two things: 1) machines has the processing capability of the MCU's required for the parts, and 2) the overall dimensions of the parts are less than or equal to the envelope of the machines. The output pattern and the rulesets involved are given below respectively:

```
          part                                        candidate
                                                       machine
-----------------------------------------------------------------------------------------------
```

```
Rules in Inference Ruleset part_machine

 1 part.1 candidate machine.2 selected m_c_unit.1 if
        part.1 need m_c_unit.1 and
        machine.2 'has m_c_unit' m_c_unit.1 and
        part.1 'has lengh' lengh.1 'has diameter' diameter.1 and
        machine.2 'has m_c_unit' m_c_unit.1 and
        machine.2 'has max_lengh' lengh.2 'has max_diameter' diameter.2 and
        lengh.2 >= lengh.1 and
        diameter.2 >= diameter.1 .
 2 part.1 candidate machine.2 selected m_c_unit.1 if
        part.1 need m_c_unit.1 and
        machine.2 'has m_c_unit' m_c_unit.1 and
        part.1 'has lengh' lengh.1 'has width' width.1 'has depth' depth.1 and
        machine.2 'has m_c_unit' m_c_unit.1 and
        machine.2 'has max_lengh' lengh.2 'has max_width' width.2 'has max_depth' depth.2 and
        lengh.2 >= lengh.1 and
        width.2 >= width.1 and
        depth.2 >= depth.1 .
```

MODULE 3 (TO FIND OUT THE MACHINES WHICH ARE CANDIDATE FOR MORE THAN ONE CELL)

The input to this module is the output from the previous module in which the candidate machines for all the parts in different cells have been determined. The output from the module will be the bottleneck machines as with as the cells where the machines are required. The output form called 'sarmd_form' will be showing the results from this module. The procedure file for this module is 'do_mach_cell'. The inference rules used for inferencing the results from the module are as below:

```
Rules in Inference Ruleset cell_mach

  1 cell.1 'has machine' machine.1 if
          cell.1 'has part' part.1 and
          part.1 candidate machine.1 .

Rules in Inference Ruleset mach_cell

  1 machine.3 'has candidate' cell.1 'has candidate' cell.2 if
          cell.1 'has machine' machine.3 and
          cell.2 'has machine' machine.3 .
```

The format of the output is given below:

```
            machine                                   candidate
                                                        cell
--------------------------------------------------------------------------------
```

MODULE 4 (TO ASSIGN THE BOTTLE-NECK MACHINES TO THE CELLS)

The rules utilised in this module are part_mcus and part_machine. These rules are already given in the description of the previous modules. In the previous module, the bottle-neck machines were found. In this module the bottle-neck machines are assigned to those cells where they have more utility. The procedure file for this module is 'do_count_m_c_units'. The output from the module will look like as given below:

```
              cell                                    assigned
                                                      machine
--------------------------------------------------------------------------------
```

MODULE 5 (TO FIND OUT THE ASSIGNED MACHINES TO THE CELLS)

After resolving the assignment of the bottle-neck machines to the cells, this module is responsible for displaying all the machine resources assigned to the different cells. The name of the procedure file for this module is 'do_assigned_mach_cell'. The output form designed to display the results is called 'cell_machines_form'. The output from the module will be as given below:

```
            CELL                                    HAS MACHINE
-------------------------------------------------------------------------------
```

MODULE 6 (TO FIND OUT FOR WHICH PART'S MCU'S, THERE IS NOT ANY MACHINE AVAILABLE)

The rules utilised in this module are again part_mcus and part_machine. These rules are already given in the description of the previous modules. The module outputs the parts MCU'S, for which machines are not available. The message of "FOR ALL THE PART'S MCU'S, MACHINES ARE AVAILABLE" will be displayed, if machines are available for all the required MCU'S. The name of the procedure file for this module is 'do_not_used_m_c_unit'. The output format from the module will be as below, if MCU'S are found for which machines are not available.

```
            part                                    not_available
                                                      m_c_unit
-------------------------------------------------------------------------------
```

MODULE 7 (TO FIND OUT WHETHER ANY CELL CAN BE MERGED INTO ANY OTHER CELL)

Sometimes it may be possible that the processing requirements of a group of parts are a subset of the processing requirements of another cell. The module has been designed to deal with this type of situation. The module checks for every part family whether the processing requirements of this part family are available in any other cell. In that case, this particular part family can be merged to that cell which can provide the processing facilities to the family. If the above mentioned situation occurs then the message "CELL X HAS BEEN MERGED TO CELL Y" will be displayed. The output form designed to display the results on the screen is 'cells_form' for this module. The procedure file responsible for this module is 'do_subset'.

MODULE 8 (TYPE ANY COMMAND)

This module accomplishes any command entered to the system. It might be for displaying the component records in the application, tables defined in the application, rules written in the application, entities defined, or any other inquiry from the system.

Selection END ends the session.

The software programs developed for implementation of the cell design are listed in appendix F. The hierarchy of the designed programs is shown in figure F.1 in the appendix.

CHAPTER 9

CONCLUDING DISCUSSION

9.1 INTRODUCTION

Before going into resolving the question "how" to group and what classifying attributes to be decided for grouping, one needs to define the function of grouping or in other words what objectives are intended to be achieved from this grouping. In this thesis part grouping has focused on making the Process Planning function efficient.

The discussion in this chapter reports the author's experimental work that was carried out on developing a Part Grouping system called CAFBG System, development of HYCAPP —a Hybrid Process Planning system and cell design. Moreover, the discussion also centres on a critical review of the research work. The experiments used real life design data or parts, described in chapter 5.2, to test the feasibility and viability of this research. These parts are being manufactured by GEC Alsthom Large Machines Limited, Rugby, England. The design data for thirty components has been interpreted and loaded in the knowledge base of the GENERIS Expert System shell. The design data is representative of the variety of parts belonging to both the major classes (i.e. rotational and prismatic). The parts vary a great deal in the geometries, sizes and other technological requirements. The parts are generated by a wide range of machining processes. A large variety of machines from traditional types to advanced machining centres are involved in machining of these machined parts. A manufacturing system capability model that consists of the machines involved for the machining of the above mentioned parts has also been created for a variety of tests related to part grouping and development of HYCAPP System.

The experimental work started with the experiments performed to evaluate the demerits inherent in the grouping methods used traditionally, and then moves through the experiments carried out to check the validity of the proposed CAFBG System, the validity check of the approach for stopping the grouping process. Furthermore, discussion on the development of HYCAPP System and cell design will also be taken.

Programming language 'C' was used when needed outside the Expert System shell. The Expert System tool has proved quite effective in the implementation of this research.

9.2 FEATURE-BASED PART DATA MODEL

Features are considered a communication medium between design and manufacturing. It was therefore decided to describe part in terms of features in this research. Feature-based component design information has been modelled, described in chapter 3.5.2 and implementation in chapter 3.7, in the Expert System environment. The representation of the feature-based component data model in the knowledge base of the Expert System has provided a good basis for the reasoning process required in transforming the design information of the components into their processing requirements; extracting the proposed classifying attributes, after reasoning out for the part family formation purposes; incorporating the decision-making process in developing a Hybrid Process Planning system and making it suitable for designing the manufacturing cells. The feature-based component data model is has been pictorially depicted in figure 3.10 and representation in GENERIS is given in appendix A.

9.3 PART GROUPING

Most research for part family formation and manufacturing cell design has concentrated on either Classification and Coding (C&C) or Production Flow Analysis (PFA) procedures. A classification and coding system scrutinises the design features of the parts from product codes. Those parts with similar codes are then formed into the same family. Much time and effort is involved in coding parts and creating an elaborate database; nevertheless, it is believed that this approach produces a weak connection between component features and machine groupings. They do not take into account the manufacturing information while making the decisions. On the other hand, PFA does not use part design information to identify part families. Instead, PFA is used to analyse the operation sequence and machine routing for the parts produced in the given shop. It groups parts with identical or similar routing.

To code all the design information of the component for achieving part grouping from the manufacturing point of view is very difficult using C&C systems because there exist many different types of machining regions (features) whose manufacturing requirements depend on other different technological parameters like size, surface finish, tolerances, connectivity etc. C&C systems therefore, do not result in realistic part grouping from the manufacturing point of view.

PFA based grouping does not lead to favourable results as well. There are mainly three reasons. Firstly, highly flexible machines are available these days, they can handle a large variety of diverse processes. In the context of batch manufacturing, there is need to keep the two issues separate i.e. identifying the part families and then allocating the machining cells to them. Secondly, components are grouped based on the machines they visit. It does not give much information about the two components visiting the same machine, how similar they are or in other words how far they share processing facilities provided by that particular machine. Grouping based on the machines they visit, therefore, does not give a realistic indication of how similar components are similar in their manufacturing requirements and work contents. Thirdly, due to the lack of a rationalisation mechanism in the manufacturing routings, different machines can be allocated for the machining of similar components. This shows similar components to be dissimilar and places them in different groups. Moreover, PFA and C&C systems both do not deal with the recent features approach.

Part grouping has been described in chapter 4.4 and its implementation is given in chapter 5.

9.3.1 DISCUSSION ON THE PART GROUPING

Experiments described in section 5.4, performed to evaluate the demerits involved in different part grouping techniques are described and critically evaluated in the following. These experiments led the author proposing the classifying attributes that have been used for part grouping in this research. Then discussion on part grouping based on CAFBG System has been taken at the end under this heading.

A set of thirty components was tried as a case study and the results of all the four methods i.e. Feature-based grouping, Process-based grouping, Machine-based grouping and CAFBG System are evaluated. This is discussed in the following. Moreover, discussion on part grouping based on composite component has been also taken in the following section.

9.3.1.1 PARTITIONING AROUND MEDIODS

The experiments performed under this heading, described in chapter 5.3.1, were to check the viability of this part grouping approach. As already said elsewhere in the thesis that a composite component can be used to represent the processing needs of a part family or capabilities of a manufacturing cell which it can offer to process a part family. We will call it grouping around cell capabilities in the following discussion. Description of the topic is given in chapters 4.4.1 and 5.4.1. According to this approach, cell capabilities were defined based on:

a) Processes, and
b) Features.

Cell capabilities based on processes included:

i) The processes the cell can offer,
ii) The materials the cell has the capability to machine,
iii) The size or envelope of the components the cell can accommodate.

and the cell capabilities based on features included:

i) The features the cell can machine and the extent of surface finish for each feature,
ii) The materials the cell has the capability to machine,
iii) The size or envelope of the components the cell can accommodate.

After defining the cell capabilities, parts were grouped around the mediods based on both criteria i.e. processes, and features. In this approach, parts are grouped to the cells for which they have more similarity level. In the case a part is at same similarity level with more than one cell, the number of set-ups required to machine the part in the cells were considered. The part will go to the cell where it can be processed with less number of set-ups requirements. A number of problems are found inherent in this method of part grouping. Firstly, this technique can give favourable part grouping results if the cell capabilities defined are exclusive but this is not possible as a limited number of machining processes are needed by an infinite number of machined components. Secondly, parts grouped in a part family/cell achieved this way are not necessarily similar in processing needs to each other. Parts utilising 100% and 10% of the cell capabilities respectively can be in same group. Finally, it is big problem to first define the mediods around which parts are to be grouped without knowing the number of part families and the processing requirements of the components in each part family. Therefore, this part grouping approach does not lead to realistic part grouping from the manufacturing point of view.

9.3.1.2 PART GEOMETRY-BASED GROUPING

In this technique, parts manufacturing features, discussed in chapters 4.4.2 and 5.4.2, were taken as classifying attributes for the part grouping. Findings of this research has proved that part grouping based on simply features cannot lead to realistic grouping results. A number of reasons can be cited in support of the point. Firstly, a large variety of features which might potentially exist on the components. A large number of classifying attributes i.e. feature types will not lead to proper grouping results as the fewest possible number of classifying attributes are considered better to achieve well balanced families. Secondly, the same machining methods can be used for a number of different feature types. The grouping based on feature types dictates that the user differentiate between the parts if different feature types exist on them. It does not take into account the similarity in their machining methods. Thirdly, a feature with different technological requirements (different size, accuracy etc.) might lead to different machining methods, different tools and different machine types. The grouping based simply on part geometry does not take into account those variants such as feature size and accuracy required on the feature. Finally, there are also component attributes like feature connectivity, overall dimensions etc. which are needed in determining the machining methods of the parts. Therefore, the experiment performed, as given in chapter 5.3 and appendix D, to group parts based on the part geometry did not result in favourable grouping.

9.3.1.3 PROCESS-BASED GROUPING

Another experiment performed for the part grouping was that in which processes were selected as classifying attributes. The description of the topic is given in chapters 4.4.3, 5.3 and 5.4.3. These processes are, usually, taken from their Process Plans. The experiment, described in chapter 5.3, showed that process based grouping would not lead to reasonable grouping results. There are number of reasons why grouping based on the processes involved in component machining will not lead to realistic results.

Firstly, these Process Plans are usually written by more than one planner which result in non-standard plans and different solutions for the processing of a part by different planners. In the process of grouping one needs to compare the components based on any standard criteria. Secondly, other component attributes like feature connectivity and feature patterns are not taken into account in the process of grouping. These attributes are also needed for determining the machine resources where the parts are to be manufactured.

9.3.1.4 MACHINE-BASED GROUPING

PFA technique identifies the part families and associated groupings of machine tools. An experiment was conducted, presented in chapter 5.3, for the part grouping based on the machines used for their processing which can be termed as machine-based grouping. Machine-based grouping, described in chapters 4.4.4 and 5.4.4, also does not lead to the grouping results which can bring together the similar parts in the groups from the manufacturing point of view. A number of reasons why this technique is not successful has already been described while discussing PFA.

9.3.1.5 GROUPING BASED ON THE CAFBG SYSTEM

Development of the CAFBG System for part grouping, to make the Process Planning function efficient, is described in chapter 4.5 and is core of this research. The system was aimed at overcoming the shortcomings found in the existing part grouping techniques. Some of the techniques are described in the previous sections. The aim of this research was to automate the part grouping activity which can,

- bring together similar machined parts in the same part families from the manufacturing point of view. This part grouping can make the Process Planning activity more efficient,
- the input to the part grouping process is part features and
- it is suitable for an existing or a specified manufacturing system.

As mentioned above, one of the aims of the part grouping was to use part features as an input. Part grouping based on part features (part geometry-bases grouping) proved unsuitable. One of the major reasons why the part geometry-based grouping was not suitable is that a feature other than geometry contains technological information like size, accuracy etc. as well as geometry. Considering simply geometry of a feature is not enough for the part grouping process. Feature technological information plays an important role in deciding its manufacturing methods. It is very difficult to encompass all this variety of information in part geometry-based grouping. However this combination of information at feature level can be handled if a feature is transformed into manufacturing processes. Also, the machining processes available are limited in number. A term called TSF (technical solution at feature level) has been used in this research for an ordered set of operations performed sequentially to fulfil the geometric and technological requirements (size, tolerances, surface finish etc.) of a feature. Another term called FTD (Feature Transition Diagram) has been used in this thesis which is the representation of all possible solutions at feature level determined based on its geometric and technological constraints. Details of the above mentioned terms are given in chapter 4.5. Keeping this in view a need for a standard criteria which can be used in the part grouping, a single solution or a single TSF was sorted out for each feature. This is a step towards rationalisation. Problems encountered because of a lack of rationalisation mechanism in the part grouping process have already been discussed elsewhere in the thesis.

Simply translating the features into processes selected at feature level does not help in the selection of different machine resource capabilities required for machining the component features. The same features, sometimes, located on different positions of the component require different machining capabilities. Feature connectivity information which is how the features are connected together on the component body is vital for identifying the machining requirements of the features. As already mentioned elsewhere in the thesis, connectivity analysis plays an important role in assigning the processing facilities to the features, as well as downstream activities of the Process Planning function like set-up determination, operation sequencing, etc. Machine resources attached without the connectivity information would not be realistic ones.

One of the problems faced by machine-based grouping was that it brings together those parts visiting the same machine/machines. This approach does not necessarily lead to a grouping of similar components. The reason for this is that parts may use the machine/machines completely or only partially, but will be placed in a same group. In order to address this problem, a concept of dividing machine tools capabilities was introduced which is MCU's (Machine Capability Units). The MCU level is in between the machine tool level and process level. It therefore provides better links between part features machining requirements and machine tools capabilities while solving the connectivity problem of part features. Conversion of part features machining requirement into MCU's is one of the classifying attribute of the system. Machining requirements in terms of MCU's are machine independent, therefore solution does not need changing when new resources are added. The MCU concept also provides the mechanism to address an existing or specified manufacturing system. Other classifying attributes include:

- Dimensional ratio/ratios of the component
- Number of approach directions (AD's) to machine the component
- Features pattern

Dimensional ratio/ratios of the component are responsible for identifying its basic shape. The basic shape of the component dictates the machining methods. Different components with different shapes need different machining and tooling methods. These ratios help in characterising the parts for selecting the machine resources, potentially candidate operations and even sometimes work holding methods. A features pattern can have implications for tooling and sometimes machine allocation problems as well. A features pattern may dictate that a special machine or tooling is required. As far as the number of approach directions (AD's) to machine the component is concerned, it does not have any direct implication in selecting the production methods, yet this is new concept in the Group Technology application and it can be used for the comparison of work contents needed in the machining of parts. The implications of the classifying attributes have already been discussed in detail elsewhere in the thesis. A reference model for the CAFBG System is shown in figure 4.9.

Experimental results, described in chapter 5.3, have proved that part grouping based on the CAFBG System show realistic grouping results from the manufacturing point of view.

GROUPING ALGORITHM DEVELOPMENT

Cluster analysis is one of the most frequently applied mathematical tools in Group Technology (GT). Cluster analysis is rearranging the groups of data points that possess strong internal similarities.

One of the hierarchical clustering procedure called "*Agglomerative hierarchical clustering*" has been, discussed in chapter 4.5.8, is applied in this research. Hierarchical clustering procedures are among the best-known methods because of their conceptual simplicity. Agglomerative hierarchical clustering procedures start with n singleton clusters and form the sequence by successively merging clusters.

Sometimes a fuzzy cluster analysis approach is used for part grouping. Mainly two different approaches to fuzzy cluster analysis, namely fuzzy equivalence and fuzzy classification, are employed in the process of part family formation. In fuzzy equivalence analysis, a part-characteristics matrix is prepared in a same way as given in chapter 4.5.6.1. A membership is defined for each classifying attribute. The membership values are within a close interval [0,1] being a subjective one. The similarity coefficient matrix is calculated in a similar fashion as discussed in section 4.5.6.3. In the clustering process after a max.min (union) operation of two fuzzy similarity relations, a λ value is considered which is a similarity threshold for the parts in order to be in the same part family. In fuzzy classification, a number of groups is decided first. This information can be represented in an incidence matrix $\{a_{ij}\}$ where c rows represent the groups and n columns represent the parts. A membership value is assigned to an element $\{a_{ij}\}$ which varies within a closed interval [0,1]. The value of membership function which is between 0 and 1, reflects to which extent a part belongs to each part family. Then a procedure for minimising the objective function is used in the grouping process. As mentioned in both the clustering approaches a membership function needs to be defined which is a subjective one. In the clustering approach used by this research a membership function, the value of which is defined within a close interval [0,1] has been used which is similar to the fuzzy method. Moreover a procedure for calculating a similarity matrix is used which works in both hard clustering and fuzzy clustering.

Therefore it can be argued that clustering technique used in this research can provide the benefits usually achieved in the fuzzy cluster analysis approach for dealing with the uncertainties and ambiguities inherent in a manufacturing system.

STOPPING THE CLUSTERING PROCESS

The same Agglomerative clustering procedure has been adopted for finding out the optimal number of groups in the grouping process. As already mentioned in chapter 4.5.9 that according to the procedure the number of clusters are exactly equal to the number of samples under consideration. The next is a partition into n-1 clusters, the next a partition into n-2, and so on until the nth, in which all the samples form one cluster. At each iteration, a distance between each pair is calculated and the closest pair of groups is merged. In the whole clustering process it can be calculated at which stage the groups are most dissimilar. This is time to stop the grouping process as groups are well separated now. If further merging happens the two most dissimilar groups will merge, and this should be avoided.

In the real world situation part families achieved after grouping are optimised at the time of cell design. The optimisation depends on a number of factors like, company's available resources, level of automation available i.e. material handling system available, components batch-sizes in the families. A number of families can be merged into one family if the number of components in these individual families is small and, also, batch sizes in these families are small. When the information like the batch sizes of the components, company's available resources is not available, the decision for optimum grouping is to be made based on the information in hand that is the classifying attributes of the components. The author takes the view that the grouping function in the cluster analysis process should be stopped at that level when well separated clusters are formed in the absence of any other optimisation criteria.

The method proposed to calculate the optimal number of groups in the cluster analysis process was tested on a set of three examples given by well known researchers in the area to check the validity of the method, described in chapter 4.5.9.1. The results found are exactly the same as calculated by other methods. Therefore, it can be argued that the proposed method has ability to calculate the optimal number of groups in the cluster analysis process.

PATTERN RECOGNITION FOR NEW COMPONENTS

Once part families are identified, the CAFBG System is capable of assigning a new component to existing part families if its similarity level is equal or more than the defined level for assigning the components to the part families. Pattern recognition has been described in chapter 4.5.10 and its implementation is given in chapter 5.3.

9.4 THE CAFBG SYSTEM FOR EFFICIENT PROCESS PLANNING

The Process Planning function can be made efficient if true similar components from the manufacturing point of view are grouped into part families. By the term efficient, the author means that when the part grouping is a realistic from the manufacturing point of view then planning the similar components will be easy i.e machine tools, cutting tools etc. will be similar for the components in a part family. As said earlier, the planning logic compares the component processing requirements imposed by both geometric and technological constraints with the processing system capabilities to select feasible solutions. In the proposed CAFBG System components processing requirements are co-ordinated with the existing or specified manufacturing system. This aspect is also helpful in the Process Planning function as plans are written while considering the existing production system. MCU's are co-ordinating between component's machining requirements with the production system capabilities. MCU is one of the classifying attributes selected in the CAFBG System. Other classifying attributes are dimensional ratios, number of AD's and features pattern. Dimensional ratios exhibit overall shape of a component thus showing potential machining methods. A features pattern can provide information for tooling and sometimes machine allocations. The classifying attribute number of AD's can potentially be used for the evaluation of work contents needed in the machining of the component thus partially covering the features connectivity aspect. As the proposed classifying attributes help a great deal in determining the manufacturing requirements of a component the grouping results based on this criteria (the CAFBG System) have resulted in realistic part grouping from the manufacturing point of view. If true similar components come together in part families, generating the planning function becomes easy and productive. Discussion on the topic has been taken in chapter 4.6.

9.5 PROCESS PLANNING IN GT

It has been demonstrated that developing a Generative planning system becomes easier and more manageable once realistic part families are achieved. Part families are realistic in the sense that their precise processing boundaries can be identified for developing the planning logic. A Hybrid planning system called HYCAPP System, described in chapter 6, has been developed successfully for a family of components made from prismatic blocks of metal. Implementation of the system is described in chapter 7.2.

9.5.1 COMPONENT GROUPING REQUIREMENTS

Generative planning is a complex task. It is obvious that it is very difficult to develop a Generative planning system which can encapsulate the manufacturing logic for all the domains of components in the world. Hybrid planning systems are being developed to handle the problem by developing the manufacturing logic for the part families, thus making the problem easier and more manageable. A pre-requisite in both the Variant and Hybrid planning systems is realistic part grouping from the manufacturing point of view. A Hybrid planning system actually utilises potentials of both the Generative and Variant systems. Aim of developing a Hybrid planning system is mainly:

1) To decompose a major task into sub-tasks and
2) To work within a defined domain of similar components.

Both the computer-aided planning systems i.e. the Variant and Hybrid are based on GT. The topic has been described in chapter 6.2. The planning function in both the systems can be made productive if true part families from the manufacturing point of view are identified. This research has proved that once true part families are achieved then developing a Generative planning becomes easier as planning is to be done within a defined boundary. Therefore, it can be argued that GT plays an important role in making Process Planning activity productive and efficient.

9.5.2 THE HYCAPP SYSTEM

A planning logic in a Generative system compares the component requirements in terms of geometric and technological both with the processing system capabilities to select feasible solutions. Therefore it is of paramount importance to have knowledge representation of part and manufacturing system capabilities precisely in the system before the planning logic is developed.

A feature-based component data model has been created in the knowledge base of the Expert System shell. The data model has not only been used in the part grouping function but also successfully in Generative planning and cell design applications. A manufacturing system capability model has also been modelled in the knowledge base of the Expert System. Flow diagram for information flow in the system is shown in figure 7.1. A reference model for the system is shown in figure 6.6.

A modular structure approach has been adopted in the development of the system. Under a modular approach, the major task is decomposed into sub-tasks and modules can be developed for these individual tasks. By using a modular approach, the task of development as well as expansion and modification of the system can be made easy and manageable. However, the modules can communicate with each other through the data input and output.

Other than developing the main module for generating the Process Plans, there are other modules developed to address the activities like creating the records for a component, editing the records for a component, deleting the records for a component, loading any component of the family for which plans are to be generated, checking whether any part can be planned or not etc.

9.5.3 THE HYCAPP SYSTEM CAPABILITIES

In the HYCAPP System, stock is taken as a Rectangular Boss feature. Six free surfaces and six PAD's which are normal vectors to these free surfaces are associated with the Boss in the knowledge base. Features are the geometric shapes to be cut from the stock, as said before.

The HYCAPP planning system can attach a set of TSF's to each feature which is picked up from its FTD based on its geometric and technological requirements. The TSF's are then transformed into MCUE's with the association of connectivity information of the feature on a component body. MCUE's exhibit the precise nature of machining requirements and a true picture of set-ups of a component. Then the total resources needed to machine the component are minimised and an appropriate solution is attached to each component feature.

Real and imaginary faces of the features can be utilised to calculate the relationship of parent and child between the features. There is need here to calculate the parent/child relationships among the features in order to calculate the PAD's for the features. A child feature like its parent feature may come up with more than one potential parent feature. True parent features need to be identified for this child feature so that its PAD's can be established. The system can do this reasoning process, details being given in chapter 7.2.1.2. With the information of the real and imaginary faces of the feature along with its connectivity and component PAD's, the system can calculate the PAD's of the feature.

After attaching the MCUE's to each feature PAD's, component AD's (which in this case represent the number of set-ups) are determined by clustering the MCUE's from common component PAD's. Here features MCUE's may appear in more than one cluster in different component PAD's. The final optimised component AD's are selected by minimising the number of clusters. This is done based on step by step selection of clusters containing the maximum number of feature state elements and then removing those feature state elements from the remaining clusters in the other component PAD's.

Operation optimisation of the features achieved in this research is a step in advancing the Process Planning research. This is contrary to the operation optimisation activity of the features planned at local level for the determination of set-ups, usually adopted by researchers. Planning at local level results in a chain of operations. Sometimes for the same operation, the component might need to visit another machine tool. The set-ups determined that way will not be realistic ones. The movement to another machine will add other set-ups in the planning. The author believes that it is good idea in advancing the set-up planning if, after matching, the true MCU's which are responsible for machining the features are attached to the features in the beginning and then optimised. This will exhibit the realistic picture of set-ups and the processing needs of the components.

The sequencing of operations is quite complicated job. A lengthy program has been developed to achieve the sequencing of operations. The sequencing of operations, implemented in the software, is based upon by considering the following categories of constraints:

1. Sequencing based on operation constraints;
2. Sequencing based on geometric constraints;
3. Sequencing based on 'good manufacturing practice' constraints.

At the end, a composite component of the planned part family, described in chapter 7.4, as well as the boundary of the system have been established. The system is capable of generating plans for components outside the family if they lie within the defined boundary, as described in chapter 7.5.

9.6 CELL DESIGN

This section discusses how the proposed classifying attributes for the part grouping are not only used to make the Process Planning function efficient but can also help in designing manufacturing cells.

As already said, cellular manufacturing is actually a subdivision of the manufacturing system or clusters of machines which can independently process a family of similar components.

Traditionally, similarities between parts have been established on the basis of which machines are required to produce them. This approach is not advantageous because the same operation can be carried out on several machines, and a great deal of flexible machines are available at the moment where quite diversified types of operations can be performed on each individual machine. In other words, the increased capabilities of the machine tools make it almost impossible to group parts based on the machines they require for their machining. An example of a five axis machining centre can be cited here. This centre is visited by nearly all types of components. Therefore, it is preferable to form part families based on the classifying attributes of the parts themselves and later assign the families to the relevant machine cells where they can be processed.

The composite component of the part family based on the CAFBG System represents the classifying attributes of among others, the MCU's, the shape of the parts, and the presence of features pattern. These attributes of the composite component are quite enough for exhibiting the machining requirements of the family of the components. MCU's represent the actual processing needs of the parts in a part family. The basic shape of the component dictates the machining methods as said before. Components basic geometrical shapes or dimensional ratios help in selecting the machining resources for them. A features pattern can have implications for tooling and sometimes machining resource allocations, as discussed earlier. Therefore, a composite component for a part family established on the CAFBG System can provide a better representative picture of the processing needs of the family and is helpful in cell design.

It has been shown in this work that a composite component represented in terms of MCU's can represent better processing requirements of a part family. By considering the working envelopes of the machines, machining resources can be identified for the part families. A machine is assigned to the cell where the utility of its MCU's being utilised by the parts in a cell is maximum. The utility should be calculated while considering also the batch sizes of the parts to be machined but this aspect was not implemented in the software. Though other classifying attributes present on the composite component of the part family, such as the overall shapes of the components and features pattern also have implications in identifying the machining resources, this aspect was also not implemented in the software.

Cell design is not just assigning the machines to a cell, but a number of other factors also need to be addressed (cell performance adjustment). Most of these factors are discussed in section 8.3.3 but were not implemented in the software, because a lot of problems in cell design are not discovered until the system is in place, as mentioned in the section.

Cell design has been discussed and its implementation given in chapter 8.

CHAPTER 10

CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER WORK

10.1 INTRODUCTION

The research described in this thesis has explored part grouping with focus on:

1. Part grouping for efficient Process Planning.
2. Part description being feature-based in part grouping function.
3. Part grouping to suit an existing or specified production system environment.

Different experiments have been performed to check the viability of different existing part grouping approaches like part grouping: around mediods; based on part geometry; based on machining processes; and based on machines. The investigation carried out has resulted in a new set of classifying attributes which can overcome some of the shortcomings inherent in the above mentioned part grouping techniques a great deal. An experimental software called CAFBG (computer-aided feature-based grouping) System has been implemented to automate the part grouping function based on these attributes. The CAFBG System is capable of grouping any machined components which can be manufactured in a given machine shop, provided the machining facility has been specified to the system. A prototype Hybrid planning system called HYCAPP System for a part family generated based on the attributes, has been developed to demonstrate potential benefits achieved in programming a CAPP when system boundaries are defined. Moreover, work has been done in the area of cell design to show that part grouping based on the proposed classifying attributes is helpful even in cell design activity. The research which has been done has led to the conclusions and recommendations for further work, which are made in this chapter.

10.2 CONCLUSIONS

1. A new set of classifying attributes have been developed to improve part grouping for the process planning function. These attributes are: 1) part processing requirements in terms of MCU's (Machine Capability Units); 2) dimensional ratio/ratios of a part; 3) number of AD's (approach directions) to machine the part; and 4) features pattern. Grouping based on these attributes or criteria outperform other grouping methodologies in a number of ways:
    1) ***The MCU*** is the most significant amongst these classifying attributes because its use makes the solution of the part grouping problem independent of specific machine tools, whereas other grouping methodologies are machine tool specific. The MCU provides a more generic and flexible machining solution because any group of parts formed by using MCU's can be processed on any machine tools which collectively include the required MCU's.
    2) ***The number of AD's***, being new in the GT paradigm, is a measure of complexity and work content of parts and has direct bearing on number of set-ups and fixturing requirements. Such a facility is not available in other part grouping methodologies.
    3) The classifying attribute ***features pattern*** considers the importance of part features pattern in the part grouping function. This provides an important advantage over other grouping techniques by enabling special toolings and special machine resource requirements to be identified in the part grouping process.
    4) Part ***dimensional ratio/ratios***, based on the Opitz classification and coding system, represent the basic geometrical structure or shape of parts. This classification system provides a good basis for classifying parts according to their basic geometrical shapes. For example cylindrical parts, where length-to-diameter ratio is less than or equal to 0.5, can be classified as discs. For diameters up to approximately 300 mm, the part would be gripped in a lathe chuck, for larger diameters

the part would be clamped on the table of a vertical borer. Potential machining features involved and thus the machining processes required may be predicted on the basis of the basic shape of the parts. This is a further important factor in the selection of production methods for the part family.

2. An experimental software called CAFBG System, for the implementation of part grouping based on the above mentioned attributes, has been produced. This is a knowledge based expert system that can automatically provide a number of functions. Given the feature-based part description as an input, it converts this into the format required for the cluster analysis program, after extracting the required attributes from the knowledge base. It then does part grouping, and finds the optimal number of part families. The part groups are then reloaded in the knowledge base. New parts can be added to existing part families based on their similarity, by pattern recognition, with parts in the existing families. The inadequacies of traditional part grouping techniques are largely overcome when the criteria developed in the CAFBG System are implemented. Discussion and comparison between results based on traditional methods and those produced by the CAFBG System are given in detail in chapter 5.

3. One of the purposes for the part grouping, to suit an existing or specified production system, has been achieved by introducing the MCU concept. The MCU concept provides better links between part features processing requirements and production system capabilities while observing the connectivity aspect of part features.

4. A procedure has been devised and implemented to find out an optimal number of families in the grouping process. This is contrary to the approaches where clustering techniques are usually based on a pre-defined number of groups or fixed minimum level of similarity, at which grouping process should be stopped.

5. It has been shown how a feature-based component description can be modelled in an Expert System environment which can be used for part grouping, Hybrid process planning and cell design.

6. The viability of part grouping around mediods (defined cell capabilities around which parts can be grouped) has been investigated and found to have limitations for part grouping. The reasons being: it is not applicable to define the capabilities of machining cells exclusively; parts in a family formed by this approach cannot be taken as similar because their machining work contents are not always the same; there is a big problem in defining the mediods without knowing the number of part families and the processing requirements of the parts in each part family before hand.

7. The experiments performed in part grouping, using form features as the classifying attribute, has proved that this part grouping technique has demerits because: a large variety of feature types exist on parts (selecting a large number of attributes will not result in proper part grouping); similar processing methods are available for different features; the processing method of a feature is not simply linked to the feature type but also other parameters like its size, accuracy etc.

8. The experiments carried out to check the viability of the part grouping technique, using the machining processes as the classifying attribute, has shown that this method of grouping results in poor part grouping because: alternative processing solutions can be allocated to the part features; the same processes can be carried out on several machines; this grouping approach does not take into account information like feature connectivity and the features pattern that are needed for selecting the true processing facilities.

9. The experiments conducted while taking machine tools as classifying attribute has proved that this approach has limitations, the main reasons being: alternative machine tools can be selected for the machining of parts; parts utilising a machine capability completely or only partially are taken as similar; availability of versatile machine tools recently in the market where most of the part features can be done on a single machine.

10. It has been shown, by developing a prototype Hybrid planning system, that programming a Generative planning system is easier and more manageable if it is written for a set of similar parts in a part family, thus constraining the programming within a part family domain. This is because the wider the part set, the more difficult is development of the logic.

11. It has been found that the MCU concept not only helps in achieving well balanced part families but is also helpful in process planning and cell design applications because part machining requirements can be precisely represented in terms of MCU's.

10.3 RECOMMENDATIONS FOR FURTHER WORK

1. There is presently no procedure available in which the structural aspect of a part can be completely addressed. A mechanism needs to be investigated which can completely encapsulate the connectivity or structural aspects of part features. This will be helpful in defining the boundary of a Hybrid planning system from the structural aspect of components present in a part family. This will also help in characterising the parts based on this aspect.

2. The approach taken in this research attaches TSF's (technical solution at feature level) to features, but assumes that features can be machined equally and effectively from each of their EAD's (external access directions). There is a need to develop this such that EAD's are selected to be both feasible and optimal in terms of time and cost of machining.

3. In an incidence matrix for cluster analysis a membership function for each classifying attribute is defined within a closed interval [0,1] which is subjective. An investigation needs to be carried out to develop a cluster analysis approach which can allocate different weights to different classifying attributes in an incidence matrix, according to their priority or importance in the clustering function.

4. The functions of raw material selection and machine selection are addressed in the thesis but are not presently implemented in the HYCAPP System. If such facilities are implemented in the software, these will greatly enhance the system capability.

5. The output of the HYCAPP System is capable of development to enable it to be utilised for part programming to realise the integration of CAPP and CAM.

REFERENCES

Allen, D. K, 1979; *Classification and Coding - Theory and Application.* Computer-Aided Manufacturing Laboratory, BrighamYoung University, Provo, Utah.

Allen, D. K, and Smith, P. R., 1980; *Computer-aided process planning.* (Provo, Utah: Brigham Young University, Computer Aided Manufacturing Laboratory).

Allen, D. K., and Smith, P. R., 1984; *Computer Aided Process Planning.* (Provo, UT: Brigham Young University, Computer Aided Manufacturing Laboratory).

Alder, G. M., McGeough, J. A., Spencer, C. A., Hon, K. K. B., and Ismail H. S., 1986; *Selection of machining processes by intelligent knowledge-based systems.* International Conference on Computer-aided Production Engineering, Edinburg, April 1986.

Al-Qattan, I., 1990; *Designing flexible manufacturing cells using a branch and bound mathod.* International Journal of Production Research, Vol. 28, No. 2, pp. 325-36.

Alting, L., and Zhang, H., 1989; *Computer Aided Process Planning: the state-of-the-art survey.* International Journal of Production Research, Vol. 27, No. 4, pp. 553-585.

Anon 1981; *Process planning problems: First see how with SEE-WHY.* The Production Engineer, Vol. 60, No. 12, pp. 14-18.

Armstrong, G. T., Carey, C. G., and Pennington, A. D., 1984; *Numerical Code Generation from a Geometric Modelling System.* Solid Modelling by Computers, Plenum Press.

Arn, E. A., 1975; *Group Technology.* Springer — Verlag, New Yark.

Askin, R. G., and Chiu, K. S., 1990; *A graph partitioning procedure for machine assignment and cell formation in group technology.* International Journal of Production Research, Vol. 28, No. 8, pp. 1555-1572.

REFERENCES

Ballakur, A., and Steudel, H. J., 1987; *A within cell utilisation based heuristic for designing cellular manufacturing system.* International Journal of Production Research, Vol. 25, No. 5, pp. 639-665.

Barash, M. M., 1980; *Computer-integrated Manufacturing.* Winter Annual Meeting of ASME, November 1980, pp 37-50.

Barkocy, B. E., and Zdeblick, W. J., 1984; *A Knowledge-Based System for Machining Operation Planning.* Proceedings of AUTOFACT 6 (Dearborn, MI: Society of Manufacturing Engineers).

Barr, A., and Feigenbaum, E., 1981; *The Handbook of Artificial Intelligence.* Vol. 1, Kaufman Publishers, Los Altos, CA.

Barr, A., and Feigenbaum, E., 1982; *The Handbook of Artificial Intelligence.* Vol. 2, Kaufman Publishers, Los Altos, CA.

Baxter, R., 1985; *Planning for CIM.* The Production Engineer, Vol. 64 (2), pp. 21.

Bedworth, D. D., Henderson, M. R., and Wolfe, P. M., 1991; *Computer-Integrated Design and Manufacturing.* McGraw-Hill Book Co., Singapore.

Beigel, J., 1986; *The future of Artificial Intelligence in manufacturing.* Proceedings of the 8th Annual Conference on Computers and Industrial Engineering.

Besant, C. B., and Lui, C. W. K., 1986; *Computer-Aided Design and Manufacture.* Third Edition, Ellis Horwood Limited, England.

Beynon, D. P., 1993; *Knowledge Engineering for Information Systems.* McGraw-Hill Book Company.

Black, J. T., 1983; *Cellular Manufacturing Systems Reduce Setup Time Making Small Lot Production Economical.* Industrial Engineering, Vol. 15.

Bo, K., 1989; *Hardware for Computer Graphics and Ccomputer Aided Design.* University of Trondheim, Norway.

Boe, W. J., and Cheng , C. H., 1991; *A close neighbour algorithm for designing cellular manufacturing systems.* International Journal of Production Research, 29(10), pp. 2097-2116.

REFERENCES

Boerma, J. R., and Kals, J. J., 1988; *FIXES, a System for Automatic Selection of Set-ups and Design of Fixtures.* Annals of the CIRP, Vol. 37/1/1988, pp. 443-446.

Boerma, J. R., and Kals, J. J., 1989; *Fixtures Design with FIXES: The Automatic Selection of Positioning, Clamping and Support Features for Prismatic Parts.* Annals of the CIRP, Vol. 38/1/1989, pp. 399-402.

Bonnet, A., 1985; *A.I. Promise and performance.* Prentice Hall.

Boothroyd, G., 1981; *Fundamentals of Metal Machining and Machine Tools.* McGraw-Hill Book Co, Singapore.

Boothroyd, G., and Knight, W. A., 1989; *Fundamentals of Machining and Machine Tools.* Marcel Dekker, New York.

Brimson, J. A., and Downey, P. J., 1986; *Feature Technology: A Key to Manufacturing Integration.* CIM Review, spring 1986.

Broaden, R. J., 1985; *Understanding and Planning for Computer-Integrating Manufacturing (CIM).* MSc Thesis, Department of Management Sciences, UMIST.

Broaden, R. J., and Dale, B. G., 1986a; *What is Computer-Integrated Manufacturing?* International Journal of Operations and Production Management, Vol. 6 (3).

Broaden, R. J., and Dale, B. G., 1986b; *Development of a Model for use in Planning for Computer-Integrated Manufacturing.* International Journal of Operations and Production Management, Vol. 6 (5).

Brooks, S. L., Hummel, K. E., and Wolf, M. L., 1987; *XCUT: A Rule Based Expert System for the Automated Process Planning of Machined Parts.* Bendix Kensas City Technical Report BDX-613-3768 (June 1987).

Buchanan, B. G., and Shortliffe, E. H., 1984; *Rule-Based Expert Systems.* Addison-Wesley, Reading, MA.

Burbidge, J. L., 1963; *Production flow analysis.* The Production Engineer. Vol. 42, No. 12.

REFERENCES

Burbidge, J. L., 1964; *Dragons in Pursuit of the Economic Batch Quantity*. Operation Research, Vol. 15, No. 4.

Burbidge, J. L., 1971; *Production flow analysis*. The Production Engineer, Vol. 50, April/May, pp. 139-152.

Burbidge, J. L., 1974; *Production flow analysis on the computer*. The Production Engineer.

Burbidge, J. L., 1975: *The Introduction of Group Technology*. Heinemann Ltd., London.

Burbidge, J. L., 1977; *A manual method of Production flow analysis*. The Production Engineer, October 1977.

Burbidge, J. L., 1979; *Group Technology in the Engineering Industry*. Mechanical Engineering Publications, London.

Burbidge, J. L., 1980; *The Introduction of Group Technology*. John Wiley & Sons, New Yark.

Burbidge, J. L., and Zelenovic, D. M., 1983; *Using PFA to Plan GT for a New Factory*. Material Flow, Vol. 1, pp. 129-140.

Burbidge, J. L., 1988; *Production flow analysis for planning group technology*. Oxford Science Publications, UK.

Burbidge, J. L., 1989; *Production Flow Analysis*. Oxford: Clarendon Press, 1989.

Burbidge, J. L., 1991; *Production Flow Analysis for Planning Group Technology*. J. Ops. Mgmt., Vol. 10, No. 1, Jan. 1991.

Burch, J. G., Strater, F. R., and Grudnitski, G., 1983; *Information Systems Theory and Practice*. John Wiley, New Yark

Butterfield, W. R., Green, M. K., Scott, D. C., and Stoker, W. J., 1985; *Part features for process planning*. Report C85, 3, CAM-I Inc., Arington, Texas.

CAM-I report 1984; XPS-E: *An Experimental System for Process Planning*. CAM-I report (August 1984).

Carlier, J., and Peters J., 1985; *MOPS - A Machining centre Operation Planning System.* Annals of the CIRP, Vol. 34/1/1985, pp. 409-411..

Carrie, A., 1973; *Numerical Toxonomy applied to group technology and plant layout.* International Journal of Production Research, Vol. 11, No. 4, pp. 399-416.

Case, K., and Gao, J. 1993; *Feature Technology - An Overview.* International Journal of Computer Integrated Manufacturing, Vol. 6 Nos. 1 & 2 pp. 2-12.

Chan, H. M., and Milner D. A., 1982; *Direct clustering algorithm for group formation in cellular manufacture.* Journal of Manufacturing Systems, 1, pp. 65-74.

Chan, S. C., and Voelcker, H. B., 1986; *An introduction to MPL - a new machining process/programming language.* Proceedings IEEE Conference on Robotics and Automation.

Chandrasekharan, M. P., and Rajagopalan, R., 1986; *An ideal seed non-hierarchical clustering algorithm for cellular manufacturing.* International Journal of Production Research, 24 (2), pp. 451-463.

Chandrasekharan, M. P., and Rajagopalan, R., 1986a; *MODROC: an extension of rank order clustering for group technology.* International Journal of Production Research, 24 (2), pp. 1221-1234.

Chandrasekharan, M. P., and Rajagopalan, R., 1987; *ZODIAC - an algorithm for concurrent formation of part families and machine-cells.* International Journal of Production Research, 25 (6), pp. 835-850.

Chang, T. C., 1980; *Interfacing CAD and CAM— a study of hole design.* M.S. Thesis, Virginia Polytechnic and State University, Blacksburg, Virginia, USA.

Chang, T. C., and Wysk, R. A., 1981; *An Integrated CAD/Automated Process Planning System.* AIIE Transactions, Vol. 13, No. 3, pp. 223-233, September 1981.

Chang, T. C., and Wysk, R. A., 1984; *Integrating CAD and CAM Through Automated Process Planning.* International Journal of Production Research, 22 (5), pp. 97-114.

Chang, T. C., and Wysk, R. A., 1985; *An Introduction to Automated Process Planning Systems.* Prentice-Hall, Inc.

Chang, T. C., Anderson, D. C., and Mitchel, O. R., 1988; *QTC— An Integrated Design/Manufacturing/Inspection System for Prismatic Parts.* Proceedings of the 1988 ASME International Computers in Engineering Conference, pp. 417-426, July 31 - August 4, 1988.

Chang, T. C, 1990; *Expert Process Planning for Manufacturing.* Addison-Wesley Publishing Company, 1990.

Charniak, E., and McDermott, D., 1985; *Introduction to Artificial Intelligence.* Addison-Wesely, USA.

Chen, B., 1982; *ROMAPT : A new link between CAD and CAM.* Computer Aided Design, 14, pp. 261-266.

Chen, S. J., Hinduja, S. and Barrow, G., 1989; *Automatic Tool Selection for Rough Turning Operations.* International Journal of Machine Tools and Manufacturing, Vol. 29. No. 4.

Cheung, Y. P., and Dowd, A. L., 1988; *Artificial Intelligence in Process Planning.* Computer-Aided Engineering Journal, August 1988.

Chevalier, P. W., 1983; *Group Technology: The Connecting Link to Integration of CAD and CAM.* CASA/SME Autofact Europe Conference, September 1983.

Chevalier, P. W., 1984; *Group Technology as a CAD/CAM Integrater in Batch Manufacturing.* Int. J. Operations and Production Management, Vol. 4, No. 3.

Choi, B. K., and Barash, M. M., and Anderson, D. C., 1984; *Automatic recognition of machined surfaces from a 3D model.* Computer Aided Design, 16 (2), pp. 81-86.

Choi, B. K., and Barash, M. M., 1985; *STOPP: an approach to CADCAM integration.* Computer-aided Design, Volume 17, Number 4, May 1985.

Choobineh, F., 1988; *A Framework For the design of Cellular Manufacturing Systems.* International Journal of Production Research., Vol 26, No 7, pp. 1161- 1172.

Chryssolouris, G., and Guillot, M., 1990; *A Comparison of Statistical and Artificial Intelligence Approaches to the Selection of Process Parameters in Intelligent Machining.* ASME Journal of Engineering for Industry, Vol. 112, pp. 122-131.

Chu, C. H., and Wang, H. P., 1988; *The Use of Artificial Intelligence in Process Planning.* International Journal of Operations and Production Management, pp. 5-18.

Chu, C. H., and Mayshing, T., 1990; *A comparison of three array-based clustering techniques for manufacturing cell formation.* International Journal of Production Research, Vol. 28(8), pp. 1417-1433.

Chu, C. H., and Hayya, J. C., 1991; *A fuzzy clustering approach to manufacturing cell formation.* International Journal of Production Research, Vol. 29(7), pp. 1475-1487.

Chung, J. C., 1988; *Feature Based Geometry Construction for Geometric Reasoning.* ASME-Computers in Eng. Conference.

Chuo, T., and Ito, Y., 1991; *A New Concept of CAPP Based on Flair of Experienced Engineers — Analysis of Decision-Making Processes of Experienced Process Engineers.* Annals of the CIRP, Vol. 40/1/1991.

Clark, A. L. and South, N. E., 1987; *Feature-Based Design of Mechanical Parts.* SME AUTOFACT 1987, Detroit, pp. 1.69- 1.76.

Cohen, P. R., and Feigenbaum, E., 1982; *The Handbook of Artificial Intelligence.* Vol. 3, Kaufman Publishers, Los Altos, CA.

Coombs, M. J., 1984; *Developments in expert systems.* Academic Press Inc. (London) Ltd., 1984.

Crookall, J. R., and Baldwin, K. I.,1972; *An investigation into application of grouping principles and cellular manufacturing using Monte Carlo simulation.* Annals of the CIRP, 1, 3.

Cunningham, J. J. and Dixon, J. R., 1988; *Designing with features: the origin of features.* Computers in Engineering, Vol. 1.

Cutkosky, M. R., and Tenenbaum, J. M., 1987; *CAD/CAM Integration Through Concurrent Process and Product Design.* Winter Annual Meeting of ASME, 1987, PED-Vol. 25, pp. 1-10, Boston, USA.

Cutkosky, M. R., *et al.* 1988; *Features in Process Based Design.* proc. Computers in Engineering, ASME.

REFERENCES

Dahel, N. E., and Smith, S. B., 1993; *Designing flexibility into cellular manufacturing systems.* International Journal of Production Research, Vol. 31(4), pp. 933-945.

Dallery, Y., and Yao, D. D., 1988; *Modelling a system of flexible manufacturing cells, Modelling and Design of Flexible Manufacturing Systems* (Kusiak, A., Ed.), pp. 289-300, Elsevier.

Darvishi, A. R. and Gill, K. F. 1988; *Knowledge Representation Database for the Development of a Fixture Design Expert System.* Proceedings of the Institute of Mechanical Engineers, Vol. 202, No. B1.

Davies, D. L., and Bouldin, D. W., 1979; *A Cluster Separation Measure.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 1, No. 2, pp. 224-227.

Davies, B. J., and Darbyshire, I. L., 1984; The use of expert systems in process planning. Annals of the CIRP, 33/1/1984, pp. 303-306.

De Beer, C., Van Gerwen, R., and De Witte, J., 1976; *Analysis of engineering production systems as a base for production oriented reconstruction.* Annals of CIRP, Vol. 25/1/1976, pp. 439-441.

Descotte, Y., and Latombe, J. C., 1981; *GARI: A Problem Solver that Plans how to Machine Mechanical Parts.* Proc. 7th Int. Joint. Conf. on Artificial Intelligence, pp. 766-772.

Descotte, Y., and Latombe, J. C., 1985; *GARI: An expert system for process planning.* Solid Modelling by Computers, M. S. Pickett and J. W. Boyse, (Eds.) (New York: Plenum Press).

Detollenaere, M., Dewolf, W., Vanacker, H., Pinte, J., and Williams, D., 1988; *ESOP: An Expert System for Operation Planning on Machining Centres.* CAM-I Report M-88-PPP-02.

DeWitte, J., 1980; *The use of similarity coefficients in production flow analysis.* International Journal of Production Research, 18 (4), pp. 503-14.

Dixon, J. R., 1986; *Artificial Intelligence and Design: A Mechanical Engineering View.* Proceedings of AAAI-86 Fifth National Conference on Artificial Intelligence, Vol. 2, Philadelphia, PA, August 1986, pp. 872-877.

Dixon, J. R., 1988; *Designing with features: building manufacturing knowledge into more intelligent CAD systems.* ASME manuf. Inter 88, Atlanta GA, April 1988.

Doukidis, G. I., and Whitley, E. A., 1988; *Developing Expert Systems.* Chartwell-Bratt (Publishing and Training) Ltd.

Duda, R. O., and Hart, P. E., 1973; *Pattern Classification and Scence Analysis.* John Wiley & Sons, Inc.

Dumolin, W. J., and Santen, W. J., 1983; *Cellular Manufacturing becomes a philosophy of management of component facility.* Industrial Engineering, Nov. 1983, Vol. 15.

Dunn, M. S., and Mann, W. S., 1978; *Computerised Production Process Planning.* Proceedings of the 15th NCS Annual Meeting and Technical Conference.

Edwards, G. A. B., 1971; *Readings in Group Technoloy.* The Machinery Publishing Co., London.

Edwards, G. A. B., and Koenigsberger, F., 1973; *Group Technology — The cell system and machine tools.* The production Engineer, July/August 1973.

El-Midany, T. T., and Davies B. J., 1981; *AUTOCAP - A Dialogue System for Planning and Sequence of Operations.* International Journal of Machine Tool Design and Research, 21 (3/4).

El-Midany, T. T., and Davies B. J., 1983; *MIPLAN: Implementation at Union Switch and Signal.* Association for Integrated Manufacturing Technology (Numerical Control Society) 20th Annual Meeting and Technical Conference, April 1983.

El-Essawy, I. G., and Torrance, J., 1972; *Component flow analysis — an effective approach to production systems design.* The Production Engineers, Vol. 51, pp. 165-170.

Emerson, C., and Ham, I., 1982; *An automated Coding and Process Planning System Using a DEC. PDP-10.* Computers and Industrial Engineering, Vol. 6, No. 2, pp. 159-168.

Eversheim, W., and Fuchs, H., 1979; *Integrated Generation of Drawings.* Process Plans and NC Taps, SME Technical Paper No. MS79-178 (Dearborn, MI: Society of Manufacturing Engineers).

Eversheim, W., Fuchs, H., and Zons, K. H., 1980; *Automatic Process Planning with Regard to Production by Application of the System AUTAP for Control Problems.* Computer Graphics in Manufacturing Systems, 12th CIRP International Seminar on Manufacturing Systems, Belgrade.

Eversheim, W., 1991; *Structured Modelling of Manufacturing Processes as NC-Data Preparation.* Annals of the CIRP, Vol. 40/1/1991.

Evre, A. H., and Cals, H .J. J., 1986; *XPLANE, A Generative Computer Aided Process Planning System for Part Manufacturing.* Annals of the CIRP, Vol. 35/1/1986, pp. 325-329.

Evre, A. H. van 't, and Houten, F. J. A. M. van, 1988; *PART, a parallel approach to computer aided process planning.* 4th Int. Conf. on Computer Aided Engineering (CAPE), Edinburgh, Proceedings, pp. 281-288.

Faught, W. S., 1986; *Application of AI in Engineering.* IEEE Computer, Vol. 19, No. 7, pp. 17-27, July, 1986.

Feigenbaum, E. A., 1982; *Knowledge Engineering for the 1980's.* Department of Computer Science, Standford University, Stanford California.

Forsyth R., 1984; *Expert systems principles and case studies.* Chapman and Hall computing.

Francis, R. L., and White, J. A., 1974; *Facility Layout and Location: An Analytical Approach.* Prentice Hall.

Frazier, G. V., and Gaither, N., 1991; *Seed selection procedures for cell formation heuristics.* International Journal of Production Research, 29(11), pp. 2227-2237.

Fu, K. S., 1980; *Recent developments in Pattern Recognition.* IEEE Transactions on Computers, Vol. C-29, No. 10, October 1980.

Gallagher, C.C., and Knight, W.A. 1973; *Group Technology.* Butterworth & Co. Ltd.

Gallagher, C.C., and Knight, W.A. 1986; *Group Technology Production Methods in Manufacture.* Ellis Horwood.

Gandhi, A., and Myklebust, A., 1989; *A natural language approach to feature based modelling.* Mobil Research and Development Corporation, Princeton, New Jersey.

GENERIS 1991; Release 3.1, User Guide; Instrumatic Data Systems Ltd. Unit 4, First Avenue, Globe Park, Marlow, Bucks, SL7 1YA, USA.

Gettelman, K., 1971; *Organisation Production for Parts — Not Processes.* Modern Machine Shop, Nov. pp. 50-61.

Gindy, N. N. Z., 1989; *A hierarchical structure for form features.* International Journal of Production Research, Vol. 27, No. 12, pp. 2089-2103.

Gindy, N. N. Z., and Ratchev, T. M., 1991; *Product and machine tools data models for computer aided process planning systems.* proceedings of the IFIP TC5 Forth Conference on Computer Applications and Engineering- CAPE'91, Bordeaux, France, 10-12 September, 1991.

Gindy, N. N. Z., Huang, X., and Ratchev, T. M., 1993; *Feature-based component model for computer-aided process planning systems.* International Journal of Computer Integrated Manufacturing, Vol. 6 Numbers 1 & 2 pp. 20-26.

Glusti, F., and Santochi, M., 1989; *KAPLAN: A Knowledge-Based Approach to Process Planning of Rotational Parts.* Annals of the CIRP, Vol. 38/1/1989.

Gombinski, J., 1963; *Classification and Coding and their Industrial Applications.* Engineering Materials and Design Conference, 25 Nov. Paper No. 16.

Gombinski, J., 1969; *Fundamental Aspect of Component Classification.* Annals of the CIRP, Vol. 17, pp. 367-375.

Gongaware, T. A., and Ham, I., 1981; *Cluster analysis applications for group technology manufacturing systems.* Manufacturing Engineering Transactions, pp. 503-508.

Graves, G. R., Balaji, Y., and Parks, C. M., 1988; *An interface architecture for CAD/CAPP integration using knowledge-based systems and feature recognition algorithms.* International Journal of Computer Integrated Manufacturing, Vol. 1, No. 2, pp. 89-100.

Grayer, A. R., 1976; *A Computer link between Design and Manufacturing.* PhD Dissertation, University of Cambridge.

Groover, M. P., and Zimmer, E. W., 1984; *CAD/CAM: computer-aided design and manufacturing.* Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632.

Groover, M. P., 1987; *Automation, Production Systems, and Computer Integrated Manufacturing.* Prentice-Hall, Inc.

Haan, R. A., 1977; *Group Technology Coding and Classification Applied to NC Part Programming.* Proceedings of the 14th Numerical Control Society Annual Meeting and Technical Conference, Pittsburgh, Pennsylvania.

Halvei, G. and Weill, R. 1985; *Influence of Manufacturing Tolerances on Fixturing of Machined Parts in Processes Planning Systems.* First CIRP Working Seminar on CAPP, 1985.

Ham, I., 1980; *Current Trends and Future Prospects of Group Technology Applications Related to Integrated Computer Aided Manufacturing.* Proceedings of Conference on International Manufacturing Engineering, 1980, published by the Institution of Engineers, Australia, August 1980.

Ham, I., Hitomi, K., and Yoshika, T., 1985; *Group Technology: Applications to Production Management.* Kluwer-Nijhoff, Publishing, USA.

Ham, I., Gonclaves, E., and Han, C., 1988a; *An integrated approach to group technology part family data base design based on AI techniques.* Annals of the CIRP, Vol. 37/1/1988, pp. 433-437.

Ham, I., and Lu, S. C. Y., 1988; *Computer-Aided Process Planning: The Present and the Future.* Annals of the CIRP , Vol. 37/2/1988, pp. 591-601.

Ham I., and Stephen C.Y., 1989; *Machine learning techniques for Group Technology applications.* Annals of the CIRP, Vol. 38/1/1989.

Han, C., Ham, I., ; *Multiobjective Cluster Analysis for Part Family Formations.* Journal of Manufacturing Systems, Vol. 5, No. 4.

Harhalakis, G., Nagi, R., and Proth, J. M., 1990; *An efficient heuristic in manufacturing cell formation for group technology applications.* International Journal of Production Research, 28 (1) , pp. 185-198.

Harmon, P., and King, D., 1985; *Expert Systems.* John Wiley & Sons, Inc.

Hayes, C. C., 1987; *Observing Machinist's Planning Methods: Using Goal Interactions to Guide Planning.* Proceedings of the Ninth Annual Cognitive Science Meeting (July 1987), pp. 952-958.

Hayes, C. C., 1988; *Automated Process Planning System for Prismatic Parts, Exper Systems: Strategies and Solutions in Manufacturing Design and Planning*. Edited by Andrew Kusiak.

Haywood, W., 1990; *CIM: Revolution in Progress*. Proceedings of Final IIASA Conference on Computer Integrated Manufacturing: Technology, Organisation and People in Transition, Laxenburg, July 1990.

Hegland, D. E., 1981; *Out in Front with CAD/CAM at Lockheed-Georgia*. Production Engineering.

Henderson, M. R., 1984a; *Extraction of Feature Information From Three-Dimensional CAD Data*. PhD Dissertation, Purdue University.

Henderson, M. R., 1984b; *Feature Recognition in Geometric Modelling*. Proc. CAM-I 13th meeting, pp 5.1-5.12.

Henderson, M. R., Chuang, S. H., Ganu, P. and Gavankar, P., 1990; *Graph-based feature extraction*. Arizona State University.

Henderson M.R., and Chang G. J., 1988; *FRAPP: Automated Feature Recognition and Process Planning from Solid Model Data*. Computers in Engineering, ASME, Vol. 1.

Hinde, C. J., and Alton, N. G., 1987; *Process planning using a truth maintenance system*. Proceedings of the 2nd UMIST/ACME Workshop on Advanced Research in Computer Aided Manufacturing.

Hinduja, S., and Huang, H., 1989; *OP-PLAN: an automated operation planning system for turned components*. Journal of Engineering Manufacture, Vol. 203, No. B3.

Hitomi, K., 1979; *Manufacturing Systems Engineering*. Taylor & Francis Ltd., London.

Hordeski, M. F., 1986; *CAD/CAM Techniques*. Reston Publishing Company Inc., a Prentice-Hall Company, Reston Virginia.

Horrington, J., 1973; *Computer Integrating Manufacturing*. Industrial Press, New York.

Houtzeel, A., 1981; *Integrating CAD/CAM through Group Technology*. Proceedings of the 18th Numerical Control Society Annual Meeting and Technical Conference, Dallas, Texas, USA.

Hummel, K. E., and Brooks, S. L. 1986; *Symbolic Representations of Manufacturing Features for Automated Process Planning Systems.* Proceedings of Winter Annual Meeting of the ASME, Anaheim, CA, December 1986, pp. 233-243.

Hyde, W. F., 1981; *Improving Productivity by Classification, Coding, and Data Base Standardization: The Key to Maximising CAD/CAM and Group Technology.* Marcel Dekker.

Hyer, N. L., and King, R. E., 1984; *Group Technology at Work.* Society of Manufacturing Engineers.

Hyer, N. L., 1984a; *The potential of group technology for US. manufacturing.* Journal of Operations Management, 4, (3), pp. 183-202.

Hyer, N. L., and Wemmerlöv, U., 1984b; *Group technology and Productivity.* Harvard Business Review (July/August 1984).

Hyer, N. L., and Wemmerlöv, U., 1985; *Group Technology-Oriented Coding Schemes: Structures, Application and Implementation.* Production and Inventory Management, Vol. 26, No. 2, pp. 55-78.

Hyer, N. L., (Editor) 1987; *Capabilities of Group Technology.* Michigan, USA.

Hyer, N. L., and Wemmerlöv, U., 1989; *Group Technology in the US manufacturing industry: a survey of current practices.* International Journal of Production Research, 27 (8), pp. 1287-1304.

Imamura, S., Kojima, T., Sekiguchi, H., and Inoui, K., 1988; *A Study on the Object Oriented Product Model — Representation of Geometry and Dimension.* Annals of the CIRP, Vol. 37/1/1988, pp. 127-130.

Inui, M., *et al.* 1987a; *Automatic process planning for sheet metal parts with bending simulation.* Proceedings of ASME-WAM.

Inui, M., *et al.* 1987b; *Extended process planning capabilities with dynamic manipulation of product models.* 19th CIRP International Seminar on Manufacturing Systems, Penn. State, USA, 1-2 June.

Iwata, K., Kakino, Y., Oba, F., and Sugimura, N., 1980; *Development of non-part family type computer aided production planning system CIMS/PRO.* Advanced Manufacturing Technology, edited by P. Blake. (Amsterdam: North-Holland Publishing Company, pp. 177-184.

Iwata, K., and Sugimura, N., 1986; *Development of Product model for Design and Manufacturing of Machine Products.* North American Metal Research Conference 14th Proceedings (NAMRC).

Iwata, K. and Sugimura, N., 1987; *An Integrated CAD/CAPP System with "Know-hows" on Machining Accuracies of Parts.* Transactions of ASME, Vol. 109, pp. 128-133, May 1987.

Ivanov, E. K., 1968; *Group Production Organisation and Technology.* Business Publications Limited.

Jackson, P., 1986; *Introduction to Expert Systems.* Addison-Wesley.

Jackson, P., 1990; *Introduction to Expert Systems.* 2nd Edn., Addison-Wesley, Reading, MA.

Johnson, P. E., 1986; *Early applications get user approval.* Expert Systems User, November 1986.

Joneja, A., and Chang, T. C., 1991; *Search anatomy in feature-based automated process planning.* Journal of Design and Manufacturing, Vol. 1, pp. 7-15.

Joshi, S., Chang, T. C., and Liu, C. R., 1986; *An Automated Process Planning system Structure Based on AI.* Computers in Engineering, Vol. 1, Proceedings of the 1986 ASME Intenational Computers in Engineering Conference and Exhibition, July 20-24, Chicago, Illinois, USA.

Joshi, S., Narenda, N. V., and Chang, T. C., 1988a; *Expert process planning system with solid model interface.* International Journal of Production Research, Vol. 26, No. 5, pp. 863-885.

Joshi, S., and Chang, T. C., 1988; *Graph-based heuristicsfor recognition of machined features from a 3-D solid model.* Report TR-ERC 88-4, April, Engineering Research Centre for Manufacturing Systems, Purdue University, Indiana.

Kandel A., 1982; *Fuzzy Techniques in Pattern Recognition.* John Wiley & Sons.

Kanumury, M., Shah, J., and Chang, T. C., 1988; *An Automatic Process Planning System for QTC - An Integrated CAD/CAM System.* Purdue Engineering Research Centre for Intelligent Manufacturing Systems.

Kaufmann, A., 1975; *Introduction to the theory of fuzzy subsets.* Academic Press.

Kaufman, L., 1990; *Finding Groups in Data: An Introduction to Cluster Analysis*, John Wiley & Sons, Inc.

Kazuaki and Lwata, 1987; *Knowledge-based Flexible Part Classification System for CAD/CAM.* Annals of the CIRP, Vol. 36/1/1987.

Kekre, S., 1987; *Performance of a Manufacturing Cell with Increased Product Mix.* IEEE Transactions, Vol. 19, No. 3, pp. 329-339.

Kernighan . W., and Ritchie, 1981; *Introduction to the 'C' programming language.* Prentice-Hall, London.

Kim, J. Y., Grady, P. and Young, R. E., 1991; *Feature taxonomies for rotational parts: a review and proposed taxonomies.* Int. J. Computer Integrated Manufacturing, Vol. 4, No. 6, pp. 341-350.

King, J. R., 1980; *Machine-component grouping in production flow analysis: an approach using a rank order clustering algorithm.* International Journal of Production Research, 18, pp. 213-237.

King, J. R., and Nakornchai, V., 1982; *Machine-component grouping formation in group technology: review and extension.* International Journal of Production Research, 20, pp. 117-133.

Kiprianou, L. K., 1980; *Shape Classification in Computer-Aided Design.* PhD Dissertation, Cambridge University.

Kimura, F., *et al.* 1984; *A Study on Product Modelling for Integration of CAD/CAM, in Kochan, D., Integration of CAD/CAM.* North-Holland.

Kotler, R. A., 1980a; *Computerised Process Planning — Part1.* Army Man Tech Journal, Vol. 4, No. 5, pp. 20-29.

Kotler, R. A., 1980b; *Computerised Process Planning — Part2.* Army Man Tech Journal, Vol. 4, No. 4, pp. 28-36.

Kramer, T. R., and Jun, J., 1986; *Software for an Automated Machining Workstation.* Proceedings of the Third Biannual International Machine Tool Technical Conference, Session 12.

Kumar, K. R., Kusiak, A., and Vannelli, A., 1985; *Grouping of parts and components in flexible manufacturing system.* European Journal of Operational Research, 24, pp. 387-397.
Kumar, K. R., and Vannelli, A., 1987; *Strategic subcontracting for efficient disaggregated manufacturing.* International Journal of Production Research, 25 (12), pp. 1715-1728.

Kumara, S. R. T., Joshi, S., Kashyap, R. L., Moodie, C. L., and Chang, T. C., 1986; *Expert systems in industrial engineering.* International Journal of Production Research, Vol. 24, No. 5, pp. 1107-1125.

Kusiak, A., 1985; *The part families problem in flexible manufacturing systems.* Annals of Operations Research, 3, pp. 279- 300.

Kusiak, A., 1987; *The generalised group technology concept.* International Journal of Production Research, 25 (4), pp. 561-569.

Kusiak, A., and Chow, W. S., 1987; *Efficient solving of the group technology problem.* Journal of Manufacturing systems, 6 (2), pp. 117-24.

Kusiak, A., and Heragu, S. S., 1987; *Group technology.* Computers in Industries, 9, pp. 87-91.

Kusiak, A., 1988; *Knowledge-based group technology.* Expert systems, Kusiak, A., (ed.), SME, pp. 259-276.

Kusiak, A., and Cho, M., 1992; *Similarity coefficient algorithms for solving the group technology problem.* International Journal of Production Research, 30 (11), pp. 2633-2646.

Kyo Lee, J. W., and Jhang, M. L., 1989; *Pattern Recognition and Process Planning prismatic workpieces by Knowledge Based Approach.* Annals of the CIRP, Vol. 38/1/1989.

Kyprianou L. K., 1976; *Shape classification in computer aided design.* PhD Dissertation, University of Cambridge, Cambridge, England.

Lee, Y., and Fu, K., 1987; *Machine Understanding of CSG: Extraction ad Unification of Manufacturing Features.* IEEE, Computers Graphics Applications, January, 1987.
Lee, H., and Garcia-Diaz, A., 1993; *A network flow approach to solve clustering problems in group technology.* International Journal of Production Research, 31 (3), pp. 603-612.

Li, J., Ding, Z., and Lei, W., 1986; *Fuzzy Cluster Analysis and Fuzzy Pattern Recognition Methods for Formation of Part Families.* Proceedings of 14th North American Manufacturing Research Conference.

Li, J. K., and Zhang, C., 1989; *Operational dimensions and tolerances calculation in CAPP systems for precision manufacturing.* Annals of the CIRP, Vol. 38/1/1989, pp. 403-406.

Lim, B. S., and Knight, J. A. G., 1987; *A foundation for the knowledge-base computer integrated manufacturing system.* Artificial Intelligence, Vol. 2, pp. 11-22.

Link, C. H., 1978; *CAM-I Automated Process Planning.* SME Technical Paper No. MS78-213 (Dearborn, MI: Society of Manufacturing Engineers).

Lissaman, A. J., and Martin, S. J., 1982; *Principles of Engineering Production.* (2nd Edition) Hodder and Stonghton.

Logendran, R., 1992; *A model for duplicating bottleneck machines in the presence of budgetary limitations in cellular manufacturing.* International Journal of Production Research, 30 (3), pp. 694-683.

Luby, S. C., Dixon, J. R., and Simmons, M. K., 1986; *Creating and using a features database.* Computers in Mechanical Engineering, Vol. 5, No. 3, November 1986, pp. 25-33.

Mantyla, M., 1988; *An Introduction to Solid Modelling.* Computer Science Press.

Marks, P. C., 1987; *Expert System-Based Generative Process Planning.* Conference Proceedings Nov. 9-12, Detroit, Michigan, Autofact 87.

Maropoulos, P. G., and Hinduja, S., 1990; *Automatic Tool Selection for Finish Turning.* Proc. Instn. Mech. Eng., Vol. 204, pp. 43-51.

Matsushima, K. , Okada, N., and Sata, T., 1982; *The integration of CAD and CAM by application of Artificial-Intelligence techniques.* Annals of the CIRP, Vol. 31/1/1982, pp. 329-332.

McAuley, J., 1972; *Machine grouping for efficient production.* The Production Engineering, 52, pp. 53-57.

McCormick, W. T., Schweitzer, P. J., and White, T. E., 1972; *Problem decomposition and data recognition by a clustering technique.* Operation Research, 20, pp. 993-1008.

Medland, A. J., and Mullineux, G., 1988; *Principle of CAD.* Chapman and Hall Publication.

Melkote, R. and Taylor, D. L., 1988; *An Implementation of Rule Based Selection of Milling Cutters, Feed Rates and Spindle Speed.* ASME International Computers in Engineering Conference.

Milacic, V. R., 1985; *SAPT— expert system for manufacturing process planning. Proceedings of the Symposium on Computer Aided/Intelligent Process Planning.* The Winter Annual Meeting of the ASME, Miami Beach, Florida, U.S.A., November 17-22, pp. 43-54.

Milacic, V. R., 1986; *How to build expert systems.* Annals of the CIRP, Vol. 35/2/1986, pp. 445-450.

Millar, G. W., 1984; *Group Technology - Framework for CAD/CAM Integration.* AUTOFACT 1984, California, USA.

Minsky, M., 1968; *Matter, Mind and Models in Semantic Information Processing.* MIT Press.

Mitrofanov, S. P., 1966; *Scientific Principles of Group Technology.* (Russian text published in 1959), Translated into English by E. Harris, National lending Library for Science and Technology, UK..

Mosier, C. T., 1989; *An experiment investigating the application of clustering procedures and similarity coefficients to the GT machine cell formation problem.* International Journal of Production Research, 27 (10), pp. 1811-35.

Mouleeswaran, C. B., 1984; *PROPLAN: A knowledge based expert system for manufacturing process planning.* Master's Thesis, University of Illionois at Chicago, USA.

Mullineux , G., 1986; CAD: *Computational Concepts and Methods.* Kogan Page, London.

Muthsam, H., and Mayer, C., 1990; *An expert system for process planning of prismatic workpieces.* Proc. Ist. Conf. Artificial Intelligence and Expert Systems in Manufacturing, 211-220, March 1990.

Nandkeolyar, U., and Christy, D. P., 1992; *An investigation of the effect of machine flexibility and number of part families on system performance.* International Journal of Production Research, 30 (3), pp. 513-26.

Nau, D. S., and Chang, T. C., 1980; *Prospects for process selection using artificial intelligence.* CASE, SME, pp. 214-228.

Nau, D. S., and Chang, T. C., 1983; *Prospects for process selection using artificial intelligence.* Computers in Industry, pp. 253-263.

Nau, D. S., and Chang, T. C., 1985; *A knowledge based approach to generative process planning.* Proceedings of the Symposium on Computer Aided/Intelligent Process Planning, pp. 65-72, The Winter Annual Meeting of the ASME, Miami Beach, Florida, U.S.A., November 17-22.

Nau, D. S., and Chang, T. C., 1985b; *Hierarchical Presentation of Problem-solving Knowledge in a Frame-based Process Planning System.* Journal of Intelligent Systems, Vol. 1, No. 1, pp. 29-44.

Nau, D. S., and Gray, M., 1986; *SIPS: An Application of Hierarchical Knowledge Clustering to Process Planning.* Symposium on Integrated and Intelligent Manufacturing, ASME Winter Annual Meeting, Anabeim, California, Dec. 1986, pp. 219-225.

Nau, D. S., and Gray, M., 1987; *An Application of Hierarchical Knowledge Clustering to Process Planning.* ASME-AWM.

Nau, D. S., 1987b; *Automated Process Planning Using Hierarchical Abstraction.* Texas Instrruments Call for Papers on AI for Industrial Automation (July 1987).

Niebel, B. W., Draper, A. B., and Wysk, R. A., 1989; *Modern Manufacturing Process Engineering.* McGraw-Hill, USA.

Nilsson, N. J., 1980; *Principles of Artificial Intelligence.* Tioga Publishing Company, Palo Alto, Calif.

Nolen, J., 1989; *Computer-Automated Process Planning for World-Class Manufacturing.* Marcel Dekker.

Opitz, H., 1970; *A Classification to Describe Workpieces.* Pergamon Press.

Opitz, H., and Wiendahl, H. P., 1971; *Group technology and manufacturing systems for small and medium quantity production.* International Journal of Production Research, Vol. 9, No. 1, pp. 181-203.

Patel, R. M., and Mcleod, A. J., 1988; *Engineering feature description in mechanical engineering design.* Computer Aided Engineering Journal, October 1988.

Peknenik, J., and Grum, J., 1980; *A new development to part classification for Group Technology.* International Conference on Manufacturing Engineering, Melbourne, August 1980.

PDES 1988; *Product Data Exchange Specification*, First Working Draft, NISTIR 88-4004, National Institute of Standards and Technology. National Engineering Laboratory, Gaithersburg, MD 20899, Dec. 1988.

Perotti, G., Tornincasa, S., and Oberto, G., 1991; *Semantic Techniques for Representation and Identification of Part Families.* Annals of the CIRP, Vol. 40/1/1991.

Petropoulos, P. G., 1973; *Optimal Selection of Machining Rate Variables by Geometric Programming.* International Journal of Production Research, Vol. 4, pp. 305-314.

Phillips R. H., and Gomayel, J. I., 1980; *Computerised Process Planning for Metal Cutting.* Proceedings of the Eighth North American Manufacturing Research Conference.

Phillips, R. H., and Mouleeswaran, C. B., 1985; *A Knowlegde-Based Approach to Generative Process Planning.* Proceedings of CASA/AME Autofact'85 Conference.

Pratt, M. T., 1984; *Solid Modelling and Interface Between Design and Manufacture.* IEEE Trans. Computer Graphics and Applications, 4 (7), pp. 52-59.

Pratt, M. J., and Wilson, P. R., 1985; *Requirements for support of form features in a solid modelling system.* CAM-I, R-85-ASPP-01.

Pratt, M. J., 1993; *Applications of feature recognition in the product life-cycle.* International Journal of Computer Integrated Manufacturing, Vol. 6 Numbers 1 & 2, pp. 13-19.

Pullen, R. D., 1976; *A survey of cellular manufacturing cells.* Production Engineer, 55, pp. 451-454.

Purcheck, G. F. K., 1975a; *A linear programming method for the combinatorial grouping of an incomplete power set.* Journal of Cybernetics, 5, 51.

Purcheck, G. F. K., 1975; *A mathematical classification as a basis for the design of group-technology production cells.* The Production Engineer, pp. 35-48.

Purcheck, G. F. K., 1985; *Machine-component group formation: an heuristic method for flexible production cells and flexible manufacturing systems.* International Journal of Production Research, 23, pp. 911-943.

Rajagopalan, R. and Batra, J. L., 1975; *Design of cellular production systems: a graph-theoretic approach.* International Journal of Production Research, Vol. 13, No. 6, pp. 567-579.

Rajagopalan, R. and Batra, J. L., 1975; *Design of cellular production systems: a graph-theoretic approach.* International Journal of Production Research, Vol. 13, No. 6, pp. 567-579.

Rajamani, D., Singh, N., and Aneja, Y. P., 1990; *Integrated design of cellular manufacturing systems in the presence of alternative process plans.* International Journal of Production Research, Vol. 28, No. 6, pp. 1541-1554.

Rajamani, D., Singh, N., and Aneja, Y. P., 1992; *A model for cell formation in manufacturing systems with sequence dependence.* International Journal of Production Research, Vol. 30, No. 6, pp. 1227-1235.

Ranky, P. G., 1983; *The Design and Operation of FMS.* IFS (Publications) Ltd., UK.

Ranky, P. G., 1986; *Computer Integrated Manufacturing.* Prentice-Hall International, UK, Ltd.
Ranky, P. G., 1990; *Flexible Manufacturing Cells and Systems in CIM.* Biddles Limited, UK.

Ranson, G. M., 1972; *Group Technology: A foundation for better total company operation.* McGraw-Hill Book Company (UK) Ltd.

Rayson, P. T., 1985; *A review of expert systems principles and their role in manufacturing systems.* Robotica, 3, 279.

Requicha, A. A. G., and Voelcker, H. B., 1982; *Solid Modelling: A Historical Summary and Contemporary Assessment.* Computer Graphics and Applications, IEEE.

Requicha, A. A. G., and Voelcker, H. B., 1983 ; *Solid Modelling : Current Status and Research Directions.* Computer Graphics and Applications, IEEE.

Rich, E., 1983; *Artificial Intelligence*. McGraw-Hill, New Yark.

Romesburg, H. C., 1984; *Cluster Analysis for Researchers*. Lifetime Learning Publications, Belmount,California 94002, a division of Wadsworth, Inc.

Roy, U. and Liu, C. R. 1988; *An Expert System Approach to the Machine Tool Sequencing and Machining Sequencing Problems in Computer Aided Process Planning*. School of Industrial Engineering, Purdue University.

Roth, H. P., Zeh, K. P., and Muthsam, H., 1988; *Artificial Intelligence in Future Manufacturing*. Proc. 7th Int. Conf. on Flexible Manufacturing Systems, pp. 249-263.

Sadowski, R., 1984; *Computer-integrated Manufacturing Systems with Applying Systems Approach to the Factory of the Future*. Industrial Engineering, Vol. 16 (1), pp 35-40.

Sakurai H., and Gassard D.C., 1988; *Shape Feature Recognition from 3D Solid models*. Computers in Engineering, 1988, ASME, Vol. 1.

Schaffer, G., 1980; GT via Automated Process Planning. American Machinist, May 1980, pp. 119-122.

Schaffer, G. H., 1981; *Implementing CIM*. American Machinist Special Report 736, pp. 153-154.

Schlechtendahl, E. G., 1988; *Computer Aided Design*. Marcel Decker.

Schonberger, R. J., 1983; *Plant Layout Becomes Product Oriented with Cellular Just In Time Production Concepts*. Industrial Engineering, Vol. 15.

Schutzer, D., 1987; *Artificial Intelligence*. Van Nostrand Reinhold Company, New Yark.

Seifoddini, H., and Wolfe, P. M., 1986; *Application of the similarity coefficient method in group technology*. IIE Transactions, 18 (3), pp. 271-277.

Seifoddini, H., and Wolfe, P. M., 1987; *Selection of a threshold value based on material handling cost in machine-component grouping*. IIE Transactions, 19 (3), pp. 266-270.

Seifoddini, H. 1989a; *A note on the similarity coefficient method and the problem of improper machine assignment in group technology applications.* International Journal of Production Research, 27 (7), pp.1161-1165.

Seifoddini, H., 1989b; *Duplication process in machine cells formation in group technology.* IIE Transactions, 21 (4), pp. 382-388.

Seifoddini, H., 1989c; *Single linkage versus average linkage clustering in machine cells formation applications.* Computers and Industrial Engineering, 16 (3), pp. 419-426.

Seifoddini, H., 1990; *A probabilistic model for machine cell formation.* Journal of Manufacturing Systems, 9 (1), pp. 69-75.

Shaffer, G. H., 1986; *Artificial Intelligence: a tool for smart manufacturing.* American Mechinist, Vol. 130, No. 8.

Shafer, S. M., Kern, G. M., and Wei, J. C., 1992; *A Mathematical Programming Approach for dealing with Exceptional Elements.* International Journal of Production Research, Vol 30, No 5, pp. 1029-1036.

Shafer, S. M., and Rogers, D. F., 1993; *Similarity and distance measures for cellular manufacturing. Part I. A survey.* International Journal of Production Research, Vol 31, No 5, pp. 1133 - 1142.

Shah, J. J., Sreevalsan, P., Rogers, M. T., and Mathew, A., 1988; *Current Status of Feature Technology.* Report R-88-GM-04.1, CAM-I Inc. Arlington Texas, USA.

Shah, J. J., Rogers, M. T., 1988b; *Functional Requirements and Conceptual Design of the Feature-Based Modelling System.* Computer-Aided Engineering Journal, February, pp. 9-15.

Shah, J. J., Rogers, M. T., 1988c; *Geometric Modelling for Automated Manufacturing.* 16th North American Manufacturing Research Conf. Proceedings, May 24-27, University of Illinois Urbana, Illinois.

Shah, J. J., Rogers, M. T., 1990; *Feature based modelling shell: design and implementation.* Computers in Engineering Journal, Jan. 1990.

Shaw, N. K., Bloor, M. S. and Pennington, A. D., 1989; *Product Data Models.* Research in Engineering Design, Vol. 1, No. 1, pp. 43-50.

Shiko, G., 1992; *A process planning-oriented approach to part family formation problem in group technology applications.* International Journal of Production Research., 1992, Vol. 30, No. 8, pp. 1739-1752.

Simon, H. A., 1969; *The Sciences of Artificial.* MIT Press, Cambridge, MA.

Snead, C. S., 1989; *Group Technology: Foundation for Competitive Manufacturing.* Van Nostrand Reinhold, London, England.

Solaja, V. B., and Vrosevic, S. M. 1973; *The Method of Hypothetical Group Technology Production Lines.* Annals of the CIRP, Vol. 22/1/1973.

Spur, G., et. al. 1978; *CAPPSY— A Dialogue System for Computer-Aided Manufacturing Planning.* Proceedings, 19th MIDR Conference.

Spur, G., et. al. 1986; *Product Models as Basis for Integrated Design and Manufacturing.* Int. J. Mach. Tool Des. Res., Vol. 26, No. 2, pp. 171-178.

Srinivasan, G., Narendran, T. T., and Mahadevan, B., 1990; *An assignment model for the part families problem in group technology.* International Journal of Production Research, 28 (1), pp. 145-152.

Stecke, K. E., 1983; *Formulation and solution of nonlinear integer production planning problems for flexible manufacturing systems.* Management Science, 29, pp. 273-288.

Subramanyam , S., and Lu, S. C-Y., 1988; *An Integrated AI/OR Approach To Operation Planning Using Process Performance Models.* 16 NAMRC North Americam Manufacturing Research Conference Proceedings, May 24-27, 1988, University of Illinois Urbana, Illinois.

Sutherland, I. E., 1963; *Sketch Pad, A Man Machine Graphic Communication System AFIPS.* SJCC 23, pp. 329-346.

Tabucanon, M . T., and Ojha, R., 1987; *ICRMA — a heuristic approach for intercell flow reduction in cellular manufacturing.* Material Flow, 4 (18), pp. 189-197.

Tam, K. Y. *An operation sequence based similarity coefficient for part families formations.* Journal of Manufacturing Systems, 9 (1), pp. 55-68.

Tanner, J. P., 1985; *Manufacturing Engineering: An Introduction to the Basic Functions.* Marcel Dekker Inc, New York.

Tempelhof, K., 1980; *A System of Computer Aided Process Planning for Machining Parts.* (New Yark: Elsevier Publishers).

Thorley, R. H., 1971; *Group Technology — A complete manufacturing system.* Inaugural Lecture, 14 Oct. 1971, University of Aston, Birmingham.

Tilley, S., 1992; *Integration of CAD/CAM and Production Control for Sheet Metal Components Manufacturing.* Annals of the CIRP , Vol. 41/1/92, pp. 177-180.

Tornincasa, S., and Zompi, A., 1991; *Feature-based object oriented approach in automated manufacturing.* Proc. of "Computer Integrated Manufacturing" ICCIM, 91 Manufacturing Enterprises of 21st cantury, organised by "GINTIC Institute of CIM" Nuayang Technical University, Singapore, 2-4 October, 1991.

Tsang, J. P., 1987; *The Propel Process Planner.* 19th CIRP International Seminar on Manufacturing Systems, Penn. State University, June 1987, pp. 71-77.

Tulkoff, J., 1981; *Lockheed's GENPLAN.* Proceedings of 18th Numerical Control Society Annual Meeting and Technical Conference, pp. 417-421, Dallas, Texas, USA.

Unger, M. B., and Ray, S. R., 1988; *Feature-Based Process Planning in the AMRF.* Computers in Engineering, ASME, Vol. 1.

Vakharia, A. J., 1986; *Methods of Cell Formation in Group Technology: A Framework for Evaluation.* Journal of Operations Management—Special Combined Issue, Vol. 6, No. 3, pp. 257-271.

Vakharia, A. J., and Wemmerlöv, U., 1990; *Designing a cellular manufacturing system: a materials flow approach based on operation sequences.* IIE Transactions, 22 (1), pp. 84-97.

Vannelli, A., and Kumar, R., 1986; *A method for finding minimal bottle-neck cells for grouping part-machine families.* International Journal of Production Research, 24 ( ), pp. 387-400.

Vernadat, F., 1984; *Communications: a key requirement in Computer-integrated Manufacturing.* Second Annual Phoenix Conference on Computers and Communications, Phoenix, Arizona, USA, March 1984, pp. 193-198.

Waghodekar, P. H., and Sahu, S., 1984; *Machine-component cell formation in group technology: MACE.* International Journal of Production Research, 22 (6), pp. 937-948.

Waldman, H., 1983; *Process Planning at Sikorsky.* CAD/CAM Technology (Summer 1983).

Walker, H., and West, R., 1990 ; *Integrating CAD and CAM through the Feature Based Methodology for Design and Manufacture.* Proceedings Matador Conference, 1990.

Wang, H. P., and Wysk, R. A., 1986; *An expert system for machining data selection.* Computers and Industrial Engineering. 10, No. 2, pp. 99-107.

Wang, H. P., and Wysk, R. A., 1987; *Intelligent Reasoning for process planning.* Computer in Industry, pp. 293-309, June/July, 1987.

Wang, H. P., and Wysk, R. A., 1988a; *AIMSI —a prelude to new generation of integrated CAD/CAM systems.* International Journal of Production Research, 1988, Vol 26, pp. 119-31.

Wang, H. P., and Wysk, R. A., 1988b; *A knowledge-based approach for automated process planning.* International Journal of Production Research, 1988, Vol 26, No 6, pp. 999-1014.

Wang, H. P., and Li, G. K., 1991; *Computer-Aided Process Planning.* Elsevier Science Publishers, B. V.

Warren G. M. H., and Moodie, C. L., 1993; *Cellular Manufacturing.* Report TAP 930104, Technical Assistant Program (TAP), Purdue University, January 4, 1993.

Waterman, D. A., 1986; *A Guide to Expert Systems.* Addison-Wesley, Reading, Massachusetts.

Weill, R., 1988; *Integrating Dimensions and Tolerancing in computer-aided process planning.* Robotics and Computer-Integrated Manufacturing, Vol. 4, pp. 41-48.

Wemmerlöv, U., and Hyer, N. L., 1986; *Procedures for the part family/machine group identification problem in cellular manufacturing.* Journal of Operations Management, Vol. 6, No. 2, pp. 125-147.

Wemmerlöv, U., and Hyer, N. L., 1987; *Research issues in cellular manufacturing.* International Journal of Production Research., Vol 25, pp. 413-431.

Wemmerlöv, U., 1989; *Cellular Manufacturing in the US Industry: A survey of users.* International Journal of Production Research., Vol 27, pp. 1511-1530.

Wild, R., 1985; *Essentials of Production and Operations Management.* Second Edition, Holt, Rinehart and Winston Ltd: Eastbourne, East Sussex, BN21 3UN.

Willis, D., Donaldson, I. A., and Murray, J. L., 1988; *A Knowledge Based System for Process Planning based on solid modeller.* Heriot-Watt University.

Winston, P. H., 1984; *Artificial Intelligence.* Addison-Wesley, Reading, MA.

Wolfe, P. M., 1985; *Computer-Aided Process Planning: is link between CAD and CAM.* Industrial Engineer, pp. 72-77, August 1985.

Woo, T. C., 1977; *Computer aided recognition of volumetric designs.* D. McPherson (Editor), Advances in Computer aided Manufacturing (Amsterdam: North-Holland Publishing Co.), pp. 121-35.

Woo, T. C., 1982; *Feature Extraction by Volume Decomposition.* Proc. Con. on CAD/CAM Tech. in Mech. Eng., Cambridge, USA.

Woodwark, J. R., 1988; *Shape Models in Computer-Integrated Manufacture: A Review.* UKSC Report 183, IBM-UK, 01/1988.

Wu, H. L., Venugopal, R., and Barash, M. M., 1989; *Design of a cellular manufacturing system: a syntactic pattern recognition approach.* Journal of Manufacturing Systems, 5 (2), pp. 81-87.

Wysk, R. A., 1977; *An automated process planning and selection program: APPAS.* PhD Thesis, Purdue University, West Lafayette, Indiana, U.S.A.

Wysk, R. A., Miller, D., and Davis, R., 1977b; *The Integration of Process Selection and Machine Requirements Planning.* Proceedings, AIIE Systems Engineering Conference.

Wysk, R. A., Barash, M. Y., and Moodie, C. M., 1980; *Unit Machining Operations*. Trans. of ASME, Journal of Eng. for Industry, Vol. 102, Nov. 1980.

Wysk, R. A., 1985; *Automated Process Planning Systems —An Overview of Ten Years of Activities.* Ist CIRP working seminar on CAPP, Jan. 1985.

Wysk, R. A., Chang, T. C., and Ham, I., 1986; *Automated process planning systems—an overview of ten years of activities*. Annals of the CIRP, Vol. 35/1/1986.

Xu, H. and Wang, H. P., 1989; *Part family formation for GT applications based on fuzzy mathematics*. International Journal of Production Research, 27 (9), pp. 1637-1651.

Young, R. I. M., 1991; *Machine Planning in a Product Model Environment.* PhD Dissertation, Department of Manufacturing Engineering, Loughborough University of Technology.

Zeng, C. P., and Yang, G. X., 1992; *STJLBM-1 Expert System Geometry a Part Classification and Coding System.* Information Technology for advanced Manufacturing Systems, Olling, G. J., and Deng, Z., (Editors) Elsevier Science Publications, B. v. (North-Holland).

Zust, R., and Taiber, J., 1990; *Knowledge-Based Process Planning System for Prismatic Workpieces in a CAD/CAM Environment.* Annals of the CIRP, Vol. 39/1/90.

Appendix A

DESIGN INTERPRETATION OF COMPONENTS

Design interpretation of a set of thirty components used for grouping case studies based on different criteria is given in this appendix.

COMPONENT DESIGN DATA

NO'S.	DRW. NO'S	FEATURES	PROCESSES	M/S
1.	C 6204230			
		LEN. = 324.5, DIA. = 47		
		St. r_boss	T1	SMU
			G1	UNV
		S_feature		
		Face	F1	SMU
		Centre	C11	"
		Screw	T7	SMU
		Keyway	W7	EDG
	No. of AD's are = 3			
2.	B 615.1225.9100.22			
		LEN. = 1487, DIA. = 133		
		St. r_boss	T1	SMU
			T1	END
		S_feature	G1	MGR
		Face	F1	END
		Centre	C11	"
		Screw	T7	SMU
		Keyway	W7	KEY
		St. r_pocket (Radial)	D2	"
	No. of AD's are = 3			
3.	B615.2205.9200.12			
		LEN. = 1861, DIA. = 228.6		
		St. r_boss	T1	SHI
			G1	MGR
		S_feature		
		Face	F1	END
		Centre	C11	"
		Screw	T7	SHI
		Keyway	W7	"
		St. r_pocket (Radial)	D2	"
		St. r_hole (Arms)		
		S_feature		
		Thread	D3	MRD
	No. of AD's are = 3			
4.	B615.1325.9240.02			
		LEN. = 1280, DIA. = 140.178		
		St. r_boss	T1	SHI
			G1	MGR
		S_feature		
		Face	F1	END
		Centre	C11	"
		Keyway	W7	SHI
		St. r_pocket (Radial)	D2	"
	No. of AD's are = 3			

COMPONENT DESIGN DATA

NO'S.	DRW. NO'S	FEATURES	PROCESSES	M/S
5.	B615.2205.9200.02			
		LEN. = 1753, DIA. = 228.6		
		St. r_boss	T1	SHI
			G1	MGR
		S_feature		
		Face	F1	END
		Centre	C11	"
		Screw	T7	SHI
		Keyway	W7	"
		St. r_pocket (Radial)	D2	"
		St. r_hole (Arms)		
		S_feature		
		Thread	D3	MRD
	No. of AD's are = 3			
6.	B615.2205.9200.13			
		LEN. = 1841, DIA. = 228.6		
		St. r_boss	T1	SHI
			G1	MGR
		S_feature		
		Face	F1	END
		Centre	C11	"
		Screw	T7	SHI
		Keyway	W7	"
		St. r_pocket (Radial)	D2	"
		St. r_hole (Arms)		
		S_feature		
		Thread	D3	MRD
	No. of AD's are = 3			
7.	D 5534208			
		LEN. = 3185, DIA. = 490		
		St. r_boss	T1	HEI
		S_feature		
		Face	F1	HEI
		Centre	C11	KTW
	No. of AD's are = 3			
8.	D 5050731			
		LEN. = 4305, DIA. = 444		
		St. r_boss	T1	HEI
		S_feature		
		Face	F1	HEI
		Centre	C11	KTW
	No. of AD's are = 3			
9.	C 1591 1516			
		LEN. = 16, DIA. = 761.24		
		St. r_boss	T2	BUL
		S_feature		
		Keyway	W7	MAK
		St. r_hole (bore)	T2	BUL

COMPONENT DESIGN DATA

NO'S.	DRW. NO'S	FEATURES	PROCESSES	M/S
		S_feature		
		Chamfer	C15	"
		St. r_hole (p.c.pattern)		
		S_feature		
		Thread	D3	"
	No. of AD's are = 3			
10.	C 1329 2505			
		LEN. = 35, DIA. = 385		
		St. r_boss	T2	LAN
		St. r_hole (bore)	"	"
		S_feature		
		Chamfer	C15	"
		St. r_hole (p.c.pattern)	D2	MAK
		Thru. rectangular slot	S2	SLT
	No. of AD's are = 3			
11.	D 9431.2187			
		LEN. = 25, DIA. = 146		
		St. r_boss	T2	SMU
		St. r_hole (bore)	"	"
		S_feature		
		Screw	T7	"
		St. r_hole	D2	MRD
		St. r_hole		
		S_feature		
		Thread	D3	"
	No. of AD's are = 3			
12.	X 2655627			
		LEN. = 19.05, DIA. = 111		
		St. r_boss	T2	QKO
		St. r_hole (bore)	"	"
		S_feature		
		Screw	T7	"
		Thru. partial round slot	M2	HOR
	No. of AD's are = 3			
13.	E 9358.5122			
		LEN. = 12, DIA. = 150		
		St. r_boss	T2	QKO
		St. r_pocket (bore)	"	"
		St. r_hole (bore)	"	"
		St. r_hole (p.c.pattern)	D2	MRD
		Partial round step	M2	
	HOR			
	No. of AD's are = 3			

COMPONENT DESIGN DATA

NO'S.	DRW. NO'S	FEATURES	PROCESSES	M/S
14.	X 4104181			
	Thin component			
		LEN. = 110, BRED. = 95, TH = 38		
		Surface	M2	VER
		Thru. rectangular slot	M2	VER
		Contoured step	M2	HOR
		St. r_hole (straight pattern)		
		S_feature		
		Thread	D3	WTD
	No. of AD's are = 3			
15.	E 9811.7109			
		LEN. = 32, DIA. = 25		
		St. r_boss	T12	CAP
		St. r_hole	"	"
	No. of AD's are = 3			
16.	K 4102201			
	Thin component			
		LEN. = 80, BRED. = 60, TH = 40		
		St. r_hole (straight pattern)		
		S_feature		
		Thread	D3	WTD
	No. of AD's are = 1			
17.	K 4053852			
		LEN. = 144, BRED. = 60, TH = 12		
		Surface	M2	VER
		Thru. rectangular slot	M2	VER
		St. r_hole	D2	WTD
	No. of AD's are = 1			
18.	K 4062662			
		LEN. = 156, BRED. = 60, TH = 12		
		Surface	M2	VER
		Thru. rectangular slot	M2	VER
		St. r_hole	D2	WTD
	No. of AD's are = 1			
19.	K 4029050			
		LEN. = 274, BRED. = 96, TH = 10		
		St. r_pocket (straight pattern)	D2	WTD
		S_feature		
		C/bore	C8	"
		C/s,k	C7	"
	No. of AD's are = 1			

COMPONENT DESIGN DATA

NO'S.	DRW. NO'S	FEATURES	PROCESSES	M/S
20.	E 5040061			
		LEN. = 30, BRED. = 15, TH = 6		
		St. r_hole		
		S_feature		
		Thread	D3	WTD
	No. of AD's are = 1			
21.	E 9625.3321			
		LEN. = 127, BRED. = 127, TH = 20		
		St. r_hole (straight pattern)	D2	WTD
		S_feature		
		Spotface	S22	WTD
	No. of AD's are = 1			
22.	D 5572195			
		LEN. = 710, BRED. = 410, TH = *30		
		St. r_hole (straight pattern)	D2	MAK
	No. of AD's are = 1			
23.	Y 4100566			
		LEN. = 270, BRED. = 225, TH = *25		
		St. r_hole (straight pattern)	D2	WTD
	No. of AD's are = 1			
24.	E 5001058			
		LEN. = 410, BRED. = 410, TH = 25		
		St. r_hole (straight pattern)	D2	WTD
	No. of AD's are = 1			
25.	E 5000033			
		LEN. = 56, BRED. = 12, TH = 8		
		St. r_hole	D2	WTD
		S_feature		
		C/s,k	C7	WTD
	No. of AD's are = 1			
26.	M 2936444			
		LEN. = 317.5, BRED. = 177.8, TH = 15.88		
		Surface	M2	HOR
		St. r_hole (straight pattern)	D2	MRD
		St. r_pocket	D2	MRD
	No. of AD's are = 6			
27.	X 4045777			
		LEN. = 500, BRED. = 120, TH = 25		
		St. r_hole (straight pattern)	D2	WTD
	No. of AD's are = 1			

COMPONENT DESIGN DATA

NO'S.	DRW. NO'S	FEATURES	PROCESSES	M/S
28.	E 5572065			
		LEN. = 1328, BRED. = 65, TH = 40		
		Triangular step	M2	VER
		St. r_hole (axial)		
		S_feature		
		Thread	D3	KTW
	No. of AD's are = 3			
29.	E 8082.8829			
		LEN. = 652, BRED. = 30.68, TH = 11.43		
		Rectangualr step	M2	HOR
		Surface	"	"
		Partial r_step	"	"
	No. of AD's are = 6			
30.	E 55333488			
		LEN. = 1328, BRED. = 65, TH = 40		
		Rectangualr step	M2	HOR
		Surface	"	"
	No. of AD's are = 4			

Appendix B

Table of Contents

B TABLE STRUCTURE FOR THE FEATURE-BASED PART DATA MODEL, THE MACHINING SYSTEM CAPABILITY DATA MODEL AND PART DESCRIPTION MODULE

B.1 TABLE STRUCTURE

A detailed representation of tables designed for storing the feature-based component data model and the machine capability data model, in GENERIS software is given in the following:

Table parts

Main Subject part
Valuedness Single Valued Generic On

Relationship	Property	Datatype	Status	Index
has drawing_number	drawing_number	Text	Mandatory	None
has description	description	Text	Mandatory	None
drawn by	person	entity	Optional	None
revision	date	date	Mandatory	None
required	heat_treatment	text	Optional	None
has batch_size	batch_size	Integer	Optional	None
has mass	mass	Decimal	Optional	None
has material	material	Entity	Optional	None
has bhn	bhn	Integer	Optional	None
x_value	position	Decimal	Mandatory	None
y_value	position	Decimal	Mandatory	None
z_value	position	Decimal	Mandatory	None
g_x_value	Orientation	Decimal	Mandatory	None
g_y_value	Orientation	Decimal	Mandatory	None
g_z_value	Orientation	Decimal	Mandatory	None
has length	length	Decimal	Mandatory	None
has lengthuptol	lengthuptol	Decimal	Mandatory	None
has lengthlotol	lengthlotol	Decimal	Mandatory	None
has widthuptol	widthuptol	Decimal	Optional	None
has widthlotol	widthlotol	Decimal	Optional	None
has depthuptol	depthuptol	Decimal	Optional	None
has depthlotol	depthlotol	Decimal	Optional	None
has diameter	diameter	Decimal	Optional	None
has diauptol	diauptol	Decimal	Optional	None
has dialotol	dialotol	Decimal	Optional	None
has pattern	pattern	Entity	Optional	None
has no_of_AD	no_of_AD	Integer	Optional	None

Table feature list

Main Subject part
Valuedness Multi Valued Generic On

Relationship	Property	Datatype	Status	Index
has featurecode	featurecode	Entity	Optional	None
has feature	feature	Entity	Optional	None

Table feature data

Main Subject featurecode
Valuedness Single Valued Generic On

Relationship	Property	Datatype	Status	Index
x_position	position	Decimal	Mandatory	None
y_position	position	Decimal	Mandatory	None
z_position	position	Decimal	Mandatory	None
x_orientation	orientation	Decimal	Mandatory	None
y_orientation	orientation	Decimal	Mandatory	None
z_orientation	orientation	Decimal	Mandatory	None
fealengh	lengh	Decimal	Mandatory	None
fealengthuptol	lengthuptol	Decimal	Optional	None
fealengthlotol	lengthlotol	Decimal	Optional	None
feawidth	width	Decimal	Optional	None
feawidthuptol	widthuptol	Decimal	Optional	None
feawidthlotol	widthlotol	Decimal	Optional	None
feadepth	depth	Decimal	Optional	None
feadepthuptol	depthuptol	Decimal	Optional	None
feadepthlotol	depthlotol	Decimal	Optional	None
feadiameter	diameter	Decimal	Optional	None
feadiauptol	diauptol	Decimal	Optional	None
feadialotol	dialotol	Decimal	Optional	None
has depth_axis	depth_axis	Entity	Mandatory	None
depth_symmetry	symmetry	Entity	Mandatory	None
has ent_ext_relation	ent_ext_relation	Entity	Mandatory	None
has form_variation	form_variation	Entity	Mandatory	None
has straightness	straightness	Decimal	Optional	None
has flatness	flatness	Decimal	Optional	None
has roundness	roundness	Decimal	Optional	None
has cylindricity	cylindricity	Decimal	Optional	None
has surfinish	surfinish	Decimal	Optional	None
has axiscode	axiscode	Entity	Optional	None
fearadius	radius	Decimal	Optional	None
feaangle	angle	Decimal	Optional	None

Table surfaces

Main Subject featurecode
Valuedness Multi Valued Generic On

Relationship	Property	Datatype	Status	Index
has real	featurecode	Entity	Optional	None
has imagin	featurecode	Entity	Optional	None

Table axiscodes

Main Subject axiscode
Valuedness Multi Valued Generic On

Relationship	Property	Datatype	Status	Index
has concentric	axiscode	Entity	Optional	None
has concentricity	concentricity	Decimal	Optional	None
has symmetric	axiscode	Entity	Optional	None
has symmetryvalue	symmetryvalue	Decimal	Optional	None

Table sec features

Main Subject featurecode
Valuedness Multi Valued Generic On

Relationship	Property	Datatype	Status	Index
has sec_featurecode	sec_featurecode	Entity	Optional	None
has sec_feature	sec_feature	Entity	Optional	None

Table sec features data

Main Subject sec_featurecode
Valuedness Multi Valued Generic On

Relationship	Property	Datatype	Status	Index
has type	type	Text	Optional	None
seclength	length	Decimal	Optional	None
seclengthuptol	lengthuptol	Decimal	Optional	None
seclengthlotol	lengthlotol	Decimal	Optional	None
secwidth	width	Decimal	Optional	None
secwidthuptol	widthuptol	Decimal	Optional	None
secwidthlotol	widthlotol	Decimal	Optional	None
secdepth	depth	Decimal	Optional	None
secdepthuptol	depthuptol	Decimal	Optional	None
secdepthlotol	depthlotol	Decimal	Optional	None
secdiameter	diameter	Decimal	Optional	None
secdiauptol	diauptol	Decimal	Optional	None
secdialotol	dialotol	Decimal	Optional	None
secpitch	pitch	Decimal	Optional	None
secradius	radius	Decimal	Optional	None
secangle	angle	Decimal	Optional	None

Table_comp surfaces

Main Subject facecode
Valuedness Multi Valued Generic On

Relationship	Property	Datatype	Status	Index
has parallel	featurecode	Entity	Optional	None
has parallelism	parallelism	Decimal	Optional	None
has perpendicular	featurecode	Entity	Optional	None
has perpendicularity	perpendicularity	Decimal	Optional	None
has angle	featurecode	Entity	Optional	None
has angularity	angularity	Decimal	Optional	None
runout_wrt	featurecode	Entity	Optional	None
has runout	runout	Decimal	Optional	None

Table_nor surfaces

Main Subject featurecode
Valuedness Multi Valued Generic On

Relationship	Property	Datatype	Status	Index
has nor_vector	nor_vector	Entity	Optional	None

THE MACHINING SYSTEM CAPABILITY DATA MODEL

Table_tot system

Main Subject firm
Valuedness Multi Valued Generic On

Relationship	Property	Datatype	Status	Index
has machine	machine	Entity	Optional	None

Table_machines

Main Subject machine
Valuedness Multi Valued Generic On

Relationship	Property	Datatype	Status	Index
has m_c_unit	m_c_unit	Entity	Optional	None

Table m_c_units

Main Subject m_c_unit
Valuedness Multi Valued Generic On

Relationship	Property	Datatype	Status	Index
has operation	operation	Entity	Optional	None

B.2 PART DESCRIPTION MODULE

In this section, the procedure file 'input_part' which is for describing the feature-based component data model has been listed. Following the file, other procedure files 'do_round_boss', 'do_round_hole', 'do_slot' etc. are attached. These programs are called from the main program 'input_part' when an instance of any particular type of feature input happens.

PROCEDURE FOR USING THE SOFTWARE

The directory in which the cell design software programs are developed is called 'fahd'. After changing the home directory to the directory mentioned, the system starts by typing in 'generis' on the prompt. GENERIS Expert System opens its window for its use. The next command to be entered is 'do generisinit'. A user defined window called 'WELCOME' is invoked. The password will be asked to enter to get the main menu. The password is simply <return>. After entering the password, the main menu will be displayed in another user defined window called 'MAIN MENU'. Main menu facilitates the selection of different system modules. Title of the menu window is 'SELECTION'. Select the menu "TO INPUT THE PART" to input the part description.

B.2.1 PROCEDURE FILE 'input_part'

```
* DO GENERIS FILE 'input_part'
*
* PROCEDURE FILE TO CREATE COMPONENT DATA MODEL IN THE GENERIS
APPLICATION
*
CREATE LOCAL dec00 DECIMAL
CREATE LOCAL dec01 DECIMAL
CREATE LOCAL dec02 DECIMAL
CREATE LOCAL dec03 DECIMAL
CREATE LOCAL dec04 DECIMAL
CREATE LOCAL dec05 DECIMAL
CREATE LOCAL dec06 DECIMAL
CREATE LOCAL dec07 DECIMAL
CREATE LOCAL dec08 DECIMAL
CREATE LOCAL dec09 DECIMAL
CREATE LOCAL dec10 DECIMAL
CREATE LOCAL dec11 DECIMAL
CREATE LOCAL dec12 DECIMAL
CREATE LOCAL dec13 DECIMAL
CREATE LOCAL dec14 DECIMAL
CREATE LOCAL dec15 DECIMAL
CREATE LOCAL dec16 DECIMAL
CREATE LOCAL dec17 DECIMAL
CREATE LOCAL dec18 DECIMAL
CREATE LOCAL dec19 DECIMAL
CREATE LOCAL dec20 DECIMAL
CREATE LOCAL dec21 DECIMAL
CREATE LOCAL tex0 TEXT
CREATE LOCAL tex1 TEXT
CREATE LOCAL tex2 TEXT
CREATE LOCAL tex3 TEXT
CREATE LOCAL int0 INTEGER
CREATE LOCAL int1 INTEGER
CREATE LOCAL int2 INTEGER
CREATE LOCAL int3 INTEGER
CREATE LOCAL int4 INTEGER
CREATE LOCAL int5 INTEGER
CREATE LOCAL int6 INTEGER
CREATE LOCAL int7 INTEGER
CREATE LOCAL countno INTEGER
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
CREATE LOCAL nam4 NAME
CREATE LOCAL nam5 NAME
CREATE LOCAL nam6 NAME
CREATE LOCAL nam7 NAME
CREATE LOCAL nam8 NAME
CREATE LOCAL nam9 NAME
CREATE LOCAL nam10 NAME
CREATE LOCAL nam11 NAME
CREATE LOCAL nam12 NAME
CREATE LOCAL nam13 NAME
CREATE LOCAL nam14 NAME
CREATE LOCAL nam15 NAME
*
CLEAR
WINDOW NEW WELCOME SIZE 37,92 AT 2,2
POSITION 1,15
MESSAGE "FILE TO CREATE RECORDS IN THE APPLICATION"
LABEL sarmad
LABEL again
POSITION 5,1
MESSAGE "PLEASE INPUT THE PART NAME:?"
READ tex2
IF tex2 = BLANK
    POSITION 5,30
    MESSAGE "YOU HAVE NOT INPUT THE PART"
    POSITION 6,1
    GOTO again
ENDIF
POSITION 5,1
IF TELL tex2 IS part
    LET nam0 = tex2
ELSE
    FACT tex2 IS part
    LET nam0 = tex2
ENDIF
IF TELL ANY RECORDS IN parts FOR ^nam0
    POSITION 5,0
    MESSAGE "" ^nam0 "   ALREADY EXISTS: DO YOU WANT TO REENTER THIS PART:?
    y/YES "
    POSITION 6,0
    MESSAGE "y FOR ENTER ANOTHER PART AND r FOR ENTERING THE FEATURES FOR
    THIS PART"
    READ tex0
    IF tex0 = y
        DELETE RECORDS IN parts FOR ^nam0
    ELSE
        IF tex0 = r
            GOTO robin
        ELSE
            CLEAR SIZE 36,85 AT 2,0
            GOTO again
        ENDIF
    ENDIF
ENDIF
POSITION 7,0
MESSAGE "PLEASE INPUT THE DRAWIMG NUMBER FOR " ^nam0" :?"
READ tex0
POSITION 8,0
MESSAGE "PLEASE INPUT THE BATCH SIZE FOR " ^nam0" :?"
READ int7
POSITION 9,0
MESSAGE "PLEASE INPUT THE DESCRIPTION  FOR " ^nam0" :?"
READ tex1
POSITION 10,0
MESSAGE "PLEASE INPUT THE MASS  FOR " ^nam0" :?"
READ dec00
POSITION 11,0
MESSAGE "PLEASE INPUT THE MATERIAL  FOR " ^nam0" :?"
READ tex2
IF tex2 = BLANK
    GOTO fahd3
ELSE
    POSITION 10,31
    MESSAGE "YOU HAVE NOT INPUT THE MATERIAL  FOR " ^nam0" "
    POSITION 12,0
    GOTO again1
    IF TELL tex2 IS material IN parts
        LET nam1 = tex2
    ELSE
        FACT tex2 IS material
        LET nam1 = tex2
    ENDIF
ENDIF
LABEL fahd3
POSITION 12,0
MESSAGE "PLEASE INPUT THE BHN OF THE MATERIAL FOR " ^nam0" :?"
READ int0
POSITION 13,0
MESSAGE "PLEASE INPUT THE DATUM X_POSITION FOR " ^nam0" :?"
READ dec01
POSITION 14,0
MESSAGE "PLEASE INPUT THE DATUM Y_POSITION FOR " ^nam0" :?"
READ dec02
```

```
POSITION 15,0
MESSAGE "PLEASE INPUT THE DATUM Z_POSITION FOR " ^nam0" :?"
READ dec03
POSITION 16,0
MESSAGE "PLEASE INPUT THE X_ORIENTATION FOR " ^nam0" :?"
READ dec19
POSITION 17,0
MESSAGE "PLEASE INPUT THE Y_ORIENTATION FOR " ^nam0" :?"
READ dec20
POSITION 18,0
MESSAGE "PLEASE INPUT THE Z_ORIENTATION FOR " ^nam0" :?"
READ dec21
POSITION 19,0
MESSAGE "PLEASE INPUT THE OVERALL LENGTH FOR " ^nam0" :?"
READ dec04
POSITION 20,0
MESSAGE "PLEASE INPUT THE UPPER TOLERANCE FOR THE LENGHT FOR " ^nam0" :?"
READ dec05
POSITION 21,0
MESSAGE "PLEASE INPUT THE LOWER TOLERANCE FOR THE LENGHT FOR " ^nam0" :?"
READ dec06
POSITION 22,0
MESSAGE "PLEASE INPUT THE OVERALL WIDTH FOR " ^nam0" :?"
READ dec07
POSITION 23,0
MESSAGE "PLEASE INPUT THE UPPER TOLERANCE FOR THE WIDTH FOR " ^nam0" :?"
READ dec08
POSITION 24,0
MESSAGE "PLEASE INPUT THE LOWER TOLERANCE FOR THE WIDTH FOR " ^nam0" :?"
READ dec09
POSITION 25,0
MESSAGE "PLEASE INPUT THE OVERALL DEPTH FOR " ^nam0" :?"
READ dec10
POSITION 26,0
MESSAGE "PLEASE INPUT THE UPPER TOLERANCE FOR DEPTH FOR " ^nam0" :?"
READ dec11
POSITION 27,0
MESSAGE "PLEASE INPUT THE LOWER TOLERANCE FOR THE DEPTH FOR " ^nam0" :?"
READ dec12
POSITION 28,0
MESSAGE "PLEASE INPUT THE OVERALL DIAMETER FOR " ^nam0" :?"
READ dec13
POSITION 29,0
MESSAGE "PLEASE INPUT THE UPPER TOLERANCE FOR DIAMETER FOR " ^nam0" :?"
READ dec14
POSITION 30,0
MESSAGE "PLEASE INPUT THE LOWER TOLERANCE FOR THE DIAMETER FOR " ^nam0"
:?"
READ dec15
POSITION 31,0
MESSAGE "DO THE PATTERN OF ROUND HOLES OR ROUND POCKETS EXIST FOR "
^nam0" :? y/YES"
READ tex2
IF tex2 = y
    POSITION 32,0
    MESSAGE "WHICH PATTERN TYPE EXIST ?"
    POSITION 33,0
    MESSAGE "PLEASE ENTER ------> 1 <----- FOR STRAIGHT PATTERN"
    POSITION 34,0
    MESSAGE "PLEASE ENTER ------> 2 <----- FOR PCD PATTERN"
    POSITION 35,0
    MESSAGE "PLEASE ENTER ------> 3 <----- FOR BOTH PATTERN"
    READ int1
    IF int1 = 1
        LET nam2 = straight1
    ELSE
        IF int1 = 2
            LET nam2 = pcd
        ELSE
            IF int1 = 3
                LET nam2 = pcd_straight1
            ENDIF
        ENDIF
    ENDIF
ENDIF
POSITION 36,0
MESSAGE "PLEASE ENTER NUMBER OF THE AD's"
READ int1
FACT IN parts
^nam0
^tex0

^tex1
^dec00
^nam1
^int0
^dec01
^dec02
^dec03


^dec04
^dec05
^dec06
^dec07
^dec08
^dec09
^dec10
^dec11
^dec12
^dec13
^dec14
^dec15
^nam2
^int1

LABEL robin
CLEAR SIZE 36,85 AT 4,0
POSITION 4,0
MESSAGE "DO YOU WANT TO ENTER THE FEATURE LIST AND THEIR DATA FOR THE
PART " ^nam0" ? ---> y/YES "
```

```
READ tex3
IF tex3 = y
    IF TELL ANY RECORDS IN TABLE fea_list FOR ^nam0
        POSITION 5,0
        MESSAGE "SOME FEATURES ALREADY EXIST IN THE TABLE"
        POSITION 6,0
        MESSAGE "---> DO YOU WANT TO ENTER AGAIN ? ----> ENTER 'y' OR "
        POSITION 7,0
        MESSAGE "---> DO YOU WANT TO ENTER MORE AND RETAIN ALREADY
        EXISTING ? ----> ENTER 'r' "
        READ tex0
        IF tex0  = y
            DELETE RECORDS IN TABLE fea_list FOR ^nam0
            LET int3 = 0
        ELSE
            IF tex0 = r
                FETCH RECORDS IN table fea_list FOR ^nam0
                LET int3 = $COUNT
            ENDIF
        ENDIF
    ELSE
        LET int3 = 0
    ENDIF
    LABEL star
    LET tex1 = ^nam0:"f":int3+1
    IF TELL tex1 IS featurecode

    ELSE
        FACT tex1 IS featurecode
    ENDIF
    LET nam2 = tex1
    IF TELL ANY RECORDS IN fea_list FOR ^ nam2
        CLEAR SIZE 36,85 AT 4,0
        POSITION 4,1
        MESSAGE"THE FEATURECODE "^ nam2" ALREADY EXISTS : DO YOU WANT TO
        RE-ENTER :"
        POSITION 5,0
        MESSAGE" y/YES and n/NO"
        READ tex0
        IF tex0 = y
            IF TELL ANY RECORDS IN fea_data FOR ^ nam2
                DELETE RECORDS IN TABLE fea_data FOR ^ nam2
            ENDIF
        ELSE
            LET int3 = int3 + 1
            CLEAR SIZE 36,85 AT 4,0
            GOTO star
        ENDIF
    ENDIF
    LABEL again2
    POSITION 6,1
    MESSAGE"*******************************************************"
    POSITION 7,1
    MESSAGE"PLEASE ENTER THE FEATURE ?                            *"
    POSITION 8,1
    MESSAGE"*******************************************************"
```

```
POSITION 9,1
MESSAGE"FOR FEATURE 'surface' JUST TYPE ---------------> s          *"
POSITION 10,1
MESSAGE"FOR FEATURE 'round_hole' JUST TYPE -----------> h          *"
POSITION 11,1
MESSAGE"FOR FEATURE 'round_pocket' JUST TYPE --------> p          *"
POSITION 12,1
MESSAGE"FOR FEATURE 'rectangular_slot' JUST TYPE ------> r          *"
POSITION 13,1
MESSAGE"FOR FEATURE 'triangular_step' JUST TYPE --------> l          *"
POSITION 14,1
MESSAGE"FOR FEATURE 'contoured_step' JUST TYPE ---- --> c          *"
POSITION 15,1
MESSAGE"FOR FEATURE 'axial_round_pocket' JUST TYPE --> x          *"
POSITION 16,1
MESSAGE"FOR FEATURE 'round_boss' JUST TYPE -----------> b          *"
POSITION 17,1
MESSAGE"*********************************************************"
POSITION 18,0
MESSAGE"IF THERE IS ANY OTHER FEATURE WHICH IS NOT MENTIONED
POSITION 19,0
MESSAGE"ABOVE, PLEASE INPUT --------------------- -----> w          *"
POSITION 20,0
MESSAGE"*********************************************************"
POSITION 21,0
READ tex1
IF tex1 = BLANK
    CLEAR SIZE 36,85 AT 4,0
    POSITION 5,0
    MESSAGE "YOU HAVE NOT ENTERED THE FEATURE"
    POSITION 6,1
    GOTO again2
ELSE
    IF tex1 = s
        LET tex1 = surface
    ELSE
        IF tex1 = h
            LET tex1 = round_hole
        ELSE
            IF tex1 = p
                LET tex1 = round_pocket
            ELSE
                IF tex1 = r
                    LET tex1 = rectangular_slot
                ELSE
                    IF tex1 = l
                        LET tex1 = triangular_step
                    ELSE
                        IF tex1 = c
                            LET tex1 = contoured_step
                        ELSE
                            IF tex1 = e
                                LET tex1 = free_surface
                            ELSE
                                IF tex1 = x
                                    LET tex1 = axial_round_pocket
```

```
                                        ELSE
                                            IF tex1 = b
                                                LET tex1 = round_boss
                                            ELSE
                                                IF tex1 = w
                                                    POSITION 22,0
                                                    MESSAGE"PLEASE INPUT THE FEATURE
                                                    FOR " ^nam0" ?"
                                                    POSITION 23,0
                                                    READ tex1
                                                ENDIF
                                            ENDIF
                                        ENDIF
                                    ENDIF
                                ENDIF
                            ENDIF
                        ENDIF
                    ENDIF
                ENDIF
            ENDIF
        ENDIF
        IF TELL tex1 IS feature
            LET nam3 = tex1
        ELSE
            FACT tex1 IS feature
            LET nam3 = tex1
        ENDIF
        FACT ^nam0 'has featurecode' ^nam2 'has feature' ^nam3
        CLEAR SIZE 36,85 AT 4,0
        POSITION 25,0
        MESSAGE"DO YOU WANT TO ENTER THE DATA FOR THE FEATURE " ^nam3 " ? y FOR
        YES"
        READ tex0
        IF tex0 = y
*******************************************
* TO ENTER THE RECORDS IN TABLE fea_data
*******************************************
            CLEAR SIZE 36,85 AT 4,0
            POSITION 4,1
            MESSAGE " NOW THE DATA FOR THE FEATURE " ^nam3 "  WILL BE ENTERED"
            IF ^nam3 = round_boss
                DO do_round_boss
            ELSE
                IF ^nam3 = round_hole
                    DO do_round_hole
                ELSE
                    IF ^nam3 = round_pocket
                        DO do_round_hole
                    ELSE
                        IF ^nam3 = rectangular_slot
                            DO do_slot
                        ELSE
                            IF ^nam3 = triangular_step
                                DO do_slot
                            ELSE
                                IF ^nam3 = contoured_step
                        DO do_slot
                    ELSE
                        IF ^nam3 = surface
                            DO do_surface
                        ELSE
                            IF ^nam3 = axial_round_pocket
                                DO do_round_hole
                            ELSE
                                CREATE RECORDS IN fea_data FOR ^nam2
                            ENDIF
                        ENDIF
                    ENDIF
                ENDIF
            ENDIF
        ENDIF
    ENDIF
  ENDIF
ENDIF
***********************************************************
* NOW RECORDS IN THE TABLE sec_features WILL BE ENTERED
***********************************************************
  CLEAR SIZE 36,85 AT 4,0
  POSITION 8,1
  MESSAGE"DO YOU WANT TO ENTER THE SECONDARY FEATURES FOR THE
  FEATURE "^nam3"? y/YES  "
  READ  tex0
  IF tex0 = y
     LET int5 = 0
     LABEL a3
     LET tex1 = ^ nam2:"sec":int5+1
     IF TELL tex1 IS sec_featurecode

     ELSE
        FACT tex1 IS sec_featurecode
     ENDIF
     LET nam11 = tex1
     CLEAR SIZE 36,85 AT 2,0
     LABEL again3
     POSITION 6,1
     MESSAGE"***********************************************************"
     POSITION 7,1
     MESSAGE"PLEASE ENTER THE "int5 + 1" SEC_FEATURE FOR FEATURE ? "^ nam2"
     *"
     POSITION 8,1
     MESSAGE"***********************************************************"
     POSITION 9,1
     MESSAGE"FOR SEC_FEATURE 'thread' JUST TYPE ----------------> h       *"
     POSITION 10,1
     MESSAGE"FOR SEC_FEATURE 'axial_thread' JUST TYPE ----------> x       *"
     POSITION 11,1
     MESSAGE"FOR SEC_FEATURE 'keyway' JUST TYPE ----------------> k       *"
     POSITION 12,1
     MESSAGE"FOR SEC_FEATURE 'chamfer' JUST TYPE ---------------> c       *"
     POSITION 13,1
     MESSAGE"FOR SEC_FEATURE 'screw' JUST TYPE -----------------> s       *"
     POSITION 14,1
```

```
MESSAGE"FOR SEC_FEATURE 'centre' JUST TYPE ------------------> t       *"
POSITION 15,1
MESSAGE"**************************************************************"
POSITION 16,1
MESSAGE"IF THERE IS ANY OTHER SECONDARY FEATURE WHICH IS NOT
MENTIONED"
POSITION 17,0
MESSAGE"ABOVE, PLEASE INPUT -----------------------------------> w       *"
POSITION 18,0
MESSAGE"**************************************************************"
READ tex3
IF tex3 = BLANK
    POSITION 19,0
    MESSAGE "YOU HAVE NOT ENTERED THE SEC_FEATURE"
    POSITION 6,1
    GOTO again
ENDIF
IF tex3 = h
    LET tex3 = thread
ELSE
    IF tex3 = x
        LET tex3 = axial_thread
    ELSE
        IF tex3 = k
            LET tex3 = keyway
        ELSE
            IF tex3 = c
                LET tex3 = chamfer
            ELSE
                IF tex3 = s
                    LET tex3 = screw
                ELSE
                    IF tex3 = t
                        LET tex3 = centre
                    ELSE
                        IF tex3 = w
                            POSITION 20,0
                            MESSAGE"PLEASE INPUT THE SECONDARY FEATURE
                            "
                            POSITION 21,0
                            READ tex3
                        ENDIF
                    ENDIF
                ENDIF
            ENDIF
        ENDIF
    ENDIF
ENDIF
IF TELL tex3 IS feature
    LET nam12 = tex3
ELSE
    FACT tex3 IS feature
    LET nam12 = tex3
ENDIF
FACT ^ nam2 'has sec_featurecode' ^nam11 'has sec_feature' ^nam12
CLEAR SIZE 36,80 AT 4,0
```

```
        POSITION 20,0
        MESSAGE"DO YOU WANT TO ENTER THE SECONDARY FEATURE DATA FOR
        "^nam12 "? y/YES"
        READ tex0
        IF tex0 = y
            IF ^nam12 = thread
            DO do_thread
            ELSE
                IF ^nam12 = axial_thread
                DO do_thread
                ELSE
                    CREATE RECORDS IN sec_features_data FOR ^nam11
                ENDIF
            ENDIF
        ENDIF
        CLEAR SIZE 36,80 AT 4,0
        POSITION 20,0
        MESSAGE"MORE SECONDARY FEATURES TO BE ENTERED FOR THIS FEATURE?
        y/YES  "
        READ  tex1
        IF tex1 = y
            LET int5 = int5 + 1
            CLEAR SIZE 36,85 AT 2,0
            GOTO a3
        ELSE
            CLEAR SIZE 36,85 AT 2,0
        ENDIF
    ENDIF
****************************************************************
* NOW RECORDS IN THE TABLE surfaces WILL BE ENTERED
****************************************************************
    CLEAR SIZE 36,85 AT 3,0
    POSITION 4,0
    MESSAGE"DO YOU WANT TO ENTER THE REAL & IMAGINARY SURFACES FOR THE
    FEATURECODE " ^nam2 " ?"
    POSITION 5,0
    MESSAGE" RETURN FOR NO AND y FOR YES   "
    READ tex0
    IF tex0  = y
        LET int4 = 0
        LABEL a2
        POSITION 6,0
        MESSAGE "PLEASE ENTER FEATURECODE 'has real' FOR  --> "^nam2 "? "
        READ tex2
        IF (tex2 = BLANK)
            LET nam10 = BLANK
        ELSE
            IF TELL tex2 IS featurecode
                LET nam10 = tex2
            ELSE
                FACT tex2 IS featurecode
                LET nam10 = tex2
            ENDIF
        ENDIF
        POSITION 7,0
        MESSAGE "PLEASE ENTER FEATURECODE 'has imagin' " ^nam2 " "
    READ tex2
    IF (tex2 = BLANK)
        LET nam9 = BLANK
    ELSE
        LET nam9 = tex2
    ENDIF
    IF (nam10 = BLANK)

    ELSE
        POSITION 8,0
        MESSAGE"PLEASE ENTER THE SURFACE FINISH ON REAL SURFACE? "
        READ dec00
    ENDIF
    FACT IN surfaces
    ^nam2
    ^nam10
    ^nam9
    ^dec00

    IF (nam10 = BLANK)

    ELSE
        CLEAR SIZE 36,85 AT 2,0
        POSITION 4,0
        MESSAGE"PLEASE ENTER THE NOR_VECTOR FOR THE FEATURECODE "
        ^nam10 " ? "
        POSITION 5,0
        READ nam13
        IF nam13 = BLANK

        ELSE
            FACT IN nor_surfaces
            ^nam10
            ^nam13

        ENDIF
    ENDIF
    CLEAR SIZE 36,85 AT 2,0
    POSITION 4,0
    MESSAGE"MORE DATA TO BE INPUT FOR REAL AND IMAGINARY
    FEATURECODE FOR " ^nam2 " ? "
    POSITION 5,0
    MESSAGE"-----> y/YES "
    READ tex0
    IF tex0 = y
        GOTO a2
    ENDIF
ENDIF
CLEAR SIZE 36,85 AT 3,0
POSITION 4,0
MESSAGE " HAVE YOU GOT MORE FEATURE LIST FOR THE PART " ^nam0" : y/YES "
READ tex0
IF tex0 = y
    LET int3 = int3 + 1
    CLEAR SIZE 36,85 AT 2,0
    GOTO star
    ENDIF
ENDIF
CLEAR SIZE 36,85 AT 2,0
POSITION 4,1
MESSAGE " DO YOY WANT TO INPUT THE REQUIRED DATA FOR THE PART "^nam0" ?
y/YES"
POSITION 5,1
READ tex0
IF tex0 = y
    POSITION 6,5
    MESSAGE "PLEASE WAIT ..."
    LET int6 = 6
    LET nam14 = free_surface
    LET int2 = 0
    WHILE (int6 > 0)
        LET int2 = int2 + 1
        LET tex1 = "s":int2
        LET tex2 = "PAD":int2
        LET nam13 = tex1
        LET nam15 = tex2
        FACT IN fea_list
        ^nam0
        ^nam13
        ^nam14

        FACT IN surfaces
        ^nam13
        ^nam13

        FACT IN nor_surfaces
        ^nam13
        ^nam15

        LET int6 = int6 - 1
    ENDWHILE
ENDIF
CLEAR SIZE 36,85 AT 2,0
POSITION 4,1
MESSAGE " HAVE YOU GOT MORE PART TO ENTER : y/YES and n/NO"
READ tex0
IF tex0 = y
    GOTO sarmad
ENDIF
CLEAR
WINDOW MAIN_MENU
DELETE WINDOW WELCOME
```

B.2.2 PROCEDURE FILE 'do_round_boss'

```
* DO GENERIS FILE 'do_round_boss'
*
POSITION 6,0
MESSAGE "PLEASE ENTER THE X POSITION FOR ORIGIN OF THE FEATURE "
^nam3":? "
READ dec11
POSITION 7,0
MESSAGE "PLEASE ENTER THE Z POSITION FOR ORIGIN OF THE FEATURE "
^nam3":?"
READ dec12
POSITION 8,0
MESSAGE "PLEASE ENTER THE X ORIENTATION OF THE FEATURE " ^nam3":?"
READ dec9
POSITION 9,0
MESSAGE "PLEASE ENTER THE Z ORIENTATION OF THE FEATURE " ^nam3":?"
READ dec10
POSITION 10,0
MESSAGE "PLEASE ENTER THE DIAMETER FOR " ^nam3":?"
READ dec03
POSITION 11,0
MESSAGE "PLEASE ENTER THE POSITIVE DIAMETER TOLERENCE FOR "
^nam3":?"
READ dec04
POSITION 12,0
MESSAGE "PLEASE ENTER THE NEGATIVE DIAMETER TOLERENCE FOR "
^nam3":?"
READ dec05
POSITION 13,0
MESSAGE "PLEASE ENTER THE DEPTH FOR " ^nam3":? "
READ dec00
POSITION 14,0
MESSAGE "PLEASE ENTER THE POSITIVE DEPTH TOLERENCE FOR " ^nam3":?"
READ dec01
POSITION 15,0
MESSAGE "PLEASE ENTER THE NEGATIVE LENGTH DEPTH TOLERENCE FOR "
^nam3":?"
READ dec02
POSITION 16,0
MESSAGE "PLEASE ENTER THE DEPTH AXIS INFORMATION FOR " ^nam3":?"
POSITION 17,0
MESSAGE "FOR STRAIGHT, PLEASE INPUT -----------> d "
POSITION 18,0
MESSAGE "FOR NOT STRAIGHT, PLEASE INPUT -----> s "
POSITION 19,0
READ tex0
IF tex0 = d
    LET nam4 = straight
ELSE
    IF tex0 = s
        LET nam4 = not_straight
    ENDIF
ENDIF
POSITION 20,0
```

```
MESSAGE "PLEASE ENTER THE SYMMETRY INFORMATION FOR " ^nam3":?"
POSITION 21,0
MESSAGE "FOR SYMMETRIC, PLEASE INPUT -----------> d "
POSITION 22,0
MESSAGE "FOR NOT SYMMETRIC, PLEASE INPUT -----> s "
POSITION 23,0
READ tex0
IF tex0 = d
    LET nam5 = symmetric
ELSE
    IF tex0 = s
        LET nam5 = not_symmetric
    ENDIF
ENDIF
POSITION 24,0
MESSAGE "PLEASE ENTER THE ENTRY/EXIT RELATION INFORMATION FOR "
^nam3":?"
POSITION 25,0
MESSAGE "FOR SAME, PLEASE INPUT -------------> s "
POSITION 26,0
MESSAGE "FOR NOT SAME, PLEASE INPUT -------> d "
POSITION 27,0
READ tex0
IF tex0 = s
    LET nam6 = same
ELSE
    IF tex0 = d
        LET nam6 = not_same
    ENDIF
ENDIF
POSITION 28,0
MESSAGE "PLEASE ENTER THE FORM VARIATION INFORMATION FOR "
^nam3":?"
POSITION 29,0
MESSAGE "FOR CONSTANT, PLEASE INPUT -----------> c "
POSITION 30,0
MESSAGE "FOR TAPERED, PLEASE INPUT -------------> s "
POSITION 31,0
MESSAGE "FOR CONCAVE, PLEASE INPUT ------------> v "
POSITION 32,0
MESSAGE "FOR CONVEX, PLEASE INPUT --------------> x "
POSITION 33,0
MESSAGE "FOR COUNTOURED, PLEASE INPUT --------> u "
POSITION 34,0
READ tex0
IF tex0 = c
    LET nam7 = constant
ELSE
    IF tex0 = s
        LET nam7 = tapered
    ELSE
        IF tex0 = v
            LET nam7 = concave
        ELSE
            IF tex0 = x
                LET nam7 = convex
            ELSE
                IF tex0 = u
                    LET nam7 = contoured
                ENDIF
            ENDIF
        ENDIF
    ENDIF
ENDIF
POSITION 35,0
MESSAGE "PLEASE ENTER THE ROUNDNES VALUE FOR " ^nam3":?"
READ dec07
POSITION 36,0
MESSAGE "PLEASE ENTER THE CYLINDRICITY VALUE FOR " ^nam3":? "
READ dec08
POSITION 37,0
MESSAGE "PLEASE ENTER THE SURFACE FINISH VALUE FOR " ^nam3":? "
READ dec18
LET tex1 = ^nam2:"axis"
IF TELL tex1 IS axiscode

ELSE
    FACT tex1 IS axiscode
ENDIF
LET nam8 = tex1
FACT IN TABLE fea_data
^nam2
^dec11
^dec12
^dec9
^dec10
^dec00
^dec01
^dec02
^dec03
^dec04
^dec05
^nam4
^nam5
^nam6
^nam7
^dec07
^dec08
^dec18
^nam8

CLEAR SIZE 36,80 AT 4,0
```

B.2.3 PROCEDURE FILE 'do_round_hole'

```
* DO GENERIS FILE 'do_round_hole'
*
POSITION 6,0
MESSAGE "PLEASE ENTER THE X POSITION FOR ORIGIN OF THE FEATURE " ^nam3":? "
READ dec11
POSITION 7,0
MESSAGE "PLEASE ENTER THE Y POSITION FOR ORIGIN OF THE FEATURE " ^nam3":? "
READ dec13
POSITION 8,0
MESSAGE "PLEASE ENTER THE Z POSITION FOR ORIGIN OF THE FEATURE " ^nam3":?"
READ dec12
POSITION 9,0
MESSAGE "PLEASE ENTER THE X ORIENTATION OF THE FEATURE " ^nam3":?"
READ dec9
POSITION 10,0
MESSAGE "PLEASE ENTER THE Y ORIENTATION OF THE FEATURE " ^nam3":?"
READ dec14
POSITION 11,0
MESSAGE "PLEASE ENTER THE Z ORIENTATION OF THE FEATURE " ^nam3":?"
READ dec10
POSITION 12,0
MESSAGE "PLEASE ENTER THE DIAMETER FOR " ^nam3":?"
READ dec03
POSITION 13,0
MESSAGE "PLEASE ENTER THE POSITIVE DIAMETER TOLERENCE FOR " ^nam3":?"
READ dec04
POSITION 14,0
MESSAGE "PLEASE ENTER THE NEGATIVE DIAMETER TOLERENCE FOR " ^nam3":?"
READ dec05
POSITION 15,0
MESSAGE "PLEASE ENTER THE DEPTH FOR " ^nam3":? "
READ dec00
POSITION 16,0
MESSAGE "PLEASE ENTER THE POSITIVE DEPTH TOLERENCE FOR " ^nam3":?"
READ dec01
POSITION 17,0
MESSAGE "PLEASE ENTER THE NEGATIVE LENGTH DEPTH TOLERENCE FOR " ^nam3":?"
READ dec02
POSITION 18,0
MESSAGE "PLEASE ENTER THE DEPTH AXIS INFORMATION FOR " ^nam3":?"
POSITION 19,0
MESSAGE "FOR STRAIGHT, PLEASE INPUT ---------> d "
POSITION 20,0
MESSAGE "FOR NOT STRAIGHT, PLEASE INPUT -----> s "
POSITION 21,0
READ tex0
IF tex0 = d
    LET nam4 = straight
ELSE
    IF tex0 = s
        LET nam4 = not_straight
    ENDIF
ENDIF
POSITION 22,0
```

```
MESSAGE "PLEASE ENTER THE SYMMETRY INFORMATION FOR " ^nam3":?"
POSITION 23,0
MESSAGE "FOR SYMMETRIC, PLEASE INPUT ------------> d "
POSITION 24,0
MESSAGE "FOR NOT SYMMETRIC, PLEASE INPUT -----> s "
POSITION 25,0
READ tex0
IF tex0 = d
    LET nam5 = symmetric
ELSE
    IF tex0 = s
        LET nam5 = not_symmetric
    ENDIF
ENDIF
POSITION 26,0
MESSAGE "PLEASE ENTER THE ENTRY/EXIT RELATION INFORMATION FOR " ^nam3":?"
POSITION 27,0
MESSAGE "FOR SAME, PLEASE INPUT ------------> s "
POSITION 28,0
MESSAGE "FOR NOT SAME, PLEASE INPUT -----> d "
POSITION 29,0
READ tex0
IF tex0 = s
    LET nam6 = same
ELSE
    IF tex0 = d
        LET nam6 = not_same
    ENDIF
ENDIF
POSITION 30,0
MESSAGE "PLEASE ENTER THE FORM VARIATION INFORMATION FOR " ^nam3":?"
POSITION 31,0
MESSAGE "FOR CONSTANT, PLEASE INPUT -----------> c "
POSITION 32,0
MESSAGE "FOR TAPERED, PLEASE INPUT ---------------> s "
POSITION 33,0
MESSAGE "FOR CONCAVE, PLEASE INPUT -------------> v "
POSITION 34,0
MESSAGE "FOR CONVEX, PLEASE INPUT ----------------> x "
POSITION 35,0
MESSAGE "FOR COUNTOURED, PLEASE INPUT --------> u "
POSITION 36,0
READ tex0
IF tex0 = c
    LET nam7 = constant
ELSE
    IF tex0 = s
        LET nam7 = tapered
    ELSE
        IF tex0 = v
            LET nam7 = concave
        ELSE
            IF tex0 = x
                LET nam7 = convex
            ELSE
                IF tex0 = u
```

```
                    LET nam7 = contoured
                ENDIF
            ENDIF
        ENDIF
    ENDIF
ENDIF
CLEAR SIZE 36,80 AT 5,0
POSITION 6,0
MESSAGE "PLEASE ENTER THE ROUNDNES VALUE FOR " ^nam3":?"
READ dec07
POSITION 7,0
MESSAGE "PLEASE ENTER THE CYLINDRICITY VALUE FOR " ^nam3":? "
READ dec08
POSITION 8,0
MESSAGE "PLEASE ENTER THE SURFACE FINISH VALUE FOR " ^nam3":? "
READ dec18
LET tex1 = ^nam2:"axis"
IF tell tex1 IS axiscode

ELSE
    FACT tex1 IS axiscode
ENDIF
LET nam8 = tex1
FACT IN TABLE fea_data
^nam2
^dec11
^dec13
^dec12
^dec9
^dec14
^dec10
^dec00
^dec01
^dec02
^dec03
^dec04
^dec05
^nam4
^nam5
^nam6
^nam7
^dec07
^dec08
^dec18
^nam8

CLEAR SIZE 36,80 AT 4,0
```

B.2.4 PROCEDURE FILE 'do_slot'

```
* DO GENERIS FILE 'do_slot'
*
POSITION 6,0
MESSAGE "PLEASE ENTER THE X POSITION FOR ORIGIN OF THE FEATURE " ^nam3":? "
READ dec11
POSITION 7,0
MESSAGE "PLEASE ENTER THE Y POSITION FOR ORIGIN OF THE FEATURE " ^nam3":? "
READ dec13
POSITION 8,0
MESSAGE "PLEASE ENTER THE Z POSITION FOR ORIGIN OF THE FEATURE " ^nam3":?"
READ dec12
POSITION 9,0
MESSAGE "PLEASE ENTER THE X ORIENTATION OF THE FEATURE " ^nam3":?"
READ dec09
POSITION 10,0
MESSAGE "PLEASE ENTER THE Y ORIENTATION OF THE FEATURE " ^nam3":?"
READ dec14
POSITION 11,0
MESSAGE "PLEASE ENTER THE Z ORIENTATION OF THE FEATURE " ^nam3":?"
READ dec10
POSITION 12,0
MESSAGE "PLEASE ENTER THE LENGTH FOR " ^nam3":?"
READ dec03
POSITION 13,0
MESSAGE "PLEASE ENTER THE POSITIVE LENGTH TOLERENCE FOR " ^nam3":?"
READ dec04
POSITION 14,0
MESSAGE "PLEASE ENTER THE NEGATIVE LENGTH TOLERENCE FOR " ^nam3":?"
READ dec05
POSITION 15,0
MESSAGE "PLEASE ENTER THE WIDTH FOR " ^nam3":?"
READ dec15
POSITION 16,0
MESSAGE "PLEASE ENTER THE POSITIVE WIDTH TOLERENCE FOR " ^nam3":?"
READ dec16
POSITION 17,0
MESSAGE "PLEASE ENTER THE NEGATIVE WIDTH TOLERENCE FOR " ^nam3":?"
READ dec17
POSITION 18,0
MESSAGE "PLEASE ENTER THE DEPTH FOR " ^nam3":? "
READ dec00
POSITION 19,0
MESSAGE "PLEASE ENTER THE POSITIVE DEPTH TOLERENCE FOR " ^nam3":?"
READ dec01
POSITION 20,0
MESSAGE "PLEASE ENTER THE NEGATIVE LENGTH DEPTH TOLERENCE FOR " ^nam3":?"
READ dec02
POSITION 21,0
MESSAGE "PLEASE ENTER THE DEPTH AXIS INFORMATION FOR " ^nam3":?"
POSITION 22,0
MESSAGE "FOR STRAIGHT, PLEASE INPUT -----------> d "
POSITION 23,0
MESSAGE "FOR NOT STRAIGHT, PLEASE INPUT -----> s "
```

```
POSITION 24,0
READ tex0
IF tex0 = d
    LET nam4 = straight
ELSE
    IF tex0 = s
        LET nam4 = not_straight
    ENDIF
ENDIF
POSITION 25,0
MESSAGE "PLEASE ENTER THE SYMMETRY INFORMATION FOR " ^nam3":?"
POSITION 26,0
MESSAGE "FOR SYMMETRIC, PLEASE INPUT -----------> d "
POSITION 27,0
MESSAGE "FOR NOT SYMMETRIC, PLEASE INPUT -----> s "
POSITION 28,0
READ tex0
IF tex0 = d
    LET nam5 = symmetric
ELSE
    IF tex0 = s
        LET nam5 = not_symmetric
    ENDIF
ENDIF
POSITION 29,0
MESSAGE "PLEASE ENTER THE ENTRY/EXIT RELATION INFORMATION FOR " ^nam3":?"
POSITION 30,0
MESSAGE "FOR SAME, PLEASE INPUT -----------> s "
POSITION 31,0
MESSAGE "FOR NOT SAME, PLEASE INPUT -----> d "
POSITION 32,0
READ tex0
IF tex0 = s
    LET nam6 = same
ELSE
    IF tex0 = d
        LET nam6 = not_same
    ENDIF
ENDIF
POSITION 33,0
MESSAGE "PLEASE ENTER THE FORM VARIATION INFORMATION FOR " ^nam3":?"
POSITION 34,0
MESSAGE "FOR CONSTANT, PLEASE INPUT -----------> c "
POSITION 35,0
MESSAGE "FOR TAPERED, PLEASE INPUT ---------------> s "
POSITION 36,0
MESSAGE "FOR CONCAVE, PLEASE INPUT --------------> v "
POSITION 37,0
MESSAGE "FOR CONVEX, PLEASE INPUT ----------------> x "
POSITION 38,0
MESSAGE "FOR COUNTOURED, PLEASE INPUT --------> u "
POSITION 39,0
READ tex0
IF tex0 = c
    LET nam7 = constant
ELSE
```

```
    IF tex0 = s
        LET nam7 = tapered
    ELSE
        IF tex0 = v
            LET nam7 = concave
        ELSE
            IF tex0 = x
                LET nam7 = convex
            ELSE
                IF tex0 = u
                    LET nam7 = contoured
                ENDIF
            ENDIF
        ENDIF
    ENDIF
ENDIF
CLEAR SIZE 36,80 AT 5,0
POSITION 8,0
MESSAGE "PLEASE ENTER THE STRAIGHTNESS VALUE FOR " ^nam3":?"
READ dec06
POSITION 9,0
MESSAGE "PLEASE ENTER THE FLATNESS VALUE FOR " ^nam3":?"
READ dec08
POSITION 10,0
MESSAGE "PLEASE ENTER THE SURFACE FINISH VALUE FOR " ^nam3":?"
READ dec18
POSITION 11,0
MESSAGE "PLEASE ENTER THE RADIUS VALUE FOR " ^nam3":?"
POSITION 12,0
READ dec07
LET tex1 = ^nam2:"axis"
IF tell tex1 IS axiscode

ELSE
    fact tex1 IS axiscode
ENDIF
LET nam8 = tex1
FACT IN TABLE fea_data
^nam2
^dec11
^dec13
^dec12
^dec09
^dec14
^dec10
^dec03
^dec04
^dec05
^dec15
^dec16
^dec17
^dec00
^dec01
^dec02
^nam4
^nam5
```

```
^nam6
^nam7
^dec06
^dec08
^dec18
^nam8
^dec07

CLEAR SIZE 36,80 AT 4,0
```

B.2.5 PROCEDURE FILE 'do_surface'

```
* DO GENERIS FILE 'do_surface'
*
POSITION 6,0
MESSAGE "PLEASE ENTER THE X POSITION FOR ORIGIN OF THE FEATURE " ^nam3":? "
READ dec11
POSITION 7,0
MESSAGE "PLEASE ENTER THE Y POSITION FOR ORIGIN OF THE FEATURE " ^nam3":? "
READ dec13
POSITION 8,0
MESSAGE "PLEASE ENTER THE Z POSITION FOR ORIGIN OF THE FEATURE " ^nam3":?"
READ dec12
POSITION 9,0
MESSAGE "PLEASE ENTER THE X ORIENTATION OF THE FEATURE " ^nam3":?"
READ dec09
POSITION 10,0
MESSAGE "PLEASE ENTER THE Y ORIENTATION OF THE FEATURE " ^nam3":?"
READ dec14
POSITION 11,0
MESSAGE "PLEASE ENTER THE Z ORIENTATION OF THE FEATURE " ^nam3":?"
READ dec10
POSITION 12,0
MESSAGE "PLEASE ENTER THE LENGTH FOR " ^nam3":?"
READ dec03
POSITION 13,0
MESSAGE "PLEASE ENTER THE POSITIVE LENGTH TOLERENCE FOR " ^nam3":?"
READ dec04
POSITION 14,0
MESSAGE "PLEASE ENTER THE NEGATIVE LENGTH TOLERENCE FOR " ^nam3":?"
READ dec05
POSITION 15,0
MESSAGE "PLEASE ENTER THE WIDTH FOR " ^nam3":?"
READ dec15
POSITION 16,0
MESSAGE "PLEASE ENTER THE POSITIVE WIDTH TOLERENCE FOR " ^nam3":?"
READ dec16
POSITION 17,0
MESSAGE "PLEASE ENTER THE NEGATIVE WIDTH TOLERENCE FOR " ^nam3":?"
READ dec17
POSITION 18,0
MESSAGE "PLEASE ENTER THE STRAIGHTNESS VALUE FOR " ^nam3":?"
READ dec06
POSITION 19,0
MESSAGE "PLEASE ENTER THE FLATNESS VALUE FOR " ^nam3":?"
```

```
READ dec08
POSITION 20,0
MESSAGE "PLEASE ENTER THE SURFACE FINISH VALUE FOR " ^nam3":?"
READ dec18
FACT IN TABLE fea_data
^nam2
^dec11
^dec13
^dec12
^dec09
^dec14
^dec10
^dec03
^dec04
^dec05
^dec15
^dec16
^dec17

^dec06
^dec08
^dec18

CLEAR SIZE 36,80 AT 4,0
```

B.2.6 PROCEDURE FILE 'do_thread'

```
* DO GENERIS FILE 'do_thread'
*
POSITION 6,0
MESSAGE "PLEASE ENTER THE TYPE OF THE THREAD:?"
READ tex0
POSITION 7,0
MESSAGE "PLEASE ENTER THE NOMINAL DIAMETER FOR  THE THREAD:?"
READ dec02
POSITION 8,0
MESSAGE "PLEASE ENTER THE LENGTH FOR THE THREAD:?"
READ dec03
FACT IN TABLE sec_features_data
^nam11
^tex0
^dec03
^dec02

CLEAR SIZE 36,80 AT 4,0
```

Appendix C

Table of Contents

C PART GROUPING AROUND COMPOSITE COMPONENT/CELL CAPABILITIES

C.1 SOFTWARE IMPLEMENTATION

This section briefly discusses the major components of the software developed for grouping the parts around the mediods. Modules can interact with each other through the data input and output. The directory in which the software programs are developed is called 'aroog'. After changing the home directory to the directory mentioned, the system starts by typing in 'generis' on the prompt. The GENERIS Expert System opens its window for use. The next command to be entered is 'do generisinit'. A user defined window called 'WELCOME' is invoked. The password will be asked for to get the main menu. The password is simply <return>. After entering the password, the main menu will be displayed in another user defined window called 'MAIN MENU'. The main menu facilitates the selection of different system modules. The title of the menu window is 'SELECTION' and the menu will comprise of the following modules:

1. TO CREATE THE RECORDS FOR THE PARTS
2. TO CREATE THE CAPABILITIES OF A CELL
3. TO DELETE THE CAPABILITIES OF ANY CELL
4. TO GROUP THE COMPONENTS AROUND THE CELLS
5. END

The details of the main menu are as below:

Selection 1 is for inputting design data of the component into the system. It works interactively and writes the feature-based component data in the system in the appropriate tables.

Selection 2 is to define the cell capabilities/composite component around which the components are to be grouped. By selecting this choice or selection, another menu called menu_1 will be displayed which provides the chance to define the composite component based on processes and also composite component based on the features. The title of the menu is 'CELL CAPABILITIES'. The sub-menu will look like:

1. TO CREATE THE CAPABILITIES OF A CELL BASED ON PROCESSES
2. TO CREATE THE CAPABILITIES OF A CELL BASED ON FEATURES
3. END

In the above mentioned menu, *selection 1* is to create the cell capabilities based on the processes. *Selection 2* will define the composite component based on the features and *selection 3* is to end the session. Cell capabilities based on processes include:

a) The processes the cell can offer,
b) The materials the cell has the capability to machine,
c) The size or envelope of the components the cell can accommodate.

and the cell capabilities based on features can be given in the following:

a) The features the cell can machine and the extent of surface finish for each feature,
b) The materials the cell has the capability to machine,
c) The size or envelope of the components the cell can accommodate.

Selection 3 in the main menu is to delete the cell capabilities such that a new composite component can be defined. *Selection 4* in the main menu is to group the parts around the centre of groups or composite component already defined. By hitting this selection, another sub-menu will appear. The title of sub-menu window is 'CELL ASSIGNING'. The menu is given in the following:

1. TO ASSIGN THE COMPONENTS TO THE CELLS BASED ON PROCESSES
2. TO ASSIGN THE COMPONENTS TO THE CELLS BASED ON FEATURES
3. END

In the above mentioned menu, *selection 1* is responsible for grouping the components around the composite components which are defined based on the processes, while selection 2 will be grouping the components around the composite components which are defined based on the features. *Selection 3* is to end the session.

Selection 1 will invoke the procedure file called 'do_process_main' which comprises the four sub-modules. The name of the sub-modules are; 1) do_result1, 2) do_potential_cell, 3) do_cell_similarity, and 4) do_op_cell. Sub-module 1 is responsible for attaching the TSF's to each feature of the part. The user defined form to show the results from this module is 'result1_form'. The output from this module will be parts and their features along with the processing needs for each feature. The result from sub-module 2 show for each individual feature for all the parts against each cell whether they can be processed or not. The answer will be in terms of yes or no. The output heading from this module will look like:

Part	Featurecode	Potential Cell	Whether can be done

The answer will be yes, if the feature can be processed, the material of the component can be processed in the cell, and the size of the component is within the size envelope limits defined for the cell. Sub-module 3 calculates the similarity level of each component to each cell in the percentage terms. The output heading from this module will show:

Cell	Part	Similarity Level

The output form showing the results from this module is 'cell_similarity_form'. The last sub-module assigns the components to the cells for which they have more similarity level. The output from this module will be cells and the components grouped around these cells.

In the case of *selection 2* which is 'TO ASSIGN THE COMPONENTS TO THE CELLS BASED ON FEATURES', the procedure file involved is 'do_feature_main' which consists of the three sub-modules. The sub-modules are; 1) do_potential_cell1, 2) do_cell_similarity, and 3) do_op_cell. Sub-module 1 will tell feature by feature for each part which one can be processed against each cell. The output heading will be as shown below:

Part	Featurecode	Potential Cell	Whether can be done

Under the heading 'whether can be done', will be yes or no for each part feature against each cell. Sub-module 2 will show for each part, the cells and the level of similarity, and sub-module 3 will assign the components to the cells for which they have maximum level of similarity.

In the case of a tie or in other words if for any particular component the similarity level is same for more than one cell, the component should be assigned to the cell where it can be processed with least set-up involvement. This aspect has not been implemented in the software.

The main software programs developed for grouping around composite component/cell capabilities, described in section 5.3.1, are given in this appendix. The software programs hierarchy (the sequence in which they are called) designed for the grouping is given in figure C.1.

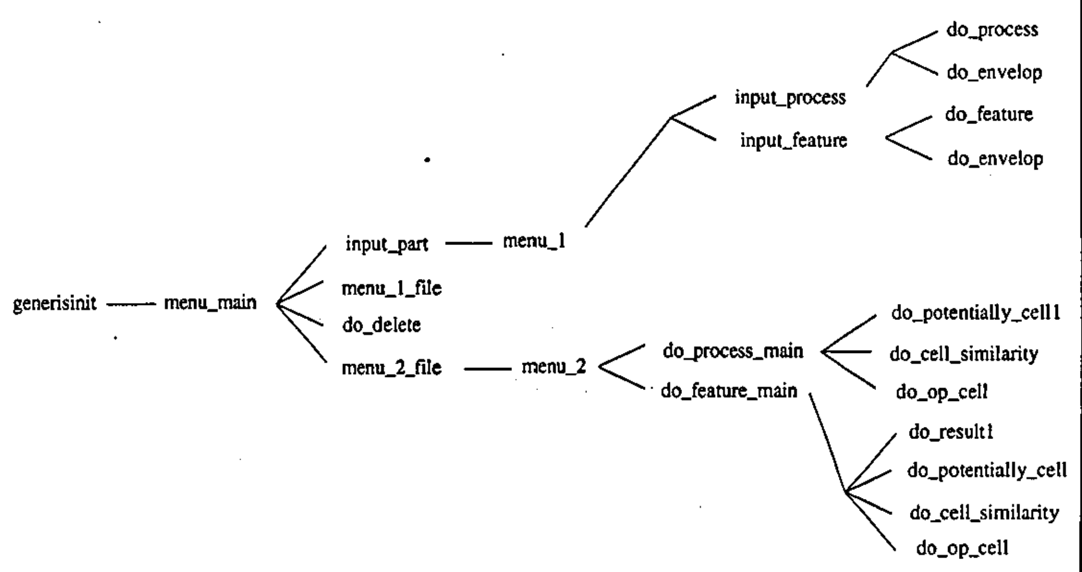


Figure C.1 Software programs hierarchy for grouping around composite component/call capabilities.

C.2 PROCEDURE FILE 'generisInit'

```
*
* DO GENERIS FILE 'generisinit'
*
* START UP FILE FOR THE DEMONSTRATION
*
WINDOW NEW WELCOME SIZE 37,92 AT 2,2
POSITION 5,0
MESSAGE "\THE APPLICATION IS OPENED ONCE YOU HAVE INPUT THE PASSWARD\ \"
OPEN fahd
CLEAR
WINDOW NEW MAIN_MENU SIZE 37,92 AT 2,2
POSITION 9,20
MESSAGE "PLEASE MAKE SELECTION AS PER REQUIREMENTS \"
POSITION 12,18
MENU menu_main
WINDOW COMMAND
DELETE WINDOW ALL
CLOSE
```

C.3 PROCEDURE MENU FILE 'menu_main'

```
*
* GENERIS MENU FILE 'menu_main'
*
TITLE SELECTION
TO CREATE THE RECORDS FOR THE PARTS:DO input_part;
TO CREATE THE CAPABILITIES OF THE CELLS:DO menu_1_file;
TO DELETE THE CAPABILITIES OF ANY CELL:DO do_delete;
TO ASSIGN THE COMPONENTS TO THE CELLS:DO menu_2_file;
END:MENU RETURN;
```

C.4 PROCEDURE FILE 'menu_1_file'

```
*
* DO GENERIS FILE 'menu_1_file'
*
WINDOW NEW CELL_CAPABILITIES SIZE 37,92 AT 2,2
POSITION 9,21
MESSAGE "PLEASE MAKE SELECTION AS PER REQUIREMENTS \"
POSITION 12,10
MENU menu_1
WINDOW MAIN_MENU
DELETE WINDOW CELL_CAPABILITIES
```

C.5 PROCEDURE FILE 'menu_1'

```
*
* GENERIS MENU FILE 'menu_1'
*
CLEAR
TITLE CELL CAPABILITIES
TO CREATE THE CAPABILITIES OF THE CELLS BASED ON PROCESSES:DO input_process;
TO CREATE THE CAPABILITIES OF THE CELLS BASED ON FEATUERS:DO input_feature;
END:MENU RETURN;
```

C.6 PROCEDURE FILE 'input_process'

```
*
* DO GENERIS FILE 'input_process'
*
* FILE TO INPUT THE PROCESSES
*
CREATE LOCAL dec00 DECIMAL
CREATE LOCAL dec01 DECIMAL
CREATE LOCAL dec02 DECIMAL
CREATE LOCAL dec03 DECIMAL
CREATE LOCAL dec04 DECIMAL
CREATE LOCAL dec05 DECIMAL
CREATE LOCAL dec06 DECIMAL
CREATE LOCAL dec07 DECIMAL
CREATE LOCAL int5 INTEGER 1
CREATE LOCAL tex0 TEXT
CREATE LOCAL tex1 TEXT
CREATE LOCAL tex2 TEXT
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
CLEAR
POSITIONAM1,15
MESSAGE "CREATING THE CELL CAPABILITIES BASED ON PROCESSES"
LABEL again
POSITION5,0
MESSAGE "DO YOU WANT TO ENTER THE CAPABILITIES OF CELL NO. "int5"  ====>   y
FOR YES"
POSITION6,0
MESSAGE " <RETURN> FOR NEXT CELL "
READ tex0
IF tex0  = y
    LABEL sarmad
    LET tex1 = "C_":int5
    IF TELL tex1 IS A cell
        LET nam1 = tex1
    ELSE
        FACT tex1 IS cell
        LET nam1 = tex1
    ENDIF
ELSE
    POSITION7,0
    MESSAGE " DO YOU WANT TO GO FOR NEXT CELL "int5 + 1" y/YES AND <RETURN>
    FOR FINISHING"
    READ tex0
    IF tex0 = y
        CLEAR SIZE 40,80 AT 4,0
        LET int5 = int5 + 1
        GO TO again
    ELSE
        GO TO shakeel
    ENDIF
ENDIF
IF TELL ANY RECORDS IN TABLE cell_operation FOR ^ nam1
    POSITION7,0
    MESSAGE "CAPABILITIES FOR CELL NO. "int5" ALREADY EXIST IN TERMS OF
    PROCESSES"
    POSITION8,0
    MESSAGE "DO YOU WANT TO RE-ENTER ?  y FOR YES AND n FOR NO  "
    READ tex0
    IF tex0 = n
        LET int5 = int5 + 1
        CLEAR SIZE 38,90 AT 4,0
        GO TO again
    ELSE
        IF tex0 = y
            DELETE RECORDS IN TABLE cell_operation FOR ^nam1
            IF TELL ANY RECORDS IN TABLE cell_material FOR ^nam1
                DELETE RECORDS IN TABLE cell_material FOR ^nam1
            ENDIF
            IF TELL ANY RECORDS IN TABLE envelop FOR ^nam1
                DELETE RECORDS IN TABLE envelop FOR ^nam1
                ENDIF
            ENDIF
        ENDIF
    ENDIF
    CLEAR SIZE 38,90 AT 2,0
    POSITIONAM2,10
    MESSAGE"NOW THE CELL CAPABILITIES WILL BE ENTERED IN TERMS OF PROCESSES "
    POSITION4,16
    MESSAGE"CELL CAPABILITIES FOR THE CELL " ^nam1" "
    IF TELL ANY RECORDS IN TABLE cell_operation FOR ^ nam1
        DELETE RECORDS IN TABLE cell_operation FOR ^nam1
    ENDIF
    POSITION6,0
    MESSAGE "DOES THE OPERATION  drilling  EXISTS ? y FOR YES"
    READ tex0
    IF tex0 = y
        LET nam2 = drilling
        DO do_process
    ENDIF
    POSITION7,0
    MESSAGE "DOES THE OPERATION  hor_milling  EXISTS ? y FOR YES"
    READ tex0
    IF tex0 = y
    LET nam2 = hor_milling
    DO do_process
    ENDIF
    POSITION8,0
    MESSAGE "DOES THE OPERATION  ver_milling  EXISTS ? y FOR YES"
    READ tex0
    IF tex0 = y
        LET nam2 = ver_milling
        DO do_process
    ENDIF
    POSITION9,0
    MESSAGE "DOES THE OPERATION  hor_drilling  EXISTS ? y FOR YES"
    READ tex0
    IF tex0 = y
        LET nam2 = hor_drilling
        DO do_process
    ENDIF
    POSITIONAM10,0
    MESSAGE "DOES THE OPERATION  turning  EXISTS ? y FOR YES"
    READ tex0
    IF tex0 = y
        LET nam2 = turning
        DO do_process
    ENDIF
    POSITIONAM11,0
    MESSAGE "DOES THE OPERATION  reaming  EXISTS ? y FOR YES"
    READ tex0
    IF tex0 = y
        LET nam2 = reaming
        DO do_process
    ENDIF
    POSITIONAM12,0
    MESSAGE "DOES THE OPERATION  taping  EXISTS ? y FOR YES"
```

```
READ tex0
IF tex0 = y
    LET nam2 = taping
    DO do_process
ENDIF
POSITIONAM13,0
MESSAGE "DOES THE OPERATION  spotfacing  EXISTS ? y FOR YES"
READ tex0
IF tex0 = y
    LET nam2 = spotfacing
    DO do_process
ENDIF
POSITIONAM14,0
MESSAGE "DOES THE OPERATION  countersinking  EXISTS ? y FOR YES"
READ tex0
IF tex0 = y
    LET nam2 = countersinking
    DO do_process
ENDIF
POSITIONAM15,0
MESSAGE "DOES THE OPERATION  counterboring  EXISTS ? y FOR YES"
READ tex0
IF tex0 = y
    LET nam2 = counterboring
    DO do_process
ENDIF
POSITIONAM16,0
MESSAGE "DOES THE OPERATION  hor_taping  EXISTS ? y FOR YES"
READ tex0
IF tex0 = y
    LET nam2 = hor_taping
    DO do_process
ENDIF
POSITIONAM20,0
MESSAGE "IS THERE ANY OTHER OPERATION WHICH WAS NOT IN THE LIST y FOR YES"
READ tex0
IF tex0 = y
    LABEL aroog
    POSITIONAM21,0
    MESSAGE "PLEASE ENTER THE OPERATION FOR THE CELL "^nam1 " ?"
    READ tex0
    IF TELL tex0 IS operation
        LET nam2 = tex0
    ELSE
        FACT tex0 IS operation
        LET nam2 = tex0
    ENDIF
    DO do_process
    POSITIONAM22,0
    MESSAGE "MORE OPERATION TO BE ENTERED FOR THE CELL "^nam1 " ?"
    READ tex0
    IF tex0 = y
        GO TO aroog
    ENDIF
ENDIF
LABEL aroog1
```

```
CLEAR SIZE 36,90 AT 6,0
POSITION6,0
MESSAGE "PLEASE ENTER THE MATERIAL WHICH THE CELL "^nam1 " CAN PROCESS ?"
READ tex0
IF tex0 = BLANK
    POSITION5,0
    MESSAGE"YOU HAVE NOT ENTERED THE MATERIAL "
    GO TO aroog1
ELSE
    IF TELL tex0 IS material
        LET nam3 = tex0
    ELSE
        FACT tex0 IS material
        LET nam3 = tex0
    ENDIF
    IF TELL ANY RECORDS IN TABLE cell_material FOR ^nam1
        DELETE RECORDS IN TABLE cell_material FOR ^nam1
    ENDIF
    FACT IN cell_material
    ^nam1
    ^nam3

    POSITION8,0
    MESSAGE "CAN THE CELL  "^nam1 " CAN PROCESS MORE MATERIAL  ? y/YES"
    READ tex0
    IF tex0 = y
        GO TO aroog1
    ENDIF
ENDIF
CLEAR SIZE 36,90 AT 5,0
POSITION5,0
MESSAGE "NOW THE DATA ABOUT THE DIMENSIONS OF THE COMPONENTS IT CAN
ENTERTAIN WILL BE ENTERED "
DO do_envelop
POSITION5,0
MESSAGE "DO YOU WANT TO ENTER THE CAPABILITIES OF ANOTHER CELL ? y FOR
YES "
READ tex0
IF tex0 = y
    LET int5 = int5 +1
    GO TO sarmad
ENDIF
LABEL shakeel
CLEAR
WINDOW CELL_CAPABILITIES
POSITION 8,28
MESSAGE"PLEASE SELECT AS PER REQUIREMENTS "
```

C.7 PROCEDURE FILE 'do_process'

```
*
* DO GENERIS FILE 'do_process'
*
FACT IN cell_operation
^n1
^n2
```

C.8 PROCEDURE FILE 'do_envelop'

```
*
* DO GENERIS FILE 'do_envelop'
*
IFTELL ANY RECORDS IN envelop FOR ^nam1
    DELETE RECORDS IN TABLE envelop FOR ^nam1
ENDIF
POSITION 6,0
MESSAGE "PLEASE ENTER THE MINIMUM LENGTH OF THE COMPONENT IT CAN
ACCOMODATE ? "
READ dec00
POSITION 7,0
MESSAGE "PLEASE ENTER THE MAXIMUM LENGTH OF THE COMPONENT IT CAN
ACCOMODATE ? "
READ dec01
POSITION 8,0
MESSAGE "PLEASE ENTER THE MINIMUM WIDTH OF THE COMPONENT IT CAN
ACCOMODATE ?"
READ dec02
POSITION 9,0
MESSAGE "PLEASE ENTER THE MAXIMUM WIDTH OF THE COMPONENT IT CAN
ACCOMODATE ? "
READ dec03
POSITION 10,0
MESSAGE "PLEASE ENTER THE MINIMUM HEIGHT OF THE COMPONENT IT CAN
ACCOMODATE ?"
READ dec04
POSITION 11,0
MESSAGE "PLEASE ENTER THE MAXIMUM HEIGHT OF THE COMPONENT IT CAN
ACCOMODATE ? "
READ dec05
POSITION 12,0
MESSAGE "PLEASE ENTER THE MINIMUM DIAMETER OF THE COMPONENT IT CAN
ACCOMODATE ?"
READ dec06
POSITION 13,0
MESSAGE "PLEASE ENTER THE MAXIMUM DIAMETER OF THE COMPONENT IT CAN
ACCOMODATE ?"
READ dec07
*
FACT IN TABLE envelop
^nam1
^dec00
^dec01
^dec02
```

```
^dec03
^dec04
^dec05
^dec06
^dec07

CLEAR SIZE 36,90 AT 2,0
```

C.9 PROCEDURE FILE 'input_feature'

```
*
* DO GENERIS FILE 'input_feature'
*
CREATE LOCAL dec00 DECIMAL
CREATE LOCAL dec01 DECIMAL
CREATE LOCAL dec02 DECIMAL
CREATE LOCAL dec03 DECIMAL
CREATE LOCAL dec04 DECIMAL
CREATE LOCAL dec05 DECIMAL
CREATE LOCAL dec06 DECIMAL
CREATE LOCAL dec07 DECIMAL
CREATE LOCAL int5 INTEGER 1
CREATE LOCAL tex0 TEXT
CREATE LOCAL tex1 TEXT
CREATE LOCAL tex2 TEXT
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
CLEAR
POSITION 1,15
MESSAGE "CREATING THE CELL CAPABILITIES BASED ON FEATURES"
LABEL again
POSITION 5,0
MESSAGE "DO YOU WANT TO ENTER THE CAPABILITIES OF CELL NO. "int5"  ====>   y
FOR YES"
POSITION 6,0
MESSAGE " <RETURN> FOR NEXT CELL"
READ tex0
IF tex0  = y
    LABEL sarmad
    LET tex1 = "C_":int5
    IF TELL tex1 IS A cell
        LET nam1 = tex1
    ELSE
        FACT tex1 IS cell
        LET nam1 = tex1
    ENDIF
ELSE
    POSITION 7,0
    MESSAGE "CAPABILITIES FOR NEXT CELL ? y/yes and <RETURN> FOR FINISH "
    READ tex0
    IF tex0 = y
        CLEAR SIZE 40,80 AT 4,0
        LET int5 = int5 + 1
        GOTO again
    ELSE
        GOTO shakeel
    ENDIF
ENDIF
IF TELL ANY RECORDS IN TABLE cell_feature FOR ^ nam1
    POSITION 7,0
    MESSAGE "CAPABILITIES FOR CELL NO. "int5" ALREADY EXIST IN TERMS OF
    FEATURES "
    POSITION 8,0
    MESSAGE "DO YOU WANT TO REENTER ?  y FOR YES AND n FOR NO  "
    READ tex0
    IF tex0 = n
        LET int5 = int5 + 1
        CLEAR SIZE 38,90 AT 4,0
        GOTO again
    ELSE
        IF tex0 = y
            DELETE RECORDS IN TABLE cell_feature FOR ^nam1
            IF TELL ANY RECORDS IN TABLE cell_material FOR ^nam1
                DELETE RECORDS IN TABLE cell_material FOR ^nam1
            ENDIF
            IF TELL ANY RECORDS IN TABLE envelop FOR ^nam1
                DELETE RECORDS IN TABLE envelop FOR ^nam1
            ENDIF
        ENDIF
    ENDIF
ENDIF
CLEAR SIZE 38,90 AT 2,0
POSITION 2,10
MESSAGE"NOW THE CELL CAPABILITIES WILL BE ENTERED IN TERMS OF FEATURES "
POSITION 4,16
MESSAGE"CELL CAPABILITIES FOR THE CELL " ^nam1" "
IF TELL ANY RECORDS IN TABLE cell_feature FOR ^ nam1
    DELETE RECORDS IN TABLE cell_feature FOR ^nam1
ENDIF
POSITION 6,0
MESSAGE "CAN THE FEATURE contoured_step BE PROCESSED IN CELL " ^nam1" ? y FOR
YES"
READ tex0
IF tex0 = y
    LET nam2 = contoured_step
    POSITION 7,0
    MESSAGE "MINIMUM SURFACE FINISH IT CAN ACCOMPLISH FOR THIS FEATURE ?"
    READ dec00
    DO do_feature
ENDIF
POSITION 8,0
MESSAGE "CAN THE FEATURE triangular_step BE PROCESSED IN CELL " ^nam1" ? y FOR
YES"
READ tex0
IF tex0 = y
    LET nam2 = triangular_step
    POSITION 9,0
    MESSAGE "MINIMUM SURFACE FINISH IT CAN ACCOMPLISH FOR THIS FEATURE ?"
    READ dec00
    DO do_feature
```

```
ENDIF
POSITION 10,0
MESSAGE "CAN THE FEATURE rectangular_slot BE PROCESSED IN CELL " ^nam1" ? y FOR
YES"
READ tex0
IF tex0 = y
    LET nam2 = rectangular_slot
    POSITION 11,0
    MESSAGE "MINIMUM SURFACE FINISH IT CAN ACCOMPLISH FOR THIS FEATURE "
    READ dec00
    DO do_feature
ENDIF
POSITION 12,0
MESSAGE "CAN THE FEATURE round_hole BE PROCESSED IN CELL " ^nam1" ? y FOR YES"
READ tex0
IF tex0 = y
    LET nam2 = round_hole
    POSITION 13,0
    MESSAGE "MINIMUM SURFACE FINISH IT CAN ACCOMPLISH FOR THIS FEATURE ?"
    READ dec00
    DO do_feature
ENDIF
POSITION 14,0
MESSAGE "CAN THE FEATURE axial_round_hole BE PROCESSED IN CELL " ^nam1" ? y FOR
YES"
READ tex0
IF tex0 = y
    LET nam2 = axial_round_hole
    POSITION 15,0
    MESSAGE "MINIMUM SURFACE FINISH IT CAN ACCOMPLISH FOR THIS FEATURE ?"
    READ dec00
    DO do_feature
ENDIF
POSITION 16,0
MESSAGE "CAN THE FEATURE round_pocket BE PROCESSED IN CELL " ^nam1" ? y FOR
YES"
READ tex0
IF tex0 = y
    LET nam2 = round_pocket
    POSITION 17,0
    MESSAGE "MINIMUM SURFACE FINISH IT CAN ACCOMPLISH FOR THIS FEATURE ?"
    READ dec00
    DO do_feature
ENDIF
POSITION 18,0
MESSAGE "CAN THE FEATURE axial_round_pocket BE PROCESSED IN CELL " ^nam1" ? y
FOR YES"
READ tex0
IF tex0 = y
    LET nam2 = axial_round_pocket
    POSITION 19,0
    MESSAGE "MINIMUM SURFACE FINISH IT CAN ACCOMPLISH FOR THIS FEATURE ?"
    READ dec00
    DO do_feature
ENDIF
POSITION 20,0
```

```
MESSAGE "CAN THE FEATURE surface BE PROCESSED IN CELL " ^nam1" ? y FOR YES"
READ tex0
IF tex0 = y
    LET nam2 = surface
    POSITION 21,0
    MESSAGE "MINIMUM SURFACE FINISH IT CAN ACCOMPLISH FOR THIS FEATURE "
    READ dec00
    DO do_feature
ENDIF
POSITION 22,0
MESSAGE "CAN THE FEATURE thread BE PROCESSED IN CELL " ^nam1" ? y FOR YES"
READ tex0
IF tex0 = y
    LET nam2 = thread
    POSITION 23,0
    MESSAGE "MINIMUM SURFACE FINISH IT CAN ACCOMPLISH FOR THIS FEATURE "
    READ dec00
    DO do_feature
ENDIF
POSITION 24,0
MESSAGE "CAN THE FEATURE round_boss BE PROCESSED IN CELL " ^nam1" ? y FOR YES"
READ tex0
IF tex0 = y
    LET nam2 = round_boss
    POSITION 25,0
    MESSAGE "MINIMUM SURFACE FINISH IT CAN ACCOMPLISH FOR THIS FEATURE "
    READ dec00
    DO do_feature
ENDIF
CLEAR SIZE 38,90 AT 5,0
POSITION 7,0
MESSAGE "IS THERE ANY OTHER FEATUER WHICH WAS NOT IN THE LIST y FOR YES"
READ tex0
IF tex0 = y
    LABEL aroog
    POSITION 9,0
    MESSAGE "PLEASE ENTER THE FEATUER FOR THE CELL "^nam1 " ?"
    READ tex0
    IF TELL tex0 IS feature
        LET nam2 = tex0
    ELSE
        FACT tex0 IS feature
        LET nam2 = tex0
    ENDIF
    DO do_feature
    POSITION 11,0
    MESSAGE "MORE FEATUERS BE ENTERED FOR THE CELL "^nam1 " ?"
    READ tex0
    IF tex0 = y
        CLEAR SIZE 36,90 AT 6,0
        GOTO aroog
    ENDIF
ENDIF
LABEL aroog1
CLEAR SIZE 36,90 AT 6,0
POSITION 6,0
```

```
MESSAGE "PLEASE ENTER THE MATERIAL WHICH THE CELL "^nam1 " CAN PROCESS ?"
READ tex0
IF tex0 = BLANK
    POSITION 5,0
    MESSAGE"YOU HAVE NOT ENTERED THE MATERIAL "
    GOTO aroog1
ELSE
    IF TELL tex0 IS material
        LET nam3 = tex0
    ELSE
        FACT tex0 IS material
        LET nam3 = tex0
    ENDIF
    IF TELL ANY RECORDS IN TABLE cell_material FOR ^nam1
        DELETE RECORDS IN TABLE cell_material FOR ^nam1
    ENDIF
    FACT IN cell_material
    ^nam1
    ^nam3

    POSITION 8,0
    MESSAGE "CAN THE CELL  "^nam1 " CAN PROCESS MORE MATERIAL  ? y/YES"
    READ tex0
    IF tex0 = y
        GOTO aroog1
    ENDIF
ENDIF
CLEAR SIZE 36,90 AT 5,0
POSITION 5,0
MESSAGE "NOW THE DATA ABOUT THE DIMENSIONS OF THE COMPONENTS IT CAN
ENTERTAIN WILL BE ENTERED "
DO do_envelop
POSITION 5,0
MESSAGE "DO YOU WANT TO ENTER THE CAPABILITIES OF ANOTHER CELL ? y FOR
YES "
READ tex0
IF tex0 = y
    LET int5 = int5 +1
    GOTO sarmad
ENDIF
LABEL shakeel
CLEAR
WINDOW CELL_CAPABILITIES
POSITION 8,28
MESSAGE"PLEASE SELECT AS PER REQUIREMENTS "
```

C.10 PROCEDURE FILE 'do_feature'

```
*
* DO GENERIS FILE 'do_feature'
*
FACT IN cell_feature
^n1
^n2
^dec00
```

C.11 PROCEDURE FILE 'do_delete'

```
* DO GENERIS FILE 'do_delete'
*
* TO DELETE THE CELL CAPABILITIES OF ANY CELL
*
CREATE LOCAL tex0 TEXT
CREATE LOCAL tex1 TEXT
CREATE LOCAL int5 INTEGER1
CREATE LOCAL nam1 NAME
CLEAR
WINDOW NEW WELCOME  SIZE 40,85 AT 2,2
POSITION 1,20
MESSAGE "DELETING THE CELL CAPABILITIES"
LABEL again
POSITION 5,0
MESSAGE "DO YOU WANT TO DELETE THE CAPABILITIES OF CELL NO. " int5"  ====>   y
FOR YES"
READ tex0
IF tex0  = y
    LABEL sarmad
    LET tex1 = "C_":int5
    IF TELL tex1 IS cell
        LET nam1 = tex1
    ELSE
        POSITION 6,0
        MESSAGE"CELL "^tex1" DOES NOT EXIST "
        CLEAR SIZE 38,90 AT 4,0
        LET int5 = int5 + 1
        GOTO again
    ENDIF
    IF TELL ANY RECORDS IN TABLE cell_feature FOR ^ nam1
    DELETE RECORDS IN TABLE cell_feature FOR ^nam1
    ELSE
        POSITION 7,0
        MESSAGE"RECORDS FOR CELL "^nam1" IN TABLE cell_feature DO NOT EXIST "
        HOLD 2
    ENDIF
    IF TELL ANY RECORDS IN TABLE cell_operation FOR ^nam1
        DELETE RECORDS IN TABLE cell_operation FOR ^nam1
    ELSE
        POSITION 8,0
        MESSAGE"RECORDS FOR CELL "^nam1" IN TABLE cell_operation DO NOT EXIST "
        HOLD 2
    ENDIF
    IF TELL ANY RECORDS IN TABLE cell_material FOR ^nam1
        DELETE RECORDS IN TABLE cell_material FOR ^nam1
    ELSE
        POSITION 9,0
        MESSAGE"RECORDS FOR CELL "^nam1" IN TABLE cell_material DO NOT EXIST "
        HOLD 2
    ENDIF
    IF TELL ANY RECORDS IN TABLE envelop FOR ^nam1
        DELETE RECORDS IN TABLE envelop FOR ^nam1
    ELSE
```

```
        POSITION 10,0
        MESSAGE"RECORDS FOR CELL "^nam1" IN TABLE envelop DO NOT EXIST "
        HOLD 2
    ENDIF
ELSE
    POSITION 6,0
    MESSAGE"DO YOU WANT TO QUIT ? y FOR YES n FOR NO"
    READ tex0
    IF tex0 = y
        GOTO aroog1
    ELSE
        IF tex0 = n
            LET int5 = int5 + 1
            CLEAR SIZE 38,90 AT 4,0
            GOTO again
        ENDIF
    ENDIF
ENDIF
CLEAR SIZE 38,90 AT 4,0
POSITION 5,0
MESSAGE "DO YOU WANT TO DELETE THE CAPABILITIES OF ANOTHER CELL ? y FOR
YES "
READ tex0
IF tex0 = y
    LET int5 = int5 +1
    GOTO sarmad
ENDIF
LABEL aroog1
CLEAR
WINDOW MAIN_MENU
```

C.12 PROCEDURE FILE 'menu_2_file'

```
* DO GENERIS FILE 'menu_2_file'
*
WINDOW NEW CELL_ASSIGNMENT SIZE 37,92 AT 2,2
POSITION 9,21
MESSAGE "PLEASE MAKE SELECTION AS PER REQUIREMENTS \"
POSITION 12,10
MENU menu_2
WINDOW MAIN_MENU
DELETE WINDOW CELL_ASSIGNMENT
```

C.13 PROCEDURE MENU FILE 'menu_2'

```
* GENERIS MENU FILE 'menu_2'
*
CLEAR
TITLE CELL ASSIGNING
TO ASSIGN THE COMPONENTS TO THE CELLS BASED ON PROCESSES:DO
do_process_main;
```

```
TO ASSIGN THE COMPONENTS TO THE CELLS BASED ON FEATURES:DO do_feature_main;
END:MENU RETURN;
```

C.14 PROCEDURE FILE 'do_process_main'

```
* DO GENERIS FILE 'do_process_main'
*
* TO ASSIGN THE COMPONENTS TO THE CELLS BASED ON PROCESSES
WINDOW NEW WELCOME SIZE 37,92 AT 2,2
*
CLEAR
POSITION 2,22
MESSAGE"ASSIGNING THE COMPONENTS TO THE CELLS "
POSITION 3,22
MESSAGE"************************************* "
DO do_result1
DO do_potential_cell
DO do_cell_similarity
DO do_op_cell
CLEAR
POSITION 8,28
MESSAGE"PLEASE SELECT AS PER REQUIREMENTS "
POSITION 9,28
MESSAGE"********************************* "
```

C.15 PROCEDURE FILE 'do_result1'

```
* DO GENERIS FILE 'do_result1'
*
* TO CREATE RECORDS IN TABLE 'result1'
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int5 INTEGER 0
CREATE LOCAL int6 INTEGER 0
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
CREATE LOCAL nam4 NAME
IF TELL ANY RECORDS IN result1
    DELETE RECORDS IN result1
ENDIF
DISABLE ALL
ENABLE oper
FETCH NEW featurecode has feature1 feature potential1 operation
LET fetchno = $FETCH
LET int6 = $COUNT
```

```
WHILE int6 > 0
    LET int1 = int1 + 1
    LET nam0  = FETCH VALUE (fetchno,int1,1)
    IF nam0 = BLANK
        LET nam0 = nam4
    ELSE
        LET nam4 = nam0
    ENDIF
    LET nam1  = FETCH VALUE (fetchno,int1,2)
    IF nam1 = BLANK
        LET nam1 = nam3
    ELSE
        LET nam3 = nam1
    ENDIF
    LET nam2  = FETCH VALUE (fetchno,int1,3)
* n is featurecode
* nam1 is feature
* nam2 is operation
    FACT IN result1
    ^nam0
    ^nam1
    ^nam2

    LET int6 = int6 -1
ENDWHILE
DISABLE ALL
DELETE FETCH fetchno
FETCH NEW featurecode FOR 'has surfinish' surfinish <= 0.5
LET fetchno1 = $FETCH
LET int6 = $COUNT
LET int1 = 0
WHILE int6 > 0
    LET int1 = int1 + 1
    LET nam0 = FETCH VALUE (fetchno1,int1,1)
    FETCH NEW feature 'has feature' FOR ^nam0
    LET fetchno = $FETCH
    LET int5 = $COUNT
    WHILE int5 > 0
        LET int2 = int2 + 1
        LET nam1  = FETCH VALUE (fetchno,int2,3)
        FACT IN result1
        ^nam0
        ^nam1
        reaming

        LET int5 = int5 -1
    ENDWHILE
    DELETE FETCH fetchno
    LET int6 = int6 -1
ENDWHILE
DELETE FETCH fetchno1
DISABLE ALL
DISPLAY RECORDS IN result1
```

C.16 PROCEDURE FILE 'do_potential_cell'

```
* DO GENERIS FILE 'do_potential_cell'
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL fetchno2 INTEGER
CREATE LOCAL fetchno3 INTEGER
CREATE LOCAL fetchno4 INTEGER
CREATE LOCAL lengh_dec DECIMAL
CREATE LOCAL width_dec DECIMAL
CREATE LOCAL depth_dec DECIMAL
CREATE LOCAL dia_dec DECIMAL
CREATE LOCAL min_lengh DECIMAL
CREATE LOCAL min_width DECIMAL
CREATE LOCAL min_depth DECIMAL
CREATE LOCAL min_dia DECIMAL
CREATE LOCAL max_lengh DECIMAL
CREATE LOCAL max_width DECIMAL
CREATE LOCAL max_depth DECIMAL
CREATE LOCAL max_dia DECIMAL
CREATE LOCAL i INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int3 INTEGER 0
CREATE LOCAL int4 INTEGER 0
CREATE LOCAL int5 INTEGER 0
CREATE LOCAL int6 INTEGER 0
CREATE LOCAL int7 INTEGER 0
CREATE LOCAL int8 INTEGER 0
CREATE LOCAL counter INTEGER
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
CREATE LOCAL nam4 NAME
CREATE LOCAL tex0 TEXT
CREATE LOCAL flag TEXT
IF TELL ANY RECORDS IN result1
    IF TELL ANY RECORDS IN results
        DELETE RECORDS IN results
    ENDIF
    FETCH NEW RECORDS IN parts
    LET fetchno = $FETCH
    LET int0 = $COUNT
    WHILE int0 > 0
        LET int1 = int1 + 1
        LET nam0  = FETCH VALUE (fetchno,int1,1)
        LET nam1  = FETCH VALUE (fetchno,int1,5)
*       nam0 is part
*       nam1 is material
        IF nam1 = BLANK
            LET nam1 = steel
        ENDIF
        LET lengh_dec = FETCH VALUE (fetchno,int1,7)
        LET width_dec = FETCH VALUE (fetchno,int1,10)
        LET depth_dec = FETCH VALUE (fetchno,int1,13)
        LET dia_dec   = FETCH VALUE (fetchno,int1,16)
        FETCH NEW RECORDS IN envelop
        LET fetchno1 = $FETCH
        LET int2 = $COUNT
        LET int3 = 0
        WHILE int2 > 0
            LET int3 = int3 + 1
            LET nam2  = FETCH VALUE (fetchno1,int3,1)
* nam2 is cell
            FETCH NEW RECORDS IN envelop FOR ^nam2
            LET fetchno4 = $FETCH
            LET min_lengh = FETCH VALUE (fetchno4,1,2)
            LET max_lengh = FETCH VALUE (fetchno4,1,3)
            LET min_width = FETCH VALUE (fetchno4,1,4)
            LET max_width = FETCH VALUE (fetchno4,1,5)
            LET min_depth = FETCH VALUE (fetchno4,1,6)
            LET max_depth = FETCH VALUE (fetchno4,1,7)
            LET min_dia   = FETCH VALUE (fetchno4,1,8)
            LET max_dia   = FETCH VALUE (fetchno4,1,9)
            DELETE FETCH fetchno4
            LET flag = down
            IF dia_dec = BLANK
                IF lengh_dec >= min_lengh
                    IF lengh_dec <= max_lengh
                        IF width_dec >= min_width
                            IF width_dec <= max_width
                                IF depth_dec >= min_depth
                                    IF depth_dec <= max_depth
                                        LET flag = high
                                    ENDIF
                                ENDIF
                            ENDIF
                        ENDIF
                    ENDIF
                ENDIF
            ELSE
                IF lengh_dec >= min_lengh
                    IF lengh_dec <= max_lengh
                        IF dia_dec >= min_dia
                            IF dia_dec <= max_dia
                                LET flag = high
                            ENDIF
                        ENDIF
                    ENDIF
                ENDIF
            ENDIF
            FETCH NEW RECORDS IN fea_list FOR ^nam0
            LET fetchno2 = $FETCH
            LET int4 = $COUNT
            LET int5 = 0
            WHILE int4 > 0
                LET int5 = int5 + 1
                LET nam3  = FETCH VALUE (fetchno2,int5,2)
* nam3 is featurecode
```

```
                IF TELL ANY ^n 'has featurecode' ^nam3 'has feature' external_surface

                ELSE
                    IF TELL ANY ^nam2 can_process ^nam1
                        FETCH NEW RECORDS IN result1 FOR ^nam3
                        LET fetchno3 = $FETCH
                        LET int6 = $COUNT
                        LET int8 = int6
                        LET int7 = 0
                        LET counter = 0
                        LET tex0 = BLANK
                        WHILE int6 > 0
                            LET int7 = int7 + 1
                            LET nam4  = FETCH VALUE (fetchno3,int7,3)
* nam4 is operation
                            IF TELL ANY ^nam2 'has facility' ^nam4
                                LET counter = counter + 1
                            ENDIF
                            IF int8 = counter
                                IF flag = high
                                    LET tex0  = yes
                                ENDIF
                            ENDIF
                            LET int6 = int6 - 1
                        ENDWHILE
                        DELETE FETCH fetchno3
                        FACT IN results
                        ^nam0
                        ^nam3
                        ^nam2
                        ^tex0

                    ENDIF
                ENDIF
                LET int4 = int4 - 1
            ENDWHILE
            DELETE FETCH fetchno2
            LET int2 = int2 - 1
        ENDWHILE
        DELETE FETCH fetchno1
        LET int0 = int0 - 1
    ENDWHILE
    DELETE FETCH fetchno
    DISPLAY RECORDS IN results
ELSE
    MESSAGE"RECORDS IN TABLE result1 DO NOT EXIST"
    HOLD 5
ENDIF
```

C.17 PROCEDURE FILE 'do_cell_similarity'

```
* DO GENERIS FILE 'do_cell_similarity'
*
* TO CREATE RECORDS IN TABLE 'cells'
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int3 INTEGER 0
CREATE LOCAL dec DECIMAL
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
IF TELL ANY RECORDS IN results
    IF TELL ANY RECORDS IN cells
        DELETE RECORDS IN cells
    ENDIF
    LABEL begin
    FETCH NEW RECORDS IN results
    LET fetchno = $FETCH
    LET nam0  = FETCH value (fetchno,1,1)
    LET nam1  = FETCH value (fetchno,1,3)
* nam0 is part
* nam1 is cell
    DELETE FETCH fetchno
    FETCH NEW RECORDS IN results FOR ^nam0 FOR 'coded as' featurecode FOR 'potential cell'
    ^nam1
    LET fetchno = $FETCH
    LET int2 = $COUNT
    FETCH NEW RECORDS IN results FOR ^nam0 FOR 'coded as' featurecode FOR 'potential cell'
    ^nam1 'has member1' member1
    LET fetchno = $FETCH
    LET int3 = $COUNT
    LET dec = int3/int2
    FACT IN cells
    ^nam1
    ^nam0
    ^dec

    DELETE RECORDS IN results FOR ^nam0 'potential cell' ^nam1
    IF TELL ANY RECORDS IN results
        GOTO begin
    ENDIF
    DISPLAY RECORDS IN cells
ELSE
    HOLD 2
    MESSAGE"RECORDS IN TABLE 'results' DO NOT EXIST"
ENDIF
```

C.18 PROCEDURE FILE 'do_op_cell'

```
* DO GENERIS FILE 'do_op_cell'
*
* TO CREATE RECORDS IN TABLE 'op_cell'
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int3 INTEGER 0
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL dec  DECIMAL
CREATE LOCAL dec1 DECIMAL
CREATE LOCAL big  DECIMAL
IF TELL ANY RECORDS IN cells
    IF TELL ANY RECORDS IN op_cell
        DELETE RECORDS IN op_cell
    ENDIF
    FETCH NEW RECORDS IN parts
    LET fetchno = $FETCH
    LET int0 = $COUNT
    WHILE int0 > 0
        LET int1 = int1 + 1
        LET nam0 = FETCH VALUE (fetchno,int1,1)
* nam0 is part
        FETCH NEW RECORDS IN cells FOR ^nam0
        LET fetchno1 = $FETCH
        LET int2 = $COUNT
        LET int3 = 0
        LET big = 0
        WHILE int2 > 0
            LET int3 = int3 + 1
            LET dec = FETCH VALUE (fetchno1,int3,3)
            IF(big < dec)
                LET big = dec
                LET nam1  = FETCH VALUE (fetchno1,int3,1)
                LET dec1 = FETCH VALUE (fetchno1,int3,3)
* nam1 is cell
* dec1 is similarity value
            ENDIF
            LET int2 = int2 - 1
        ENDWHILE
        DELETE FETCH fetchno1
        FACT IN op_cell
        ^nam1
        ^nam0
        ^dec1

        LET int0 = int0 - 1
    ENDWHILE
    DISPLAY RECORDS IN op_cell
ELSE
```

```
    MESSAGE"RECORDS IN cells DO NOT EXIST  ---> SKIPPING"
    HOLD 2
ENDIF
```

C.19 PROCEDURE FILE 'do_feature_main'

```
* DO GENERIS FILE 'do_feature_main'
*
* TO ASSIGN THE COMPONENTS TO THE CELLS BASED ON FEATURES
*
WINDOW NEW WELCOME SIZE 37,92 AT 2,2
CLEAR
POSITION 2,22
MESSAGE"ASSIGNING THE COMPONENTS TO THE CELLS "
POSITION 3,22
MESSAGE"*****************************************"
DO do_potential_cell1
DO do_cell_similarity
DO do_op_cell
CLEAR
POSITION 8,28
MESSAGE"PLEASE SELECT AS PER REQUIREMENTS "
POSITION 9,28
MESSAGE"************************************ "
```

C.20 PROCEDURE FILE 'do_potential_cell1'

```
* DO GENERIS FILE 'do_potential_cell1'
*
* TO CREATE RECORDS IN TABLE 'results'
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL fetchno2 NTEGER
CREATE LOCAL fetchno3 INTEGER
CREATE LOCAL fetchno4 INTEGER
CREATE LOCAL lengh_dec DECIMAL
CREATE LOCAL width_dec DECIMAL
CREATE LOCAL depth_dec DECIMAL
CREATE LOCAL dia_dec DECIMAL
CREATE LOCAL min_lengh DECIMAL
CREATE LOCAL min_width DECIMAL
CREATE LOCAL min_depth DECIMAL
CREATE LOCAL min_dia DECIMAL
CREATE LOCAL max_lengh DECIMAL
CREATE LOCAL max_width DECIMAL
CREATE LOCAL max_depth DECIMAL
CREATE LOCAL max_dia DECIMAL
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int3 INTEGER 0
CREATE LOCAL int4 INTEGER 0
CREATE LOCAL int5 INTEGER 0
CREATE LOCAL int6 INTEGER 0
CREATE LOCAL int7 INTEGER 0
CREATE LOCAL int8 INTEGER 0
CREATE LOCAL counter INTEGER
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
CREATE LOCAL nam4 NAME
CREATE LOCAL nam5 NAME
CREATE LOCAL tex0 TEXT
CREATE LOCAL flag TEXT
IF TELL ANY RECORDS IN results
    DELETE RECORDS IN results
ENDIF
FETCH NEW RECORDS IN parts
LET fetchno = $FETCH
LET int0 = $COUNT
WHILE int0 > 0
    LET int1 = int1 + 1
    LET nam0  = FETCH VALUE (fetchno,int1,1)
    LET nam1  = FETCH VALUE (fetchno,int1,5)
* nam0 is part
* nam1 is material
    IF nam1 = BLANK
        LET nam1 = steel
    ENDIF
    LET lengh_dec = FETCH VALUE (fetchno,int1,7)
    LET width_dec = FETCH VALUE (fetchno,int1,10)
    LET depth_dec = FETCH VALUE (fetchno,int1,13)
    LET dia_dec   = FETCH VALUE (fetchno,int1,16)
    FETCH NEW RECORDS IN envelop
    LET fetchno1 = $FETCH
    LET int2 = $COUNT
    LET int3 = 0
    WHILE int2 > 0
        LET int3 = int3 + 1
        LET nam2  = FETCH VALUE (fetchno1,int3,1)
*       nam2 is cell
        FETCH NEW RECORDS IN envelop FOR ^nam2
        LET fetchno4 = $FETCH
        LET min_lengh = FETCH VALUE (fetchno4,1,2)
        LET max_lengh = FETCH VALUE (fetchno4,1,3)
        LET min_width = FETCH VALUE (fetchno4,1,4)
        LET max_width = FETCH VALUE (fetchno4,1,5)
        LET min_depth = FETCH VALUE (fetchno4,1,6)
        LET max_depth = FETCH VALUE (fetchno4,1,7)
        LET min_dia   = FETCH VALUE (fetchno4,1,8)
        LET max_dia   = FETCH VALUE (fetchno4,1,9)
        DELETE FETCH fetchno4
        LET flag = down
        IF dia_dec = BLANK
            IF lengh_dec >= min_lengh
                IF lengh_dec <= max_lengh
```

```
                    IF width_dec >= min_width
                         IF width_dec <= max_width
                              IF depth_dec >= min_depth
                                   IF depth_dec <= max_depth
                                        LET flag = high
                                   ENDIF
                              ENDIF
                         ENDIF
                    ENDIF
               ENDIF
          ENDIF
     ELSE
          IF lengh_dec >= min_lengh
               IF lengh_dec <= max_lengh
                    IF dia_dec >= min_dia
                         IF dia_dec <= max_dia
                              LET flag = high
                         ENDIF
                    ENDIF
               ENDIF
          ENDIF
     ENDIF
     FETCH NEW RECORDS IN fea_list FOR ^n
     LET fetchno2 = $FETCH
     LET int4 = $COUNT
     LET int5 = 0
     WHILE int4 > 0
          LET int5 = int5 + 1
          LET nam3  = FETCH VALUE (fetchno2,int5,2)
          LET nam5  = FETCH VALUE (fetchno2,int5,3)
* nam3 is featurecode
* nam5 is feature
          LET int8 = 0
          LET counter = 0
          LET tex0 = BLANK
          IF TELL ANY ^nam0 'has featurecode' ^nam3 'has feature' external_surface

          ELSE
               IF TELL ANY ^nam2 can_process ^nam1
                    IF TELL ANY ^nam2 'provide facility' ^nam5
                         LET counter = counter + 1
                         IF TELL ANY ^nam3 has sec_feature feature
                              FETCH NEW RECORDS IN sec_features FOR ^nam3
                              LET fetchno3 = $FETCH
                              LET int6 = $COUNT
                              LET int8 = int6
                              LET int7 = 0
                              WHILE int6 > 0
                                   LET int7 = int7 + 1
                                   LET nam4  = FETCH VALUE (fetchno3,int7,3)
*                                  nam4 is sec_feature
                                   IF TELL ANY ^nam2 'provide facility' ^nam4
                                        LET counter = counter + 1
                                   ENDIF
                                   LET int6 = int6 - 1
                              ENDWHILE
                    DELETE FETCH fetchno3
                ENDIF
                IF (counter = int8 + 1)
                    IF flag = high
                        LET tex0  = yes
                    ENDIF
                ENDIF
            ENDIF
        ENDIF
        FACT IN results
        ^nam0
        ^nam3
        ^nam2
        ^tex0

      ENDIF
      LET int4 = int4 - 1
    ENDWHILE
    DELETE FETCH fetchno2
    LET int2 = int2 - 1
  ENDWHILE
  DELETE FETCH fetchno1
  LET int0 = int0 - 1
ENDWHILE
DELETE FETCH fetchno
DISPLAY RECORDS IN results
```

Appendix D

Table of Contents

D THE SOFTWARE IMPLEMENTATION FOR THE PART GROUPING AND PATTERN RECOGNITION

D.1 INTRODUCTION

The main software programs developed, the data files involved and implementation in the part grouping based on part geometry, processes, machines and the CAFBG System, are given in this appendix. The software programs hierarchy (the sequence in which they are called) designed for the grouping is given in figure D.1. On the top of the figure, the main menu is shown, which groups parts based on the different criteria discussed in chapter 4. Only the software programs and the data files involved in the CAFBG System are listed in the following to avoid repetition of programs, as programs involved in all the grouping techniques are similar. However, results from different grouping criteria are attached in the appendix. Furthermore, important software programs are attached for the pattern recognition described in section 5.3.5.1. The main start up file 'generisinit', the main menu procedure file called 'menu_main' and 'file_command' procedure file are given in the beginning.

The software programs developed for all the part grouping discussed in chapter 4 except 'Grouping around composite component/cell capabilities' are in a directory called 'fahd'. After changing the home directory to the directory mentioned, the system starts by typing in 'generis' on the prompt. The GENERIS Expert System opens its window for its use. The next command to be entered is 'do generisinit'. A user defined window called 'WELCOME' is invoked. The password will be asked for to get the main menu. The password is simply <return>. After entering the password, the main menu will be displayed in another user defined window called 'MAIN MENU'. The main menu facilitates the selection of different system modules. The title of the menu window is 'SELECTION' and the menu will comprise of the following modules:

1. TO INPUT THE PART GEOMETRY INFORMATION
2. PART GROUPING BASED ON THE PART GEOMETRY
3. PART GROUPING BASED ON THE PROCESSES
4. PART GROUPING BASED ON THE MACHINES
5. PART GROUPING BASED ON CAFBG SYSTEM
6. PATTERN RECOGNITION FOR A NEW PART
7. TYPE ANY COMMAND

The modules mentioned above are self explanatory.

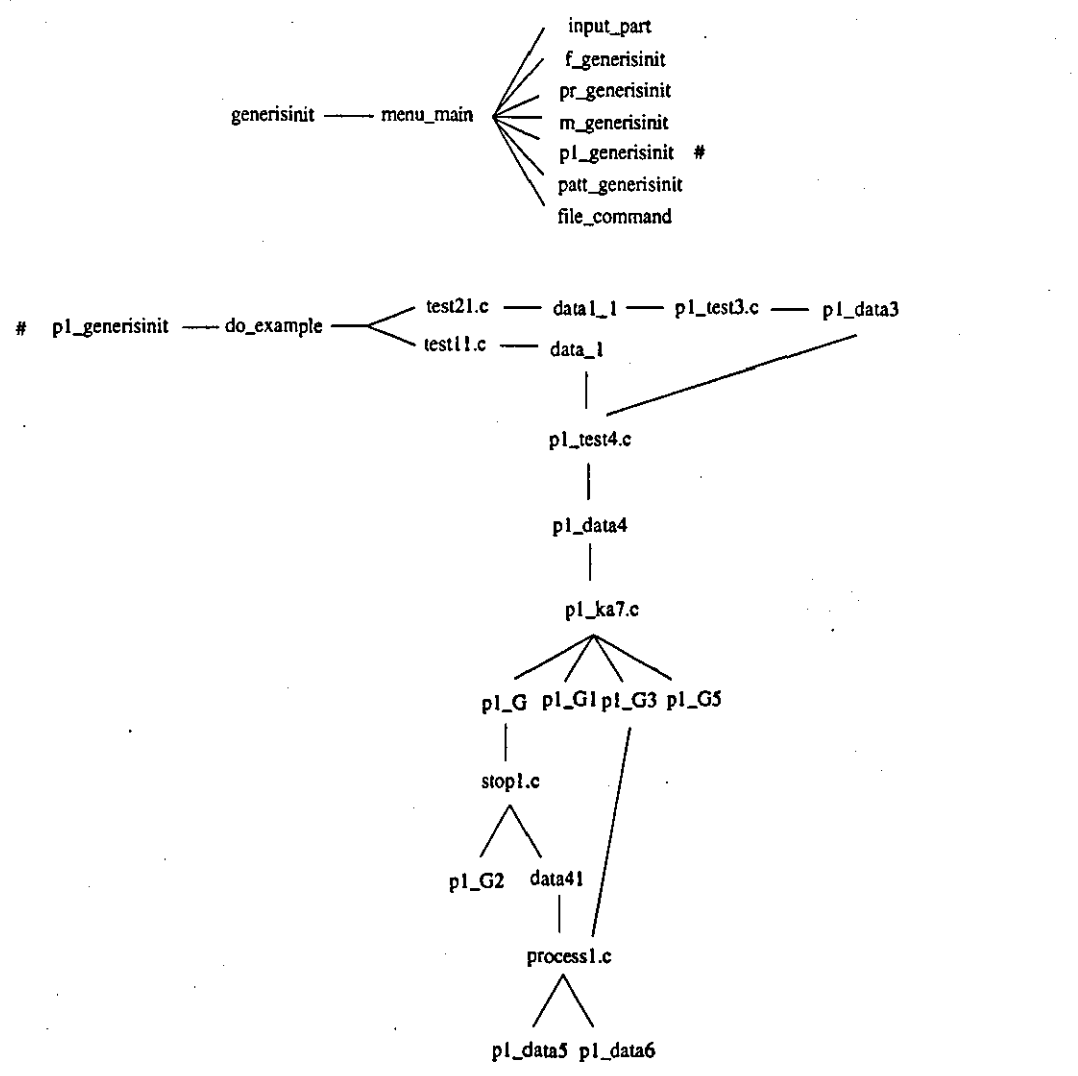


Figure D.1 Main menu for grouping based on different grouping techniques and hierarchy for the software programs and data files involved in the CAFBG System

D.2 PROCEDURE FILE 'generisinit'

```
*
* DO GENERIS FILE 'generisinit'
*
* START UP FILE FOR THE DEMONSTRATION
*
WINDOW NEW MAIN_MENU SIZE 37,92 AT 2,2
POSITION 5,12
MESSAGE "\THE APPLICATION IS OPENED ONCE YOU HAVE INPUT THE PASSWORD\ \"
OPEN fahd
CLEAR
POSITION 9,24
MESSAGE "PLEASE MAKE SELECTION AS PER REQUIREMENTS \"
POSITION 12,20
MENU menu_main
WINDOW COMMAND
DELETE WINDOW ALL
DELETE WINDOW ALL
CLOSE
```

D.3 PROCEDURE MENU FILE 'menu_main'

```
*
* GENERIS MENU FILE 'menu_main'
*
TITLE SELECTION
TO INPUT THE PART GEOMETRY INFORMATION:DO input_part;
PART GROUPING BASED ON THE PART GEOMETRY:DO f_generisinit;
PART GROUPING BASED ON THE PROCESSES:DO pr_generisinit;
PART GROUPING BASED ON THE MACHINES: DO m_generisinit;
PART GROUPING BASED ON CAFBG SYSTEM:DO p1_generisinit;
PATTERN RECOGNITION FOR A NEW PART:DO patt_generisinit
TYPE ANY COMMAND:DO file_command;
END:MENU RETURN;
```

D.4 PROCEDURE FILE 'file_command'

```
*
* GENERIS DO FILE 'file_command'
*
CREATE LOCAL tex1 TEXT
LABEL again
POSITION 12,15
MESSAGE " TYPE ANY COMMAND  "
READ tex1
IF tex1 = BLANK
    POSITION 13,15
    MESSAGE "YOU HAVE NOT INPUT THE COMMAND  "
    HOLD 2
    CLEAR
    GOTO again
ENDIF
DO COMMAND tex1
CLEAR
RETURN
```

D.5 PART GEOMETRY-BASED GROUPING

SOFTWARE IMPLEMENTATION

The extraction of grouping data from the GENERIS application was not made in this case. Other main programs designed for the implementation of software are given below:

1) The incidence matrix file called 'f_data2' is the input to the cluster analysis software program 'f_ka7.c' written in C language. The program works in a loop until all the components are merged into one group or the value of similarity between the groups during the clustering process becomes zero. At each

iteration, it calculates the similarity matrix between the grouping centres existing at that iteration, merges the two most similar groups and then updates the new centre of group. This program writes the results in two files, the details being:

file 'f_W1':	After each iteration the number of groups formed and components in each group and un-grouped components are written in this file.
file 'f_W':	After each iteration the number of groups and similarity level at which two groups merge are written in this file.

2) 'f_W' is the input to the next C program designed called 'f_stop', which calculates the optimal number of groups, or in other words the number of groups at which grouping process should be stopped. It calculates the difference or distance between the similarity levels of every two consecutive iterations. It picks up two consecutive iterations, when the distance between them is maximum. Stopping at this iteration suggests that if the grouping process goes further, the two most dissimilar groups during the whole grouping process will merge. Therefore, grouping should be stopped at that iteration and the number of groups at that point will be optimal. This number showing the optimal number of groups is written in a separate file called 'f_W2'.

D.5.1 PROCEDURE DO FILE 'f_generisinit'

```
*
* GENERIS DO FILE 'f_generisinit'
*
WINDOW NEW INTERFACE DEMONSTRATION SIZE 37,92 AT 2,2
POSITION 5,0
MESSAGE "\THE APPLICATION IS OPENED ONCE YOU HAVE INPUT THE PASSWORD\ \"
OPEN fahd
CREATE LOCAL tex0 TEXT
CLEAR
POSITION 2,7
MESSAGE"GENERIS INTERFACE DEMONSTRATION FOR GROUPING THE PARTS BASED
ON PART GEOMETRY"
POSITION 3,7
MESSAGE"**********************************************************************"
HOLD 2
DISPLAY FILE f_data2
POSITION 8,14
MESSAGE"NOW GROUPING IS BEING DONE BY THE CLUSTER ANALAYSIS PROGRAM "
POSITION 10,36
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix f_ka7
ENDIF
DISPLAY FILE f_W1
DISPLAY FILE f_W
CLEAR SIZE 40,80 AT 8,2
```

```
POSITION 8,10
MESSAGE"CALCULATING THE NUMBER OF GROUPS AT WHICH GROUPING SHOULD BE
STOPPED"
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix f_stop
ENDIF
DISPLAY FILE f_W2
DISPLAY FILE f_W1
CLEAR SIZE 40,80 AT 8,2
POSITION 8,24
MESSAGE"INTERFACING DEMONSTRATION COMPLETE ..."
HOLD 2
WINDOW MAIN_MENU
DELETE WINDOW INTERFACE DEMONSTRATION
```

D.6 PROCESS-BASED GROUPING

SOFTWARE IMPLEMENTATION

For the implementation of process-based grouping, the following major programs have been written:

1) The incidence matrix file called 'pr_data2' is the input to the cluster analysis software program 'pr_ka7.c' written in C language. The program works in the loop until all the components are merged into one group or the value of similarity between the groups during the clustering process becomes zero. At each iteration, it calculates the similarity matrix between the grouping centres existing at that iteration, merges the two most similar groups and then updates the new centre of group. This program writes the results in two files, the details being:

   file 'pr_U1': After each iteration the number of groups formed and components in each group and un-grouped components are written in this file.
   file 'pr_U': After each iteration the number of groups and similarity level at which two groups are merged are written in this file.

2) File 'pr_U' is the input to the next designed C program 'pr_stop.c', which calculates the optimal number of groups or, in other words, the number of groups at which the grouping process should be stopped. It calculates the difference or distance between the similarity levels of every two consecutive iterations. It picks up two consecutive iterations, when the distance between them is maximum. Stopping at this iteration suggests that if we go further, the two most dissimilar groups during the whole grouping process will merge.

Therefore, grouping should be stopped at that iteration and the number of groups at that point of time will be optimal. This number showing the optimal number of groups is written in a separate file 'pr_U2'.

D.6.1 PROCEDURE FILE 'pr_generisinit'

```
*
* GENERIS DO FILE 'pr_generisinit'
*
WINDOW NEW INTERFACE DEMONSTRATION SIZE 37,92 AT 2,2
CREATE LOCAL tex0 TEXT
CLEAR
POSITION 2,7
MESSAGE"GENERIS INTERFACE DEMONSTRATION FOR GROUPING THE PARTS BASED
ON PROCESSES"
POSITION 3,7
MESSAGE"******************************************************************"
DISPLAY FILE pr_data2
CLEAR SIZE 40,80 AT 8,2
POSITION 8,14
MESSAGE"NOW GROUPING IS BEING DONE BY THE CLUSTER ANALAYSIS PROGRAM "
POSITION 10,36
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix pr_ka7
ENDIF
DISPLAY FILE pr_U1
DISPLAY FILE pr_U
CLEAR SIZE 40,80 AT 8,2
POSITION 8,10
MESSAGE"CALCULATING THE NUMBER OF GROUPS AT WHICH GROUPING SHOULD BE
STOPPED"
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix pr_stop
ENDIF
DISPLAY FILE pr_U2
DISPLAY FILE pr_U1
CLEAR SIZE 40,80 AT 8,2
POSITION 8,23
MESSAGE"INTERFACING DEMONSTRATION COMPLETE ..."
HOLD 2
WINDOW MAIN_MENU
DELETE WINDOW INTERFACE DEMONSTRATION
```

D.7 MACHINE-BASED GROUPING

SOFTWARE IMPLEMENTATION

For the implementation of machine-based grouping, the following main programs have been designed:

1) The component design information is input through a user interface which writes the information in the appropriate tables in the GENERIS knowledge base after having dialogues with the user. This interface has been designed in the GENERIS high level language.

2) An interface program written in C high level programming language extracts the information about the parts and the machines visited by them from the GENERIS knowledge base. While running this piece of software, it asks the user to input the database name, 'fahd'. After entering the database name, it will ask for the query. The query this time to be entered is 'part machine'. The output will be written in the file called 'm_data'.

3) After reading information about the machines used by the parts from the file 'm_data', another software program 'm_part_machine.c' designed in C programming language writes in the file 'm_data1', after processing this information into the format needed for the grouping of the parts. This is also called an incidence matrix for the cluster analysis.

4) The incidence matrix is the input to the cluster analysis software program 'm_ka7.c' written in C language. At each iteration, it calculates the similarity matrix between the grouping centres existing at that iteration, merges the two most similar groups and then updates the new centre of group. This program writes the results in two files, the details being:

file 'm_G1':	After each iteration the number of groups formed and components in each group and un-grouped components are written in this file.
file 'm_G':	After each iteration the number of groups and similarity level at which two groups merge are written in this file.

5) Input to the next designed C program 'm_stop.c' is data file 'm_G', which calculates the optimal number of groups. It calculates the difference or distance between the similarity levels of every two consecutive iterations. It picks up two consecutive iterations, when the distance between them is maximum. Stopping at this iteration suggests that if we go further, the two most dissimilar groups during the whole grouping process will merge. Therefore, grouping should be stopped at that iteration and the number of groups at that point will be optimal. This number showing the optimal number of groups is written in a separate file 'm_G2'.

D.7.1 PROCEDURE FILE 'm_generisinit'

```
*
* GENERIS DO FILE 'm_generisinit'
*
WINDOW NEW INTERFACE DEMONSTRATION SIZE 37,92 AT 2,2
POSITION 5,0
MESSAGE "\THE APPLICATION IS OPENED ONCE YOU HAVE INPUT THE PASSWARD\\"
OPEN fahd
CREATE LOCAL tex0 TEXT
CLEAR
POSITION 2,7
MESSAGE"GENERIS INTERFACE DEMONSTRATION FOR GROUPING THE PARTS BASED
ON MACHINES"
POSITION 3,7
MESSAGE"*******************************************************************"
POSITION 4,7
MESSAGE"PLEASR ENTER <RETURN> TWICE AFTER HAVING A LOOK ON THE
RESULTS"
POSITION 6,7
MESSAGE"NOW THE RECORDS ABOUT PARTS AND MACHINES USED BY THEM WILL BE
DISPLAYED"
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    DISPLAY RECORDS IN part_machine
ENDIF
CLEAR SIZE 37,92 AT 6,2
POSITION 8,10
MESSAGE"FETCHING THE MACHINES REQUIRED FOR THE COMPONENTS FROM THE
GENERIS "
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix m_test1
ENDIF
DISPLAY FILE m_data
CLEAR SIZE 37,92 AT 8,2
POSITION 8,14
MESSAGE"CONVERTING THE FETCHED DATA INTO THE FORMAT REQUIRED "
POSITION 9,14
MESSAGE"BY THE CLUSTER ANALYSIS PROGRAM "
POSITION 11,34
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix m_part_machine
ENDIF
DISPLAY FILE m_data1
CLEAR SIZE 37,92 AT 8,2
```

```
POSITION 8,14
MESSAGE"NOW GROUPING IS BEING DONE BY THE CLUSTER ANALAYSIS PROGRAM "
POSITION 10,36
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix m_ka7
ENDIF
DISPLAY FILE m_G1
DISPLAY FILE m_G
CLEAR SIZE 37,92 AT 8,2
POSITION 8,10
MESSAGE"CALCULATING THE NUMBER OF GROUPS AT WHICH GROUPING SHOULD BE
STOPPED"
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix m_stop
ENDIF
DISPLAY FILE m_G2
DISPLAY FILE m_G1
CLEAR SIZE 37,92 AT 8,2
POSITION 8,23
MESSAGE"INTERFACING DEMONSTRATION COMPLETE ..."
HOLD 2
WINDOW MAIN_MENU
DELETE WINDOW INTERFACE DEMONSTRATION
```

D.8 GROUPING BASED ON THE CAFBG SYSTEM

SOFTWARE IMPLEMENTATION

For the implementation of the system, the following main programs have been designed:

1) Component design information is input through a user interface which writes the information in the appropriate tables in the GENERIS knowledge base after dialogue with the user. This interface has been designed in the GENERIS high level language. The name of the interface program is 'input_part'.

   In order to simplify the reasoning process in the software, part features are input after associating the connectivity information with them. For example, instead of just putting Round-Hole for a feature whose position is at the extreme end of a long turned shaft and is not co-axial, Hor-Round-Hole is input. In the case of a co-axial Round-Pocket feature on the same long turned shaft, the input for the feature will be C-Round-Pocket.

2) Each feature geometry is associated with a FTD. FTD generates a number of chain of processes called TSF's of the feature and TSF's are decided based on the feature technological constraints. A single TSF is assigned to a feature in our case. A TSF is decided with the help of complement information like feature geometric parameters/type, technological constraints and position, and then matched with the MCU's available in the machine shop/cell. The inference rules responsible for attaching the operations to the component features are given below:

```
Rules in Inference Ruleset 'oper'

  1 featurecode.1 'has feature1' feature.1 potential1 operation.1 if
        part.1 'has featurecode' featurecode.1 'has feature' feature.1 and
        feature.1 'has mc_method' operation.1 .
  2 featurecode.1 'has feature1' feature.2 potential1 operation.1 if
        part.1 'has featurecode' featurecode.1 'has feature' feature.1 and
        featurecode.1 'has sec_featurecode' sec_featurecode.2 'has sec_feature' feature.2 and
        feature.2 'has mc_method' operation.1 .


Rules in Inference Ruleset 'oper1'

  1 part.1 'need operation' operation.1 if
        part.1 'has featurecode' featurecode.1 and
        featurecode.1 'has feature1' feature.1 potential1 operation.1 .


Rules in Inference Ruleset 'oper2'

  1 featurecode.1 'has feature1' feature.1 potential1 boring if
        part.1 'has featurecode' featurecode.1 'has feature' round_hole and
        featurecode.1 feadiameter diameter.1 and
        diameter.1 >= 50.0000 .
  2 featurecode.1 'has feature1' feature.1 potential1 boring if
        part.1 'has featurecode' featurecode.1 'has feature' round_pocket and
        featurecode.1 feadiameter diameter.1 and
        diameter.1 >= 50.0000 .
  3 featurecode.1 'has feature1' feature.1 potential1 grinding if
        part.1 'has featurecode' featurecode.1 'has feature' round_boss and
        featurecode.1 'has surfinish' surfinish.1 and
        surfinish.1 <= 1.0000 .
  4 featurecode.1 'has feature1' feature.1 potential1 wibbling if
        part.1 'has featurecode' featurecode.1 'has feature' feature.1 and
        featurecode.1 'has sec_featurecode' sec_featurecode.2 'has sec_feature' keyway and
        sec_featurecode.2 'has type' woodruff .
  5 featurecode.1 'has feature1' feature.1 potential1 slotting if
        part.1 'has featurecode' featurecode.1 'has feature' feature.1 and
        featurecode.1 'has sec_featurecode' sec_featurecode.2 'has sec_feature' keyway and
        sec_featurecode.2 'has type' straight .
```

3) Every feature geometry is associated with a fixed number of EAD's of the features belonging to the component PAD's (potential approach directions). Sometimes features have parent/child relationship i.e. a child feature can only be processed after its parent feature has been processed. This parent/child

relationship is calculated by using the real and imaginary surfaces of the adjacent features.

After attaching the MCUE's to each feature as shown in figure 4.7 in chapter 4, component AD's (approach directions) can be determined by clustering the MCUE's (operations) from the common component PAD's. MCUE's may appear in more than one cluster in component different PAD's. The final component AD's are selected by minimising the number of clusters. This is done based on step by step selection of clusters containing the maximum number of MCUE's and then removing those MCUE's from the remaining clusters in the other component PAD's. The result is the number of AD's from which all component features can be processed. The module calculating the number of AD's for processing the components is described in chapter 7.

4) An interface program called 'test2.c', written in C high level programming language extracts the proposed parameters/characteristics of the components from the GENERIS knowledge base. While running this piece of software, it asks the user to input the database name, 'fahd'. After entering the database name, it will ask the user to enter the query. The query this time to be entered is 'part 'has length' length 'has width' width 'has depth' depth 'has diameter' diameter pattern no_of_AD'. The output will be written in the file called 'p_data1'.
Another software program called 'test1.c' has been designed to extract the information about the processing needs of the components. This program will also ask for the database name. The database name 'fahd' will be entered. The query to be entered is "part 'need m_c_unit' m_c_unit". The output file in this case is 'p_data'.

file p_data1 part dimensions to calculate the dimensional ratios, pattern type, number of AD's,
file 'p_data' part processing requirements in terms of MCU's.

5) File 'p_data1' contains inconsistent data i.e. diameter, width and pattern are not in the format which can be read and transferred to the format required by the cluster analysis program. The program 'p_test3.c' has been designed to transform it into the readable format. This program will write the output data in the 'p_data3' file.

6) After reading information about the parts from both the files (p_data and p_data3), the software program 'p_test4.c' written in C programming language writes in a file 'p_data4' after processing this information into the format needed for the grouping of the parts. This is also called the incidence matrix for the cluster analysis.

7) The incidence matrix is the input to the cluster analysis software program 'p_ka7.c' written in C language. The program works in the loop, until all the components are merged into one group or the value of similarity between the groups during the clustering process becomes zero. At each iteration, it

calculates the similarity matrix between the grouping centres existing at that iteration, merges the two most similar groups and then updates the new centre of group. This program writes the results in two files, the details being:

file 'p_G1':	After each iteration the number of groups formed and components in each group and un-grouped components are written in this file.
file 'p_G':	After each iteration the number of groups and similarity level at which two groups are merged are written in this file.
file 'p_G3':	Similar type of file as file 'p_G1', latter used for reading the results for optimal number of groups.

8) Input to the next designed C program 'p_stop.c' is file 'p_G', which calculates the optimal number of groups or, in other words, the number of groups at which the grouping process should be stopped. It calculates the difference or distance between the similarity levels of every two consecutive iterations. It picks up two consecutive iterations, when the distance between them is maximum. Stopping at this iteration suggests that if we go further, the two most dissimilar groups during the whole grouping process will merge. Therefore, grouping should be stopped at that iteration and the number of groups at that point will be optimal. This number showing the optimal number of groups is written in two files; 1) 'p_G2' and 2) 'data4'.

9) The next designed program in C language called 'process.c' reads the optimal number from the file 'data4' and after reading the results for the optimal number of grouping from the file 'p_G3', transforms the results into the format readable by the GENERIS software. The output is written in two files called 'p_data5' and 'p_data6'. File 'p_data5' contains the information about the cells and their member parts. Whereas file 'p_datat6' consists of information about the cells and processing requirements of their centre of groups. GENERIS writes the grouping results in appropriate tables after reading these two files.

D.8.1 PROCEDURE FILE 'p1_generisinit'

```
*
* GENERIS DO FILE 'p1_generisinit'
*
WINDOW NEW INTERFACE DEMONSTRATION SIZE 37,92 AT 2,2
CREATE LOCAL tex0 TEXT
CLEAR
POSITION 2,10
MESSAGE"GENERIS INTERFACE DEMONSTRATION FOR GROUPING THE PARTS "
POSITION 3,10
MESSAGE"********************************************************** "
POSITION 8,10
MESSAGE"ASSIGNING PROCESSING REQUIREMENTS IN TERMS OF MCU'S TO THE
PARTS"
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
IF TELL ANY RECORDS IN part_process
   DELETE RECORDS IN part_process
ENDIF
```

```
DO do_example
DISPLAY RECORDS IN part_process
CLEAR SIZE 37,80 AT 8,2
ENABLE part_c_unit
DISPLAY part need m_c_unit
MESSAGE"MCU'S ASSIGNMENT COMPLETE ..."
HOLD 3
CLEAR SIZE 37,80 AT 8,2
MESSAGE"FETCHING THE COMPONENT LEVEL DATA FROM THE GENERIS "
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
POSITION 12,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix test21
    DISPLAY FILE data1_1
ENDIF
CLEAR SIZE 40,80 AT 8,2
POSITION 8,10
MESSAGE"FETCHING THE PROCESSING REQUIREMENTS FOR THE COMPONENT FROM
THE GENERIS "
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix test11
    DISPLAY FILE data_1
ENDIF
DISABLE ALL
CLEAR SIZE 40,80 AT 6,2
POSITION 8,10
MESSAGE"CONVERTING THE FETCHED DATA INTO THE FORMAT REQUIRED "
POSITION 9,10
MESSAGE"BY THE CLUSTER ANALYSIS PROGRAM "
POSITION 11,32
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix p1_test3
    unix p1_test4
    DISPLAY FILE p1_data4
ENDIF
CLEAR SIZE 40,80 AT 8,2
POSITION 8,10
MESSAGE"NOW GROUPING IS BEING DONE BY THE CLUSTER ANALAYSIS PROGRAM "
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
```

```
    unix p1_ka7
    DISPLAY FILE p1_G1
    DISPLAY FILE p1_G
ENDIF
CLEAR SIZE 40,80 AT 8,2
POSITION 8,10
MESSAGE"CALCULATING THE NUMBER OF GROUPS AT WHICH GROUPING SHOULD BE
STOPPED"
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix stop11
    DISPLAY FILE data41
    DISPLAY FILE p1_G1
ENDIF
CLEAR SIZE 40,80 AT 8,2
POSITION 8,10
MESSAGE"CONVERTING THE GROUPING RESULTS INTO THE FORMAT REQUIRED BY
THE GENERIS"
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix process11
    DISPLAY FILE p1_data5
    DISPLAY FILE p1_data6
ENDIF
CLEAR SIZE 40,80 AT 8,2
POSITION 8,10
MESSAGE"CREATING DATABASE FOR THE CELLS AND PARTS IN THE GENERIS"
POSITION 9,10
MESSAGE"*************** (GROUPING RESULTS) **********************"
POSITION 11,32
MESSAGE"PLEASE WAIT ..."
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    IF TELL ANY RECORDS IN cells
        DELETE RECORDS IN cells
    ENDIF
    DO p1_data6
ENDIF
CLEAR SIZE 40,80 AT 8,2
POSITION 8,46
MESSAGE"NOW"
POSITION 10,10
MESSAGE"CREATING DATABASE FOR THE CELLS AND MCU'S IN THE GENERIS"
POSITION 11,10
MESSAGE"*************** (GROUPING RESULTS) **********************"
POSITION 13,32
```

```
MESSAGE"PLEASE WAIT ..."
POSITION 16,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    IF TELL ANY RECORDS IN cell_MCU
        DELETE RECORDS IN cell_MCU
    ENDIF
    DO p1_data5
ENDIF
DISPLAY RECORDS IN cells
DISPLAY RECORDS IN cell_MCU
CLEAR SIZE 40,80 AT 8,2
POSITION 8,20
MESSAGE"INTERFACING DEMONSTRATION COMPLETE ..."
HOLD 2
WINDOW MAIN_MENU
DELETE WINDOW INTERFACE DEMONSTRATION
```

D.8.2 PROCEDURE FILE 'do_example'

```
*
* DO GENERIS FILE 'do_example'
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL nam NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
*
IF TELL ANY RECORDS IN part_process
    DELETE RECORDS IN part_process
ENDIF
ENABLE oper oper1 oper2
FETCH NEW part need operation operation
LET fetchno = $FETCH
LET int0 = $COUNT
WHILE int0 > 0
    LET int1 = int1 + 1
    LET nam  = FETCH VALUE (fetchno,int1,1)
    IF nam = BLANK
        LET nam = nam2
    ELSE
        LET nam2 = nam
    ENDIF
    LET nam1  = FETCH VALUE (fetchno,int1,2)
    FACT IN part_process
    ^nam
    ^nam1

    LET int0 = int0 - 1
ENDWHILE
DISABLE ALL
DELETE FETCH fetchno
```

D.8.3 PROGRAM 'test21.c'

```
#include <stdio.h>
#include "dsl.h"
static double deref_doub(ptr)
register char *ptr;
{
    double doub;
    register int i;
    register char *dest = (char *)(&doub);
    for(i=0;i<sizeof(doub);i++)
        *dest++ = *ptr++;
    return (doub);
}

main()

{
int i, j, k;
int num;
int posn;
int filler_flag;
double *dechold;
char s1[20];
Thead_output *head_output;
Tdata_output *data_output;
char dbname[61];
char query[101];
FILE *file2 ;
file2=fopen("data1_1","w");
printf("Please give database name (max. 60 characters)\n");
gets(dbname);
printf("\nPlease supply the query (max. 100 characters)\n");
gets(query);

num = F_fetch(dbname,query,&head_output,&data_output);
if(num < 0)
{
    printf("error in fetch %d\n",num);
    printf("%s\n%s\n",dbname,query);
    exit(0);
}

for(i = 0; i < num; i++)
{
    for(j = 0; j < head_output[0].num_columns; j++)
    {
        posn = head_output[0].num_columns * i + j;
        filler_flag = 0;
        if(data_output[posn].depth > 0)
        break;
        fill:
        if(data_output[posn].data.integer == FILLER)
        {
```

```
        for(k = 0; k < head_output[0].num_columns; k++)
        {
            if(data_output[posn+k-j].data.integer != FILLER) break;
        }
        if(k == head_output[0].num_columns)
        {
            posn += head_output[0].num_columns;
            goto fill
        }
        while(data_output[posn].data.integer == FILLER)
        {
            posn -= head_output[0].num_columns;
            if(posn < 0)
            {
                filler_flag++;
                break;
            }
        }
    }

    if(filler_flag ) break;
        if(data_output[posn].data.integer == UNKNOWN ||
            data_output[posn].data.integer == BLANK ||
            data_output[posn].data.integer == CONFLICT)
            {
            if( j == 3 || j == 5 || j == 6 || j == 7 || j == 8 || j == 9)
                fprintf(file2,"0.0 ");
            else
                fprintf(file2,"UNKNOWN ");
                if( (j + 1) % head_output[0].num_columns == 0 )
                    fprintf(file2,"\n") ;
            continue;
            }

            switch(head_output[j].col_type)
            {
                case TINT:
                    fprintf(file2,"%d  ",data_output[posn].data.integer);
                    break;
                case TDEC:
                    dechold = data_output[posn].data.dec_ptr;
                    fprintf(file2,"%f  ",deref_doub(dechold));
                    break;
                case TTIME:
                    timeout(data_output[posn].data.integer,s1);
                    fprintf(file2,"%s  ",s1);
                    break;
                case TDATE:
                    dateout(data_output[posn].data.integer,s1);
                    fprintf(file2,"%s  ", s1);
                    break;
                default:
                    fprintf(file2,"%s  ", data_output[posn].data.text_ptr);
                    break;
            }
            if( (j + 1) % head_output[0].num_columns == 0 )
```

```
                    fprintf(file2,"\n");
                }
            }

    free_output_structures();

    exit(0);
}
```

D.8.4 DATA FILE 'data1_1'

```
p1  324.500000  UNKNOWN 0.0 47.000000  0.0 3
p2  1487.000000  UNKNOWN 0.0 133.000000  0.0 3
p3  1861.000000  UNKNOWN 0.0 228.600000  straight1  3
p4  1280.000000  UNKNOWN 0.0 140.178000  0.0 3
p5  1753.000000  UNKNOWN 0.0 228.600000  straight1  3
p6  1841.000000  UNKNOWN 0.0 228.600000  straight1  3
p7  3185.000000  UNKNOWN 0.0 490.000000  0.0 3
p8  4305.000000  UNKNOWN 0.0 444.000000  0.0 3
p9  16.000000  UNKNOWN 0.0 761.240000  pcd  3
p10  35.000000  UNKNOWN 0.0 385.000000  pcd  3
p11  25.000000  UNKNOWN 0.0 146.000000  0.0 3
p12  19.050000  UNKNOWN 0.0 111.000000  0.0 3
p13  12.000000  UNKNOWN 0.0 150.000000  pcd  4
p14  110.000000  95.000000  38.000000  UNKNOWN straight1  3
p15  32.000000  UNKNOWN 0.0 25.000000  0.0 3
p16  80.000000  80.000000  60.000000  UNKNOWN straight1  1
p17  144.000000  60.000000  12.000000  UNKNOWN 0.0 1
p18  156.000000  60.000000  12.000000  UNKNOWN 0.0 1
p19  274.000000  96.000000  10.000000  UNKNOWN straight1  1
p20  30.000000  15.000000  6.000000  UNKNOWN 0.0 1
p21  127.000000  127.000000  20.000000  UNKNOWN straight1  1
p22  710.000000  410.000000  20.000000  UNKNOWN straight1  1
p23  270.000000  225.000000  25.000000  UNKNOWN straight1  1
p24  410.000000  410.000000  25.000000  UNKNOWN straight1  1
p25  56.000000  12.000000  8.000000  UNKNOWN 0.0 1
p26  317.500000  177.800000  15.880000  UNKNOWN straight1  6
p27  500.000000  120.000000  25.000000  UNKNOWN straight1  1
p28  1328.000000  65.000000  40.000000  UNKNOWN 0.0 3
p29  652.000000  30.680000  11.430000  UNKNOWN 0.0 6
p30  1328.000000  65.000000  40.000000  UNKNOWN 0.0 4
```

D.8.5 DATA FILE 'data_1'

```
p1  MCU_4
p1  MCU_8
p1  MCU_9
p2  MCU_1
p2  MCU_4
p2  MCU_8
p2  MCU_9
p3  MCU_1
p3  MCU_4
p3  MCU_8
```

```
p3 MCU_9
p4 MCU_1
p4 MCU_4
p4 MCU_8
p4 MCU_9
p5 MCU_1
p5 MCU_4
p5 MCU_8
p5 MCU_9
p6 MCU_1
p6 MCU_4
p6 MCU_8
p6 MCU_9
p7 MCU_4
p8 MCU_4
p9 MCU_1
p9 MCU_2
p9 MCU_4
p9 MCU_5
p9 MCU_9
p10 MCU_1
p10 MCU_2
p10 MCU_4
p10 MCU_5
p10 MCU_10
p11 MCU_1
p11 MCU_2
p11 MCU_4
p11 MCU_5
p12 MCU_2
p12 MCU_4
p12 MCU_5
p12 MCU_6
p13 MCU_2
p13 MCU_4
p13 MCU_5
p13 MCU_6
p14 MCU_1
p14 MCU_6
p14 MCU_7
p15 MCU_2
p15 MCU_4
p16 MCU_1
p17 MCU_1
p17 MCU_7
p18 MCU_1
p18 MCU_7
p19 MCU_1
p20 MCU_1
p21 MCU_1
p22 MCU_1
p23 MCU_1
p24 MCU_1
p25 MCU_1
p26 MCU_1
p26 MCU_6
```

p27 MCU_1
p28 MCU_3
p28 MCU_7
p29 MCU_6
p30 MCU_6

D.8.6 PROGRAM 'p1_test3.c'

```
/* Program to read the data from the files 'data1_1' and 'data_1' and transforms the data into the
format required by the cluster analysis */
#include <stdio.h>
#include <string.h>
#define G 100
#define C 50
#define val 0.2
#define val1 1.0
#define val2 0.5
struct rec
{
    char part[4];
    float lengh;
    char width[12];
    float depth;
    char diameter[12];
    char pattern_type[11];
    int no_of_AD;
    struct
    {
        char part[4];
        char process[20];
    } spatt[130];
};
main()
{
    struct rec patt[40] ;
    FILE *file1, *file2,*file3;
    int l, i  = 0,j ,apatt[130],k =0;
    float dat[C][C];
    char STR[G],string1[4],string2[21], buff[6];
    static char nab2[]="0.0";
    static char nab1[]= "UNKNOWN";
    file1 =fopen("data_1","r");
    file2 =fopen("data1_1","r");
    file3=fopen("p1_data3","w");
    while(fgets(STR, 100, file2) != NULL)
    {
        sscanf(STR,"%s          %f          %s          %f          %s          %s
        %d",patt[i].part,&patt[i].lengh,patt[i].width,&patt[i].depth,patt[i].diameter,patt[i].pattern_ty
        pe,&patt[i].no_of_AD);
        j = 0;
        if ((strcmp(patt[i].part,buff)) == 0)
        {
            strcpy(patt[i].spatt[j].part,string1) ;
            strcpy(patt[i].spatt[j].process,string2) ;
            j++;
            strcpy(buff,"rrr") ;
        } 
        while(fgets(STR, 50, file1) != NULL)
        {
            sscanf(STR," %s",buff);
            sscanf(STR," %s %s ", string1,string2);
            if ((strcmp(patt[i].part,buff)) == 0)
            {
                sscanf(STR,"%s %s ", patt[i].spatt[j].part,patt[i].spatt[j].process);
                ++;
            }
            else
                break;
        }
        apatt[i] = j;
        i++;
    }
    l = i ;
    k =0;
    for (i = 0 ; i < l ; i++)
    {
    if ((strcmp(patt[i].width,"UNKNOWN")) == 0)
    {
        for(j=0;j< 8;j++)
        patt[i].width[j] = nab2[j] ;
    }
    if ((strcmp(patt[i].pattern_type,"0.0")) == 0)
    {
        for(j=0;j< 9;j++)
        patt[i].pattern_type[j] = nab1[j] ;
    }
    if ((strcmp(patt[i].diameter,"UNKNOWN")) == 0)
    {
        for(j=0;j< 9;j++)
        patt[i].diameter[j] = nab2[j] ;
    }
    fprintf(file3,"%s          %.2f          %s          %.2f          %s          %s
    %d",patt[i].part,patt[i].lengh,patt[i].width,patt[i].depth,patt[i].diameter,patt[i].pattern_type,patt[i]
    .no_of_AD);
    fprintf(file3,"\n");
    fclose(file1);
    fclose(file2);
    fclose(file3);
}
```

D.8.7 DATA FILE 'p1_data3'

```
p1 324.50 0.0 0.00 47.000000 UNKNOWN 3
p2 1487.00 0.0 0.00 133.000000 UNKNOWN 3
p3 1861.00 0.0 0.00 228.600000 straight1 3
p4 1280.00 0.0 0.00 140.178000 UNKNOWN 3
p5 1753.00 0.0 0.00 228.600000 straight1 3
p6 1841.00 0.0 0.00 228.600000 straight1 3
p7 3185.00 0.0 0.00 490.000000 UNKNOWN 3
p8 4305.00 0.0 0.00 444.000000 UNKNOWN 3
```

```
p9 16.00 0.0 0.00 761.240000 pcd 3
p10 35.00 0.0 0.00 385.000000 pcd 3
p11 25.00 0.0 0.00 146.000000 UNKNOWN 3
p12 19.05 0.0 0.00 111.000000 UNKNOWN 3
p13 12.00 0.0 0.00 150.000000 pcd 4
p14 110.00 95.000000 38.00 0.0 straight1 3
p15 32.00 0.0 0.00 25.000000 UNKNOWN 3
p16 80.00 80.000000 60.00 0.0 straight1 1
p17 144.00 60.000000 12.00 0.0 UNKNOWN 1
p18 156.00 60.000000 12.00 0.0 UNKNOWN 1
p19 274.00 96.000000 10.00 0.0 straight1 1
p20 30.00 15.000000 6.00 0.0 UNKNOWN 1
p21 127.00 127.000000 20.00 0.0 straight1 1
p22 710.00 410.000000 20.00 0.0 straight1 1
p23 270.00 225.000000 25.00 0.0 straight1 1
p24 410.00 410.000000 25.00 0.0 straight1 1
p25 56.00 12.000000 8.00 0.0 UNKNOWN 1
p26 317.50 177.800000 15.88 0.0 straight1 6
p27 500.00 120.000000 25.00 0.0 straight1 1
p28 1328.00 65.000000 40.00 0.0 UNKNOWN 3
p29 652.00 30.680000 11.43 0.0 UNKNOWN 6
p30 1328.00 65.000000 40.00 0.0 UNKNOWN 4
```

D.8.8 PROGRAM 'p1_test4.c'

```
/* Program to read the data from the files 'p1_data3' and 'data_1' and transforms the data into the
format required by the cluster analysis */

#include <stdio.h>
#include <string.h>
#define G 100
#define C 50
#define val 0.2
#define val1 1.0
#define val2 0.5
struct records
{
    char part[4];
    float lengh;
    float width;
    float depth;
    float diameter;
    char pattern_type[11];
    int no_of_AD;
    struct
    {
        char part[4];
        char m_c_unit[20];
    } sec[130];
};
main()
{
    struct records primary[40];
    FILE *file1, *file2,*file3;
    int l, i  = 0,j ,a[130],k =0;
```

```
float dat[C][C];
char STR[G],string1[4],string2[21], buff[6];
file1 =fopen("data_1","r");
file2 =fopen("p1_data3","r");
file3=fopen("p1_data4","w");
while(fgets(STR, 100, file2) != NULL)
{
    sscanf(STR,"%s          %f          %f          %f          %f          %s
    %d",primary[i].part,&primary[i].lengh,&primary[i].width,&primary[i].depth,&primary[i].di
    ameter,primary[i].pattern_type,&primary[i].no_of_AD);
    j = 0;
    if ((strcmp(primary[i].part,buff)) == 0)
    {
        strcpy(primary[i].sec[j].part,string1) ;
        strcpy(primary[i].sec[j].m_c_unit,string2) ;
        j++; strcpy(buff,"rr") ;
    }
    while(fgets(STR, 50, file1) != NULL)
    {
        sscanf(STR," %s",buff);
        sscanf(STR," %s %s ", string1,string2);
        if ((strcmp(primary[i].part,buff)) == 0)
        {
            sscanf(STR,"%s %s ", primary[i].sec[j].part,primary[i].sec[j].m_c_unit);
            j++;
        }
        else
            break;
    }
    a[i] = j;
    i++;
}
l = i ;
k =0;
for (i = 0 ; i < l ; i++)
{
    if(primary[i].diameter != 0.0)
    {
        if(primary[i].lengh/primary[i].diameter <= 0.5)
            dat[i][0] = val1;
        else
            if(primary[i].lengh/primary[i].diameter          <          3.0          &&
            primary[i].lengh/primary[i].diameter > 0.5)
                dat[i][1] = val1;
            else
                if(primary[i].lengh/primary[i].diameter >= 3.0)
                    dat[i][2] = val1;
    }
    else
    {
        if(primary[i].lengh/primary[i].width <= 3 && primary[i].lengh/primary[i].depth >= 4)
            dat[i][3] = val1;
        else
            if(primary[i].lengh/primary[i].width > 3)
                dat[i][4] = val1;
            else
            if(primary[i].lengh/primary[i].width        <=        3        &&
            primary[i].lengh/primary[i].depth < 4)
                dat[i][5] = val1;
}
if ((strcmp(primary[i].pattern_type,"straight1")) == 0)
    dat[i][33] = val1;
else
    if ((strcmp(primary[i].pattern_type,"pcd")) == 0)
        dat[i][32] = val1;
    else
        if ((strcmp(primary[i].pattern_type,"pcd_straight1")) == 0)
        {
            dat[i][32] = val1;
            dat[i][33] = val1;
        }
if(primary[i].no_of_AD == 1)
    dat[i][34] = val;
else
    if(primary[i].no_of_AD == 2)
        dat[i][35] = val;
    else
        if(primary[i].no_of_AD == 3)
            dat[i][36] = val;
        else
            if(primary[i].no_of_AD == 4)
                dat[i][37] = val;
            else
                if(primary[i].no_of_AD == 5)
                    dat[i][38] = val;
                else
                    if(primary[i].no_of_AD == 6)
                        dat[i][39] = val;

for (j = 0 ; j < a[k] ; j++)
{
    if ((strcmp(primary[i].sec[j].m_c_unit,"MCU_1")) == 0)
        dat[i][6] = val1;
    if ((strcmp(primary[i].sec[j].m_c_unit,"MCU_2")) == 0)
        dat[i][7] = val1;
    if ((strcmp(primary[i].sec[j].m_c_unit,"MCU_3")) == 0)
        dat[i][8] = val1;
    if ((strcmp(primary[i].sec[j].m_c_unit,"MCU_4")) == 0)
        dat[i][9] = val1;
    if ((strcmp(primary[i].sec[j].m_c_unit,"MCU_5")) == 0)
        dat[i][10] = val1;
    if ((strcmp(primary[i].sec[j].m_c_unit,"MCU_6")) == 0)
        dat[i][11] = val1;
    if ((strcmp(primary[i].sec[j].m_c_unit,"MCU_7")) == 0)
        dat[i][12] = val1;
    if ((strcmp(primary[i].sec[j].m_c_unit,"MCU_8")) == 0)
        dat[i][13] = val1;
    if ((strcmp(primary[i].sec[j].m_c_unit,"MCU_9")) == 0)
        dat[i][14] = val1;
    if ((strcmp(primary[i].sec[j].m_c_unit,"MCU_10")) == 0)
        dat[i][15] = val1;
        }
        k++;
    }
    fprintf(file3,"%d\n",l);
    fprintf(file3,"24\n");
    for (i = 0 ; i < l ; i++)
    {
        for (j = 0 ; j < 24 ; j++)
        {
            fprintf(file3,"%.1f ",dat[i][j] );
        }
        fprintf(file3,"\n");
    }

    fclose(file1);
    fclose(file2);
    fclose(file3);
}
```

D.8.9 DATA FILE 'p1_data4'

```
30
24
0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 1.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 1.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 1.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
1.0 0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
1.0 0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
1.0 0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0
0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.2
0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0
```

D.8.10 PROGRAM 'p1_ka7.c'

```
/* CLUSTER ANALYSIS PROGRAM FOR GROUPING THE COMPONENTS  */

#include <stdio.h>
#define min(y,z) ((y) < (z)? (y):(z))
#define NMAX 35
#define M 36
main()
{
    FILE *file2,*file1,*file3,*file4, *file5;

    int ngrp, b, un_no, flag1, ii, mat[NMAX], mat2[NMAX], var1, var2, i, j, k, ncol,
    ung_com[NMAX], group[NAMX][NMAX], union(), rowmax, N_ROW, N_COL,
    g_data[NMAX][NMAX];

    float grouped[NMAX][M], un_grp[NMAX][M], xnum, xdemon, sim[NMAX][NMAX], lar,
    uno(), data[NMAX][M], copy[NMAX][M];

    file5=fopen("p1_G5","w");
    file4=fopen("p1_G3","w");
    file3=fopen("p1_G","w");
    file2=fopen("p1_G1","w");
    file1=fopen("p1_data4","r");

/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                   READ THE DATA FILE
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
    fscanf(file1,"%d\n%d\n",&N_ROW,&N_COL);
    for(i=0 ; i< N_ROW;i++)
    {
        for(j=0 ; j< N_COL;j++)
            fscanf(file1,"%f",&data[i][j]);
    }

/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*   COPY DATA MATRIX INTO ANOTHER MATRIX FOR LATER USE
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * **/

    for(i=0 ; i< N_ROW;i++)
        for(j=0 ; j< N_COL;j++)
            copy[i][j] = data[i][j] ;
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * **
* CREATE ANOTHER BINARY MATRIX WHICH WILL GIVE EACH COMPONENT A GROUP
* IN  THE BEGINNING AND LATER THIS MATRIX WILL BE USED TO VISUALISE THE
*  COMPONENTS, THE GROUPS THEY BELONG TO.
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
    for(i=0 ; i< N_ROW;i++)
    {
        for(j=0 ; j< N_ROW;j++)
        {
            if(i==j)
                group[i][j] = 1;
            else
                group[i][j ] = 0;
        }
    }
    rowmax= N_ROW;
    fprintf(file3,"%d  1.0000\n",rowmax);
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 * PROGRAM WORKS IN DO LOOP IN THE FOLLOWING
 * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
    do
    {
/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 *          PRINT CURRENT DATA MATRIX
 * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */

        fprintf(file2,"\nPRINTING CURRENT DATA MATRIX:\n\n");
        for(i=0;i<rowmax;i++)
        {
            for(j=0;j<N_COL;j++)
            {
                fprintf(file2,"%.3f ",data[i][j]);
            }
            printf(file2,"\n");
        }
        fprintf(file2,"\n");

/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 *   CALCULATE THE SIMILARITY MATRIX
 * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */

        for(i = 0 ; i < rowmax ; i++)
        {
            sim[i][i] = 1.0 ;
            for(j=i+1;j<rowmax;j++)
            {
                if(j== (rowmax-1))
                    sim[j][j] =1.0 ;
                xnum=0.0;
                xdemon=0.0;
                for(ncol=0;ncol<N_COL;ncol++)
                {
                    xnum += min(data[i][ncol], data[j][ncol]);
                    xdemon += ((data[i][ncol])+(data[j][ncol]));
                }

                sim[i][j] = xnum/(0.5*xdemon) ;
                sim[j][i]=sim[i][j];
            }
        }

/* * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
 *         PRINT THE SIMILARITY MATRIX
 * * * * * * * * * * * * * * * * * * * * * * * * * * * * * */
        fprintf( file2,"SIMILARITY MATRIX \n") ;
        for(i = 0 ; i < rowmax ; i++)
        {
            for(j = 0 ; j < rowmax ; j++)
                fprintf( file2," %.3f",sim[i][j]) ;
            fprintf( file2," \n") ;
        }

/******************************************
* CALCULATE THE MOST SIMILARITY PAIR OF COMPONENTS AND
* SIMILARITY VALUE BETWEEN THEM
******************************************/
        var1 = var2 = 0;
        lar=0.0;
        for(i=0;i<rowmax;i++)
        {
            for(j=i+1;j<rowmax;j++)
            {
                if(lar <sim[i][j] )
                {
                    lar= sim[i][j];
                    var1 = i; var2 = j;
                }
                else
                    continue;
            }
        }

/****************************************
* PRINT THE MOST SIMILARITY PAIR OF COMPONENTS AND
* SIMILARITY VALUE BETWEEN THEM
****************************************/

        if(lar== 0.0 ) goto finish;
        fprintf(file3,"%d %.4f\n",rowmax-1,lar);
        fprintf(file2,"\n AT THIS ATERATION, SIMILARITY BETWEEN GROUP NO.---> %d
        \n",var1+1);
        printf(file2," AND GROUP NO.---> %d IS MAXIMUM AND ITS VALUE IS ===>
        %.3f\n",var2+1,lar);

/*******************************************
* CALCULATE THE NEW CENTRE OF GROUP AFTER COMBINING MOST
* SIMILARITY PAIR OF GROUPS, DELETE THE PAIR OF GROUPS AND
* THIS WAY THE DATA MATRIX WILL BE REDUCED BY ONE GROUP
*******************************************/

        for(ncol=0;ncol<N_COL;ncol++)
            data[var1][ncol]= uno(data[var1][ncol],data[var2][ncol]);
        for(j=0;j<N_ROW;j++)
            group[var1][j]= un(group[var1][j],group[var2][j]);
        for(k=var2;k<(rowmax-1);k++)
        {
        for(ncol=0 ;ncol < N_COL ;ncol ++)
            data[k][ncol]= data[k+1][ncol];
        for(j=0 ; j< N_ROW;j++)
            group[k][j]= group[k+1][j];
        }

/*****************************************
* PRINT THE GROUPING SITUATION AT CURRENT ITERATION
* I.E. NUMBER OF GROUPS AND THEIR MEMBERS
*****************************************/
```

```
        fprintf(file2,"\n PRINT THE GROUPING SITUATION AT CURRENT ITERATION");
        fprintf(file2,"\n i.e. NUMBER OF GROUPS AND THEIR MEMBERS\n\n");

        fprintf(file2,"    ");
        for(i=0 ; i<(rowmax-1);i++)
        {
            fprintf(file2,"  %d",i+1);
        }
        fprintf(file2,"\n=======");
        for(i=0 ; i< N_ROW;i++)
        fprintf(file2,"===");
        fprintf(file2,"\n");
        for(i=0 ; i<(rowmax-1);i++)
        {
            fprintf( file2," %d -",i+1) ;
            for(j=0 ; j< N_ROW;j++)
                fprintf( file2,"  %d",group[i][j]);
            fprintf(file2,"\n");
        }


/*************************************** ************** *
* RECORDING THE INFORMATION ABOUT THE GROUPED AND UNGROUPED
* COMPONENTS
*******************************************************/

        un_no = 0 ;
        ngrp = 0 ;
        for(i = 0 ; i < (rowmax-1) ;i++)
        {
            flag1 = 0 ;                    /* Reset the flag */
            b = 0 ;
            for(j=0 ; j< N_ROW;j++)
            {
                if (group[i][j]  !=  0)
                {
                    mat1[b] = j+1 ;
                    mat2[b] = j ;
                    b++ ;
                }
            }
             if(b == 1)
            {
                flag1 = 1 ;                /* Set the flag */
                ung_com[un_no] = mat1[0] ;
                ii =  mat2[0] ;
                for(k=0 ; k < N_COL;k++)
                    un_grp[un_no][k] = copy[ii][k] ;
                un_no ++ ;
            }
            if(b > 1)
            {
                g_data[ngrp][0] = b ;
                for (k= 0 ; k < N_COL ; k++ )
                grouped[ngrp][k] = data[i][k] ;
        for (k= 1 ; k < (b+1) ; k++ )
        {
            ii = (k-1) ;
            g_data[ngrp][k] = mat1[ii];
        }
        ngrp ++ ;
    }
}                                   /* LOOP i FINISHES HERE */

if(rowmax <= 15)
{
    fprintf( file2,"\nTOTAL GROUPS IS /ARE  %d   \n",ngrp);
    fprintf( file4,"TOTAL %d \n ",rowmax-1) ;
    fprintf( file5,"TOTAL %d \n ",rowmax-1) ;
    for(o=0 ; o < ngrp ;o++)
    for(k=0 ; k < (g_data[o][0])+1 ;k++)
    for(o=0 ; o < ngrp ;o++)
    {
        fprintf( file2," GROUP NO ==>  %d has components %d \n",o+1,g_data[o][0]);
        fprintf( file4," GROUP %d components %d  \n",o+1,g_data[o][0]);

        for(k=0 ; k < (g_data[o][0]) ;k++)
        {
            fprintf( file2," %d   ",g_data[o][k+1]);
            fprintf( file4," %d   ",g_data[o][k+1]);
        }

        fprintf( file2,"COMPONENTS CHARACTERISTICS ARE  \n\n");
        for(k=0 ; k < (g_data[o][0]) ;k++)
        {
            for(j=0;j<N_COL;j++)
            {
                fprintf( file2,"%.3f ",copy[g_data[o][k+1]-1][j]);
                fprintf( file5,"%.3f ",copy[g_data[o][k+1]-1][j]);
            }
            fprintf( file2,"\n");
        }
        fprintf( file2,"\n");
        fprintf( file2,"\nCOMPONENTS CHARACTERISTICS ARE  \n");
        for(k=0 ; k < N_COL ;k++)
        {
            fprintf( file2,"%.3f " ,grouped[o][k]);
        }
        fprintf(file2,"\nCOMPOSITE  COMPONENT  CHARACTERISTICS  INVOLVED
        ARE \n");
        for(k=0 ; k < N_COL ;k++)
        {
            switch (k)
            {

                case 0 :if(grouped[o][k] != 0.0)
                    fprintf( file2,"L/D <= 0.5,");
                    break;
                case 1 :if(grouped[o][k] != 0.0)
                    fprintf( file2,"L/D < 3.0 && L/D > 0.5,");
                    break;
```

```
case 2:if(grouped[o][k] != 0.0)
     fprintf( file2,"L/D >= 3.0,");
     break;
case 3 :if(grouped[o][k] != 0.0)
     fprintf( file2,"L/W <= 3 && L/T >= 4,");
     break;
case 4 :if(grouped[o][k] != 0.0)
     fprintf( file2,"L/W > 3,");
     break;
case 5 :if(grouped[o][k] != 0.0)
     fprintf( file2,"L/W <= 3 && L/T < 4,");
     break;
case 6:if(grouped[o][k] != 0.0)
     fprintf( file2,"MCU_1,");
     break;
case 7 :if(grouped[o][k] != 0.0)
     fprintf( file2,"MCU_2,");
     break;
case 8 :if(grouped[o][k] != 0.0)
     fprintf( file2,"MCU_3,");
     break;
case 9 :if(grouped[o][k] != 0.0)
     fprintf( file2,"MCU_4,");
     break;
case 10 :if(grouped[o][k] != 0.0)
     fprintf( file2,"MCU_5,");
     break;
case 11 :if(grouped[o][k] != 0.0)
     fprintf( file2,"MCU_6,");
     break;
case 12 :if(grouped[o][k] != 0.0)
     fprintf( file2,"MCU_7,");
     break;
case 13 :if(grouped[o][k] != 0.0)
     fprintf( file2,"MCU_8,");
     break;
case 14 :if(grouped[o][k] != 0.0)
     fprintf( file2,"MCU_9,");
     break;
case 15 :if(grouped[o][k] != 0.0)
     fprintf( file2,"MCU_10,");
     break;
case 16 :if(grouped[o][k] != 0.0)
     fprintf( file2,"P.C.PATTERN,");
     break;
case 17 :if(grouped[o][k] != 0.0)
     fprintf( file2,"ST.PATTERN,");
     break;
case 18 :if(grouped[o][k] != 0.0)
     fprintf( file2,"AD's = 1,");
     break;
case 19 :if(grouped[o][k] != 0.0)
     fprintf( file2,"AD's = 2,");
     break;
case 20 :if(grouped[o][k] != 0.0)
     fprintf( file2,"AD's = 3,");
```

```
                    break;
                case 21 :if(grouped[o][k] != 0.0)
                    fprintf( file2,"AD's = 4,");
                    break;
                case 22 :if(grouped[o][k] != 0.0)
                    fprintf( file2,"AD's = 5,");
                    break;
                case 23 :if(grouped[o][k] != 0.0)
                    fprintf( file2,"AD's = 6,");
                    break;

            }
        }
        fprintf( file2,"\n");
        fprintf( file2,"\n");
    }
    for(k=6 ; k < 16 ;k++)
    {
        switch (k)
        {
            case 6:if(grouped[o][k] != 0.0)
                    fprintf( file4,"MCU_1 ");
                    break;
            case 7 :if(grouped[o][k] != 0.0)
                    fprintf( file4,"MCU_2 ");
                    break;
            case 8 :if(grouped[o][k] != 0.0)
                    fprintf( file4,"MCU_3 ");
                    break;
            case 9 :if(grouped[o][k] != 0.0)
                    fprintf( file4,"MCU_4 ");
                    break;
            case 10 :if(grouped[o][k] != 0.0)
                    fprintf( file4,"MCU_5 ");
                    break;
            case 11 :if(grouped[o][k] != 0.0)
                    fprintf( file4,"MCU_6 ");
                    break;
            case 12 :if(grouped[o][k] != 0.0)
                    fprintf( file4,"MCU_7 ");
                    break;
            case 13 :if(grouped[o][k] != 0.0)
                    fprintf( file4,"MCU_8 ");
                    break;
            case 14 :if(grouped[o][k] != 0.0)
                    fprintf( file4,"MCU_9 ");
                    break;
            case 15 :if(grouped[o][k] != 0.0)
                    fprintf( file4,"MCU_10 ");
                    break;
        }
    }
    fprintf(file4,"\n");
    if(un_no !=0 )
    {
```

```
                fprintf(file2,"\n      *******      UNGROUPED      COMPONENTS      ARE      :
                ************\n");
                fprintf( file2," No of Components are = %d  \n\n",un_no);
                fprintf( file2,"COMPONENTS ARE \n");
                for(j=0 ; j< un_no;j++)
                {
                    if(j == 15 || j == 30 || j == 45 || j == 60 || j == 75 || j == 90) fprintf( file2,"\n");
                        fprintf( file2," %d  ",ung_com[j]);
                }
                fprintf(  file2,"\n\nUNGROUPED  COMPONENTS  CHARACTERISTICS  ARE
                \n\n");
                for(j=0 ; j< un_no;j++)
                {

                    for(k=0 ; k < N_COL ;k++)
                    {
                        fprintf( file2,"%.3f ",un_grp[j][k]);
                    }
                    fprintf( file2,"\n");
                }
            }
        }
        rowmax-- ;
    }
    while(lar !=0.0 ||  rowmax> 0);
    finish :  ;
    fclose(file1);
    fclose(file2);
    fclose(file3);
    fclose(file4);
    fclose(file5);
}

int union(j,k)
int j,k;
{
    return (j || k) ;
}

float uno(y,z)
float y,z;
{
    return((y) > (z)? (y):(z));
}
```

D.8.11 DATA FILE 'p1_G'

```
30  1.0000
29  1.0000
28  1.0000
27  1.0000
26  1.0000
25  1.0000
24  1.0000
23  1.0000
```

```
22 1.0000
21 1.0000
20 0.9123
19 0.9091
18 0.8772
17 0.8611
16 0.8148
15 0.8148
14 0.8108
13 0.8077
12 0.7761
11 0.7143
10 0.7123
9 0.6531
8 0.6531
7 0.5238
6 0.4490
5 0.4400
4 0.4103
3 0.4000
2 0.3208
1 0.2095
```

D.8.12 DATA FILE 'p1_G1'

```
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 8

 GROUPS IS /ARE  6

 GROUP NO ==>  1 has components 6
 1   2   3   4   5   6

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_1,MCU_4,MCU_8,MCU_9,ST.PATTERN,AD's = 3,

 GROUP NO ==>  2 has components 2
 7   8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_4,AD's = 3,

 GROUP NO ==>  3 has components 5
 9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D   <=   0.5,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_9,MCU_10,P.C.PATTERN,AD's   =
3,AD's = 4,

 GROUP NO ==>  4 has components 11
 14   16   17   18   19   20   21   22   23   24   26
```

```
CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W <= 3 && L/T >= 4,L/W <= 3 && L/T < 4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's =
1,AD's = 3,AD's = 6,

 GROUP NO ==>  5 has components 2
 25   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_1,ST.PATTERN,AD's = 1,

 GROUP NO ==>  6 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_6,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
 15   28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 7

 GROUPS IS /ARE  5

 GROUP NO ==>  1 has components 8
 1   2   3   4   5   6   7   8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_1,MCU_4,MCU_8,MCU_9,ST.PATTERN,AD's = 3,

 GROUP NO ==>  2 has components 5
 9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D   <=   0.5,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_9,MCU_10,P.C.PATTERN,AD's   =
3,AD's = 4,

 GROUP NO ==>  3 has components 11
 14   16   17   18   19   20   21   22   23   24   26

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W <= 3 && L/T >= 4,L/W <= 3 && L/T < 4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's =
1,AD's = 3,AD's = 6,

 GROUP NO ==>  4 has components 2
 25   27
```

```
CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_1,ST.PATTERN,AD's = 1,

 GROUP NO ==>  5 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_6,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
 15   28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 6

 GROUPS IS /ARE  4

 GROUP NO ==>  1 has components 8
 1   2   3   4   5   6   7   8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_1,MCU_4,MCU_8,MCU_9,ST.PATTERN,AD's = 3,

 GROUP NO ==>  2 has components 5
 9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D  <=  0.5,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_9,MCU_10,P.C.PATTERN,AD's  =
3,AD's = 4,

 GROUP NO ==>  3 has components 13
 14   16   17   18   19   20   21   22   23   24   25   26   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W   <=   3   &&   L/T   >=   4,L/W   >   3,L/W   <=   3   &&   L/T   <
4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's = 1,AD's = 3,AD's = 6,

 GROUP NO ==>  4 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_6,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
```

```
 15   28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 5

 GROUPS IS /ARE  3

 GROUP NO ==>  1 has components 8
 1   2   3   4   5   6   7   8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_1,MCU_4,MCU_8,MCU_9,ST.PATTERN,AD's = 3,

 GROUP NO ==>  2 has components 5
 9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D   <=   0.5,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_9,MCU_10,P.C.PATTERN,AD's   =
3,AD's = 4,

 GROUP NO ==>  3 has components 15
 14   16   17   18   19   20   21   22   23   24   25   26   27   29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W     <=     3     &&     L/T     >=     4,L/W     >     3,L/W     <=     3     &&     L/T     <
4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's = 1,AD's = 3,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
 15   28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 4

 GROUPS IS /ARE  2

 GROUP NO ==>  1 has components 13
 1   2   3   4   5   6   7   8   9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D                        <=                        0.5,L/D                        >=
3.0,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_8,MCU_9,MCU_10,P.C.PATTERN,ST.PATTE
RN,AD's = 3,AD's = 4,

 GROUP NO ==>  2 has components 15
 14   16   17   18   19   20   21   22   23   24   25   26   27   29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE
```

```
L/W <= 3 && L/T >= 4,L/W > 3,L/W <= 3 && L/T <
4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's = 1,AD's = 3,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
 15   28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 3

 GROUPS IS /ARE  2

 GROUP NO ==>  1 has components 13
 1   2   3   4   5   6   7   8   9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D <= 0.5,L/D >=
3.0,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_8,MCU_9,MCU_10,P.C.PATTERN,ST.PATTE
RN,AD's = 3,AD's = 4,

 GROUP NO ==>  2 has components 16
 14   16   17   18   19   20   21   22   23   24   25   26   27   28   29
 30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W <= 3 && L/T >= 4,L/W > 3,L/W <= 3 && L/T <
4,MCU_1,MCU_3,MCU_6,MCU_7,ST.PATTERN,AD's = 1,AD's = 3,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  1

COMPONENTS ARE
 15

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 2

 GROUPS IS /ARE  1

 GROUP NO ==>  1 has components 29
 1   2   3   4   5   6   7   8   9   10   11   12   13   14   16
 17   18   19   20   21   22   23   24   25   26   27   28   29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D <= 0.5,L/D >= 3.0,L/W <= 3 && L/T >= 4,L/W > 3,L/W <= 3 && L/T <
4,MCU_1,MCU_2,MCU_3,MCU_4,MCU_5,MCU_6,MCU_7,MCU_8,MCU_9,MCU_10,P.C.PATTE
RN,ST.PATTERN,AD's = 1,AD's = 3,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  1

COMPONENTS ARE
```

```
 15

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 1

 GROUPS IS /ARE  1

 GROUP NO ==>  1 has components 30
 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15
 16  17  18  19  20  21  22  23  24  25  26  27  28  29  30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D <= 0.5,L/D < 3.0 && L/D > 0.5,L/D >= 3.0,L/W <= 3 && L/T >= 4,L/W > 3,L/W <= 3 && L/T
<
4,MCU_1,MCU_2,MCU_3,MCU_4,MCU_5,MCU_6,MCU_7,MCU_8,MCU_9,MCU_10,P.C.PATTE
RN,ST.PATTERN,AD's = 1,AD's = 3,AD's = 4,AD's = 6,
```

D.8.13 DATA FILE 'p1_G3'

```
TOTAL 8
 GROUP 1 components 6
1 2 3 4 5 6
MCU_1 MCU_4 MCU_8 MCU_9
 GROUP 2 components 2
7 8
MCU_4
 GROUP 3 components 5
9 10 11 12 13
MCU_1 MCU_2 MCU_4 MCU_5 MCU_6 MCU_9 MCU_10
 GROUP 4 components 11
14 16 17 18 19 20 21 22 23 24 26
MCU_1 MCU_6 MCU_7
 GROUP 5 components 2
25 27
MCU_1
 GROUP 6 components 2
29 30
MCU_6
 GROUP 7 components 1
 15
 MCU_2 MCU_4
 GROUP 8 components 1
 28
 MCU_3 MCU_7
TOTAL 7
 GROUP 1 components 8
1 2 3 4 5 6 7 8
MCU_1 MCU_4 MCU_8 MCU_9
 GROUP 2 components 5
9 10 11 12 13
MCU_1 MCU_2 MCU_4 MCU_5 MCU_6 MCU_9 MCU_10
 GROUP 3 components 11
14 16 17 18 19 20 21 22 23 24 26
MCU_1 MCU_6 MCU_7
 GROUP 4 components 2
```

```
25 27
MCU_1
 GROUP 5 components 2
29 30
MCU_6
 GROUP 6 components 1
 15
 MCU_2 MCU_4
 GROUP 7 components 1
 28
 MCU_3 MCU_7
TOTAL 6
 GROUP 1 components 8
1 2 3 4 5 6 7 8
MCU_1 MCU_4 MCU_8 MCU_9
 GROUP 2 components 5
9 10 11 12 13
MCU_1 MCU_2 MCU_4 MCU_5 MCU_6 MCU_9 MCU_10
 GROUP 3 components 13
14 16 17 18 19 20 21 22 23 24 25 26 27
MCU_1 MCU_6 MCU_7
 GROUP 4 components 2
29 30
MCU_6
 GROUP 5 components 1
 15
 MCU_2 MCU_4
 GROUP 6 components 1
 28
 MCU_3 MCU_7
TOTAL 5
 GROUP 1 components 8
1 2 3 4 5 6 7 8
MCU_1 MCU_4 MCU_8 MCU_9
 GROUP 2 components 5
9 10 11 12 13
MCU_1 MCU_2 MCU_4 MCU_5 MCU_6 MCU_9 MCU_10
 GROUP 3 components 15
14 16 17 18 19 20 21 22 23 24 25 26 27 29 30
MCU_1 MCU_6 MCU_7
 GROUP 4 components 1
 15
 MCU_2 MCU_4
 GROUP 5 components 1
 28
 MCU_3 MCU_7
TOTAL 4
 GROUP 1 components 13
1 2 3 4 5 6 7 8 9 10 11 12 13
MCU_1 MCU_2 MCU_4 MCU_5 MCU_6 MCU_8 MCU_9 MCU_10
 GROUP 2 components 15
14 16 17 18 19 20 21 22 23 24 25 26 27 29 30
MCU_1 MCU_6 MCU_7
 GROUP 3 components 1
 15
 MCU_2 MCU_4
 GROUP 4 components 1
 28
 MCU_3 MCU_7
TOTAL 3
  GROUP 1 components 13
1 2 3 4 5 6 7 8 9 10 11 12 13
MCU_1 MCU_2 MCU_4 MCU_5 MCU_6 MCU_8 MCU_9 MCU_10
 GROUP 2 components 16
14 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
MCU_1 MCU_3 MCU_6 MCU_7
 GROUP 3 components 1
 15
 MCU_2 MCU_4
TOTAL 2
  GROUP 1 components 29
1 2 3 4 5 6 7 8 9 10 11 12 13 14 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
MCU_1 MCU_2 MCU_3 MCU_4 MCU_5 MCU_6 MCU_7 MCU_8 MCU_9 MCU_10
 GROUP 2 components 1
 15
 MCU_2 MCU_4
TOTAL 1
  GROUP 1 components 30
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
MCU_1 MCU_2 MCU_3 MCU_4 MCU_5 MCU_6 MCU_7 MCU_8 MCU_9 MCU_10
```

D.8.14 DATA FILE 'p1_G5'

```
TOTAL 8
 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 1.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
1.0 0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 0.0 0.0 1.0 1.0 1.0 0.0 0.0 0.0 0.2 0.2 0.0 0.0
0.0 0.0 0.0 1.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.2 0.0 0.0 0.2
0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.2
0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
TOTAL 7
 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 1.0 0.0 0.0 0.2 0.0 0.0 0.0
1.0 0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 0.0 0.0 1.0 1.0 1.0 0.0 0.0 0.0 0.2 0.2 0.0 0.0
0.0 0.0 0.0 1.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.2 0.0 0.0 0.2
0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.2
0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
TOTAL 6
 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 1.0 0.0 0.0 0.2 0.0 0.0 0.0
1.0 0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 0.0 0.0 1.0 1.0 1.0 0.0 0.0 0.0 0.2 0.2 0.0 0.0
0.0 0.0 0.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.2 0.0 0.0 0.2
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.2
0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
TOTAL 5
 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 1.0 0.0 0.0 0.2 0.0 0.0 0.0
1.0 0.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 0.0 0.0 1.0 1.0 1.0 0.0 0.0 0.0 0.2 0.2 0.0 0.0
0.0 0.0 0.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.2 0.2 0.0 0.2
```

```
0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
TOTAL 4
 1.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 0.2 0.2 0.0 0.0
0.0 0.0 0.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.2 0.2 0.0 0.2
0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
0.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
TOTAL 3
 1.0 0.0 1.0 0.0 0.0 0.0 1.0 1.0 0.0 1.0 1.0 1.0 0.0 1.0 1.0 1.0 1.0 1.0 0.0 0.0 0.2 0.2 0.0 0.0
0.0 0.0 0.0 1.0 1.0 1.0 1.0 0.0 1.0 0.0 0.0 1.0 1.0 0.0 0.0 0.0 0.0 1.0 0.2 0.0 0.2 0.2 0.0 0.2
0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
TOTAL 2
 1.0 0.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.2 0.0 0.2 0.2 0.0 0.2
0.0 1.0 0.0 0.0 0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.0 0.0 0.0
TOTAL 1
 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.2 0.0 0.2 0.2 0.0 0.2
```

D.8.15 PROGRAM 'stop1.c'

```
/* FILE 'stop1.c' */

/* PROGRAM TO READ DATA AFTER GROUPING AND TO CALCULATE WHERE
GROUPING PROCESS SHOULD BE STOPPED */

#include <stdio.h>
#define G 20
struct record
{
    int group;
    float sim_level;
    float range;
} ;
main()
{
    struct record part[115] ;
    FILE *file1,*file2,*file3;
    int i,j,k;
    int N=0;
    float lar;
    char STR[G] ;
    file1=fopen("p1_G","r");
    file2=fopen("p1_G2","w");
    file3=fopen("data41","w");
    while(fgets(STR, 20, file1) != NULL)
    {
        sscanf(STR,"%d %f ",&part[i].group,&part[i].sim_level);
        i++;
    }
    N = i;

    for(i = 0;i < N ;i++)
    {
        fprintf(file2,"%d %.4f ",part[i].group,part[i].sim_level) ;
        fprintf(file2,"\n");
    }
```

```
i = 0;
part[i].range = 1.0 - part[i].sim_level;
for(i = 0 ; i < N ;i++)
{
    part[i+1].range = part[i].sim_level - part[i+1].sim_level;
}
for(i = 0;i< N ;i++)
{
    fprintf(file2,"%d %.4f %.4f",part[i].group,part[i].sim_level,part[i].range) ;
    fprintf(file2,"\n");
}
fprintf(file2,"\n\n");
for(i=1 ; i< N-1;i++)
{
    lar =  part[i].range ;
    k =1;
    for(j=i+1 ; j < N ;j++)
    {
         if (part[j].range > lar)
        {
            lar = part[j].range  ;
            k =  j ;
            }
        }
        if(k > i )
        {
            part[N].group  =  part[i].group,part[N].sim_level=  part[i].sim_level,part[N].range  =
            part[i].range ;
            part[i].group  =  part[k].group,part[i].sim_level=  part[k].sim_level,part[i].range  =
            part[k].range ;
            part[k].group = part[N].group,part[k].sim_level = part[N].sim_level, part[k].range =
            part[N].range ;
        }
    }
    for(i = 1;i< N ;i++)
    {
        fprintf(file2,"%d %.4f %.4f\n",part[i].group,part[i].sim_level,part[i].range) ;
    }
    for(i = 1;i< N ;i++)
    {
    if((part[i].range-part[i+1].range)<0.0001 || abs(part[i+1].range-part[i].range) < 0.0001)

    else
    {
        if(part[i].group < part[i-1].group)
        {
            fprintf(file2,"\n\nMESSAGE GROUPING SHOULD BE STOPPED AT GROUP
            NO. %d \n",part[i].group+1 );
            break;
        }
        else
        {
            fprintf(file2,"\n\nMESSAGE GROUPING SHOULD BE STOPPED AT GROUP
            NO. %d \n",part[i-1].group+1 );
            break;
        }
        }
    }


fclose(file1);
fclose(file2);
fclose(file3);

}
```

D.8.16 DATA FILE 'p1_G2'

```
N IS 30
30 1.0000
29 1.0000
28 1.0000
27 1.0000
26 1.0000
25 1.0000
24 1.0000
23 1.0000
22 1.0000
21 1.0000
20 0.9123
19 0.9091
18 0.8772
17 0.8611
16 0.8148
15 0.8148
14 0.8108
13 0.8077
12 0.7761
11 0.7143
10 0.7123
9 0.6531
8 0.6531
7 0.5238
6 0.4490
5 0.4400
4 0.4103
3 0.4000
2 0.3208
1 0.2095

30 1.0000 0.0000
29 1.0000 0.0000
28 1.0000 0.0000
27 1.0000 0.0000
26 1.0000 0.0000
25 1.0000 0.0000
24 1.0000 0.0000
23 1.0000 0.0000
22 1.0000 0.0000
21 1.0000 0.0000
20 0.9123 0.0877
```

```
19 0.9091 0.0032
18 0.8772 0.0319
17 0.8611 0.0161
16 0.8148 0.0463
15 0.8148 0.0000
14 0.8108 0.0040
13 0.8077 0.0031
12 0.7761 0.0316
11 0.7143 0.0618
10 0.7123 0.0020
9 0.6531 0.0592
8 0.6531 0.0000
7 0.5238 0.1293
6 0.4490 0.0748
5 0.4400 0.0090
4 0.4103 0.0297
3 0.4000 0.0103
2 0.3208 0.0792
1 0.2095 0.1113

7 0.5238 0.1293
1 0.2095 0.1113
20 0.9123 0.0877
2 0.3208 0.0792
6 0.4490 0.0748
11 0.7143 0.0618
9 0.6531 0.0592
16 0.8148 0.0463
18 0.8772 0.0319
12 0.7761 0.0316
4 0.4103 0.0297
17 0.8611 0.0161
3 0.4000 0.0103
5 0.4400 0.0090
14 0.8108 0.0040
19 0.9091 0.0032
13 0.8077 0.0031
10 0.7123 0.0020
24 1.0000 0.0000
27 1.0000 0.0000
23 1.0000 0.0000
8 0.6531 0.0000
29 1.0000 0.0000
25 1.0000 0.0000
22 1.0000 0.0000
15 0.8148 0.0000
21 1.0000 0.0000
26 1.0000 0.0000
28 1.0000 0.0000

MESSAGE GROUPING SHOULD BE STOPPED AT GROUP NO. 8
```

D.8.17 DATA FILE 'data41'

```
8
```

D.8.18 PROGRAM 'process1.c'

```
/* Program to read grouping results and number at which grouping is to be stopped and transform
into the format ready by generis */
#include <string.h>
#include <stdio.h>
#define G 700
#define C 40
struct records
{
    char process[20];
};
main()
{
    struct records p[G] ;
    FILE *file1,*file2,*file3,*file4;
    float a1[G][C],F=1.0, U = 1.0;
    int ain,  l=0,i  = 0,j= 0 ,k =0, m,x,num,num1,num2,num3,array1[C][82],array2[6];
    char STR[C],STR1[C],c;
    char buff[6];
    file1=fopen("p1_G3","r");
    file2=fopen("p1_data5","w");
    file3=fopen("p1_data6","w");
    file4=fopen("data41","r");
    fprintf(file2,"CREATE RECORDS IN 'cell_MCU' WITH '|' AND ':\n");
    fprintf(file3,"CREATE RECORDS IN 'cells' WITH '|' AND ':\n");
    fscanf(file4,"%d",&num3);
    while(fgets(STR, 12, file1) != NULL)
    {
        sscanf(STR,"%s %d",STR1,&num) ;
        if(strcmp(STR1,"TOTAL") == 0)
        {
            printf("%s %d\n",STR1,num);
            if(num == num3)
            {
                printf("num is %d\n",num);
                for(k = 0; k < num3 ; k++)
                {
                    if(k == 0 )
                    {
                        fgets(STR, 60, file1) ;
                        sscanf(STR,"%*s%*c%d%*c%*s%d",&num1,&num2);
                    }
                    if(k > 0 )
                        fscanf(file1,"%d%*c%*s%d",&num1,&num2);
                    printf("num1 is %d num2 is %d and num3 is %d\n",num1,num2,num3);
                    array1[k][0]= num1;
                    array1[k][1]= num2;
                    for (i = 2 ; i < num2+2 ; i++)
                        fscanf(file1,"%d",&array1[k][i]);
                    for (i = 0 ; i < num2+2; i++)
```

```
                        printf(" %d",array1[k][i]);
                    printf("\n");
                    fgets(STR, 600, file1);
                    l = 0;j=0;
                    do
                    {
                        fscanf(file1," %s ",p[j].process);

                        if(j>l)
                            l = j ;
                        array2[k]= l;
                        j++;

                    }
                    while((strcmp(p[j-1].process,"GROUP")    !=    0    )    &&    (strcmp(p[j-
                    1].process,"TOTAL") != 0 ));

                        for(l=0;l<array2[k];l++)
                        {
                            fprintf(file2,"cell | C_%d \n",k+1);
                            fprintf(file2,"'collection_of' 'm_c_unit' | %s \n",p[l].process);
                            fprintf(file2,":\n");
                        }
                    for(j=2;j<array1[k][1]+2;j++)
                    {
                        fprintf(file3,"cell | C_%d \n",k+1);
                        fprintf(file3,"'has part' 'part' | p%d \n",array1[k][j]);
                        fprintf(file3,":\n");
                    }
                }
            }
        }
        fgets(STR, 60, file1);
        if(strcmp(STR,"TOTAL 9") == 0)
            break;

    }

    fclose(file1);
    fclose(file2);
    fclose(file3);
    fclose(file4);
}
```

D.8.19 DATA FILE 'p1_data5'

```
CREATE RECORDS IN 'cell_MCU' WITH '|' AND ':'
cell | C_1
'collection_of' 'm_c_unit' | MCU_1
:
cell | C_1
'collection_of' 'm_c_unit' | MCU_4
:
cell | C_1
'collection_of' 'm_c_unit' | MCU_8
```

```
:
cell | C_1
'collection_of' 'm_c_unit' | MCU_9
:
cell | C_2
'collection_of' 'm_c_unit' | MCU_4
:
cell | C_3
'collection_of' 'm_c_unit' | MCU_1
:
cell | C_3
'collection_of' 'm_c_unit' | MCU_2
:
cell | C_3
'collection_of' 'm_c_unit' | MCU_4
:
cell | C_3
'collection_of' 'm_c_unit' | MCU_5
:
cell | C_3
'collection_of' 'm_c_unit' | MCU_6
:
cell | C_3
'collection_of' 'm_c_unit' | MCU_9
:
cell | C_3
'collection_of' 'm_c_unit' | MCU_10
:
cell | C_4
'collection_of' 'm_c_unit' | MCU_1
:
cell | C_4
'collection_of' 'm_c_unit' | MCU_6
:
cell | C_4
'collection_of' 'm_c_unit' | MCU_7
:
cell | C_5
'collection_of' 'm_c_unit' | MCU_1
:
cell | C_6
'collection_of' 'm_c_unit' | MCU_6
:
cell | C_7
'collection_of' 'm_c_unit' | MCU_2
:
cell | C_7
'collection_of' 'm_c_unit' | MCU_4
:
cell | C_8
'collection_of' 'm_c_unit' | MCU_3
:
cell | C_8
'collection_of' 'm_c_unit' | MCU_7
:
```

D.8.20 DATA FILE 'p1_data6'

```
CREATE RECORDS IN 'cells' WITH '|' AND ':'
cell | C_1
'has part' 'part' | p1
:
cell | C_1
'has part' 'part' | p2
:
cell | C_1
'has part' 'part' | p3
:
cell | C_1
'has part' 'part' | p4
:
cell | C_1
'has part' 'part' | p5
:
cell | C_1
'has part' 'part' | p6
:
cell | C_2
'has part' 'part' | p7
:
cell | C_2
'has part' 'part' | p8
:
cell | C_3
'has part' 'part' | p9
:
cell | C_3
'has part' 'part' | p10
:
cell | C_3
'has part' 'part' | p11
:
cell | C_3
'has part' 'part' | p12
:
cell | C_3
'has part' 'part' | p13
:
cell | C_4
'has part' 'part' | p14
:
cell | C_4
'has part' 'part' | p16
:
cell | C_4
'has part' 'part' | p17
:
cell | C_4
'has part' 'part' | p18
:
```

```
cell | C_4
'has part' 'part' | p19
:
cell | C_4
'has part' 'part' | p20
:
cell | C_4
'has part' 'part' | p21
:
cell | C_4
'has part' 'part' | p22
:
cell | C_4
'has part' 'part' | p23
:
cell | C_4
'has part' 'part' | p24
:
cell | C_4
'has part' 'part' | p26
:
cell | C_5
'has part' 'part' | p25
:
cell | C_5
'has part' 'part' | p27
:
cell | C_6
'has part' 'part' | p29
:
cell | C_6
'has part' 'part' | p30
:
cell | C_7
'has part' 'part' | p15
:
cell | C_8
'has part' 'part' | p28
```

D.9 SOFTWARE PROGRAMS FOR PATTERN RECOGNITION

The main software programs developed for the pattern recognition are given in the following. The software programs hierarchy (the sequence in which they are called) designed for the pattern recognition is shown in figure D.2. On the top of the figure, the main menu is shown, which groups the parts based on the different criteria discussed in chapter 4. The software programs hierarchy for patt_generisinit is expanded in the figure.

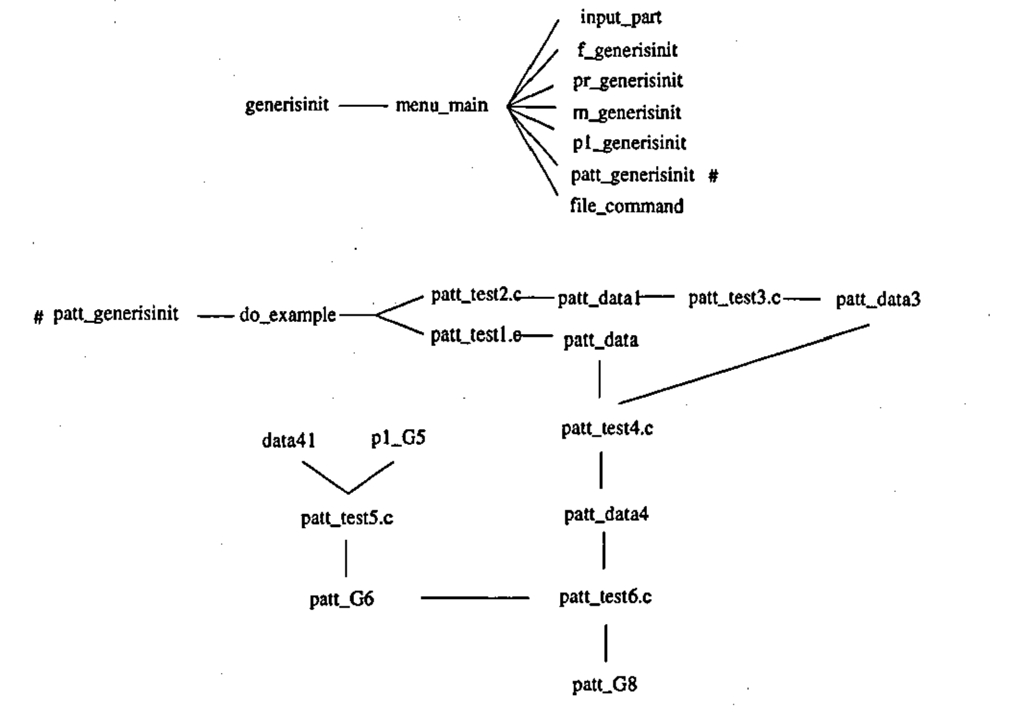


Figure D.2 Main menu for grouping based on different grouping techniques and hierarchy for the software programs and data files involved in the Pattern Recognition

New components can be assigned to existing part families based on their closeness with the centre of the groups (pattern recognition). The description of software developed for pattern recognition is described below:

1) The part to be assigned to existing part families is described in the GENERIS system through a user interface as discussed above.

2) Processing requirements to the part features in terms of MCUE's are attached with the aid of rules 'oper', 'oper1', and 'oper2' as given above. These MCUE's are converted to MCU's.

3) An interface program called 'patt_test2.c', written in C high level programming language extracts the proposed classifying attributes of the components from the GENERIS knowledge base. While running this piece of software, it asks the user to input the database name, 'fahd' and then to enter the query. The query to be entered is 'part 'has length' length 'has width' width 'has depth' depth 'has diameter' diameter pattern no_of_AD' for part_x. The part 'part_x' being the part to be assigned to the part families. The output will be written in the file called 'patt_data1'.

Another software program called 'patt_test1.c' has been designed to extract the information about the processing needs of the components. This program will also ask for the database name. The database name 'fahd' will be entered. The also ask for the database name. The database name 'fahd' will be entered. The query to be entered is "part 'need m_c_unit' m_c_unit". The output file in this case is 'patt_data'.

file patt_data1 contains part dimensions (to calculate the dimensional ratios), pattern type, and number of AD's,
file 'patt_data' consists of part processing requirements in terms of MCU's.

4) File 'patt_data1' contains inconsistent data i.e. diameter, width and pattern which is not in a format that can be easily read and transferred to the format required by the cluster analysis program. Program 'patt_test3.c' has been designed to transform it into readable format. This program will write the output data in the 'patt_data3' file.

5) After reading information about the parts from both the files (patt_data and patt_data3), software program 'patt_test4.c' designed in C programming language, writes in a file 'patt_data4' after processing this information into the format needed for the grouping of the parts.

6) The grouping results after each iteration (number of groups, the members of each group along with the composite component for each group), as well as the information about the optimal number of groups, are available already by running the cluster analysis software. These results are in the files 'p_G5' and 'data4' respectively. The program written in C called 'patt_test5.c' will read these two files and write the centre of groups for the optimal number of groups in the file 'patt_G6'.

7) Another designed software program in C called 'patt_test6.c' takes the input from the data file 'patt_data4' and 'patt_G6'. These two files contain the classifying attributes of the new part and features/characteristics of the composite components of the existing groups respectively. The program calculates the level of similarity of the new component with the centre of existing groups and will write the results in a file called 'patt_G8'. Furthermore, it will contain one of the messages given below:

(a) The component has greatest similarity with group x, with the level of similarity being y.
(b) The component cannot be assigned to any group as its similarity level with the most similar group is z.

Message (a) will be displayed if a new part is assigned to any group. A new part will be assigned to any existing group if its similarity level is equal to or more than the threshold value which is 40%. Message (b) is displayed if a new part is not being grouped to any group as the value of similarity is less than the threshold value mentioned above.

D.9.1 PROCEDURE FILE 'patt_generisinit'

```
*
* GENERIS DO FILE 'patt_generisinit'
*
WINDOW NEW INTERFACE DEMONSTRATION SIZE 37,92 AT 2,2
CREATE LOCAL tex0 TEXT
CLEAR
POSITION 2,10
MESSAGE"GENERIS INTERFACE DEMONSTRATION FOR GROUPING THE PARTS "
POSITION 3,10
MESSAGE"****************************************************** "
POSITION 8,13
MESSAGE"ASSIGNING PROCESSING REQUIREMENTS TO THE PART"
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
IF TELL ANY RECORDS IN part_process
    DELETE RECORDS IN part_process
ENDIF
DO do_example
DISPLAY RECORDS IN part_process
MESSAGE"PROCESS ASSIGNMENT COMPLETE ..."
HOLD 2
CLEAR SIZE 37,80 AT 8,2
MESSAGE"FETCHING THE COMPONENT LEVEL DATA FROM THE GENERIS "
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
POSITION 12,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix patt_test2
    DISPLAY FILE patt_data1
ENDIF
CLEAR SIZE 40,80 AT 8,2
POSITION 8,10
MESSAGE"FETCHING THE PROCESSING REQUIREMENTS FOR THE COMPONENT FROM
THE GENERIS "
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix patt_test1
    DISPLAY FILE patt_data
ENDIF
CLEAR SIZE 40,80 AT 8,2
POSITION 8,10
MESSAGE"CONVERTING THE FETCHED DATA INTO THE FORMAT REQUIRED "
POSITION 9,10
MESSAGE"BY THE CLUSTER ANALYSIS PROGRAM "
POSITION 11,32
MESSAGE"PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
```

```
IF tex0 = BLANK
    unix patt_test3
    unix patt_test4
    DISPLAY FILE patt_data3
    DISPLAY FILE patt_data4
ENDIF
CLEAR SIZE 40,80 AT 8,2
POSITION 8,10
MESSAGE"NOW PATTERN RECOGNITION FOR THE NEW PART WITH THE EXISTING
GROUPS PROCESS"
POSITION 10,32
MESSAGE"WILL BE CARRIED OUT    PLEASE WAIT ... "
POSITION 14,15
MESSAGE"PLEASE <ENTER> FOR PROGRESSING "
READ tex0
IF tex0 = BLANK
    unix patt_test5
    DISPLAY FILE patt_G6
    unix patt_test6
    DISPLAY FILE patt_G8
ENDIF
CLEAR SIZE 40,80 AT 8,2
POSITION 8,20
MESSAGE"INTERFACING DEMONSTRATION COMPLETE ..."
HOLD 2
WINDOW MAIN_MENU
DELETE WINDOW INTERFACE DEMONSTRATION
```

D.9.2 PROGRAM 'patt_test5'

```
/* Program to read incidence Matrix for the number of Groups at which Grouping should be stopped
for Pattern Recognition.*/
#include <string.h>
#include <stdio.h>
#define G 10
#define C 50
main()
{
    FILE *file1,*file2,*file4;
    float array1[G][C];
    int ain, l=0,i = 0,j= 0 ,k =0, m,x,num,num1,num2,num3,d[6];
    char STR[C],STR1[C],c;
    char buff[6];
    file1=fopen("p_G5","r");
    file2=fopen("patt_G6","w");
    file4=fopen("data4","r");
    fscanf(file4,"%d",&num3);
    fprintf(file2,"%d \n", num3);
    while(fgets(STR, 10, file1) != NULL)
    {
        sscanf(STR,"%s %d",STR1,&num) ;
        if(strcmp(STR1,"TOTAL") == 0)
        {
            if(num == num3)
            {
                for(k = 0; k < num3 ; k++)
                {
                    for (i = 0 ; i < 40 ; i++)
                    fscanf(file1,"%f",&array1[k][i]);
                }
                for(k = 0; k < num3 ; k++)
                {
                    for (i = 0 ; i < 40 ; i++)
                        fprintf(file2,"%.1f ",array1[k][i]);
                    fprintf(file2,"\n");
                }
            }

            fgets(STR, 60, file1);
            if(strcmp(STR,"TOTAL 4") == 0)
            break;

        }
    }


    fclose(file1);
    fclose(file2);
    fclose(file4);
}
```

D.9.3 PROGRAM 'patt_test6.c'

```
/* Program designed to calculate the similarity for pattern recognition */

#include <stdio.h>
#define min(y,z) ((y) < (z)? (y):(z))
#define NMAX 25
#define M 50
main()

{
    FILE *file3,*file2,*file1;
    float array1[NMAX][M], a1[NMAX][M],s[NMAX],xnum,xdemon,big;
    int i,j,K=40,N,ncol,val;
    file3=fopen("patt_G8","w");
    file2=fopen("patt_data4","r");
    file1=fopen("patt_G6","r");
    fscanf(file1,"%d\n",&N);

    for(i=0 ; i< N;i++)
    {
        for(j=0 ; j< K;j++)
            fscanf(file1,"%f",&array1[i][j]);
    }

    for(i=0 ; i< 1;i++)
    {
        for(j=0 ; j< K;j++)
            fscanf(file2,"%f",&a1[i][j]);
    }

    for(i = 0 ; i < N ; i++)
    {
        xnum=0.0;
        xdemon=0.0;
        for(ncol=0;ncol<K;ncol++)
        {
            xnum += min(a1[0][ncol], array1[i][ncol]);
            xdemon += (a1[0][ncol]+array1[i][ncol]);
        }
        printf("xnum is %.3f xdemon is %.3f for component  %d \n ",xnum,xdemon,i+1);
        s[i] = xnum/(0.5*xdemon) ;
    }
    for(i = 0 ; i < N ; i++)
    fprintf(file3," SIMLARITY WITH GROUP %d IS %.5f \n",i+1,s[i]);
    big = s[0];
    for(i = 0 ; i < N ; i++)
    {
        if(big < s[i+1])
        {
            big = s[i+1];
            val = i+1;
        }
    }
    if(big >= 0.4)
    {
        fprintf(file3,"THE COMPONENT HAS GOT MORE SIMILARITY WITH GROUP
        %d,\n",val+1);
        fprintf(file3,"WITH LEVEL OF SIMILARITY %.4f", big);
    }
    else
        fprintf(file3,"THE COMPONENT CANNOT BE ASSIGNED TO ANY CELL AS ITS
        SIMILARITY LEVEL WITH MOST SIMILAR GROUP IS %.5f",big);

    fclose(file1);
    fclose(file2);
    fclose(file3);

}
```

D.10 GROUPING RESULTS

Grouping results based on different criteria are attached in this section along with the suggested optimal number of groups at which grouping should be stopped.

D.10.1 PART GEOMETRY-BASED GROUPING

SUGGESTED OPTIMAL NUMBER OF GROUPS = 2

```
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 8

 GROUPS IS /ARE  6
```

```
GROUP NO ==>  1 has components 8
1   2   3   4   5   6   7   8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,
FACE,CENTRE,SCREW,KEYWAY,THREAD,


GROUP NO ==>  2 has components 5
9   10   11   12   15

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,THRU. RECT. SLOT,PARTIAL R_SLOT,
KEYWAY,THREAD,
CHAMFER,

GROUP NO ==>  3 has components 2
13   26

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,
PARTIAL R_STEP,SURFACE,


GROUP NO ==>  4 has components 6
14   16   17   18   20   28

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_HOLE,THRU. RECT. SLOT,
TRIANG_STEP ,SURFACE,THREAD,


GROUP NO ==>  5 has components 5
21   22   23   24   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_HOLE,

S/FACE,

GROUP NO ==>  6 has components 2
29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE


PARTIAL R_STEP,RECT. STEP,SURFACE,


******* UNGROUPED COMPONENTS ARE :  ************
```

```
 No of Components are =  2

COMPONENTS ARE
 19  25

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 7

 GROUPS IS /ARE  5

 GROUP NO ==>  1 has components 13
 1   2   3   4   5   6   7   8   9   10   11   12   15

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,THRU. RECT. SLOT,PARTIAL R_SLOT,
FACE,CENTRE,SCREW,KEYWAY,THREAD,
CHAMFER,

 GROUP NO ==>  2 has components 2
 13   26

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,
PARTIAL R_STEP,SURFACE,


 GROUP NO ==>  3 has components 6
 14   16   17   18   20   28

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_HOLE,THRU. RECT. SLOT,
TRIANG_STEP ,SURFACE,THREAD,


 GROUP NO ==>  4 has components 5
 21   22   23   24   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_HOLE,

S/FACE,

 GROUP NO ==>  5 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE


PARTIAL R_STEP,RECT. STEP,SURFACE,


 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2
```

```
COMPONENTS ARE
 19  25

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 6

 GROUPS IS /ARE  4

 GROUP NO ==>  1 has components 13
 1   2   3   4   5   6   7   8   9   10   11   12   15

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,THRU. RECT. SLOT,PARTIAL R_SLOT,
FACE,CENTRE,SCREW,KEYWAY,THREAD,
CHAMFER,

 GROUP NO ==>  2 has components 4
 13   26   29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,
PARTIAL R_STEP,RECT. STEP,SURFACE,


 GROUP NO ==>  3 has components 6
 14   16   17   18   20   28

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_HOLE,THRU. RECT. SLOT,
TRIANG_STEP ,SURFACE,THREAD,


 GROUP NO ==>  4 has components 5
 21   22   23   24   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_HOLE,

S/FACE,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
 19  25

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 5

 GROUPS IS /ARE  4

 GROUP NO ==>  1 has components 13
 1   2   3   4   5   6   7   8   9   10   11   12   15
```

```
CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,THRU. RECT. SLOT,PARTIAL R_SLOT,
FACE,CENTRE,SCREW,KEYWAY,THREAD,
CHAMFER,

 GROUP NO ==>  2 has components 4
 13   26   29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,
PARTIAL R_STEP,RECT. STEP,SURFACE,


 GROUP NO ==>  3 has components 6
 14   16   17   18   20   28

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_HOLE,THRU. RECT. SLOT,
TRIANG_STEP ,SURFACE,THREAD,


 GROUP NO ==>  4 has components 6
 21   22   23   24   25   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_HOLE,

S/FACE,C/S,K,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  1

COMPONENTS ARE
 19

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 4

 GROUPS IS /ARE  3

 GROUP NO ==>  1 has components 19
 1   2   3   4   5   6   7   8   9   10   11   12   14   15   16
 17   18   20   28

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,THRU. RECT. SLOT,PARTIAL R_SLOT,
TRIANG_STEP ,SURFACE,FACE,CENTRE,SCREW,KEYWAY,THREAD,
CHAMFER,

 GROUP NO ==>  2 has components 4
 13   26   29   30
```

```
CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,
PARTIAL R_STEP,RECT. STEP,SURFACE,


 GROUP NO ==>  3 has components 6
 21   22   23   24   25   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_HOLE,

S/FACE,C/S,K,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  1

COMPONENTS ARE
 19

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 3

 GROUPS IS /ARE  2

 GROUP NO ==>  1 has components 23
 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15
 16  17  18  20  26  28  29  30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,THRU. RECT. SLOT,PARTIAL R_SLOT,
PARTIAL                    R_STEP,TRIANG_STEP                       ,RECT.
STEP,SURFACE,FACE,CENTRE,SCREW,KEYWAY,THREAD,
CHAMFER,

 GROUP NO ==>  2 has components 6
 21   22   23   24   25   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_HOLE,

S/FACE,C/S,K,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  1

COMPONENTS ARE
 19

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 2

 GROUPS IS /ARE  2
```

```
GROUP NO ==>  1 has components 23
1  2  3  4  5  6  7  8  9  10  11  12  13  14  15
16  17  18  20  26  28  29  30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,THRU. RECT. SLOT,PARTIAL R_SLOT,
PARTIAL                          R_STEP,TRIANG_STEP                        ,RECT.
STEP,SURFACE,FACE,CENTRE,SCREW,KEYWAY,THREAD,
CHAMFER,

GROUP NO ==>  2 has components 7
19  21  22  23  24  25  27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_HOLE,R_POCKET,

C/BORE,S/FACE,C/S,K,

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 1

GROUPS IS /ARE  1

GROUP NO ==>  1 has components 30
1  2  3  4  5  6  7  8  9  10  11  12  13  14  15
16  17  18  19  20  21  22  23  24  25  26  27  28  29  30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

R_BOSS,R_HOLE,R_POCKET,THRU. RECT. SLOT,PARTIAL R_SLOT,
PARTIAL                          R_STEP,TRIANG_STEP                        ,RECT.
STEP,SURFACE,FACE,CENTRE,SCREW,KEYWAY,THREAD,
CHAMFER,C/BORE,S/FACE,C/S,K,
```

D.10.2 PROCESS-BASED GROUPING

SUGGESTED OPTIMAL NUMBER OF GROUPS = 2

```
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 8

GROUPS IS /ARE  7

GROUP NO ==>  1 has components 6
1  2  3  4  5  6

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

centring,facing,turning,screw_cutting,wibbling,grinding,drilling,taping,

GROUP NO ==>  2 has components 2
7  8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE
```

```
centring,facing,turning,

 GROUP NO ==>  3 has components 2
 9   10

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

turning,wibbling,drilling,taping,boring,chamfering,slotting,

 GROUP NO ==>  4 has components 10
 11   12   13   14   16   17   18   20   26   28

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

turning,screw_cutting,drilling,taping,boring,milling,

 GROUP NO ==>  5 has components 5
 15   22   23   24   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

turning,drilling,

 GROUP NO ==>  6 has components 2
 19   25

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

drilling,counterboring,countersinking,

 GROUP NO ==>  7 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

milling,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  1

COMPONENTS ARE
 21

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 7

 GROUPS IS /ARE  6

 GROUP NO ==>  1 has components 6
 1   2   3   4   5   6

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

centring,facing,turning,screw_cutting,wibbling,grinding,drilling,taping,

 GROUP NO ==>  2 has components 2
 7   8
```

```
CHARACTERISTICS INVOLVED IN THIS GROUP ARE

centring,facing,turning,

 GROUP NO ==>  3 has components 12
 9   10   11   12   13   14   16   17   18   20   26   28

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

turning,screw_cutting,wibbling,drilling,taping,boring,chamfering,milling,slotting,

 GROUP NO ==>  4 has components 5
 15   22   23   24   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

turning,drilling,

 GROUP NO ==>  5 has components 2
 19   25

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

drilling,counterboring,countersinking,

 GROUP NO ==>  6 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

milling,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  1

COMPONENTS ARE
 21

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 6

 GROUPS IS /ARE  5

 GROUP NO ==>  1 has components 18
 1   2   3   4   5   6   9   10   11   12   13   14   16   17   18
 20   26   28

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

centring,facing,turning,screw_cutting,wibbling,grinding,drilling,taping,boring,chamfering,milling,slo
tting,

 GROUP NO ==>  2 has components 2
 7   8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE
```

```
centring,facing,turning,

 GROUP NO ==> 3 has components 5
 15   22   23   24   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

turning,drilling,

 GROUP NO ==> 4 has components 2
 19   25

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

drilling,counterboring,countersinking,

 GROUP NO ==> 5 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

milling,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  1

COMPONENTS ARE
 21

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 5

 GROUPS IS /ARE  5

 GROUP NO ==> 1 has components 18
 1   2   3   4   5   6   9   10   11   12   13   14   16   17   18
 20   26   28

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

centring,facing,turning,screw_cutting,wibbling,grinding,drilling,taping,boring,chamfering,milling,slo
tting,

 GROUP NO ==> 2 has components 2
 7   8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

centring,facing,turning,

 GROUP NO ==> 3 has components 6
 15   21   22   23   24   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

turning,drilling,spotfacing,
 GROUP NO ==>  4 has components 2
 19   25

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

drilling,counterboring,countersinking,

 GROUP NO ==>  5 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

milling,

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 4

 GROUPS IS /ARE  4

 GROUP NO ==>  1 has components 20
 1  2  3  4  5  6  7  8  9  10  11  12  13  14  16
 17  18  20  26  28

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

centring,facing,turning,screw_cutting,wibbling,grinding,drilling,taping,boring,chamfering,milling,slo
tting,

 GROUP NO ==>  2 has components 6
 15  21  22  23  24  27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

turning,drilling,spotfacing,

 GROUP NO ==>  3 has components 2
 19   25

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

drilling,counterboring,countersinking,

 GROUP NO ==>  4 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

milling,

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 3

 GROUPS IS /ARE  3

 GROUP NO ==>  1 has components 20
 1  2  3  4  5  6  7  8  9  10  11  12  13  14  16
 17  18  20  26  28
```

```
CHARACTERISTICS INVOLVED IN THIS GROUP ARE

centring,facing,turning,screw_cutting,wibbling,grinding,drilling,taping,boring,chamfering,milling,slo
tting,

 GROUP NO ==> 2 has components 8
 15  19  21  22  23  24  25  27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

turning,drilling,counterboring,spotfacing,countersinking,

 GROUP NO ==> 3 has components 2
 29  30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

milling,

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 2

 GROUPS IS /ARE  2

 GROUP NO ==> 1 has components 28
 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15
 16  17  18  19  20  21  22  23  24  25  26  27  28

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

centring,facing,turning,screw_cutting,wibbling,grinding,drilling,taping,boring,counterboring,spotfaci
ng,countersinking,chamfering,milling,slotting,

 GROUP NO ==> 2 has components 2
 29  30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

milling,

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 1

 GROUPS IS /ARE  1

 GROUP NO ==> 1 has components 30
 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15
 16  17  18  19  20  21  22  23  24  25  26  27  28  29  30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

centring,facing,turning,screw_cutting,wibbling,grinding,drilling,taping,boring,counterboring,spotfaci
ng,countersinking,chamfering,milling,slotting,
```

D.10.3 MACHINE-BASED GROUPING

SUGGESTED OPTIMAL NUMBER OF GROUPS = 13

```
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 15
 GROUPS IS /ARE  6
 GROUP NO ==>  1 has components 4
 3  4  5  6
MACHINES INVOLVED IN THIS GROUP ARE
END,MGR,MRD,SHI,
 GROUP NO ==>  2 has components 2
 7  8
MACHINES INVOLVED IN THIS GROUP ARE
HEI,KTW,
 GROUP NO ==>  3 has components 3
 12  13  26
MACHINES INVOLVED IN THIS GROUP ARE
HOR,MRD,QKO,
 GROUP NO ==>  4 has components 8
 16  19  20  21  23  24  25  27
MACHINES INVOLVED IN THIS GROUP ARE
WTD,
 GROUP NO ==>  5 has components 2
 17  18
MACHINES INVOLVED IN THIS GROUP ARE
VER,WTD,
 GROUP NO ==>  6 has components 2
 29  30
MACHINES INVOLVED IN THIS GROUP ARE
HOR,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  9
COMPONENTS ARE
 1  2  9  10  11  14  15  22  28
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 14
 GROUPS IS /ARE  6
 GROUP NO ==>  1 has components 4
 3  4  5  6
MACHINES INVOLVED IN THIS GROUP ARE
END,MGR,MRD,SHI,
 GROUP NO ==>  2 has components 2
 7  8
MACHINES INVOLVED IN THIS GROUP ARE
HEI,KTW,
 GROUP NO ==>  3 has components 3
 12  13  26
MACHINES INVOLVED IN THIS GROUP ARE
HOR,MRD,QKO,
 GROUP NO ==>  4 has components 3
 14  17  18
MACHINES INVOLVED IN THIS GROUP ARE
HOR,VER,WTD,
 GROUP NO ==>  5 has components 8
 16  19  20  21  23  24  25  27
```

```
MACHINES INVOLVED IN THIS GROUP ARE
WTD,
 GROUP NO ==> 6 has components 2
 29  30
MACHINES INVOLVED IN THIS GROUP ARE
HOR,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  8
COMPONENTS ARE
 1  2  9  10  11  15  22  28
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 13
 GROUPS IS /ARE  7
 GROUP NO ==> 1 has components 4
 3  4  5  6
MACHINES INVOLVED IN THIS GROUP ARE
END,MGR,MRD,SHI,
 GROUP NO ==> 2 has components 2
 7  8
MACHINES INVOLVED IN THIS GROUP ARE
HEI,KTW,
 GROUP NO ==> 3 has components 2
 9  22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,MAK,
 GROUP NO ==> 4 has components 3
 12  13  26
MACHINES INVOLVED IN THIS GROUP ARE
HOR,MRD,QKO,
 GROUP NO ==> 5 has components 3
 14  17  18
MACHINES INVOLVED IN THIS GROUP ARE
HOR,VER,WTD,
 GROUP NO ==> 6 has components 8
 16  19  20  21  23  24  25  27
MACHINES INVOLVED IN THIS GROUP ARE
WTD,
 GROUP NO ==> 7 has components 2
 29  30
MACHINES INVOLVED IN THIS GROUP ARE
HOR,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  6
COMPONENTS ARE
 1  2  10  11  15  28
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 12
 GROUPS IS /ARE  7
 GROUP NO ==> 1 has components 5
 2  3  4  5  6
MACHINES INVOLVED IN THIS GROUP ARE
END,KEY,MGR,MRD,SHI,SMU,
 GROUP NO ==> 2 has components 2
 7  8
MACHINES INVOLVED IN THIS GROUP ARE
HEI,KTW,
 GROUP NO ==> 3 has components 2
 9  22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,MAK,
 GROUP NO ==> 4 has components 3
 12  13  26
MACHINES INVOLVED IN THIS GROUP ARE
HOR,MRD,QKO,
 GROUP NO ==> 5 has components 3
 14  17  18
MACHINES INVOLVED IN THIS GROUP ARE
HOR,VER,WTD,
 GROUP NO ==> 6 has components 8
 16  19  20  21  23  24  25  27
MACHINES INVOLVED IN THIS GROUP ARE
WTD,
 GROUP NO ==> 7 has components 2
 29  30
MACHINES INVOLVED IN THIS GROUP ARE
HOR,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  5
COMPONENTS ARE
 1  10  11  15  28  L/D < 3.0 && L/D > 0.5,
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 11
 GROUPS IS /ARE  7
 GROUP NO ==> 1 has components 6
 2  3  4  5  6  11
MACHINES INVOLVED IN THIS GROUP ARE
END,KEY,MGR,MRD,SHI,SMU,
 GROUP NO ==> 2 has components 2
 7  8
MACHINES INVOLVED IN THIS GROUP ARE
HEI,KTW,
 GROUP NO ==> 3 has components 2
 9  22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,MAK,
 GROUP NO ==> 4 has components 3
 12  13  26
MACHINES INVOLVED IN THIS GROUP ARE
HOR,MRD,QKO,
 GROUP NO ==> 5 has components 3
 14  17  18
MACHINES INVOLVED IN THIS GROUP ARE
HOR,VER,WTD,
 GROUP NO ==> 6 has components 8
 16  19  20  21  23  24  25  27
MACHINES INVOLVED IN THIS GROUP ARE
WTD,
 GROUP NO ==> 7 has components 2
 29  30
MACHINES INVOLVED IN THIS GROUP ARE
HOR,
 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  4
COMPONENTS ARE
 1  10  15  28
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 10
 GROUPS IS /ARE  7
 GROUP NO ==>  1 has components 6
 2   3   4   5   6   11
MACHINES INVOLVED IN THIS GROUP ARE
END,KEY,MGR,MRD,SHI,SMU,
 GROUP NO ==>  2 has components 3
 7   8   28
MACHINES INVOLVED IN THIS GROUP ARE
HEI,KTW,VER,
 GROUP NO ==>  3 has components 2
 9   22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,MAK,
 GROUP NO ==>  4 has components 3
 12   13   26
MACHINES INVOLVED IN THIS GROUP ARE
HOR,MRD,QKO,
 GROUP NO ==>  5 has components 3
 14   17   18
MACHINES INVOLVED IN THIS GROUP ARE
HOR,VER,WTD,
 GROUP NO ==>  6 has components 8
 16   19   20   21   23   24   25   27
MACHINES INVOLVED IN THIS GROUP ARE
WTD,
 GROUP NO ==>  7 has components 2
 29   30
MACHINES INVOLVED IN THIS GROUP ARE
HOR,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  3
COMPONENTS ARE
 1  10  15
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 9
 GROUPS IS /ARE  6
 GROUP NO ==>  1 has components 6
 2   3   4   5   6   11
MACHINES INVOLVED IN THIS GROUP ARE
END,KEY,MGR,MRD,SHI,SMU,
 GROUP NO ==>  2 has components 3
 7   8   28
MACHINES INVOLVED IN THIS GROUP ARE
HEI,KTW,VER,
 GROUP NO ==>  3 has components 2
 9   22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,MAK,
 GROUP NO ==>  4 has components 5
 12   13   26   29   30
MACHINES INVOLVED IN THIS GROUP ARE
```

```
HOR,MRD,QKO,
 GROUP NO ==>  5 has components 3
 14   17   18
MACHINES INVOLVED IN THIS GROUP ARE
HOR,VER,WTD,
 GROUP NO ==>  6 has components 8
 16   19   20   21   23   24   25   27
MACHINES INVOLVED IN THIS GROUP ARE
WTD,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  3
COMPONENTS ARE
 1  10  15
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 8
 GROUPS IS /ARE  5
 GROUP NO ==>  1 has components 6
 2   3   4   5   6   11
MACHINES INVOLVED IN THIS GROUP ARE
END,KEY,MGR,MRD,SHI,SMU,
 GROUP NO ==>  2 has components 3
 7   8   28
MACHINES INVOLVED IN THIS GROUP ARE
HEI,KTW,VER,
 GROUP NO ==>  3 has components 2
 9   22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,MAK,
 GROUP NO ==>  4 has components 5
 12   13   26   29   30
MACHINES INVOLVED IN THIS GROUP ARE
HOR,MRD,QKO,
 GROUP NO ==>  5 has components 11
 14   16   17   18   19   20   21   23   24   25   27
MACHINES INVOLVED IN THIS GROUP ARE
HOR,VER,WTD,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  3
COMPONENTS ARE
 1  10  15
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 7
 GROUPS IS /ARE  5
 GROUP NO ==>  1 has components 6
 2   3   4   5   6   11
MACHINES INVOLVED IN THIS GROUP ARE
END,KEY,MGR,MRD,SHI,SMU,
 GROUP NO ==>  2 has components 3
 7   8   28
MACHINES INVOLVED IN THIS GROUP ARE
HEI,KTW,VER,
 GROUP NO ==>  3 has components 3
 9   10   22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,LAN,MAK,SLT,
 GROUP NO ==>  4 has components 5
 12   13   26   29   30
MACHINES INVOLVED IN THIS GROUP ARE
HOR,MRD,QKO,
 GROUP NO ==>  5 has components 11
 14   16   17   18   19   20   21   23   24   25   27
MACHINES INVOLVED IN THIS GROUP ARE
HOR,VER,WTD,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  2
COMPONENTS ARE
 1  15
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 6
 GROUPS IS /ARE  4
 GROUP NO ==>  1 has components 6
 2   3   4   5   6   11
MACHINES INVOLVED IN THIS GROUP ARE
END,KEY,MGR,MRD,SHI,SMU,
 GROUP NO ==>  2 has components 14
 7   8   14   16   17   18   19   20   21   23   24   25   27   28
MACHINES INVOLVED IN THIS GROUP ARE
HEI,HOR,KTW,VER,WTD,
 GROUP NO ==>  3 has components 3
 9   10   22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,LAN,MAK,SLT,
 GROUP NO ==>  4 has components 5
 12   13   26   29   30
MACHINES INVOLVED IN THIS GROUP ARE
HOR,MRD,QKO,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  2
COMPONENTS ARE
 1  15
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 5
 GROUPS IS /ARE  3
 GROUP NO ==>  1 has components 6
 2   3   4   5   6   11
MACHINES INVOLVED IN THIS GROUP ARE
END,KEY,MGR,MRD,SHI,SMU,
 GROUP NO ==>  2 has components 19
 7   8   12   13   14   16   17   18   19   20   21   23   24   25   26   27   28   29   30
MACHINES INVOLVED IN THIS GROUP ARE
HEI,HOR,KTW,MRD,QKO,VER,WTD,
 GROUP NO ==>  3 has components 3
 9   10   22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,LAN,MAK,SLT,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  2
COMPONENTS ARE
 1  15
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 4
 GROUPS IS /ARE  3
 GROUP NO ==> 1 has components 7
 1  2  3  4  5  6  11
MACHINES INVOLVED IN THIS GROUP ARE
EDG,END,KEY,MGR,MRD,SHI,SMU,UNV,
 GROUP NO ==> 2 has components 19
7  8  12  13  14  16  17  18  19  20  21  23  24  25  26  27  28  29  30
MACHINES INVOLVED IN THIS GROUP ARE
HEI,HOR,KTW,MRD,QKO,VER,WTD,
 GROUP NO ==> 3 has components 3
9  10  22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,LAN,MAK,SLT,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  1
COMPONENTS ARE
 15
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 3
 GROUPS IS /ARE  2
 GROUP NO ==> 1 has components 26
1  2  3  4  5  6  7  8  11  12  13  14  16  17  18  19  20  21  23  24  25  26
27  28  29  30
MACHINES INVOLVED IN THIS GROUP ARE
EDG,END,HEI,HOR,KEY,KTW,MGR,MRD,QKO,SHI,SMU,UNV,VER,WTD,
 GROUP NO ==> 2 has components 3
9  10  22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,LAN,MAK,SLT,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  1
COMPONENTS ARE
 15
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 2
 GROUPS IS /ARE  1
 GROUP NO ==> 1 has components 3
9  10  22
MACHINES INVOLVED IN THIS GROUP ARE
BULL,LAN,MAK,SLT,

 ******* UNGROUPED COMPONENTS ARE :  ************
No of Components are =  1
COMPONENTS ARE
 15
```

D.10.4 GROUPING BASED ON THE CAFBG SYSTEM

SUGGESTED OPTIMAL NUMBER OF GROUPS = 8

```
TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 10

 GROUPS IS /ARE  8

 GROUP NO ==> 1 has components 6
 1  2  3  4  5  6
```

```
CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_1,MCU_4,MCU_8,MCU_9,ST.PATTERN,AD's = 3,

 GROUP NO ==>  2 has components 2
 7  8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_4,AD's = 3,

 GROUP NO ==>  3 has components 5
 9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D   <=   0.5,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_9,MCU_10,P.C.PATTERN,AD's   =
3,AD's = 4,

 GROUP NO ==>  4 has components 2
 14   16

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W <= 3 && L/T < 4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's = 1,AD's = 3,

 GROUP NO ==>  5 has components 3
 17   18   20

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W <= 3 && L/T >= 4,MCU_1,MCU_7,AD's = 1,

 GROUP NO ==>  6 has components 6
 19   21   22   23   24   26

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W <= 3 && L/T >= 4,MCU_1,MCU_6,ST.PATTERN,AD's = 1,AD's = 6,

 GROUP NO ==>  7 has components 2
 25   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_1,ST.PATTERN,AD's = 1,

 GROUP NO ==>  8 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_6,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
```

APPENDIX D

```
 No of Components are =  2

COMPONENTS ARE
 15  28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 9

 GROUPS IS /ARE  7

 GROUP NO ==>  1 has components 6
 1   2   3   4   5   6

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_1,MCU_4,MCU_8,MCU_9,ST.PATTERN,AD's = 3,

 GROUP NO ==>  2 has components 2
 7   8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_4,AD's = 3,

 GROUP NO ==>  3 has components 5
 9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D   <=   0.5,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_9,MCU_10,P.C.PATTERN,AD's   =
3,AD's = 4,

 GROUP NO ==>  4 has components 8
 14   16   19   21   22   23   24   26

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W  <=  3  &&  L/T  >=  4,L/W  <=  3  &&  L/T  <  4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's =
1,AD's = 3,AD's = 6,

 GROUP NO ==>  5 has components 3
 17   18   20

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W <= 3 && L/T >= 4,MCU_1,MCU_7,AD's = 1,

 GROUP NO ==>  6 has components 2
 25   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_1,ST.PATTERN,AD's = 1,

 GROUP NO ==>  7 has components 2
 29   30
```

```
CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_6,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
 15   28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 8

 GROUPS IS /ARE  6

 GROUP NO ==>  1 has components 6
 1   2   3   4   5   6

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_1,MCU_4,MCU_8,MCU_9,ST.PATTERN,AD's = 3,

 GROUP NO ==>  2 has components 2
 7   8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_4,AD's = 3,

 GROUP NO ==>  3 has components 5
 9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D   <=   0.5,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_9,MCU_10,P.C.PATTERN,AD's   =
3,AD's = 4,

 GROUP NO ==>  4 has components 11
 14   16   17   18   19   20   21   22   23   24   26

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W  <=  3  &&  L/T  >=  4,L/W  <=  3  &&  L/T  <  4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's  =
1,AD's = 3,AD's = 6,

 GROUP NO ==>  5 has components 2
 25   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_1,ST.PATTERN,AD's = 1,

 GROUP NO ==>  6 has components 2
 29   30
```

```
CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_6,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
 15  28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 7

 GROUPS IS /ARE  5

 GROUP NO ==>  1 has components 8
 1   2   3   4   5   6   7   8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_1,MCU_4,MCU_8,MCU_9,ST.PATTERN,AD's = 3,

 GROUP NO ==>  2 has components 5
 9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D  <=  0.5,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_9,MCU_10,P.C.PATTERN,AD's  =
3,AD's = 4,

 GROUP NO ==>  3 has components 11
 14   16   17   18   19   20   21   22   23   24   26

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W  <=  3  &&  L/T  >=  4,L/W  <=  3  &&  L/T  <  4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's =
1,AD's = 3,AD's = 6,

 GROUP NO ==>  4 has components 2
 25   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_1,ST.PATTERN,AD's = 1,

 GROUP NO ==>  5 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_6,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
```

```
 15  28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 6

 GROUPS IS /ARE  4

 GROUP NO ==>  1 has components 8
 1  2  3  4  5  6  7  8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_1,MCU_4,MCU_8,MCU_9,ST.PATTERN,AD's = 3,

 GROUP NO ==>  2 has components 5
 9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D   <=   0.5,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_9,MCU_10,P.C.PATTERN,AD's   =
3,AD's = 4,

 GROUP NO ==>  3 has components 13
 14   16   17   18   19   20   21   22   23   24   25   26   27

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W     <=     3     &&     L/T     >=     4,L/W     >     3,L/W     <=     3     &&     L/T     <
4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's = 1,AD's = 3,AD's = 6,

 GROUP NO ==>  4 has components 2
 29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W > 3,MCU_6,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
 15  28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 5

 GROUPS IS /ARE  3

 GROUP NO ==>  1 has components 8
 1  2  3  4  5  6  7  8

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D >= 3.0,MCU_1,MCU_4,MCU_8,MCU_9,ST.PATTERN,AD's = 3,

 GROUP NO ==>  2 has components 5
```

```
9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D   <=   0.5,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_9,MCU_10,P.C.PATTERN,AD's   =
3,AD's = 4,

 GROUP NO ==>  3 has components 15
 14   16   17   18   19   20   21   22   23   24   25   26   27   29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W    <=    3    &&    L/T    >=    4,L/W    >    3,L/W    <=    3    &&    L/T    <
4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's = 1,AD's = 3,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
 15   28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 4

 GROUPS IS /ARE  2

 GROUP NO ==>  1 has components 13
 1   2   3   4   5   6   7   8   9   10   11   12   13

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D                          <=                          0.5,L/D                          >=
3.0,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_8,MCU_9,MCU_10,P.C.PATTERN,ST.PATTE
RN,AD's = 3,AD's = 4,

 GROUP NO ==>  2 has components 15
 14   16   17   18   19   20   21   22   23   24   25   26   27   29   30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W    <=    3    &&    L/T    >=    4,L/W    >    3,L/W    <=    3    &&    L/T    <
4,MCU_1,MCU_6,MCU_7,ST.PATTERN,AD's = 1,AD's = 3,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  2

COMPONENTS ARE
 15   28


TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 3

 GROUPS IS /ARE  2

 GROUP NO ==>  1 has components 13
 1   2   3   4   5   6   7   8   9   10   11   12   13
```

```
CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D <= 0.5,L/D >= 3.0,MCU_1,MCU_2,MCU_4,MCU_5,MCU_6,MCU_8,MCU_9,MCU_10,P.C.PATTERN,ST.PATTE
RN,AD's = 3,AD's = 4,

 GROUP NO ==> 2 has components 16
 14  16  17  18  19  20  21  22  23  24  25  26  27  28  29
 30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/W <= 3 && L/T >= 4,L/W > 3,L/W <= 3 && L/T <
4,MCU_1,MCU_3,MCU_6,MCU_7,ST.PATTERN,AD's = 1,AD's = 3,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE : '************
 No of Components are =  1

COMPONENTS ARE
 15

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 2

 GROUPS IS /ARE  1

 GROUP NO ==>  1 has components 29
 1  2  3  4  5  6  7  8  9  10  11  12  13  14  16
 17  18  19  20  21  22  23  24  25  26  27  28  29  30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D <= 0.5,L/D >= 3.0,L/W <= 3 && L/T >= 4,L/W > 3,L/W <= 3 && L/T <
4,MCU_1,MCU_2,MCU_3,MCU_4,MCU_5,MCU_6,MCU_7,MCU_8,MCU_9,MCU_10,P.C.PATTE
RN,ST.PATTERN,AD's = 1,AD's = 3,AD's = 4,AD's = 6,

 ******* UNGROUPED COMPONENTS ARE :  ************
 No of Components are =  1

COMPONENTS ARE
 15

TOTAL NUMBER OF GROUPS AT THIS ITERATION ARE =====> 1

 GROUPS IS /ARE  1

 GROUP NO ==>  1 has components 30
 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15
 16  17  18  19  20  21  22  23  24  25  26  27  28  29  30

CHARACTERISTICS INVOLVED IN THIS GROUP ARE

L/D <= 0.5,L/D < 3.0 && L/D > 0.5,L/D >= 3.0,L/W <= 3 && L/T >= 4,L/W > 3,L/W <= 3 && L/T
<
4,MCU_1,MCU_2,MCU_3,MCU_4,MCU_5,MCU_6,MCU_7,MCU_8,MCU_9,MCU_10,P.C.PATTE
RN,ST.PATTERN,AD's = 1,AD's = 3,AD's = 4,AD's = 6,
```

Appendix E

Table of Contents

E SOFTWARE PROGRAMS INVOLVED IN THE HYCAPP SYSTEM

E.1 INTRODUCTION

The software programs developed for implementation of the HYCAPP System are listed in this appendix. The hierarchy (the sequence in which they are called) of the designed programs is shown in figure E.1.

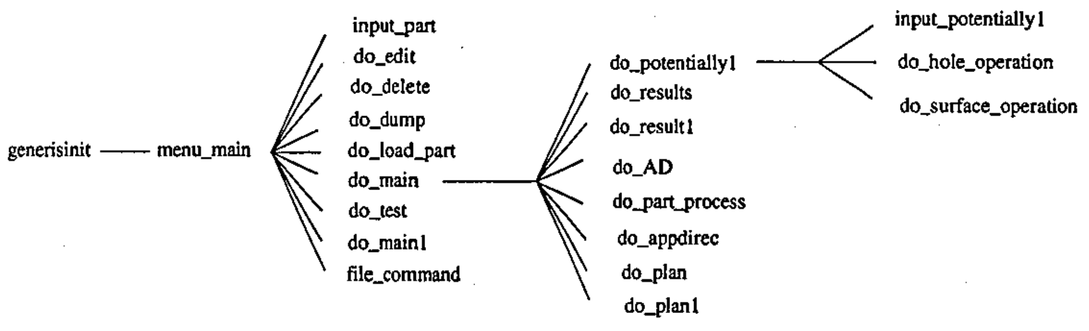


Figure E.1 Software programs hierarchy in the HYCAPP System

E.2 PROCEDURE FILE 'generisinit'

```
*
* DO GENERIS  FILE  'generisinit'
*
* START UP FILE FOR THE DEMONSTRATION
*
WINDOW NEW MAIN_MENU SIZE 37,92 AT 2,2
POSITION 5,12
MESSAGE "\THE APPLICATION IS OPENED ONCE YOU HAVE INPUT THE PASSWORD\ \"
OPEN fahd
CLEAR
POSITION 9,24
MESSAGE "PLEASE MAKE SELECTION AS PER REQUIREMENTS \"
POSITION 12,18
MENU menu_main
WINDOW COMMAND
DELETE WINDOW ALL
CLOSE
```

E.3 PROCEDURE MENU FILE 'menu_main'

```
*
* GENERIS MENU FILE 'menu_main'
*
TITLE SELECTION
TO CREATE THE RECORDS FOR A COMPONENT:DO input_part;
TO EDIT THE RECORDS FOR ANY COMPONENT:DO do_edit;
TO DELETE THE RECORDS OF ANY COMPONENT:DO do_delete;
TO DUMP THE RECORDS FOR THE COMPONENT:DO do_dump;
TO LOAD THE RECORDS OF PART FOR PROCESS PLANNING:DO do_load_part;
TO GENERATE THE PROCESS PLANS:DO do_main;
TO CHECK FOR A NEW PART WHETHER IT CAN BE PROCESSED:DO do_test;
TYPE ANY COMMAND:DO file_command;
END:MENU RETURN;
```

E.4 PROCEDURE FILE 'do_edit'

```
*
* DO GENERIS FILE 'do_edit'
*
**************************************************************
*  FILE TO EDIT THE RECORDS FOR A COMPONENT
**************************************************************
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL int0 INTEGER 1
CREATE LOCAL int1 INTEGER
CREATE LOCAL int2 INTEGER
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
CREATE LOCAL nam4 NAME
CREATE LOCAL tex0 TEXT
CREATE LOCAL dec0 DECIMAL
CREATE LOCAL dec1 DECIMAL
CREATE LOCAL dec2 DECIMAL
CREATE LOCAL dec3 DECIMAL
*
WINDOW NEW RECORD EDITING SIZE 37,92 AT 2,2
CLEAR
POSITION int0,32
MESSAGE"EDITING COMPONENT RECORDS"
POSITION int0+1,32
MESSAGE"************************"
IF TELL ANY RECORDS IN fea_list
    DISABLE ALL
    LABEL again1
    LET int0 = 3
    FETCH NEW RECORDS IN fea_list
    LET fetchno = $FETCH
    LET int2 = $COUNT
    LET nam0  = FETCH VALUE (fetchno,1,1)
    POSITION int0,27
    MESSAGE"EDITING RECORDS FOR THE COMPONENT  "^nam0 "  "
    LET int0 = int0 + 1
```

```
POSITION int0,22
MESSAGE"++++++++++++++++++++++++++++++++++++++++++++++++"
POSITION int0+1,22
MESSAGE" FEATURECODE        FEATURE"
POSITION int0+2,22
MESSAGE"**********************************************"
LET int0 = int0 + 3
LET int1 = 0
WHILE int2 > 0
    LET int1 = int1 + 1
    LET nam1 = FETCH VALUE (fetchno,int1,2)
    LET nam2 = FETCH VALUE (fetchno,int1,3)
    IF nam2 = external_surface

    ELSE
        POSITION int0,24
        MESSAGE""^nam1 " "
        POSITION int0,44
        MESSAGE""^nam2 " "
        LET int0 = int0 + 1
    ENDIF
    LET int2 = int2 - 1
ENDWHILE
DELETE FETCH fetchno
LABEL again
POSITION int0+1,10
MESSAGE"PLEASE ENTER THE FEATURECODE FOR WHICH RECORDS ARE TO BE
CHANGED?"
READ nam1
IF nam1 = BLANK
    POSITION int0+2,10
    MESSAGE"YOU HAVE NOT MENTIONED THE FEATURE TO BE EDITED"
    POSITION int0+3,10
    MESSAGE"DO YOU WANT TO QUIT? y/YES OR <RETURN>"
    READ tex0
    IF tex0 = y
        GOTO again3
    ELSE
        GOTO again
    ENDIF
ELSE
    IF TELL ANY RECORDS IN fea_data FOR ^nam1
        FETCH RECORDS IN fea_data FOR ^nam1
        FETCH RECORD 1
        LET dec0 = FIELD 8
        LET dec1 = FIELD 11
        LET dec2 = FIELD 14
        LET dec3 = FIELD 17
        LET dec4 = FIELD 28
        CLEAR SIZE 34,85 AT 3,2
        LET int0 = 4
        IF dec2 = BLANK
            POSITION int0,10
            MESSAGE"RECORDS FOR FEATURECODE "^nam1" DO NOT EXIST"
        ELSE
            POSITION int0,20
```

```
MESSAGE"FEATURECODE HAS FOLLOWING ATTRIBUTES"
IF dec3 = BLANK
    LET int0 = int0 + 1
    POSITION int0,10
    MESSAGE"LENGTH "dec0" WIDTH "dec1" DEPTH "dec2" AND SURFACE
    FINISH "dec4""
    LET int0 = int0 + 1
    POSITION int0,10
    MESSAGE"DO YOU WANT TO CHANGE ANY DATA? y/YES OR
    <RETURN>"
    READ tex0
    IF tex0 = y
        LET int0 = int0 + 1
        POSITION int0,10
        MESSAGE"DO YOU WANT TO CHANGE THE FEATURE LENGTH?
        y/YES OR <RETURN>"
        READ tex0
        IF tex0 = y
            LET int0 = int0 + 1
            POSITION int0,10
            MESSAGE"PLEASE ENTER THE NEW LENGTH: "
            READ dec0
        ENDIF
        LET int0 = int0 + 1
        POSITION int0,10
        MESSAGE"DO YOU WANT TO CHANGE THE FEATURE WIDTH?
        y/YES OR <RETURN>"
        READ tex0
        IF tex0 = y
            LET int0 = int0 + 1
            POSITION int0,10
            MESSAGE"PLEASE ENTER THE NEW WIDTH: "
            READ dec1
        ENDIF
        LET int0 = int0 + 1
        POSITION int0,10
        MESSAGE"DO YOU WANT TO CHANGE THE FEATURE DEPTH?
        y/YES OR <RETURN>"
        READ tex0
        IF tex0 = y
            LET int0 = int0 + 1
            POSITION int0,10
            MESSAGE"PLEASE ENTER THE NEW DEPTH: "
            READ dec2
        ENDIF
        LET int0 = int0 + 1
        POSITION int0,10
        MESSAGE"DO YOU WANT TO CHANGE THE FEATURE SURFACE
        FINISH? y/YES OR <RETURN>"
        READ tex0
        IF tex0 = y
            LET int0 = int0 + 1
            POSITION int0,10
            MESSAGE"PLEASE ENTER THE NEW SURFACE FINISH: "
            READ dec4
        ENDIF
        IF dec4 = BLANK
            LET dec4 = 8
        ENDIF
        FACT ^nam1 fealengh dec0 feawidth dec1 feadepth dec2 'has surfinish'
        dec4
    ENDIF
ELSE
    LET int0 = int0 + 1
    POSITION int0,10
    MESSAGE"DIAMETER "dec3" DEPTH "dec2" AND SURFACE FINISH
    "dec4""
    LET int0 = int0 + 1
    POSITION int0,10
    MESSAGE"DO YOU WANT TO CHANGE ANY DATA? y/YES OR
    <RETURN>"
    READ tex0
    IF tex0 = y
        LET int0 = int0 + 1
        POSITION int0,10
        MESSAGE"DO YOU WANT TO CHANGE THE FEATURE
        DIAMETER? y/YES OR <RETURN>"
        READ tex0
        IF tex0 = y
            LET int0 = int0 + 1
            POSITION int0,10
            MESSAGE"PLEASE ENTER THE NEW DIAMETER: "
            READ dec3
        ENDIF
        LET int0 = int0 + 1
        POSITION int0,10
        MESSAGE"DO YOU WANT TO CHANGE THE FEATURE DEPTH?
        y/YES OR <RETURN>"
        READ tex0
        IF tex0 = y
            LET int0 = int0 + 1
            POSITION int0,10
            MESSAGE"PLEASE ENTER THE NEW DEPTH: "
            READ dec2
        ENDIF
        LET int0 = int0 + 1
        POSITION int0,10
        MESSAGE"DO YOU WANT TO CHANGE THE FEATURE SURFACE
        FINISH? y/YES OR <RETURN>"
        READ tex0
        IF tex0 = y
            LET int0 = int0 + 1
            POSITION int0,10
            MESSAGE"PLEASE ENTER THE NEW SURFACE FINISH: "
            READ dec4
        ENDIF
        IF dec4 = BLANK
            LET dec4 = 8
        ENDIF
        FACT ^nam1 feadepth dec2 feadiameter dec3 'has surfinish' dec4
    ENDIF
ENDIF
```

```
            ENDIF
        ELSE
            CLEAR SIZE 34,85 AT 3,2
            LET int0 = int0 + 1
            POSITION int0,10
            MESSAGE"DATA HAS NOT BEEN ENTERED FOR "^nam1", CREATE DATA?
            y/YES OR <RETURN>"
            READ tex0
            IF tex0 = y
                CLEAR SIZE 34,85 AT 3,2
                LET int0 = int0 + 1
                POSITION int0,10
                MESSAGE"PLEASE ENTER THE FEATURE LENGTH"
                READ dec0
                POSITION int0+1,10
                MESSAGE"PLEASE ENTER THE FEATURE WIDTH, IF EXISTS"
                READ dec1
                POSITION int0+2,10
                MESSAGE"PLEASE ENTER THE FEATURE DIAMETER, IF EXISTS"
                READ dec3
                POSITION int0+3,10
                MESSAGE"PLEASE ENTER THE FEATURE DEPTH"
                READ dec2
                POSITION int0+4,10
                MESSAGE"PLEASE ENTER THE FEATURE SURFACE FINISH"
                READ dec4
                IF dec4 = BLANK
                    LET dec4 = 8
                ENDIF
                IF dec3 = BLANK
                    FACT ^nam1 fealengh dec0 feawidth dec1 feadepth dec2 'has surfinish' dec4
                ELSE
                     FACT ^nam1 feadepth dec2 feadiameter dec3 'has surfinish' dec4
                ENDIF
            ENDIF
        ENDIF
    ENDIF
    CLEAR SIZE 34,85 AT 3,2
    LET int0 = int0 + 1
    POSITION int0,10
    MESSAGE"MORE FEATURES TO BE EDITED? y/YES OR <RETURN>"
    READ tex0
    IF tex0 = y
        CLEAR SIZE 34,85 AT 3,2
        GOTO again1
    ENDIF
ELSE
    POSITION int0,10
    MESSAGE "RECORDS IN TABLE    fea_list   DO NOT EXIST"
    POSITION int0+1,10
    MESSAGE "*******************************************"
    HOLD 4
ENDIF
LABEL again3
CLEAR
WINDOW MAIN_MENU
```

E.5 PROCEDURE FILE 'do_delete'

```
*
* DO GENERIS FILE 'do_delete'
*
*****************************************************
* TO DELETE THE PART RECORDS IN THE GENERIS APPLICATION
*****************************************************
*
CLEAR
POSITION 6,28
MESSAGE "DELETING THE RECORDS ..."
DELETE RECORDS IN parts
DELETE RECORDS IN fea_list
DELETE RECORDS IN fea_data
DELETE RECORDS IN surfaces
DELETE RECORDS IN nor_surfaces
DELETE RECORDS IN sec_features
CLEAR
WINDOW MAIN_MENU
```

E.6 PROCEDURE FILE 'do_dump'

```
*
* DO GENERIS FILE 'do_dump'
*
* TO DUMP THE RECORDS FOR A COMPONENT OUTSIDE THE GENERIS APPLICATION
***********************************************************************
CREATE LOCAL tex2 TEXT
WINDOW NEW PART_DUMP SIZE 37,92 AT 2,2
LABEL again
POSITION 4,14
***********************************************************************
* ASK THE FILE NAME TO DUMP THE RECORDS IN
***********************************************************************
MESSAGE "PLEASE PROVIDE THE FILE NAME TO DUMP THE RECORDS IN ?"
READ tex2
IF tex2 = BLANK
    POSITION 5,30
    MESSAGE "YOU HAVE NOT INPUT THE FILE NAME"
    POSITION 6,1
    GOTO again
ENDIF
CLEAR
POSITION 6,32
MESSAGE "DUMPING THE RECORDS ..."
IF TELL ANY RECORDS IN parts
    DUMP RECORDS IN parts TO tex2
    DUMP RECORDS IN fea_list TO tex2
    DUMP RECORDS IN fea_data TO tex2
    DUMP RECORDS IN surfaces TO tex2
    DUMP RECORDS IN nor_surfaces TO tex2
    DUMP RECORDS IN sec_features TO tex2
    CLEAR
    POSITION 8,32
    MESSAGE "DUMPING COMPLETE "
    HOLD 2
```

```
ELSE
    CLEAR
    POSITION 8,25
    MESSAGE"RECORDS IN TABLE 'parts' DO NOT EXIST"
ENDIF
CLEAR
WINDOW MAIN_MENU
DELETE WINDOW PART_DUMP
```

E.7 PROCEDURE FILE 'do_load_part

```
*
* DO GENERIS FILE 'do_load_part'
*
******************************************************************************
* TO LOAD THE DATA FOR THE PART FOR WHICH PROCESS PLANS ARE TO BR
WRITTEN
******************************************************************************
*
CREATE LOCAL tex1 TEXT
CLEAR
IF TELL ANY RECORDS IN parts
    POSITION 12,15
    MESSAGE"DELETING THE RECORDS FOR EXISTING COMPONENT "
    POSITION 10,30
    MESSAGE"PLEASE WAIT ... "
    IF TELL ANY RECORDS IN parts
        DELETE RECORDS IN parts
    ENDIF
    IF TELL ANY RECORDS IN fea_list
        DELETE RECORDS IN fea_list
    ENDIF
    IF TELL ANY RECORDS IN fea_data
        DELETE RECORDS IN fea_data
    ENDIF
    IF TELL ANY RECORDS IN surfaces
        DELETE RECORDS IN surfaces
    ENDIF
    IF TELL ANY RECORDS IN nor_surfaces
        DELETE RECORDS IN nor_surfaces
    ENDIF
    IF TELL ANY RECORDS IN sec_features
        DELETE RECORDS IN sec_features
    ENDIF
ENDIF
CLEAR
WINDOW NEW WELCOME SIZE 37,92 AT 2,2
POSITION 1,1
MESSAGE "TO LOAD THE DATA FOR THE PART FOR WHICH PROCESS PLANS ARE TO BR
WRITTEN"
LABEL again
POSITION 3,1
MESSAGE"*****************************************************************"
POSITION 4,1
MESSAGE"PLEASE ENTER THE PART FOR WHICH PROCESS PLAN IS REQUIRE ?  *"
POSITION 5,1
```

```
MESSAGE"*****************************************************************"
POSITION 6,1
MESSAGE"FOR PART 'X4104181' JUST TYPE ---------------> p14              *"
POSITION 7,1
MESSAGE"FOR PART 'X4102201' JUST TYPE ---------------> p16              *"
POSITION 8,1
MESSAGE"FOR PART 'K4053852' JUST TYPE ---------------> p17              *"
POSITION 9,1
MESSAGE"FOR PART 'K4062662' JUST TYPE ---------------> p18              *"
POSITION 10,1
MESSAGE"FOR PART 'K4029050' JUST TYPE ---------------> p19              *"
POSITION 11,1
MESSAGE"FOR PART 'E5040061' JUST TYPE ---------------> p20              *"
POSITION 12,1
MESSAGE"FOR PART 'E9625.3321' JUST TYPE -------------> p21              *"
POSITION 13,1
MESSAGE"FOR PART 'D5572195' JUST TYPE ---------------> p22              *"
POSITION 14,1
MESSAGE"FOR PART 'Y4100566' JUST TYPE ---------------> p23              *"
POSITION 15,1
MESSAGE"FOR PART 'E5001058' JUST TYPE ---------------> p24              *"
POSITION 16,1
MESSAGE"FOR PART 'E5000033' JUST TYPE ---------------> p25              *"
POSITION 17,1
MESSAGE"FOR PART 'M2936444' JUST TYPE --------------> p26               *"
POSITION 18,1
MESSAGE"FOR PART 'X4045777' JUST TYPE --------------> p27               *"
POSITION 19,1
MESSAGE"FOR PART 'E5572065' JUST TYPE --------------> p28               *"
POSITION 20,1
MESSAGE"FOR PART 'Composite Component' JUST TYPE ----> p_x              *"
POSITION 21,1
MESSAGE"***********************************************************"
POSITION 22,1
MESSAGE "PART ?  "
READ tex1
IF tex1 = BLANK
    POSITION 21,4
    MESSAGE "YOU HAVE NOT ENTERED THE PART"
    GOTO again
ELSE
    CLEAR
    POSITION 10,20
    MESSAGE"PLEASE WAIT ... "
    POSITION 12,12
    MESSAGE"LOADING THE RECORDS FOR COMPONENT "tex1"  "
ENDIF
IF (tex1 = p14)
    DO p14_records
ELSE
    IF (tex1 = p16)
        DO p16_records
    ELSE
        IF (tex1 = p17)
            DO p17_records
        ELSE
```

```
          IF (tex1 = p18)
               DO p18_records
          ELSE
               IF (tex1 = p19)
                    DO p19_records
               ELSE
                    IF (tex1 = p20)
                         DO p20_records
                    ELSE
                         IF (tex1 = p21)
                              DO p21_records
                         ELSE
                              IF (tex1 = p22)
                                   DO p22_records
                              ELSE
                                   IF (tex1 = p23)
                                        DO p23_records
                                   ELSE
                                        IF (tex1 = p24)
                                             DO p24_records
                                        ELSE
                                             IF (tex1 = p25)
                                                  DO p25_records
                                             ELSE
                                                  IF (tex1 = p26)
                                                       DO p26_records
                                                  ELSE
                                                       IF (tex1 = p27)
                                                            DO p27_records
                                                       ELSE
                                                            IF (tex1 = p28)
                                                                 DO p28_records
                                                            ELSE
                                                                 IF (tex1 = p_x)
                                                                      DO p_x_records
                                                                 ENDIF
                                                            ENDIF
                                                       ENDIF
                                                  ENDIF
                                             ENDIF
                                        ENDIF
                                   ENDIF
                              ENDIF
                         ENDIF
                    ENDIF
               ENDIF
          ENDIF
     ENDIF
ENDIF
ENDIF
CLEAR
WINDOW MAIN_MENU
DELETE WINDOW WELCOME
```

E.8 PROCEDURE FILE 'do_test'

```
*
* DO GENERIS FILE 'do_test'
*
***************************************************************
* PROGRAM TO TEST WHETHER ANY PART CAN BE PLANNED OR NOT
***************************************************************
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL int0 INTEGER 1
CREATE LOCAL int1 INTEGER
CREATE LOCAL int2 INTEGER
CREATE LOCAL num0 INTEGER
CREATE LOCAL num1 INTEGER
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME drilling
CREATE LOCAL nam3 NAME
CREATE LOCAL nam4 NAME
CREATE LOCAL tex0 TEXT
*
WINDOW NEW TESTING SIZE 37,92 AT 2,2
CLEAR
POSITION int0,32
MESSAGE"TESTING THE COMPONENT"
POSITION int0+1,32
MESSAGE"*********************"
IF TELL ANY RECORDS IN extra_features
    DELETE RECORDS IN extra_features
ENDIF
LET int0 = 5
FETCH NEW RECORDS IN operations
LET fetchno = $FETCH
LET int2 = $COUNT
POSITION int0,10
MESSAGE"HCAPP SYSTEM HAS BEEN DESIGNED FOR THE FOLLOWING PRIMARY
FEATURES "
LET int0 = int0 + 1
POSITION int0,10
MESSAGE"*****************************************************************"
LET int0 = int0 + 3
LET int1 = 0
WHILE int2 > 0
    LET int1 = int1 + 1
    LET nam1  = FETCH VALUE (fetchno,int1,1)
    IF nam1 = nam2

    ELSE
        POSITION int0,24
        MESSAGE""^nam1 " "
        LET int0 = int0 + 1
    ENDIF
    LET nam2 = nam1
    LET int2 = int2 - 1
ENDWHILE
```

```
DELETE FETCH fetchno
POSITION int0+1,6
MESSAGE"ANOTHER LIMITATION REGARDING THE CONNECTIVITY ASPECT OF THE
COMPONENT"
POSITION int0+2,6
MESSAGE"IS THAT THE SYSTEM ONLY RECOGNIZES THE FEATURES WHOSE EADS
ARE"
POSITION int0+3,6
MESSAGE"PARALLEL TO THE FREE SURFACES"
POSITION int0+5,2
MESSAGE"SYSTEM IS NOW CHECKING WHETHER PROCESS PLANS CAN BE WRITTEN
FOR THE COMPONENT"
POSITION int0+8,22
MESSAGE"FOR FURTHERT PROCESSING ENTER <RETURN>"
READ tex0
IF TELL ANY RECORDS IN fea_list
    CLEAR SIZE 30,92 AT 4,2
    LET int0 = 5
    POSITION int0+2,24
    MESSAGE"PLEASE WAIT ..."
    FETCH NEW part 'has featurecode' featurecode 'has feature' feature BY 'has feature' feature
    LET fetchno = $FETCH
    LET int2 = $COUNT
    LET int1 = 0
    LET nam2 = drilling
    WHILE int2 > 0
        LET int1 = int1 + 1
        LET nam1 = FETCH VALUE (fetchno,int1,2)
        IF nam1 = BLANK
            LET nam1 = nam3
        ELSE
            LET nam3 = nam1
        ENDIF
        IF nam1 = free_surface

        ELSE
            IF nam1 = nam2

            ELSE
                IF TELL ANY RECORDS IN TABLE operations FOR ^nam1

                ELSE
                    POSITION int0,20
                    MESSAGE"FEATURE "^nam1" CANNOT BE PROCESSED"
                    fact in extra_features
                    ^nam1

                    1

                ENDIF
            ENDIF
        ENDIF
        LET nam2 = nam1
        LET int0 = int0 + 1
        LET int2 = int2 - 1
    ENDWHILE
```

```
DELETE FETCH fetchno
IF TELL ANY RECORDS IN extra_features
    POSITION int0,5
    MESSAGE"COMPONENT CANNOT BE PROCESSED COMPLETELY O.K. FOR
    FURTHER PROCESSING ENTER <RETURN>"
    READ tex0
ELSE
    POSITION int0,10
    MESSAGE"PART CAN BE PROCESSED O.K. FOR FURTHER PROCESSING ENTER
    <RETURN>"
    READ tex0
    GOTO fins
ENDIF
IF TELL ANY RECORDS IN extra_features
    FETCH NEW RECORDS IN extra_features
    LET fetchno = $FETCH
    LET int1 = $COUNT
    LET num0 = 0
    WHILE int1 > 0
        LET num0 = num0 + 1
        FETCH RECORD num0
        LET nam1 = FETCH VALUE (fetchno,num0,1)
        LABEL AGAIN
        CLEAR SIZE 30,92 AT 5,2
        LET int0 = 5
        POSITION int0,10
        MESSAGE"THE FEATURE "^nam1" CAN BE PROCESSED IF IT HAS GOT THE
        SOLUTION"
        LET int0 = int0 + 1
        POSITION int0,10
        MESSAGE"AMONG THE EXISTING OPERATIONS"
        FETCH feature 'has mc_method' operation BY operation 'has mc_method'
        LET num1 = 0
        LET nam2 = feature
        WHILE num1 < $COUNT
            LET num1 = num1 + 1
            FETCH RECORD num1
            LET nam0 = FIELD 2
            IF nam0 = BLANK
                LET nam0 = nam3
            ELSE
                LET nam3 = nam0
            ENDIF
            IF nam0 = nam2

            ELSE
                LET int0 = int0 + 1
                POSITION int0,24
                MESSAGE""^nam0""
            ENDIF
            LET nam2 = nam0
        ENDWHILE
        POSITION int0+2,20
        MESSAGE"PLEASE ENTER ONE"
        READ nam0
        IF nam0 = BLANK
```

```
            ELSE
                IF TELL ANY RECORDS IN extra_features FOR ^nam1
                    DELETE RECORDS IN extra_features FOR ^nam1
                ENDIF
                IF TELL ANY RECORDS IN operations FOR ^nam0
                    FACT IN operations
                    ^nam1
                    ^nam0

                ELSE
                    POSITION int0+4,20
                    MESSAGE"YOU HAVE NOT ENTERED THE OPERATION CORRECTLY
                    OK, <RETURN> FOR FURTHE PROCESSING"
                    READ tex0
                ENDIF
                POSITION int0+5,20
                MESSAGE"MORE SOLUTIONS TO BE ENTERED FOR FEATURE "^nam1"
                y/YES"
                READ tex0
                IF tex0 = y
                    GOTO AGAIN
                ENDIF
            ENDIF
            LET int1 = int1 - 1
        ENDWHILE
        IF TELL ANY RECORDS IN extra_features
            CLEAR SIZE 30,92 AT 5,2
            LET int0 = 5
            POSITION int0,6
            MESSAGE"PART CANNOT BE PROCESSED BECAUSE FOLLOWING FEATURES
            CANNOT BE PROCESSED"
            FETCH RECORDS IN extra_features
            LET num0 = 0
            WHILE num0 < $COUNT
                LET num0 = num0 + 1
                FETCH RECORD num0
                LET nam0 = FIELD 1
                LET int0 = int0 + 2
                POSITION int0,20
                MESSAGE""^nam0""
            ENDWHILE
            POSITION int0+2,15
            MESSAGE"PRESS <RETURN> FOR FURTHER PROCESSING"
            READ tex0
        ELSE
            CLEAR SIZE 30,92 AT 5,2
            LET int0 = 5
            POSITION int0,10
            MESSAGE"PART CAN NOW BE PROCESSED, PRESS <RETURN> FOR FURTHER
            PROCESSING"
            READ tex0
        ENDIF
    ENDIF
ELSE
    CLEAR
    MESSAGE"RECORDS IN TABLE fea_list DO NOT EXIST"
ENDIF
LABEL fins
WINDOW MAIN_MENU
DELETE WINDOW TESTING
```

E.9 PROCEDURE FILE 'do_main'

```
*
* DO GENERIS FILE 'do_main'
*
* **********************************************************
* MAIN FILE TO GENERATE PROCESS PLAN
* **********************************************************
*
WINDOW NEW GENERATIVE PROCESS PLANNING SIZE 37,92 AT 2,2
POSITION 2,33
MESSAGE"GENERATING PROCESS PLAN "
DO do_potentially1
DO do_results
DO do_result1
DO do_appdirec
DO do_AD
DO do_part_process
DO do_plan
DO do_plan1
CLEAR
WINDOW MAIN_MENU
DELETE WINDOW GENERATIVE PROCESS PLANNING
```

E.10 PROCEDURE FILE 'do_potentially1'

```
*
* DO GENERIS FILE 'do_potentially1'
*
* (TO CREATE RECORDS IN TABLE potentially1)
*
*****************************************************************
* TO ASSIGN THE OPERATIONS TO THE PART
*****************************************************************
CREATE LOCAL fetchno       INTEGER
CREATE LOCAL fetchno1      INTEGER
CREATE LOCAL int0          INTEGER 0
CREATE LOCAL int1          INTEGER 0
CREATE LOCAL int2          INTEGER 0
CREATE LOCAL int3          INTEGER 0
CREATE LOCAL num           INTEGER
CREATE LOCAL dec1          DECIMAL
CREATE LOCAL dec2          DECIMAL
CREATE LOCAL big           INTEGER
CREATE LOCAL nam0          NAME
CREATE LOCAL nam1          NAME
CREATE LOCAL nam2          NAME
CREATE LOCAL nam3          NAME
CREATE LOCAL nam4          NAME
CREATE LOCAL nam5            NAME
CREATE LOCAL tex             EXT
CLEAR SIZE 8,80 AT 8,2
POSITION 8,26
MESSAGE"ASSIGNING PROCESSES TO THE FEATURES"
POSITION 10,36
MESSAGE"PLEASE WAIT ... "
IF TELL ANY RECORDS IN potentially
    DELETE RECORDS IN potentially
ENDIF
IF TELL ANY RECORDS IN potentially1
    DELETE RECORDS IN potentially1
ENDIF
DISABLE ALL
ENABLE oper
FETCH NEW featurecode has feature1 feature potential1 operation
LET fetchno = $FETCH
LET int0 = $COUNT
LET int1 = 0
WHILE int0 > 0
    LET int1 = int1 + 1
    LET nam0  = FETCH VALUE (fetchno,int1,1)
    LET nam1  = FETCH VALUE (fetchno,int1,2)
    LET nam2  = FETCH VALUE (fetchno,int1,3)
    IF nam0 = BLANK
        LET nam0 = nam3
    ELSE
        LET nam3 = nam0
    ENDIF
    FACT IN potentially
    ^nam0
    ^nam1
    ^nam2

    LET int0 = int0 - 1
ENDWHILE
DISABLE ALL
LABEL rubin
LET nam0  = BLANK
LET nam4 = BLANK
LET nam5 = BLANK
FETCH NEW COUNT featurecode has feature1 feature potential1 operation by operation potential1
LET fetchno = $FETCH
LET int0 = $COUNT
LET int1 = 0
LET big = 0
WHILE int0 > 0
    LET int1 = int1 + 1
    LET int2  = FETCH VALUE (fetchno,int1,3)
    IF int2 = BLANK
        LET int2 = int3
    ELSE
        LET int3 = int2
    ENDIF
    IF(big < int2)
        LET big = int2
        LET nam1  = FETCH VALUE (fetchno,int1,2)
    ENDIF
    LET int0 = int0 - 1
ENDWHILE
DELETE FETCH fetchno
FETCH NEW COUNT featurecode has feature1 feature potential1 operation by operation potential1
LET fetchno = $FETCH
LET int0 = $COUNT
LET int1 = 0
WHILE int0 > 0
    LET int1 = int1 + 1
    LET int2 = FETCH VALUE (fetchno,int1,3)
    LET nam3  = FETCH VALUE (fetchno,int1,2)
    IF int2 = BLANK
        LET int2 = int3
    ELSE
        LET int3 = int2
    ENDIF
    IF int2 = big
        IF nam0 = BLANK
            LET nam0 = nam3
        ELSE
            IF nam4 = BLANK
                LET nam4 = nam3
            ELSE
                IF nam5 = BLANK
                    LET nam5 = nam3
                ENDIF
            ENDIF
        ENDIF
    ENDIF
    LET int0 = int0 - 1
ENDWHILE
IF nam5 = BLANK
    IF nam4 = BLANK

    ELSE
        POSITION 15,5
        MESSAGE"OPERATIONS "^nam0" AND "^nam4" HAVE EQUAL WEIGHTAGE"
        POSITION 16,5
        MESSAGE"WHICH PROCESS WOULD YOU LIKE TO SELECT? "
        POSITION 18,5
        MESSAGE"PLEASE SELECT 1) ----> FOR "^nam0" "
        POSITION 19,5
        MESSAGE"PLEASE SELECT 2) ----> FOR "^nam4" "
        POSITION 20,5
        READ int2
        IF int2 = 2
            LET nam0 = nam4
        ENDIF
    ENDIF
ELSE
    IF nam4 = BLANK

    ELSE
        POSITION 15,5
```

```
            MESSAGE"OPERATIONS "^nam0", "^nam4" and "^nam5" HAVE EQUAL WEIGHTAGE
            "
            POSITION 16,5
            MESSAGE"WHICH PROCESS WOULD YOU LIKE TO SELECT? "
            POSITION 18,5
            MESSAGE"PLEASE SELECT 1) ----> for "^nam0" "
            POSITION 19,5
            MESSAGE"PLEASE SELECT 2) ----> for "^nam4" "
            POSITION 20,5
            MESSAGE"PLEASE SELECT 3) ----> for "^nam5" "
            POSITION 21,5
            READ int2
            IF int2 = 2
                LET nam0 = nam4
            ELSE
                IF int2 = 3
                    LET nam0 = nam5
                ENDIF
            ENDIF
        ENDIF
    ENDIF
    CLEAR SIZE 40,80 AT 14,2
    DELETE FETCH fetchno
    IF nam0 = BLANK
        LET nam0 = nam1
    ENDIF
    FETCH NEW featurecode has feature1 feature potential1 operation FOR ^nam0
    LET fetchno = $FETCH
    LET int0 = $COUNT
    LET int1 = 0
    WHILE int0 > 0
        LET int1 = int1 + 1
        LET nam1 = FETCH VALUE (fetchno,int1,1)
        LET nam2 = FETCH VALUE (fetchno,int1,3)
        DO input_potentially1
        DELETE RECORDS IN potentially FOR ^nam1
        LET int0 = int0 - 1
    ENDWHILE
    DELETE FETCH fetchno
    IF TELL ANY RECORDS IN potentially
        GOTO rubin
    ENDIF
    IF TELL ANY featurecode has sec_feature thread
        FETCH RECORDS IN sec_features FOR thread
        LET num = 0
        WHILE num < $COUNT
            LET num = num + 1
            FETCH RECORD num
            LET nam1 = FIELD 1
            LET nam2 = thread
            LET nam0 = taping
            DO input_potentially1
        ENDWHILE
    ENDIF
    IF TELL ANY featurecode has sec_feature spotface
        FETCH RECORDS IN sec_features FOR spotface
        LET num = 0
        WHILE num < $COUNT
            LET num = num + 1
            FETCH RECORD num
            LET nam1 = FIELD 1
            LET nam2 = spotface
            LET nam0 = spotfacing
            DO input_potentially1
        ENDWHILE
    ENDIF
    IF TELL ANY featurecode has sec_feature countersink
        FETCH RECORDS IN sec_features FOR countersink
        LET num = 0
        WHILE num < $COUNT
            LET num = num + 1
            FETCH RECORD num
            LET nam1 = FIELD 1
            LET nam2 = countersink
            LET nam0 = countersinking
            DO input_potentially1
        ENDWHILE
    ENDIF
    IF TELL ANY featurecode has sec_feature counterbore
        FETCH RECORDS IN sec_features FOR counterbore
        LET num = 0
        WHILE num < $COUNT
            LET num = num + 1
            FETCH RECORD num
            LET nam1 = FIELD 1
            LET nam2 = counterbore
            LET nam0 = counterboring
            DO input_potentially1
        ENDWHILE
    ENDIF
    IF TELL ANY featurecode has sec_feature screw
        FETCH RECORDS IN sec_features FOR screw
        LET num = 0
        WHILE num < $COUNT
            LET num = num + 1
            FETCH RECORD num
            LET nam1 = FIELD 1
            LET nam2 = screw
            LET nam0 = screw_cutting
            DO input_potentially1
        ENDWHILE
    ENDIF
    IF TELL ANY featurecode has sec_feature chamfer
        FETCH RECORDS IN sec_features FOR chamfer
        LET num = 0
        WHILE num < $COUNT
            LET num = num + 1
            FETCH RECORD num
            LET nam1 = FIELD 1
            LET nam2 = chamfer
            LET nam0 = chamfering
            DO input_potentially1
        ENDWHILE
    ENDIF
    IF TELL ANY featurecode has sec_feature axial_thread
        FETCH RECORDS IN sec_features FOR axial_thread
        LET num = 0
        WHILE num < $COUNT
            LET num = num + 1
            FETCH RECORD num
            LET nam1 = FIELD 1
            LET nam2 = axial_thread
            LET nam0 = hor_taping
            DO input_potentially1
        ENDWHILE
    ENDIF
    IF TELL ANY RECORDS IN fea_list FOR round_hole
        LET nam2 = round_hole
        DO do_hole_operation
    ENDIF
    IF TELL ANY RECORDS IN fea_list FOR round_pocket
        LET nam2 = round_pocket
        DO do_hole_operation
    ENDIF
    IF TELL ANY RECORDS IN fea_list FOR axial_round_hole
        LET nam2 = axial_round_hole
        DO do_hole_operation
    ENDIF
    IF TELL ANY RECORDS IN fea_list FOR axial_round_pocket
        LET nam2 = axial_round_pocket
        DO do_hole_operation
    ENDIF
    IF TELL ANY RECORDS IN fea_list FOR surface
        LET nam2 = surface
        DO do_surface_operation
    ENDIF
    IF TELL ANY RECORDS IN fea_list FOR rectangular_thr_slot
        LET nam2 = rectangular_thr_slot
        DO do_surface_operation
    ENDIF
    IF TELL ANY RECORDS IN fea_list FOR rectangular_step
        LET nam2 = rectangular_step
        DO do_surface_operation
    ENDIF
    IF TELL ANY RECORDS IN fea_list FOR triangular_step
        LET nam2 = triangular_step
        DO do_surface_operation
    ENDIF
    FETCH NEW RECORDS IN fea_list
    LET fetchno = $FETCH
    LET int0 = $COUNT
    LET int1 = 0
    WHILE int0 > 0
        LET int1 = int1 + 1
        LET nam0  = FETCH VALUE (fetchno,int1,3)
    * nam0 is feature
        IF nam0 = free_surface
```

```
ELSE
    IF nam0 = triangular_step

    ELSE
        IF nam0 = rectangular_step

        ELSE
            IF nam0 = contoured_step

            ELSE
                IF nam0 = surface

                ELSE
                    IF nam0 = rectangular_thr_slot

                    ELSE
                        IF nam0 = axial_round_hole

                        ELSE
                            IF nam0 = axial_round_pocket

                            ELSE
                                IF nam0 = round_pocket

                                ELSE
                                    IF nam0 = round_hole

                                    ELSE
                                        IF nam0 = round_boss
                                            POSITION 12,5
                                            MESSAGE"LOGIC FOR ROUND BOSS
                                            FEATUER HAS NOT BEEN ENTERED"
                                            POSITION 13,5
                                            MESSAGE"SORRY! THIS FEATURE
                                            CANNOT BE PROCESSED"
                                            HOLD 5
                                        ELSE
                                            POSITION 12,5
                                            MESSAGE"DOES THE FEATURE "^nam0"
                                            FOLLOWS THE SAME PROCESSES AS THE
                                            SURFACE"
                                            POSITION 13,5
                                            MESSAGE"FEATURE FOR ITS SURFACE
                                            FINISH? y FOR YES <RETURN>
                                            OTHERWISE"
                                            READ tex
                                            IF tex = y
                                                LET nam2 = nam0
                                                DO do_surface_operation
                                            ENDIF
                                        ENDIF
                                    ENDIF
                                ENDIF
                            ENDIF
                        ENDIF
                    ENDIF
                    ENDIF
                ENDIF
            ENDIF
        ENDIF
    ENDIF
    LET int0 = int0 - 1
ENDWHILE
DISPLAY REPORT USING potentially1_form
GO
```

E.11 PROCEDURE FILE 'input_potentially1'

```
*
* DO GENERIS FILE 'input_potentially1'
*
*********************************************
* TO CREATE RECORDS IN TABLE potentially1
*********************************************
FACT IN potentially1
^nam1
^nam2
^nam0
```

E.12 PROCEDURE FILE 'do_hole_operation'

```
*
* DO GENERIS FILE 'do_hole_operation'
*
****************************************************
* TO ASSIGN OPERATIONS FOR SURFACE FINISH
****************************************************
* nam2 IS FEATURE
FETCH NEW RECORDS IN fea_list FOR ^nam2
LET int1 = 0
LET fetchno = $FETCH
LET int0 = $COUNT
WHILE int0 > 0
    LET int1 = int1 + 1
    LET nam1  = FETCH VALUE (fetchno,int1,2)
*   nam1 is featurecode
    IF TELL ANY RECORDS IN fea_data FOR ^nam1
        FETCH RECORDS IN fea_data FOR ^nam1
        FETCH RECORD 1
        LET dec2 = FIELD 17
        LET dec1 = FIELD 28
        IF dec2 = BLANK
            LET dec2 = 8
        ENDIF
        IF dec1 = BLANK
            LET dec1 = 8
        ENDIF
        IF(dec2 >= 50)
            LET nam0 = boring
            DO input_potentially1
            IF TELL ANY ^nam1 with_feature thread with_operation taping
```

```
                DELETE RECORDS IN potentially1 FOR ^nam1 with_feature thread
                with_operation taping
                FACT IN potentially1
                ^nam1
                screw
                screw_cutting

            ELSE
                IF TELL ANY ^nam1 with_feature axial_thread with_operation hor_taping
                    DELETE RECORDS IN potentially1 FOR ^nam1 with_feature axial_thread
                    with_operation hor_taping

                    FACT IN potentially1
                    ^nam1
                    screw
                    screw_cutting

                ENDIF
            ENDIF
            IF (dec1 < 0.4 )
                IF (dec1 >= 0.1)
                    LET nam0 = grinding
                    DO input_potentially1
                ENDIF
            ENDIF
            IF(dec1 <0.1)
                IF (dec1 >= 0.05)
                    LET nam0 = grinding
                    DO input_potentially1
                    LET nam0 = honing
                    DO input_potentially1
                ENDIF
            ENDIF
        ELSE
            IF (dec1 >= 0.8)
                IF(dec1 < 1.6 )
                    IF nam2 = axial_round_pocket
                        LET nam0 = hor_reaming
                        DO input_potentially1
                    ELSE
                        IF nam2 = axial_round_hole
                            LET nam0 = hor_reaming
                            DO input_potentially1
                        ELSE
                            LET nam0 = reaming
                            DO input_potentially1
                        ENDIF
                    ENDIF
                ENDIF
            ELSE
                IF (dec1 < 0.8 )
                    IF (dec1 >= 0.1)
                        IF nam2 = axial_round_pocket
                            LET nam0 = hor_reaming
                            DO input_potentially1
                        ELSE
                    IF nam2 = axial_round_hole
                        LET nam0 = hor_reaming
                        DO input_potentially1
                    ELSE
                        LET nam0 = reaming
                        DO input_potentially1
                    ENDIF
                ENDIF
                LET nam0 = grinding
                DO input_potentially1
            ENDIF
        ENDIF
    ENDIF
    IF(dec1 < 0.1)
        IF (dec1 >= 0.05)
            IF nam2 = axial_round_pocket
                LET nam0 = hor_reaming
                DO input_potentially1
            ELSE
                IF nam2 = axial_round_hole
                    LET nam0 = hor_reaming
                    DO input_potentially1
                ELSE
                    LET nam0 = reaming
                    DO input_potentially1
                ENDIF
            ENDIF
            LET nam0 = grinding
            DO input_potentially1
            LET nam0 = honing
            DO input_potentially1
        ENDIF
    ENDIF
ENDIF
ENDIF
IF TELL ANY RECORDS IN fea_data FOR ^nam1

ELSE
    IF TELL ANY RECORDS IN surfaces FOR ^nam1
        FETCH RECORDS IN surfaces FOR ^nam1
        LET num = 0
        WHILE num < $COUNT
            LET num = num + 1
            FETCH RECORD num
            LET dec1 = FIELD 4
            IF dec1 = BLAN
                LET dec1 = 8
            ENDIF
            IF (dec1 >= 0.8)
                IF(dec1 < 1.6 )
                    IF nam2 = axial_round_pocket
                        LET nam0 = hor_reaming
                        DO input_potentially1
                    ELSE
                        IF nam2 = axial_round_hole
                            LET nam0 = hor_reaming
```

```
                                DO input_potentially1
                            ELSE
                                LET nam0 = reaming
                                DO input_potentially1
                            ENDIF
                        ENDIF
                    ENDIF
                ELSE
                    IF (dec1 < 0.8 )
                        IF (dec1 >= 0.1)
                            IF nam2 = axial_round_pocket
                                LET nam0 = hor_reaming
                                DO input_potentially1
                            ELSE
                                IF nam2 = axial_round_hole
                                    LET nam0 = hor_reaming
                                    DO input_potentially1
                                ELSE
                                    LET nam0 = reaming
                                    DO input_potentially1
                                ENDIF
                            ENDIF
                            LET nam0 = grinding
                            DO input_potentially1
                        ENDIF
                    ELSE
                        IF (dec1 >= 0.05 )
                            IF (dec1 < 0.1 )
                                IF nam2 = axial_round_pocket
                                    LET nam0 = hor_reaming
                                    DO input_potentially1
                                ELSE
                                    IF nam2 = axial_round_hole
                                        LET nam0 = hor_reaming
                                        DO input_potentially1
                                    ELSE
                                        LET nam0 = reaming
                                        DO input_potentially1
                                    ENDIF
                                ENDIF
                                LET nam0 = grinding
                                DO input_potentially1
                                LET nam0 = honing
                                DO input_potentially1
                            ENDIF
                        ENDIF
                    ENDIF
                ENDIF
            ENDWHILE
        ENDIF
    ENDIF
    LET int0 = int0 - 1
ENDWHILE
DELETE FETCH fetchno
```

E.13 PROCEDURE FILE 'do_surface_operation'

```
*
* DO GENERIS FILE 'do_surface_operation'
*
* (TO ASSIGN OPERATIONS FOR SURFACE FINISH)
*
* nam2 IS FEATURE
* DISPLAY RECORDS IN fea_list FOR ^nam2
FETCH NEW RECORDS IN fea_list FOR ^nam2
LET int1 = 0
LET fetchno = $FETCH
LET int0 = $COUNT
WHILE int0 > 0
    LET int1 = int1 + 1
    LET nam1  = FETCH VALUE (fetchno,int1,2)
*   nam1 is featurecode
    IF TELL ANY RECORDS IN fea_data FOR ^nam1
        FETCH RECORDS IN fea_data FOR ^nam1
        FETCH RECORD 1
        LET dec1 = FIELD 28
        IF dec1 = BLANK
            LET dec1 = 8
        ENDIF
        IF (dec1 < 1.6 )
            IF (dec1 >= 0.1)
                LET nam0 = sur_grinding
                DO input_potentially1
            ENDIF
        ENDIF
        IF (dec1 < 0.1 )
            IF (dec1 >= 0.05)
                LET nam0 = sur_grinding
                DO input_potentially1
                LET nam0 = lapping
                DO input_potentially1
            ENDIF
        ENDIF
    ENDIF
    IF TELL ANY RECORDS IN fea_data FOR ^nam1

    ELSE
        IF TELL ANY RECORDS IN surfaces FOR ^nam1
            FETCH RECORDS IN surfaces FOR ^nam1
            LET num = 0
            WHILE num < $COUNT
                LET num = num + 1
                FETCH RECORD num
                LET dec1 = FIELD 4
                IF dec1 = BLANK
                    LET dec1 = 8
                ENDIF
                IF (dec1 < 1.6 )
                    IF (dec1 >= 0.1)
                        LET nam0 = sur_grinding
                        DO input_potentially1
                    ENDIF
                ELSE
                    IF (decl < 0.1 )
                        IF (decl >= 0.05)
                            LET nam0 = sur_grinding
                            DO input_potentially1
                            LET nam0 = lapping
                            DO input_potentially1
                        ENDIF
                    ENDIF
                ENDIF
            ENDWHILE
        ENDIF
    ENDIF
    LET int0 = int0 - 1
ENDWHILE
DELETE FETCH fetchno
```

E.14 PROCEDURE FILE 'do_results'

```
*
* DO GENERIS FILE 'do_results'
*
* (TO CREATE RECORDS IN TABLE 'results')
*
*********************************************************
* FILE TO CREATE POTENTIAL ACCESS DIRECTIONS (PADS)
*********************************************************
*
CREATE LOCAL fetchno        INTEGER
CREATE LOCAL fetchno1       INTEGER
CREATE LOCAL int0           INTEGER 0
CREATE LOCAL int1           INTEGER 0
CREATE LOCAL nam0           NAME
CREATE LOCAL nam1           NAME
CREATE LOCAL nam2           NAME
CREATE LOCAL nam3           NAME
*
CLEAR SIZE 8,80 AT 8,2
POSITION 8,25
MESSAGE"CALCULATING POTENTIAL ACCESS DIRECTIONS"
POSITION 10,38
MESSAGE"PLEASE WAIT ... "
IF TELL ANY RECORDS IN results
    DELETE RECORDS IN results
ENDIF
DISABLE ALL
ENABLE parent n_vector1
FETCH NEW featurecode mach_direction featurecode has direction nor_vector
LET fetchno = $FETCH
LET int1 = $COUNT
WHILE int1 > 0
    LET int0 = int0 + 1
    LET nam0  = FETCH VALUE (fetchno,int0,1)
    IF nam0 = BLANK
        LET nam0 = nam3
    ELSE
```

```
        LET nam3 =nam0
    ENDIF
    LET nam1  = FETCH VALUE (fetchno,int0,2)
    LET nam2  = FETCH VALUE (fetchno,int0,3)
    IF TELL ANY part has featurecode ^nam1 has feature free_surface
        FACT IN results
        ^nam0
        ^nam1
        ^nam2

    ENDIF
    LET int1 = int1 -1
ENDWHILE
DISABLE ALL
DISPLAY REPORT USING results_form
GO
```

E.15 PROCEDURE FILE 'do_result1'

```
*
* DO GENERIS FILE 'do_result1'
*
* TO CREATE RECORDS IN TABLE 'result1'
*
*******************************************************************
*  PROGRAM TO CALCULATE RESOURCE ELEMENTS REQUIRED ON EACH PAD
*******************************************************************
*
CREATE LOCAL fetchno        INTEGER
CREATE LOCAL fetchno1       INTEGER
CREATE LOCAL int0           INTEGER 0
CREATE LOCAL int1           INTEGER 0
CREATE LOCAL int2           INTEGER 0
CREATE LOCAL nam0           NAME
CREATE LOCAL nam1           NAME
CREATE LOCAL nam2           NAME
CREATE LOCAL nam3           NAME
CREATE LOCAL nam4           NAME
CREATE LOCAL nam5           NAME
CREATE LOCAL nam6           NAME
CREATE LOCAL nam7           NAME
*
CLEAR SIZE 8,80 AT 8,2
POSITION 8,20
MESSAGE"CALCULATING RESOURCE ELEMENTS REQUIRED ON EACH PAD"
POSITION 10,38
MESSAGE"PLEASE WAIT ... "
IF TELL ANY RECORDS IN result1
    DELETE RECORDS IN result1
ENDIF
IF TELL ANY RECORDS IN results
    DISABLE ALL
    FETCH NEW featurecode mach_direction featurecode has direction nor_vector with_feature
    feature with_operation operation by featurecode mach_direction
    LET fetchno = $FETCH
    LET int2 = $COUNT
```

```
        IF int2 = 0
            POSITION 12,10
            MESSAGE"RECORDS NOT FETCHED"
        ENDIF
        WHILE int2 > 0
            LET int1 = int1 + 1
            LET nam0  = FETCH VALUE (fetchno,int1,1)
            IF nam0 = BLANK
                LET nam0 = nam3
            ELSE
                LET nam3 = nam0
            ENDIF
            LET nam1  = FETCH VALUE (fetchno,int1,2)
            IF nam1 = BLANK
                LET nam1 = nam4
            ELSE
                LET nam4 = nam1
            ENDIF
            LET nam5  = FETCH VALUE (fetchno,int1,3)
            IF nam5 = BLANK
                LET nam5 = nam6
            ELSE
                LET nam6 = nam5
            ENDIF
            LET nam7  = FETCH VALUE (fetchno,int1,4)
            LET nam2  = FETCH VALUE (fetchno,int1,5)
            IF TELL ANY part has featurecode ^nam1 has feature free_surface
                FACT IN result1
                ^nam0
                ^nam1
                ^nam5
                ^nam7
                ^nam2

            ENDIF
            LET int2 = int2 - 1
        ENDWHILE
        DELETE FETCH fetchno
        DISPLAY REPORT USING result1_form
        GO

    ELSE
        MESSAGE "RECORDS IN TABLE 'results' DO NOT EXIST"
        HOLD 3
    ENDIF
```

E.16 PROCEDURE FILE 'do_AD'

```
*
* DO GENERIS FILE 'do_AD'
*
* TO FIND OUT THE AD
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL num INTEGER 1
CREATE LOCAL num1 INTEGER
CREATE LOCAL num2 INTEGER 0
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
*******************************************************
* PROCEED ONLY IF RECORDS EXIST IN TABLE op_table
*******************************************************
IF TELL ANY RECORDS IN op_table
    POSITION 8,24
    MESSAGE"CALCULATING THE NUMBER OF APPROACH DIRECTIONS"
    POSITION 10,38
    MESSAGE"PLEASE WAIT ... "
*
    FETCH NEW RECORDS IN parts
    LET fetchno1 = $FETCH
    LET nam2 = FETCH VALUE (fetchno1,1,1)
*   nam2 IS PART
    DELETE FETCH fetchno1
    FETCH NEW records IN op_table
    LET fetchno = $FETCH
    LET num1 = $COUNT
    WHILE num1 > 0
        LET num2 = num2 + 1
        LET nam0 = FETCH VALUE (fetchno,num2,4)
        IF nam1 = BLANK
            LET nam1 = nam0
        ENDIF
        IF nam0 = nam1

        ELSE
            LET num = num + 1
            LET nam1 = nam0
        ENDIF
        LET num1 = num1 - 1
    ENDWHILE
    POSITION 15,20
    MESSAGE "NUMBER OF APPROACH DIRECTIONS IS/ARE ---> "num " "
    POSITION 16,20
    MESSAGE "*********************************************"
    HOLD 5
    FACT IN parts
    ^nam2
    ^num

ELSE
    POSITION 6,20
    MESSAGE"THERE IS NO RECORD IN TABLE 'plan_table' ---> SKIPPING"
    HOLD 2
ENDIF
```

E.17 DO GENERIS FILE 'do_part_process'

```
*
**************************************************************
* TO FIND OUT THE OPERATIONS REQUIRED AT COMPONENT LEVEL
**************************************************************
* TO CREATE RECORDS IN TABLE 'part_process'
*
CREATE LOCAL fetchno        INTEGER
CREATE LOCAL fetchno1       INTEGER
CREATE LOCAL int0           INTEGER 0
CREATE LOCAL int1           INTEGER 0
CREATE LOCAL nam0           NAME screw
CREATE LOCAL nam1           NAME
CREATE LOCAL nam2           NAME
CREATE LOCAL nam3           NAME
*
CLEAR SIZE 40,80 AT 8,2
POSITION 8,19
MESSAGE"CALCULATING OPERATIONS REQUIRED AT COMPONENT LEVEL "
POSITION 10,35
MESSAGE"PLEASE WAIT ... "
IF TELL ANY RECORDS IN part_process
    DELETE RECORDS IN part_process
ENDIF
IF TELL ANY RECORDS IN potentially1
    FETCH RECORDS IN parts
    FETCH RECORD 1
    LET nam1 = FIELD 1
    FETCH NEW featurecode with_feature feature with_operation operation by operation
    with_operation
    LET fetchno = $FETCH
    LET int0  = $COUNT
    LET int1 = 0
    WHILE int0 > 0
        LET int1 = int1 + 1
        LET nam2 = FETCH VALUE (fetchno,int1,2)
*       nam2 is operation
        IF nam2 = BLANK
            LET nam2 = nam3
        ELSE
            LET nam3 = nam2
        ENDIF
        IF  nam0  = nam2

        ELSE
            FACT IN part_process
            ^nam1
            ^nam2

            LET nam0  = nam2
        ENDIF
        LET int0 = int0 - 1
    ENDWHILE
    DELETE FETCH fetchno
    DISPLAY REPORT USING part_process_form
    GO

ELSE
    CLEAR SIZE 40,80 AT 8,2
    POSITION 8,14
    MESSAGE"RECORDS IN TABLE 'op_table' DO NOT EXIST"
    HOLD 4
ENDIF
```

E.18 PROCEDURE FILE 'do_appdirec'

```
*
* DO GENERIS FILE 'do_appdirec'
*
* TO CREATE RECORDS IN TABLE 'op_table'
*
* PRE-REQUISIT RECORDS IN TABLE result1
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int3 INTEGER 0
CREATE LOCAL int4 INTEGER 0
CREATE LOCAL big INTEGER
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
CREATE LOCAL nam4 NAME
DISABLE ALL
CLEAR SIZE 40,80 at 8,2
POSITION 8,29
MESSAGE"CALCULATING APPROACH DIRECTIONS "
POSITION 10,37
MESSAGE"PLEASE WAIT ... "
IF TELL ANY RECORDS IN result1
    IF TELL ANY RECORDS IN op_table
        DELETE RECORDS IN op_table
    ENDIF
    LABEL begin
    LET int1 = 0
    LET big  = 0
    FETCH NEW COUNT featurecode 'opt direction' featurecode 'with nor_vector' nor_vector 'with
    feature' feature 'has operation' operation BY nor_vector 'with nor_vector'
    LET fetchno = $FETCH
    LET int0    = $COUNT
    WHILE int0 > 0
        LET int1 = int1 + 1
        LET int2 = FETCH VALUE (fetchno,int1,4)
        IF int2  = BLANK
            LET int2 = int4
        ELSE
            LET int4 = int2
        ENDIF
        IF(big < int2)
            LET big  = int2
            LET int3 = int1
            LET nam0 = FETCH VALUE (fetchno,int3,2)
* nam0 is nor_vector 'with nor_vector'
        ENDIF
        LET int0 = int0 - 1
    ENDWHILE
    DELETE FETCH fetchno
    FETCH NEW RECORDS IN result1 FOR 'with nor_vector' ^nam0
    LET fetchno = $FETCH
    LET int1 = 0
    LET int0 = $COUNT
    WHILE int0 > 0
        LET int1 = int1 + 1
        LET nam1 = FETCH VALUE (fetchno,int1,1)
        LET nam2 = FETCH VALUE (fetchno,int1,2)
        LET nam3 = FETCH VALUE (fetchno,int1,4)
        LET nam4 = FETCH VALUE (fetchno,int1,5)
*       nam1 is featurecode
*       nam2 is featurecode 'opt direction'
*       nam3 is feature 'with feature'
*       nam4 is operation 'has operation'
        IF TELL ANY RECORDS IN op_table FOR ^nam1 'having_feature' ^nam3 'has
        op_operation' ^nam4

        ELSE
            FACT IN op_table
            ^nam1
            ^nam3
            ^nam2
            ^nam0
            ^nam4

        ENDIF
        DELETE RECORDS IN result1 FOR ^nam1 FOR 'with feature' ^nam3 FOR 'has operation'
        ^nam4
        LET int0 = int0 - 1
    ENDWHILE
    DELETE FETCH fetchno
    IF TELL ANY RECORDS IN result1
        GOTO begin
    ENDIF
    DISPLAY REPORT USING op_table_form
    GO

ELSE
    HOLD 2
    MESSAGE" RECORDS IN TABLE result1 DO NOT EXIST"
ENDIF
```

E.19 PROCEDURE FILE 'do_plan'

```
*
* DO GENERIS FILE 'do_plan'
*
******************************************************
* TO CALCULATE FEATURE RELATIONSHIPS
******************************************************
*
* TO CREATE THE RECORDS IN TABLE 'plan_table'
*
* MAKE SURE THAT RECORDS EXIST IN TABLE 'op_table'
*
CREATE LOCAL num0 INTEGER
CREATE LOCAL num1 INTEGER
CREATE LOCAL num2 INTEGER
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL fetchno2 INTEGER
CREATE LOCAL fetchno3 INTEGER
CREATE LOCAL fetchno4 INTEGER
CREATE LOCAL fetchno5 INTEGER
CREATE LOCAL fetchno6 INTEGER
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int3 INTEGER 0
CREATE LOCAL int4 INTEGER 0
CREATE LOCAL int5 INTEGER 0
CREATE LOCAL int6 INTEGER 0
CREATE LOCAL int7 INTEGER 0
CREATE LOCAL int8 INTEGER 0
CREATE LOCAL big  INTEGER
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
CREATE LOCAL nam4 NAME
CREATE LOCAL nam5 NAME
CREATE LOCAL nam6 NAME
CREATE LOCAL nam7 NAME
*
CLEAR SIZE 40,80 AT 8,2
POSITION 8,28
MESSAGE"CALCULATING FEATURE RELATIONSHIPS "
POSITION 10,37
MESSAGE"PLEASE WAIT ... "
IF TELL ANY RECORDS IN plan_table
    DELETE RECORDS IN plan_table
ENDIF
IF TELL ANY RECORDS IN op_table
    DISABLE ALL
    ENABLE child
    LABEL begin
    LET int1 = 0
    LET big = 0
```

```
      FETCH NEW COUNT featurecode 'having_feature' feature 'has appr_direction' featurecode 'from
      nor_vector' nor_vector 'has op_operation' operation BY featurecode 'has appr_direction'
      LET fetchno = $FETCH
      LET int0 = $COUNT
      WHILE int0 > 0
           LET int1 = int1 + 1
           LET int2 = FETCH VALUE (fetchno,int1,3)
           IF int2  = BLANK
                LET int2 = int4
           ELSE
                LET int4 = int2
           ENDIF
           IF(big < int2)
                LET big = int2
                LET int3 = int1
                LET nam0  = FETCH VALUE (fetchno,int3,2)
                LET nam1  = FETCH VALUE (fetchno,int3,6)
* nam0 is 'has appr_direction' featurecode
* nam1 is operation 'has op_operation'
                LET nam3 = nam0
           ENDIF
           LET int0 = int0 - 1
      ENDWHILE
      DELETE FETCH fetchno
      LABEL RUBIN
      FETCH NEW featurecode 'has child' featurecode for ^nam3
      LET fetchno1 = $FETCH
      LET int6 = 0
      LET int5 = $COUNT
      WHILE int5 > 0
           LET int6 = int6 + 1
           LET nam2  = FETCH VALUE (fetchno1,int6,2)
*          nam2 is child featurecode
           FETCH NEW featurecode has appr_direction featurecode for ^nam2
           LET fetchno6 = $FETCH
           LET big = $COUNT
           IF big = 0

           ELSE
                LET nam6 = FETCH VALUE (fetchno6,1,2)
           ENDIF
           DELETE FETCH fetchno6
           IF nam0 = nam6
                FETCH NEW RECORDS IN op_table FOR ^nam2
                LET num0 = 0
                LET fetchno2 = $FETCH
                LET int7 = $COUNT
                WHILE int7 > 0
                     LET num0 = num0 + 1
                     LET nam4  = FETCH VALUE (fetchno2,num0,2)
                     LET nam1  = FETCH VALUE (fetchno2,num0,5)
*                    nam4 is feature
*                    nam1 is operation
                     LET nam7 = nam0
                     IF nam7 = s1
                          LET nam7 = PAD1
```

```
            ELSE
                IF nam7 = s2
                    LET nam7 = PAD2
                ELSE
                    IF nam7 = s3
                        LET nam7 = PAD3
                    ELSE
                        IF nam7 = s4
                            LET nam7 = PAD4
                        ELSE
                            IF nam7 = s5
                                LET nam7 = PAD5
                            ELSE
                                IF nam7 = s6
                                    LET nam7 = PAD6
                                ENDIF
                            ENDIF
                        ENDIF
                    ENDIF
                ENDIF
            ENDIF
            FACT IN plan_table
            ^nam2
            ^nam4
            ^nam3
            ^nam7
            ^nam1


            IF (int7 = 1)
                DELETE RECORDS IN op_table FOR ^nam2
            ENDIF
            LET int7  = int7 - 1
        ENDWHILE
        DELETE FETCH fetchno2
    ENDIF
    IF (int5 = 1)
        FETCH NEW featurecode 'has child' featurecode FOR ^nam3
        LET fetchno5  = $FETCH
        LET int8 = $COUNT
        LET num2 = 0
        WHILE int8 > 0
            LET num2 = num2 + 1
            LET nam2  = FETCH VALUE (fetchno5,num2,2)
            IF TELL ANY featurecode 'has child' featurecode FOR ^nam2
                LET nam3 = nam2
                GOTO RUBIN
            ENDIF
            LET int8 = int8 - 1
        ENDWHILE
        DELETE FETCH fetchno5
    ENDIF
    LET int5 = int5 - 1
ENDWHILE
DELETE FETCH fetchno1
IF TELL ANY records in op_table for 'has appr_direction' ^nam0
```

```
            GOTO RUBIN
        ENDIF
        IF TELL ANY records in op_table
            GOTO begin
        ENDIF
        DISPLAY REPORT USING plan_table_form
        GO
ELSE
        CLEAR SIZE 40,80 AT 8,2
        POSITION 10,15
        MESSAGE"THERE IS NO RECORD IN TABLE 'op_table' --> SKIPPING"
        HOLD 3
ENDIF
```

E.20 PROCEDURE FILE 'do_plan1'

```
*
* DO GENERIS FILE 'do_plan1'
*
* PROCEDURE FILE TO SEQUENCE THE OPERATIONS
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL fetchno2 INTEGER
CREATE LOCAL fetchno3INTEGER
CREATE LOCAL fetchno4 INTEGER
CREATE LOCAL fetchno5 INTEGER
CREATE LOCAL counter INTEGER 1
CREATE LOCAL HIF1 INTEGER
CREATE LOCAL HIF2 INTEGER
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int3 INTEGER 0
CREATE LOCAL int4 INTEGER 0
CREATE LOCAL int5 INTEGER 0
CREATE LOCAL int6 INTEGER
CREATE LOCAL int10 INTEGER 0
CREATE LOCAL int11 INTEGER 0
CREATE LOCAL int12 INTEGER 0
CREATE LOCAL int13 INTEGER 0
CREATE LOCAL int14 INTEGER
CREATE LOCAL big INTEGER
CREATE LOCAL num INTEGER
CREATE LOCAL Flg INTEGER
CREATE LOCAL Flg1 INTEGER
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
CREATE LOCAL nam4 NAME
CREATE LOCAL nam5 NAME
CREATE LOCAL nam6 NAME
CREATE LOCAL nam7 NAME
CREATE LOCAL nam8 NAME
CREATE LOCAL nam9 NAME
CREATE LOCAL nam10 NAME
CREATE LOCAL nam11 NAME
CREATE LOCAL nam12 NAME
CREATE LOCAL nam13 NAME
CREATE LOCAL nam14 NAME
CREATE LOCAL nam15 NAME
CREATE LOCAL nam16 NAME
CREATE LOCAL nam17 NAME
*
IF TELL ANY RECORDS IN plan_table
    CLEAR SIZE 40,80 AT 8,2
    POSITION 8,32
    MESSAGE"SEQUENCEING THE OPERATIONS "
    POSITION 10,38
    MESSAGE"PLEASE WAIT ... "
    FETCH RECORDS IN parts
    FETCH RECORD 1
    LET nam13 = FIELD 1
    unix hold screen rm -f plan 2>/dev/null
    WRITE REPORT USING xzz TO plan
    FOR ^nam13
    GO

    IF TELL ANY RECORDS IN plan1_table
        DELETE RECORDS IN plan1_table
    ENDIF
*****************************************************************************
* FIND OUT THE AD FROM WHICH MAXIMUM FEATURES CAN BE DONE
*****************************************************************************
    DISABLE ALL
    ENABLE child
    LABEL begin
    LET int1 = 0
    LET big = 0

    FETCH NEW COUNT featurecode containing feature parent featurecode 'from direction'
    nor_vector 'with operation' operation BY nor_vector 'from direction'
    LET fetchno = $FETCH
    LET int0 = $COUNT
    IF int0 = 0
        GOTO rubin11
    ENDIF
    WHILE int0 > 0
        LET int1 = int1 + 1
        LET int2  = FETCH VALUE (fetchno,int1,3)
        IF int2 = BLANK
            LET int2 = int4
        ELSE
            LET int4 = int2
        ENDIF
        IF(big < int2)
            LET big = int2
            LET int3 = int1
            LET nam1 = FETCH VALUE (fetchno,int3,2)
* nam1 is nor_vector 'from direction'
        ENDIF
        LET int0 = int0 - 1
    ENDWHILE
    DELETE FETCH fetchno
*****************************************************************************
* FETCH RECORDS FOR THE AD FROM WHICH MAXIMUM FEATURES CAN BE DONE IN
ORDER TO CALCULATE PARENT FEATURE
*****************************************************************************
    LABEL begin1
    FETCH NEW featurecode containing feature parent featurecode 'from direction' nor_vector 'with
    operation' operation FOR 'from direction' ^nam1
    LET fetchno = $FETCH
    LET int1 = 0
    LET int2 = $COUNT
    WHILE int2 > 0
        LET int1 = int1 + 1
        LET nam2 = FETCH VALUE (fetchno,int1,4)
*       nam2 is for parent featurecode
*****************************************************************************
* FETCH RECORDS FOR THE AD AND PARENT FEATURE ALREADY CALCULATED TO
FIND OUT THE CHILDREN FEATURES
*****************************************************************************
        LABEL AROG2

        FETCH NEW featurecode containing feature parent featurecode 'from direction' nor_vector
        'with operation' operation FOR 'from direction' ^nam1 FOR parent ^nam2
        LET fetchno1 = $FETCH
        LET int3 = $COUNT
        WHILE int3 > 0
            IF (int3 = 1)
                LET nam12 = FETCH VALUE (fetchno1,1,1)
                LET nam7 = FETCH VALUE (fetchno1,1,2)
                LET nam8 = FETCH VALUE (fetchno1,1,3)
                LET nam9 = FETCH VALUE (fetchno1,1,4)
                LET nam10 = FETCH VALUE (fetchno1,1,5)
                LET nam5 = nam10
*               nam5 is operation
                DELETE FETCH fetchno1
*****************************************************************************
* WRITE THE RECORDS IN THE TABLE plan1_table AFTER SEQUENCING
*****************************************************************************

                FACT IN plan1_table
                ^nam7
                ^nam8
                ^nam10
                ^nam12
                ^nam9


                WRITE REPORT USING xyz TO plan
                FOR ^nam12 containing ^nam9 parent ^nam8 'from direction' ^nam7 'with
                operation' ^nam10
                GO

                DELETE RECORDS IN plan_table FOR ^nam12 containing ^nam9 parent ^nam8
                'from direction' ^nam7 'with operation' ^nam10
    LET int2 = int2 - 1
    LET int3 = int3 - 1
ELSE
    DELETE FETCH fetchno1
    LET Flg = 0
    LET num = 0
    FETCH featurecode containing feature parent featurecode 'from direction'
    nor_vector 'with operation' operation for 'from direction' ^nam1 for parent ^nam2
    WHILE num < $COUNT
        LET num = num + 1
        FETCH RECORD num
        LET nam16 = FIELD 4
        IF nam16 = surface
            LET Flg = 1
        ENDIF
    ENDWHILE
    IF (nam5 = BLANK)

    ELSE
        IF Flg = 0

            IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 'with
            operation' ^nam5
                IF nam5 = hor_taping

                ELSE
                    IF nam5 = taping

                    ELSE
                        IF nam5 = reaming

                        ELSE
                            IF nam5 = spotfacing

                            ELSE
                                IF nam5 = counterboring

                                ELSE
                                    IF nam5 = countersinking

                                    ELSE
                                        IF nam5 = boring

                                        ELSE
                                            IF nam5 = grinding

                                            ELSE
                                                IF nam5 = sur_grinding

                                                ELSE
                                                    IF nam5 = honing

                                                    ELSE
                                                        IF nam5 = lapping

                                                        ELSE
```

```
                                                          IF      nam5      =
                                                          chamfering

                                                          ELSE
                                                             IF    nam5    =
                                                             screw_cutting

                                                             ELSE
                                                                GOTO rubin
                                                             ENDIF
                                                          ENDIF
                                                       ENDIF
                                                    ENDIF
                                                 ENDIF
                                              ENDIF
                                           ENDIF
                                        ENDIF
                                     ENDIF
                                  ENDIF
                               ENDIF
                            ENDIF
                         ENDIF
                      ENDIF
                   ENDIF
                ENDIF
**********************************************************************************
* CALCULATE OPERATION WHICH CAN DO MAXIMUM FEATURE HAVING SAME AD
AND SAME PARENT FEATURE
**********************************************************************************

                FETCH NEW featurecode containing feature parent featurecode 'from direction'
                nor_vector 'with operation' operation FOR 'from direction' ^nam1 FOR parent
                ^nam2 BY 'with operation' operation

                LET fetchno2 = $FETCH
                LET int4 = $COUNT
                LET nam5 = FETCH VALUE (fetchno2,1,2)
                LET nam15 = FETCH VALUE (fetchno2,1,2)
*               nam5 is operation 'with operation'
                LET int12 = 0
                WHILE int4 > 0
                   LET int12 = int12 + 1
                   LET nam12 = FETCH VALUE (fetchno2,int12,1)
                   IF nam12 = BLANK
                      LET nam12 = nam14
                   ELSE
                      LET nam14 = nam12
                   ENDIF
                   LET HIF1 = 0
                   IF TELL ANY ^nam13 'has featurecode' ^nam12 'has feature' surface
                      FETCH RECORDS IN potentially1 FOR ^nam12
                      FETCH RECORD 1
                      LET nam5 = FIELD 3
                      IF TELL ANY RECORDS IN plan_table FOR ^nam12 FOR 'with
                      operation' ^nam5
                         LET HIF1 = 1
```

```
            LET int4 = 1
        ENDIF
    ENDIF
    LET int4 = int4 - 1
ENDWHILE
DELETE FETCH fetchno2
IF(HIF1 = 1)

ELSE
    LET nam5 = nam15
ENDIF
*               nam5 is operation
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' drilling
    IF (nam5 = boring)
        LET nam5 = drilling
    ELSE
        IF (nam5 = taping)
            LET nam5 = drilling
        ELSE
            IF (nam5 = chamfering)
                LET nam5 = drilling
            ELSE
                IF (nam5 = counterboring)
                    LET nam5 = drilling
                ELSE
                    IF (nam5 = reaming)
                        LET nam5 = drilling
                    ELSE
                        IF (nam5 = countersinking)
                            LET nam5 = drilling
                        ELSE
                            IF (nam5 = spotfacing)
                                LET nam5 = drilling
                            ELSE
                                IF (nam5 = grinding)
                                    LET nam5 = drilling
                                ELSE
                                    IF (nam5 = honing)
                                        LET nam5 = drilling
                                    ENDIF
                                ENDIF
                            ENDIF
                        ENDIF
                    ENDIF
                ENDIF
            ENDIF
        ENDIF
    ENDIF
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' hor_drilling
    IF (nam5 = hor_taping)
        LET nam5 = hor_drilling
    ENDIF
ENDIF
```

```
                LABEL rubin

**************************************************************************
* FETCH FEATURES WHICH HAVE SAME AD, SAME PARENT AND REQUIRE SAME
OPERATIONS
**************************************************************************

                FETCH NEW featurecode containing feature parent featurecode 'from direction'
                nor_vector 'with operation' operation FOR 'from direction' ^nam1 FOR parent
                ^nam2 FOR 'with operation' ^nam5

                IF (HIF1 = 1)
                    FETCH NEW featurecode containing feature parent featurecode 'from direction'
                    nor_vector 'with operation' operation FOR ^nam12 FOR 'from direction' ^nam1
                    FOR parent ^nam2 FOR 'with operation' ^nam5
                ENDIF
                LET fetchno3 = $FETCH
                LET int5 = 0
                LET int6 = $COUNT
                WHILE int6 > 0
                    LET int5 = int5 + 1
                    LET nam12 = FETCH VALUE (fetchno3,int5,1)
                    LET nam7 = FETCH VALUE (fetchno3,int5,2)
                    LET nam8 = FETCH VALUE (fetchno3,int5,3)
                    LET nam9 = FETCH VALUE (fetchno3,int5,4)
                    LET nam10 = FETCH VALUE (fetchno3,int5,5)
                    LABEL shakeel

**************************************************************************
* WRITE THE RECORDS IN THE TABLE plan1_table AFTER SEQUENCING
**************************************************************************
                    FACT IN plan1_table
                    ^nam7
                    ^nam8
                    ^nam9
                    ^nam12
                    ^nam10

                    WRITE REPORT USING xyz TO plan
                    FOR ^nam12 containing ^nam10 parent ^nam8 'from direction' ^nam7 'with
                    operation' ^nam9
                    GO

                    IF ( HIF2 = 1)
                        DELETE RECORDS IN plan_table FOR ^nam12 containing ^nam10
                        parent ^nam8 'from direction' ^nam7 'with operation' ^nam9
                        IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam7 'with
                        operation' ^nam9
                            LET int6 = int6 + 1
                        ENDIF
                    ENDIF
                    IF(HIF1 = 1)
                        LET HIF1 = 0
                        DELETE RECORDS IN plan_table FOR ^nam12 'with operation' ^nam9
                    ELSE
                        IF (int6 = 1)
```

```
            DELETE RECORDS IN plan_table FOR 'with operation' ^nam5 FOR
            parent ^nam2
        ENDIF
    ENDIF
    LET int2 = int2 - 1
    LET int6 = int6 - 1
    LET int3 = int3 - 1
    LET HIF2 = 0
    IF TELL ANY featurecode 'has child' featurecode FOR ^nam12
        LET nam8 = nam12
        FETCH NEW featurecode 'has child' featurecode FOR ^nam12
        LET fetchno5 = $FETCH
        LET int13 = $COUNT
        LET int14 = 0
        WHILE int13 > 0
            LET int14 = int14 + 1
            LET nam12 = FETCH VALUE (fetchno5,int14,2)
            IF TELL ANY ^nam12 'with operation' ^nam9
                LET HIF2 = 1
                LET int13 = 1
                GOTO shakeel
            ENDIF
            LET int13 = int13 - 1
        ENDWHILE
    ENDIF
ENDWHILE
IF Flg1 = 1
    LET nam2 = nam17
ENDIF
LET Flg1 = 0
DELETE FETCH fetchno3
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR 'with
operation' ^nam9
    IF TELL ANY featurecode 'has child' featurecode FOR ^nam2
        LET num = 0
        FETCH featurecode 'has child' featurecode FOR ^nam2
        WHILE num < $COUNT
            LET num = num + 1
            FETCH RECORD num
            LET nam17 = FIELD 2
            IF TELL ANY featurecode 'has child' featurecode FOR ^nam17
                IF TELL ANY RECORDS IN plan_table FOR ^nam17 FOR 'from
                direction' ^nam1 FOR parent ^nam2 FOR 'with operation' ^nam5

                ELSE
                    LET Flg1 = 1
                    LET nam2  = nam17
                    LET nam17 = nam2
                ENDIF
            ENDIF
        ENDWHILE
    ENDIF
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' ^nam5
    GOTO rubin
```

```
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' drilling
    LET nam5 = drilling
    GOTO rubin
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' hor_drilling
    LET nam5 = hor_drilling
    GOTO rubin
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' reaming
    LET nam5 = reaming
    GOTO rubin
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' taping
    LET nam5 = taping
    GOTO rubin
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' boring
    LET nam5 = boring
    GOTO rubin
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' counterboring
    LET nam5 = counterboring
    GOTO rubin
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' countersinking
    LET nam5 = countersinking
    GOTO rubin
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' spotfacing
    LET nam5 = spotfacing
    GOTO rubin
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' screw_cutting
    LET nam5 = screw_cutting
    GOTO rubin
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' chamfering
    LET nam5 = chamfering
    GOTO rubin
ENDIF
IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
^nam2 FOR 'with operation' sur_grinding
    LET nam5 = sur_grinding
    GOTO rubin
ENDIF
```

```
                IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
                ^nam2 FOR 'with operation' grinding
                    LET nam5 = grinding
                    GOTO rubin
                ENDIF
                IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
                ^nam2 FOR 'with operation' honing
                    LET nam5 = honing
                    GOTO rubin
                ENDIF
                IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR
                parent^nam2 FOR 'with operation' lapping
                    LET nam5 = lapping
                    GOTO rubin
                ENDIF
                IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1 FOR parent
                ^nam2 FOR 'with operation' hor_taping
                    LET nam5 = hor_taping
                    GOTO rubin
                ENDIF
                IF (int3 > 0)
                    GOTO AROG2
                ENDIF
            ENDIF
        ENDWHILE
    ENDWHILE
    DELETE FETCH fetchno
    IF TELL ANY RECORDS IN plan_table FOR 'from direction' ^nam1
        GOTO begin1
    ENDIF
    IF TELL ANY RECORDS IN plan_table
        GOTO begin
    ENDIF
    DISPLAY FILE plan
ELSE
    CLEAR SIZE 40,80 at 8,2
    POSITION 8,20
    MESSAGE"THERE IS NO RECORD IN TABLE 'plan_table' --> SKIPPING"
    HOLD 3
ENDIF
LABEL rubin11
```

Appendix F

Table of Contents

F SOFTWARE PROGRAMS FOR THE IMPLEMENTATION OF CELL DESIGN

F.1 INTRODUCTION

The software programs developed for the implementation of cell design are listed in this appendix. The hierarchy of the designed programs is shown in figure F.1.

The directory in which the cell design software programs are developed is called 'rubin1'. After changing the home directory to the directory mentioned, the system starts by typing in 'generis' on the prompt. GENERIS Expert System opens its window for its use. The next command to be entered is 'do generisinit'. A user defined window called 'WELCOME' is invoked. The password will be asked to enter to get the main menu. The password is simply <return>. After entering the password, the main menu will be displayed in another user defined window called 'MAIN MENU'. Main menu facilitates the selection of different system modules. Title of the menu window is 'SELECTION'.

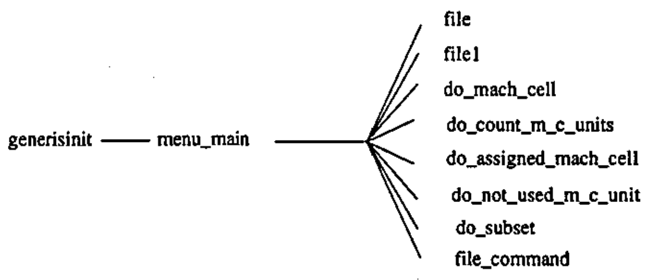


Figure F.1 Software programs hierarch in the implementation of cell design

F.2 PROCEDURE FILE 'generisinit'

```
*
* DO GENERIS FILE 'generisinit'
*
* START UP FILE FOR THE DEMONSTRATION
*
WINDOW NEW WELCOME SIZE 37,92 AT 2,2
POSITION 5,0
MESSAGE "\THE APPLICATION IS OPENED ONCE YOU HAVE INPUT THE PASSWORD \ \"
OPEN try
CLEAR
WINDOW NEW MAIN_MENU SIZE 37,92 AT 2,2
POSITION 9,20
MESSAGE "PLEASE MAKE SELECTION AS PER REQUIREMENTS \"
POSITION 12,2
MENU menu_main
WINDOW COMMAND
DELETE WINDOW ALL
CLOSE
```

F.3 PROCEDURE MENU FILE 'menu_main'

```
*
* GENERIS MENU FILE 'menu_main'
*
TITLE SELECTION
TO FIND OUT THE PROCESSING REQUIREMENTS OF THE PARTS IN TERMS OF MCU'S:do
file;
TO FIND OUT THE CANDIDATE MACHINES FOR THE PARTS:do file1;
TO FIND OUT THE MACHINES WHICH ARE CANDIDATE FOR MORE THAN ONE CELL:do
do_mach_cell;
TO ASSIGN THE BOTTLE-NECK MACHINES TO THE CELLS:do do_count_m_c_units;
TO FIND OUT THE ASSIGNED MACHINES TO THE CELLS:do do_assigned_mach_cell;
TO FIND OUT FOR WHICH PART'S MCU, THERE IS NOT ANY MACHINE AVAILABLE:do
do_not_used_m_c_unit;
TO FIND OUT WHETHER ANY CELL CAN BE MERGED INTO ANY OTHER CELL:do
do_subset;
TYPE ANY COMMAND:DO file_command;
END:menu return;
```

F.4 PROCEDURE FILE 'file'

```
*
* DO GENERIS FILE 'file'
*
ENABLE part_mcus
DISPLAY part m_c_unit 'need'
```

F.5 PROCEDURE FILE 'file1'

```
*
* DO GENERIS FILE 'file1'
*
ENABLE part_machine
DISPLAY part machine 'candidate'
```

F.6 PROCEDURE FILE 'do_mach_cell'

```
*
* DO GENERIS FILE 'do_mach_cell'
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int3 INTEGER 0
CREATE LOCAL nam3 NAME
CREATE LOCAL nam0 NAME
WINDOW NEW WELCOME SIZE 37,92 AT 2,2
POSITION 8,10
MESSAGE"FINDING OUT THE MACHINE WHICH IS CANDIDATE FOR MORE THAN ONE
CELL"
POSITION 10,30
MESSAGE"PLEASE WAIT ... "
```

```
IF TELL ANY RECORDS IN cell_machines
    DELETE RECORDS IN cell_machines
ENDIF
IF TELL ANY RECORDS IN sarmd
    DELETE RECORDS IN sarmd
ENDIF
IF TELL ANY RECORDS IN cells
    DELETE RECORDS IN cells
    DO dump_cells
ENDIF
ENABLE ALL
FETCH NEW cell 'has machine' machine
LET fetchno = $FETCH
LET int0 = $COUNT
WHILE int0 > 0
    LET int1 = int1 + 1
    LET nam3  = FETCH VALUE (fetchno,int1,2)
*nam3 is machine
    IF TELL ANY RECORDS IN TABLE sarmd FOR ^nam3

    ELSE
        FETCH NEW machine 'has candidate' cell FOR ^nam3
        LET fetchno1 = $FETCH
        LET int2 = $COUNT
        IF int2 > 1
            LET int3 = 0
            WHILE int2 > 0
                LET int3 = int3 + 1
                LET nam0  = FETCH VALUE (fetchno1,int3,2)
                FACT IN TABLE sarmd
                ^nam3
                ^nam0

                LET int2 = int2 - 1
            ENDWHILE
        ENDIF
    ENDIF
    LET int0 = int0 - 1
ENDWHILE
DISABLE ALL
DISPLAY REPORT USING sarmd_form
GO

WINDOW MAIN_MENU
DELETE WINDOW WELCOME
```

F.7 PROCEDURE FILE 'do_count_m_c_units'

```
*
* DO GENERIS FILE 'do_count_m_c_units'

* TO COUNT THE R_ELEMENTS IN A CELL
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL fetchno2 INTEGER
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int3 INTEGER 0
CREATE LOCAL int4 INTEGER 0
CREATE LOCAL int5 INTEGER 0
CREATE LOCAL int6 INTEGER 0
CREATE LOCAL big INTEGER
CREATE LOCAL count1 INTEGER 0
CREATE LOCAL nam3 NAME
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
WINDOW NEW WELCOME SIZE 37,92 AT 2,2
POSITION 8,20
MESSAGE"ASSIGNING THE MACHINES TO THE CELL"
POSITION 10,30
MESSAGE"PLEASE WAIT ... "
IF TELL ANY RECORDS IN results
    DELETE RECORDS IN results
ENDIF
IF TELL ANY RECORDS IN result1
    DELETE RECORDS IN result1
ENDIF
ENABLE part_mcus part_machine
FETCH NEW machine 'has candidate' cell
LET fetchno = $FETCH
LET int0 = $COUNT
WHILE int0 > 0
    LET int1 = int1 + 1
    LET nam0  = FETCH VALUE (fetchno,int1,1)
    IF nam0 = BLANK
        LET int0 = int0 - 1
    ENDIF
    FETCH NEW machine 'has candidate' cell FOR ^nam0
    LET fetchno1 = $FETCH
    LET int2 = $COUNT
    LET int3 = 0
    WHILE int2 > 0
        LET int3 = int3 + 1
        LET nam1  = FETCH VALUE (fetchno1,int3,2)
*       nam0 is machine
*       nam1 is cell
        FETCH NEW part candidate machine selected m_c_unit FOR ^nam0
        LET fetchno2 = $FETCH
        LET int4 = $COUNT
        LET int5 = 0
        LET count1 = 0
        WHILE int4 > 0
            LET int5 = int5 + 1
            LET nam2  = FETCH VALUE (fetchno2,int5,1)
*           nam2 is part
            IF nam2 = BLANK
                LET nam2 = nam3
            ELSE
                LET nam3 = nam2
            ENDIF
            IF TELL ANY ^nam1 has part ^nam2
                LET count1 = count1 + 1
            ENDIF
            LET int4 = int4 -1
        ENDWHILE
        FACT IN TABLE results
        ^nam1
        ^nam0
        ^count1

        LET int2 = int2 - 1
        DELETE FETCH fetchno2
    ENDWHILE
    LET int0 = int0 - 1
    DELETE FETCH fetchno1
ENDWHILE
DELETE FETCH fetchno
DISABLE ALL
LABEL again
FETCH NEW records in results
LET fetchno1 = $FETCH
LET nam0 = FETCH VALUE (fetchno1,1,2)
* nam0 is machine
DELETE FETCH fetchno1
FETCH RECORDS IN results FOR ^nam0
LET fetchno = $FETCH
LET int0 = $COUNT
LET int1 = 0
LET big = 0
IF int0 = 0
    GOTO rubin1
ENDIF
WHILE int0 > 0
    LET int1 = int1 + 1
    LET int2  = FETCH VALUE (fetchno,int1,3)
    IF int2 = BLANK
        LET int2 = int4
    ELSE
        LET int4 = int2
    ENDIF
    IF(big < int2)
        LET big = int2
        LET int3 = int1
        LET nam1 = FETCH VALUE (fetchno,int3,1)
*       nam1 is cell
    ENDIF
    LET int0 = int0 - 1
ENDWHILE
DELETE FETCH fetchno
FACT IN result1
^nam1
^nam0

DELETE RECORDS IN results FOR ^nam0
IF TELL ANY RECORDS IN results
```

```
    GOTO again
ENDIF
DISPLAY RECORD IN result1
LABEL rubin1
WINDOW MAIN_MENU
DELETE WINDOW WELCOME
```

F.8 PROCEDURE FILE 'do_assigned_mach_cell'

```
*
* DO GENERIS FILE 'do_assigned_mach_cell'
*
* TO CREATE RECORDS IN TABLE 'cell_machines'
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int3 INTEGER 0
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
CREATE LOCAL nam4 NAME
*
WINDOW NEW WELCOME SIZE 37,92 AT 2,2
POSITION 8,20
MESSAGE"ASSIGNING THE MACHINES TO THE CELLS"
POSITION 10,30
MESSAGE"PLEASE WAIT ... "
IF TELL ANY RECORDS IN cell_machines
    DELETE RECORDS IN cell_machines
ENDIF
ENABLE ALL
FETCH NEW cell 'has machine' machine
LET fetchno = $FETCH
LET int0 = $COUNT
WHILE int0 > 0
    LET int1 = int1 + 1
    LET nam0  = FETCH VALUE (fetchno,int1,1)
    IF nam0 = BLANK
        LET nam0 = nam2
    ELSE
        LET nam2 = nam0
    ENDIF
    LET nam3  = FETCH VALUE (fetchno,int1,2)
*    nam0 is cell
*    nam3 is machine
    FACT IN cell_machines
    ^nam0
    ^nam3

    LET int0 = int0 - 1
ENDWHILE
DELETE FETCH fetchno
```

```
IF TELL ANY RECORDS IN result1
    FETCH NEW RECORDS IN result1
    LET fetchno = $FETCH
    LET int0 = $COUNT
    LET int1 = 0
    WHILE int0 > 0
        LET int1 = int1 + 1
        LET nam0  = FETCH VALUE (fetchno,int1,1)
        IF nam0 = BLANK
            LET nam0 = nam2
        ELSE
            LET nam2 = nam0
        ENDIF
        LET nam3  = FETCH VALUE (fetchno,int1,2)
*       nam0 is cell
*       nam3 is machine
        FETCH NEW RECORDS IN cell_machines
        LET fetchno1 = $FETCH
        LET int2 = $COUNT
        LET int3 = 0
        WHILE int2 > 0
            LET int3 = int3 + 1
            LET nam1  = FETCH VALUE (fetchno1,int3,1)
            IF nam1 = BLANK
                LET nam1 = nam4
            ELSE
                LET nam4 = nam1
            ENDIF
            IF nam0 = nam1

            ELSE
                IF TELL ANY ^nam0 has machine ^nam3
                    DELETE RECORDS in cell_machines for ^nam1 for ^nam3
                ENDIF
            ENDIF
            LET int2 = int2 - 1
        ENDWHILE
        DELETE FETCH fetchno1
        LET int0 = int0 - 1
    ENDWHILE
    DELETE FETCH fetchno
ENDIF
DISABLE ALL
DISPLAY REPORT USING cell_machines_form
GO

WINDOW MAIN_MENU
DELETE WINDOW WELCOME
```

F.9 PROCEDURE FILE 'do_not_used_m_c_unit'

```
*
* DO GENERIS FILE 'do_not_used_m_c_unit'
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL nam3 NAME
CREATE LOCAL nam2 NAME
CREATE LOCAL nam0 NAME
*
WINDOW NEW WELCOME SIZE 37,92 AT 2,2
POSITION 8,4
MESSAGE"FINDING OUT FOR WHICH PARTS RE_ELEMENT, THERE IS NOT ANY
MACHINE AVAILABLE "
POSITION 10,32
MESSAGE"PLEASE WAIT ... "
DISABLE ALL
ENABLE part_mcus part_machine
IF TELL ANY RECORDS in not_used_m_c_units
    DELETE RECORDS IN not_used_m_c_units
ENDIF
FETCH new part m_c_unit 'need'
LET fetchno = $FETCH
LET int0 = $COUNT
WHILE int0 > 0
    LET int1 = int1 + 1
    LET nam0 = FETCH VALUE (fetchno,int1,1)
    LET nam3  = FETCH VALUE (fetchno,int1,2)
*    nam0 is part
*    nam3 is m_c_unit
    IF nam0 = BLANK
        LET nam0 = nam2
    ELSE
        LET nam2 = nam0
    ENDIF
    IF TELL ANY ^nam0 selected ^nam3

    ELSE
        FACT IN TABLE not_used_m_c_units
        ^nam0
        ^nam3

    ENDIF
    LET nam2 = nam0
    LET int0 = int0 - 1
ENDWHILE
DELETE FETCH fetchno
DISABLE ALL
IF TELL ANY RECORDS IN not_used_m_c_units
    DISPLAY RECORDS IN not_used_m_c_units
ELSE
    POSITION 15,10
    MESSAGE"FOR ALL THE PART'S MCU'S, MACHINES ARE AVAILABLE"
    HOLD 4
ENDIF
```

```
WINDOW MAIN_MENU
DELETE WINDOW WELCOME
```

F.10 PROCEDURE FILE 'do_subset'

```
*
* DO GENERIS FILE 'do_subset'
*
* TO FIND OUT THE SUBSETS IN THE GROUPING PROCESS

* RECORDS IN TABLE 'results' ARE WRITTEN TO FIND OUT THE EXISTING CELLS FOR
WHICH SEARCH IS TO BE MADE

* THEN RECORDS IN TWO DIFFERENT CELLS ARE TAKEN IN TERN WHETHER THE
PROCESSES AVAILABLE IN ONE CELL ARE AVAILABLE IN ANOTHER CELL, IF SO, CELL
CAN BE MERGED
*
CREATE LOCAL fetchno INTEGER
CREATE LOCAL fetchno1 INTEGER
CREATE LOCAL fetchno2 INTEGER
CREATE LOCAL fetchno3 INTEGER
CREATE LOCAL int0 INTEGER 0
CREATE LOCAL int1 INTEGER 0
CREATE LOCAL int2 INTEGER 0
CREATE LOCAL int3 INTEGER 0
CREATE LOCAL int4 INTEGER 0
CREATE LOCAL int5 INTEGER 0
CREATE LOCAL int6 INTEGER 0
CREATE LOCAL int7 INTEGER 0
CREATE LOCAL int8 INTEGER 0
CREATE LOCAL count1 INTEGER
CREATE LOCAL nam0 NAME
CREATE LOCAL nam1 NAME p68
CREATE LOCAL nam2 NAME
CREATE LOCAL nam3 NAME
*
WINDOW NEW WELCOME SIZE 37,92 AT 2,2
POSITION 3,10
MESSAGE"FINDING OUT WHETHER ANY CELL CAN BE MERGED INTO ANY OTHER
CELL"
POSITION 5,30
MESSAGE"PLEASE WAIT ... "
IF TELL ANY RECORDS IN cells
    DELETE RECORDS IN cells
    DO dump_cells
ELSE
    DO dump_cells
ENDIF
DISPLAY report using cells_form
GO

IF TELL ANY RECORDS IN results
    DELETE RECORDS IN results
ENDIF
FETCH NEW RECORDS IN cells
LET fetchno = $FETCH
LET int0 = $COUNT
WHILE int0 > 0
    LET int1 = int1 + 1
    LET nam0 = FETCH VALUE (fetchno,int1,1)
    IF nam1 = nam0

    ELSE
        FACT IN results
        ^nam0

        1

    ENDIF
    LET nam1 = nam0
    LET int0 = int0 - 1
ENDWHILE
DELETE FETCH fetchno
FETCH NEW RECORDS IN results
LET fetchno = $FETCH
LET int0 = $COUNT
LET int1 = 0
LET int5 = int0
WHILE int0 > 0
    LET int1 = int1 + 1
    LET nam0  = FETCH VALUE (fetchno,int1,1)
    FETCH NEW RECORDS IN results
    LET fetchno3 = $FETCH
    LET int5 = $COUNT
    LET int6 = 0
    WHILE int5 > 0
        LET int6 = int6 + 1
        LET nam2  = FETCH VALUE (fetchno3,int6,1)
        IF nam2 = nam0

        ELSE
            FETCH NEW RECORDS IN cell_process FOR ^nam2
            LET fetchno1 = $FETCH
            LET int2 = $COUNT
            LET int7 = int2
            LET count1 = count1 + 1
            LET int3 = 0
            WHILE int2 > 0
                LET int3 = int3 + 1
*               nam1 is process
                IF TELL ANY ^nam0 'has process' ^nam1
                    LET count1 = count1 + 1
                ENDIF
                IF count1 = int7
                    FETCH NEW RECORDS IN cells FOR ^nam2
                    LET fetchno2 = $FETCH
                    LET int4 = $COUNT
                    LET int8 = 0
                    WHILE int4 > 0
                        LET int8 = int8 + 1
                        LET nam3 = FETCH VALUE (fetchno2,int8,2)
*                       nam3 is part
```

```
                        FACT IN cells
                        ^nam0
                        ^nam3


                        IF int4 = 1
                            CLEAR SIZE 40,85 AT 5,0
                            POSITION 8,20
                            MESSAGE "CELL "^nam2" HAS BEEN MERGED TO CELL
                            "^nam0" "
                        ENDIF
                        LET int4 = int4 - 1
                    ENDWHILE
                    DELETE FETCH fetchno2
                    DELETE RECORDS IN cells FOR ^nam2
                ENDIF
                LET int2 = int2 - 1
            ENDWHILE
        ENDIF
        LET int5 = int5 - 1
    ENDWHILE
    DELETE FETCH fetchno3
    LET int0 = int0 - 1
ENDWHILE
DELETE FETCH fetchno
DISPLAY REPORT USING cells_form
GO

WINDOW MAIN_MENU
DELETE WINDOW WELCOME
```