THE APPLICATION OF KNOWLEDGE BASED EXPERT SYSTEMS TO MECHANICAL ENGINEERING DESIGN

by

Hezekiah Olushola Olatunde WILLIAMSON-TAYLOR

Doctor of Philosophy

The University of Aston
in Birmingham

July 1990

The University of Aston in Birmingham

The Application of Knowledge Based Expert Systems to Mechanical Engineering Design

Hezekiah Olushola Olatunde WILLIAMSON-TAYLOR

Doctor of Philosophy

July1990

SUMMARY

Investigation of the different approaches used by Expert Systems researchers to solve problems in the domain of Mechanical Design and Expert Systems was carried out. The techniques used for conventional formal logic programming is compared with those used when applying Expert Systems concepts. A literature survey of design processes was also conducted with a view to adopting a suitable model of the design process. A model, comprising a variation on two established ones, was developed and applied to a problem within what are described as class 3 design tasks. The research explored the application of these concepts to Mechanical Engineering Design problems and their implementation on a microcomputer using an Expert System building tool. It was necessary to explore the use of Expert Systems in this manner so as to bridge the gap between their use as a control structure and for detailed analytical design. The former application is well researched into and this thesis discusses the latter. Some Expert System building tools available to the author at the beginning of his work were evaluated specifically for their suitability for Mechanical Engineering design problems. Microsynics was found to be the most suitable on which to implement a design problem because of its simple but powerful Semantic Net Knowledge Representation structure and the ability to use other types of representation schemes. Two major implementations were carried out. Two concepts were proposed in the thesis for the modelling and implementation of design systems involving many equations. The method proposed enable equation manipulation and analysis using a combination of frames, semantic nets and production rules. The use of semantic nets for purposes other than for psychology and natural language interpretation, is quite new and represents one of the major contributions to knowledge by the author. The development of a purpose-built shell program for this type of design problem was recommended as an extension of the research. Microsynics may usefully be used as a platform for this development.

KEY WORDS: Computer aided design, Expert Systems, Intelligent Knowledge based systems, Expert design system, IKBS. Mechanical design system.

DEDICATION

This work is dedicated to:

my fiancee, Joyce Ololade Awofisayo for her patience, encouragement, moral support and her belief in me.
my mother, Mrs Florence Modupe Taylor and sister, Miss Henrietta Olarewaju Taylor for their strength and dedication.

ACKNOWLEDGEMENTS

The author gratefully acknowledge the trust, support and encouragement given by his supervisor, Professor T.H.E.Richards. Especially for giving the author the opportunity to undertake this research project at a time when his morale and confidence was extremely low.

The author wishes to thank Professor Foster and Dr J.E.T.Penny for their interest and advice at the early stage of the project.

Acknowledgement is also due the University for their financial support.

Lastly, the author will like to thank his friends and colleagues at the department for their help and advice.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

SYMBOLS/ VARIABLE CODE	MEANING
a	Centre distance (mm)
b	Face width (mm)
c / SI	Spring index
c	Clearance (mm)
d / WD	Wire diameter (mm)
d	Pitch circle diameter (mm)
db	Base circle diameter (mm)
do	Outside diameter (mm)
dr	Root diameter (mm)
fn	Natural frequency of spring (cycle/min)
h	Whole depth (mm)
ha	Addendum (mm)
hf	Deddendum (mm)
k / SS	Spring rate (N/mm)
k	Addendum modification factor
m	Module (mm)
mn	Normal module (mm)
n / WC	Active number of coils
p	Circular pitch (mm)
pb	Base circular pitch (mm)
pn	Normal circular pitch (mm)
rr	Root radius (mm)
t	Tooth thickness (mm)

u	Gear ratio
v	Pitch line velocity (m/s)
vf	Velocity factor
z	Number of teeth
zv	Virtual number of teeth
CC	A constant whose value depends on whether a spring is lateral guided or not laterally guided.
CC	A constant whose value depends on the gear tooth form.
D / DM	Mean coil diameter (mm)
E	Elastic modulus (N/mm^2)
EH	A constant whose value depends on whether the spring end is closed or open.
EH1	A constant whose value depends on the spring end condition
F / PMAX	Maximum load (N)
Fd	Dynamic force (N)
Fe	Axial force (N)
Fl / FL	Free length (mm)
Fm	Mean load (N)
Fmax	Maximum load (N)
Fmin	Minimum load (N)
Fo	Allowable endurance load (N)
Fs	Separating force (N)
Ft	Tangential force (N)
Fv	Variable load (N)
Fw	Wear force (N)

G / GG	Shear modulus (N/mm^2)
K / WF	Wahl factor
Ka	Application ratio
$K_F\alpha$	Load factor for bending stress across facewidth (and $K_F\beta$)
KK	Load stress factor
Kv	Dynamic factor
Lc	Solid length (mm)
LP	Length in position (mm)
N / NN	Total number of coils
OPL	Operating length (mm)
P	Power (kW)
RD	Relative deflection (mm)
SL	Slenderness ratio
X1	A constant whose value depends on the pitch line velocity.
Yf	Form factor
Y_M	Material quality factor.
Y_N	Life factor.
Y_R	Surface condition factor.
Ys	Stress correction factor
Yx	Size factor.
$Y\delta$	Sensitivity factor.
Z_E	Elasticity factor.
Z_H	Geometric factor.
Z_L	Lubrication factor.

Z_M	Material quality factor.
Z_N	Life factor.
Z_R	Roughness factor.
Z_V	Speed factor.
Z_W	Work harding factor.
Z_X	Size factor.
Z_ε	Contact ratio factor
α	Pressure angle (normally 20 deg)
α_n	Normal pressure angle (deg)
β	Helix angle (deg)
β_n	Normal helix angle (deg)
δ / DFL	Deflection (mm)
δ_{max1}	Deflection under maximum stress (mm)
δ_{max2}	Maximum physical deflection (mm)
τ / QQ	Maximum shear stress (N/mm^2)
σ_e	Surface endurance limit (N/mm^2)
σ_m	Mean stress (N/mm^2)
σ_y	Yield stress (N/mm^2)
σ_{rel}	Endurance strength for released loading (N/mm^2)

σ_B	Ultimate tensile stress (N/mm^2)
σ_F / σ_i	Induced bending stress (N/mm^2)
σ_{Fo}	Endurance strength for released bending load (N/mm^2)
σ_{FP}	Allowable stress (N/mm^2)
σ_H	Induced contact stress (N/mm^2)
σ_{HP}	Allowable contact stress (N/mm^2)
σ_R	Residual stress (N/mm^2)
ε	Total contact ratio
$\varepsilon\alpha$	Transverse contact ratio
$\varepsilon\beta$	Overlap ratio
1	Suffix (pinion)
2	Suffix (Wheel)

CHAPTER ONE

INTRODUCTION

CHAPTER ONE

INTRODUCTION

1.1 Aim of Research

The aim of the research was to examine the application of Expert System concepts to Mechanical Engineering Design problems and their implementation on a microcomputer using an Expert Systems building tool. The broad approach to the task was to investigate the different approaches used by Expert Systems researchers to solve problems in this and related domains, the assessment of different types of Expert Systems tools and knowledge representation techniques, the study of design methodologies and then to marry together the two different approaches. In the process of investigating the application of Expert Systems to Engineering Designs, the techniques used for conventional formal logic programming were compared with those used when applying Expert Systems concepts.

A survey of the literature on design processes was conducted with the view to adopting a suitable model. Design was defined by Finkelstein [1] as a creative process which starts from a requirement and defines a system and its methods of realisation or implementation, so as to satisfy the requirements. Two commonly used general models were identified and adopted with some modifications for the

present work. Shigley's model [2] was based on six design stages: Recognition of need, definition of problems, synthesis, analysis and optimisation, evaluation and presentation. This model was incorporated into that proposed by Pahl and Beitz [3]. This consisted of five phases: the recognition of need, definition and clarification of task, conceptual phase, embodiment phase and detailed design phase. The modified model is discussed in detail in chapter 2 and its application to so called class 3 design tasks, [3],[4], in conjuction with Expert System concepts is described subsequently. Appendix A1 gives a brief description of the classes of design.

A study of the literature on Expert Systems and their applications to a variety of problem areas led to the need for a study of various tools (languages and shells) and their capabilities. Some Expert System building tools available to the author at the beginning of his work were evaluated specifically for their suitability for Mechanical Engineering design problems. Such tools included Sage, MicroExpert, Expert Ease, Microsynics, ES/P Advisor, TIMM, Lisp and Prolog. Expert System programs are based on various concepts and Knowledge Representation schemes. The major representational schemes are: Declarative such as Logic, Semantic nets; Procedural such as Production systems; Frames which combine declarative schemes with procedural ones. The Shell packages evaluated represent examples of the different kinds of techniques used for Expert Systems development. Microsynics was found to be the

most suitable on which to implement a design problem because of its simple but powerful Semantic Net Knowledge Representation structure and the ability to use other types of representation schemes in conjunction with it. The development of various facilities required in tackling design problems was also reasonably convenient.

The question of marrying together the two concepts and approaches was addressed. To do this, the iterative nature of design has to be limited to a certain extent if the system, within the limitations of the knowledge representation techniques used, was to "remember" all that had been done, to keep track of the processes yet to be executed and also give advice and explanations based on the state of the process at the time (also taking into consideration what has been done). To limit the scope of iteration, the subject of variable inter-relationships and limits on value were carefully examined and procedures developed to cater for this. Factors due to manufacturing processes and safety codes where used to enforce legitimate limitations on the values acceptable for processsing. Adapting the knowledge representation system used in Microsynics to track down executed functions was another approach used in bringing the two concepts involved together. Design is iterative in nature and this makes it a difficult domain for Expert Systems application since equation solving and formula processing are also involved.
Two major case studies were carried out.
The first, dealt with helical compression springs using the

first of the two proposed concepts for the system structure required for the application of Expert Systems to this domain. The second case study was on gear design applying the second proposed concept. The detailed discussion of the two concepts are presented in chapter 5.

1.2 Layout of the Thesis

The thesis comprises eight chapters. The early ones give the necessary background and understanding to the approach adopted in the course of the research. The layout of the diagrams is such that at the end of every sub-section (for example, Sub-section 4.5.3) there are diagrams relating to that sub-section.

The first chapter introduces the reader to the thesis and the main subjects concerned with the research.

The second chapter discusses the detail approach to Mechanical Engineering design. It outlines the interrelationship between the modelling of a problem for implementation and the design processes normally used for Engineering components and systems. It then deals with the contribution of computers to Engineering design which leads to the discussion of Expert Systems and their concepts.

The third chapter discusses Expert Systems techniques with the aim of identifying the requirements, differences and

knowledge representation techniques of Expert Systems. Because a representation scheme cannot be said fully to satisfy the requirements of the problem domain, the different types of representation and their contributions are emphasized. The use of declarative representation (such as logical schemes (for example, Fortran)) for Computer-aided design programs has proved very successful, making the introduction of new techniques for programming in this domain difficult to accept. Because of this, the distinction between conventional programs and Expert System programs is addressed. Also, the need to understand that all designs do not necessarily require the same representation techniques is discussed. Having discussed the design process and Expert Systems aspects of the research, the work of other Expert Systems researchers is reviewed in the following chapter.

The fourth chapter gives a review of literature on Expert Systems research that have been conducted in the area of mechanical engineering design and identifies the knowledge gaps which this current research addressed. The first four chapters thus provide the platform on which the remainder of the work was built.

Chapter 5 discusses the general philosophy and approach to the current research and its various implementations. It demonstrates how the different concepts and techniques discussed in the previous two chapters were merged into one and what, if any, features where discarded. In particular,

what the present author has called Level and Frame based knowledge grouping concepts are introduced.

Chapter 6 deals with case study (1). It is concerned with incorporating the previously established concepts in a knowledge based program for helical compression spring design implemented on the Apricot computer.

Chapter 7, deals with the second case study which was the implementation on the Apricot computer of a gear pair design system.

Chapter 8 provides a discussion of what was achieved in the course of the research, what contribution has been made to the advancement of knowledge and understanding in the application of Expert Systems concepts to the solution of engineering design problems and draws conclusions on the research done. Recommendations for future work in this area was also discussed.

There are four appendices (A1 to A4). Appendix A1 lists the three classes of design and gives a brief explanation of each. The second appendix discusses the major parts of an Expert System, including the reasoning strategies. Appendix A3 discusses the guidelines on the concept of knowledge grouping. The last, Appendix A4, gives sample frames on information presented to the user for one of the case studies presented in the thesis. The last section of this thesis contains the references.

CHAPTER TWO

THE ENGINEERING DESIGN PROCESS

CHAPTER TWO

THE ENGINEERING DESIGN PROCESS

2.1 Introduction:

Pahl and Beitz [3] and Beakley and Leach [5], defined engineering design as a creative process, Rzevski [6] added to this definition by stating that design is also an investigative, logical-based and decision making process. Duderstadt et al [7], defined design as a complex activity that makes stringent demands on the designers skill, experience, knowledge and creativity. Eder [8] in his paper defined engineering design as the use of scientific principles, technical information and imagination in the definition of a mechanical structure, machine or system to perform prespecified functions with the maximum economy and efficiency. Various other researchers (for example, Cross et al [9], Gasparski [10] and Dilnot [11]) gave various definitions for design with all agreeing that design is a creative process. Hence, the engineering design process can be said to consist of series of steps starting from a need and terminating with the creation of a useful product. To assist in this process, engineers use design models as an aid. These models and the model adopted by the present author are discussed in the next section. Subsequent

sections describe the functions of each design activity within the adopted design model. It begins with the definition and classification of design problems in section 2.3 then progresses through the phases of conceptual design (section 2.4), embodiment design (section 2.5) and finally to detail design (section 2.6). Section 2.7 discusses component design and the contribution and importance of computers to present day engineering design applications is discussed in section 2.8. Having viewed the essentials of the design process, the next chapter will consider Expert Systems concepts and their relevance for engineering prior to addressing particular implementations for the design of a typical component and a simple system.

2.2 Design models

Engineers use design models to assist in the systematic execution of design activities necessary to arrive at an optimal solution. A model is a representation of a system or process in the form of an icon or symbols. The model of a design process thus represents the various design activities in the form of a diagram (or flow chart) representation. Hence, the model of a design process shall be referred to as a design model in this thesis. With the aid of design models, the design process can be viewed in perspective and it can be broken down into smaller problems. Also the interactions between design phases can be appreciated. However, the limitations of design models

are that there are differences between the sequence of activities as represented by them and the sequence of design activities that may be executed in practice. Design models are not unique [12]. That is, different models may be used on the same problem.

The various design activities were grouped by Pahl and Beitz into four phases. They are, the clarification of the task, the conceptual phase, embodiment phase and the detail design phase. Shigley [2] presented a five phase design model. They are definition of problem, synthesis, analysis and optimization, and evaluation. The present author adopted a six phase design model shown in Fig 2.1. It represents a modification of the design models presented by Pahl and Beitz and Shigley. Woodson [12] stated that a good design model should have separate terms for seperate actions. This also represents the views of the present author and for that reason, a modified model of Pahl and Beitz [3] and Shigley [2] was considered as suitable. Also, the sequence of activities as represented by this modified model best suit the design of components.

Other commonly used design models included the one proposed by Beakley and Leach [5]. He grouped the design activities into four phases namely, the feasibility phase, the preliminary phase, the intermediate phase and the detail design phase. The feasibility phase includes the definition of the problem, identification of design constraints and problem formulation. The preliminary phase includes the

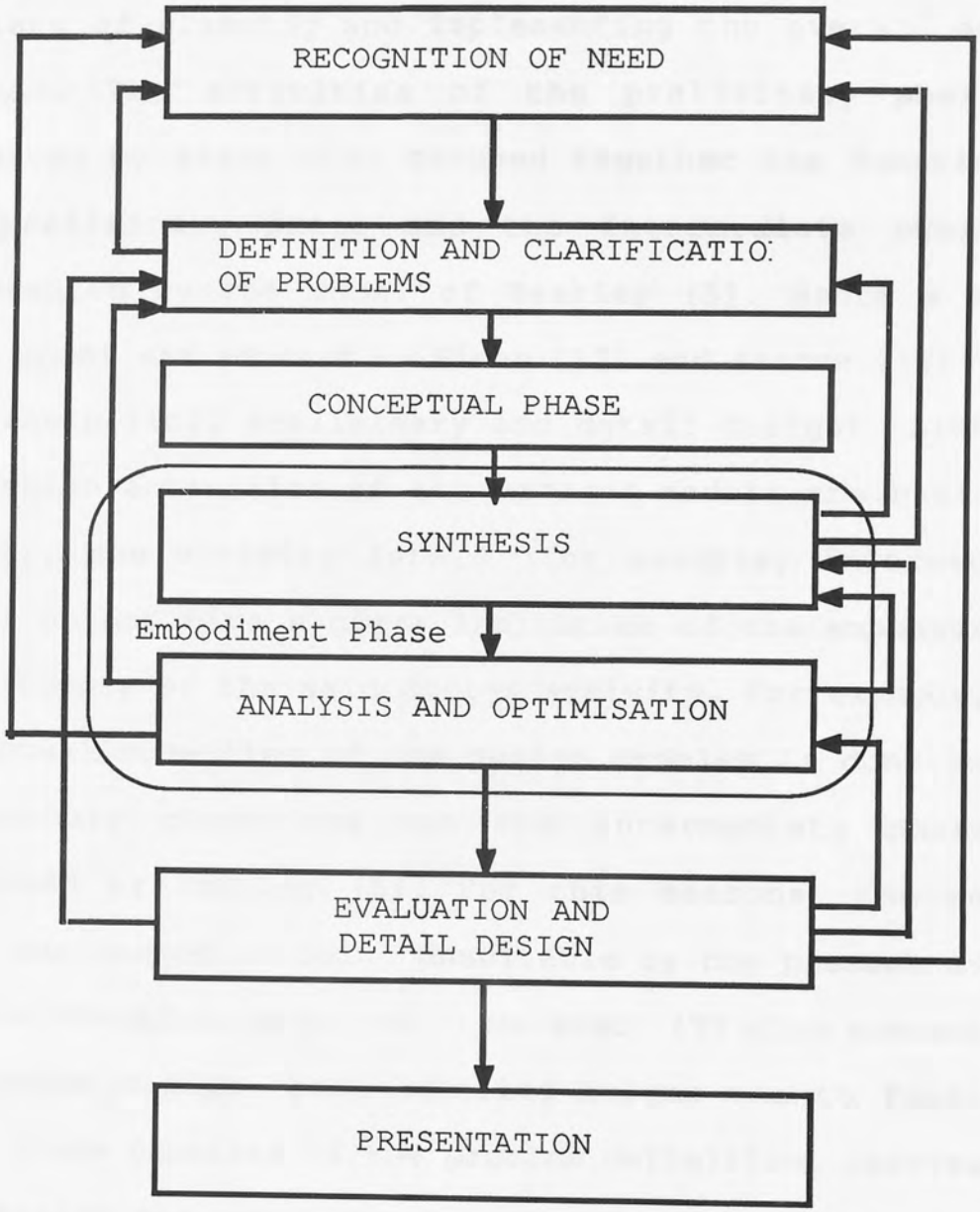


Fig 2.1 Modified design model showing the design phases

conceptual stage of design and the intermediate phase consists of planning and implementing the overall design concept. The activities of the preliminary phase as conceived by Simon [13] grouped together the function of the preliminary phase and the intermediate phase as represented by the model of Beakley [5]. Hence a three phase model was adopted by Simon [13] and Asimow [14] (that is, feasibility, preliminary and detail design). Although the design activities of the various models are basically similar, the activity labels (for example, intermediate phase) do not give a clear indication of the emphasis and constituents of the said design activity. For example, the conceptual modelling of the design problem is done in the preliminary phase and not the intermediate phase as presented by Beakley [5]. For this reasons, the design model was deemed as being unsuitable by the present author and was therefore rejected. Duderstadt [7] also presented a four phase design model labelled stages one to four. The first stage consists of the problem definition, information acquisition and model development. The solution synthesis falls into the next stage and the third stage consists of the solution verification and evaluation. The last stage is the presentation of the result.

The next section discusses the first phase of the adopted design model. That is, the definition and clarification phase.

2.3 Definition and clarification of problems

This stage of the design process is concerned with the establishment of design constraints and drawing up a specification because of the vagueness that characterises design problems [7]. A specification helps in defining and clarifying the design task and serves as a source of reference during the iterative process. The definition of the design task requires the known, unknown and desired information to be established. In most cases, all the information required for the next design phase to be initiated are unavailable. Therefore, an information acquisition process must be conducted. Parameter values like speed, geometrical dimensions, and their limitations are clarified. Implied constraints such as those caused by factory available facilities for manufacturing purposes and hence, the choice of material, cost and availability (of material) must be established and defined (that is, not what the client says he/she wants but why he/she wants it). The problem definition must take into account constraints imposed by national standards, guidelines and recommendations and to obtain an optimal solution, the design contraints and specification must be well understood. Future need to change the requirements either due to technical advancement or change in circumstances should be incorporated into the overall process. The next step requires the collation of all the relevant information and data into a form that lends itself to technical analysis. The information and data obtained must be

classified into either a demand or a wish [3]. This is necessary because a demand must be satisfied for the solution to be acceptable but a wish need not be. Pahl and Beitz [3] also proposed the division of wishes into major, medium or minor importance. Information and data characteristics such as quantity and quality (for example, for corrosion environment or waterproof requirements) should not be overlooked.

Pahl and Beitz [3] and Fox [15] recommended the following general method of compiling a specification:

1) Compile the requirements.

* Determine the quantitative and qualitative data.
* Ask: What objective must the solution satisfy? What properties must it have.
* Specify demands and wishes clearly.
* If possible rank wishes as being of major, medium or minor importance.

2) Arrange the requirements in clear order:

* First define the main objectives and the main characteristics.
* Then split into identifiable sub-systems, functions, assemblies etc.

In component design, the clarification phase is relatively simple. For this project the Britsh Standard requirements are considered as the demands and the data obtained from the client are taken as wishes of medium importance.
Once the task has been adequately clarified, the conceptual design phase may begin.

2.4 Conceptual Design

French [16] described this phase of the design process as the generation of broad solutions to the task, in the form of schemes or concepts. This can be done by the identification of the important tasks, the search for appropriate solution principles and their combination, [17],[18]. The conceptual design solutions based on the application of modern technology are most likely to produce the best solutions. Since this is a stage were engineering science, practical knowledge, production methods and commercial aspects are brought together, and where most important decisions are made, [16], the emphasis of general solution concepts is essential. The adoption of a general solution concept is to obtain an optimal solution. Generality is the core of the solution principle and it is essential to avoid a conventional solution concept as this tends to cloud innovative and better solution concepts. The clarification of these general solution concepts leads to the definition of the overall task. Next is the analysis and specification in respect of the required function and the essential constraint. Pahl and Beitz [3] advised that the functional relationships contained in the specification be formulated explicitly and be arranged in order of importance. Conflict between current definition and decisions already taken may occur as the definition is systematically generalized during the design process.

However, this progression creates new avenues to pursue. Once the general solution concepts have been established, they can now be laid out in a definite way. This phase is called the embodiment phase [16].

2.5 Embodiment Design.

Embodiment is the process of giving concrete form to an abstract concept or ideal. Hence, embodiment design is the process of giving concrete form to the conceptual design solutions. This may entail embodiment of more than one design concept before an acceptable design solution is found. These acceptable solutions must be developed to a stage where the functional capabilities of each sub-design or components can be fully assesed. The functional capabilities include the mode of operation, reliability and durability. The general layout of the design are determined at this phase including material selection, and the geometrical shapes of various components. The determination of these layouts is conducted in an attempt to achieve a given goal with the available materials and acceptable geometrical shapes and constraints of sub-components. Hence, the use of technical methods and principles, the application of stress analysis, differential equations, material technology etc., are needed for the assesment of components and their general layout.

The embodiment phase is the most difficult phase of the

design process in that it entails the simultaneous execution of the synthesis, analysis, optimization (that is, trade offs, maximisation, minimisation of parameters including cost, size, weight etc.), and evaluation (or assessement) of the general design on a continuous basis. Information gathering is also conducted, although it is not a major process at this phase but is needed because any additional information that may affect the design principle must be obtained. Since every aspect of a design principle is questioned at this stage, additional information may be required to clarify doubts that may arise. Because constant checks for design faults must also be conducted, the subsequent correction of any design fault means that these activities (that is, analysis, synthesis, optimization, evaluation and information gathering process) will be conducted iteratively. The iteration process must be "forward" moving. The quick identification and categorisation of problems arising helps in achieving a systematic approach to the overall principle and helps the iterative process to progress.

The close examination of design solutions may raise doubts as to the practicality of a solution concept. In some cases, the basic design concept may be seriously flawed and if such a case arises, the conceptual phase has to be re-initiated. An embodiment design solution cannot correct a bad solution concept and will cause difficulties in the detail design phase. Also, alterations to any design area must be carefully considered as it may have a serious

effect on other design areas or parameters.

A detail planning of the embodiment design phase is difficult because of activities such as checking design faults, additional information gathering when necessary, evaluation, the iterative process etc. It cannot be predicted how many times each activity will have to be conducted and in what sequence they may come after a design fault detection. The optimum design solution is also difficult to predict in advance if an open view of the final solution is adopted. This, however is the best strategy to adopt until a clear design choice emerges. The restrictions imposed by national standards and codes of practice usually help in this process. Hence, as a rule, designers must start from a general and rough design solution and proceed to a more concrete and detail design solution.

Additional factors that help the process of embodiment design is the adoption of the principles of clarity, simplicity and safety. The avoidance of ambiguity help in the fast execution of all the activities embodied in this phase. A simple design, design layout or solution principlesusually leads to a design with less parts and also to ease of understanding of the various functional capabilities. Having safety in mind reduces the chance of the design being rejected on the basis of operational safety by the National Health and Safety Executive or an equivalent national body responsible for such a function.

Safety is fundamental to the final acceptance of a design.

2.6 Detail design.

The final and a major part of the design process is detailing. This phase completes the integration of all technical aspects with final instructions about the layout, form, dimensions and surface properties of all individual components, the absolute selection of materials and closer examination of manufacturing methods and costs. The detailing process is coupled with the evaluation phase (Fig 2.1). Although the major function at this phase is detailing, continuous evaluation must be carried out at every stage of the design process and emphasised at this phase. The reason for the emphasis is because the careful evaluation of the design may reveal many design errors not detected at earlier stages.

In the following section, a brief discussion on how the above concepts were incorporated in the two case studies underlying the work of this thesis is presented.

2.7 Discussion on component design.

2.7.1 Helical compression spring design.

The need to design a helical compression spring implicitly means that there is a need to regulate the rate of

displacement between a member(s) in relation to another member(s) due to additional weight or force on the former. An example, of this application is a spring front suspension on a car. The geometric constraints placed on a spring design for this application will be dictated partly by the geometry of the working area and spring housing. Other constraints such as the working length, the initial load, the maximum applicable load will partly depend on the weight of the car (corrected for engine position) and partly on the load carry capacity of the car. The definition and clarification of the problem has to be carried out by drawing up a specification based on the particular application. The problem of the type of spring material available can be addressed at this stage or in the embodiment phase. Different spring arrangements, such as nested springs or serial springs will be considered in the conceptual stage. The rejection of some of the arrangements will be due to geometric constraints. In the embodiment phase all the different ideas on arrangements or spring material usage will have to be put together to form a realistic design proposition. More detailed analysis of the chosen design will then have to be carried out with a view to obtaining an acceptable design for the geometric and stress analysis/load requirements. The material specification used for the project was cold drawn steel [19]. The geometry of helical compression springs can be completely specified by four variables: the wire diameter, the mean coil diameter, the total number of coils and the free length. If the specification of a spring, whether in

drawn or written form is not clear, a designer may not get what he wants. In fact, he may obtain more than he actually requires (since most design formulae and methods are conservative in nature), and this may lead to waste of material and perhaps high manufacturing costs.
The approach adopted in this respect by the author, was to use conservative methods which lead to in-built safety factors. The manufacturing cost was regarded as being of less importance.

In terms of specification, the information acquisition phase considers all information as of "medium wish" because each variable has an in-built level of importance. More details about this are given in chapter 6.

2.7.2 Gear pair design.

The primary task for a gear designer, is to achieve a gear pair that transmits the required power at a specified speed for the required life. The types of gears addressed in this project are involute spur and helical gears with unmodified teeth. BS 436:part 3:1986 [20] gives all the necessary factors for modification of the bending and contact limiting stress values for gear teeth required. There are various other equations available for the approximate determination of design parameter values (for example, the Lewis and Baath equations [4]). The standard also gives the necessary types of heat treatment required to obtain the hardness parameter values needed. Chapter 7 deals with gear pair design and discusses in detail all the factors

affecting designs of this nature.

The gear pair design is a part of an existing gearbox design system. When relating such a design problem to a design model, it enables the easy identification of various phases. For example, the design of a gear pair will be constrained by geometry, power requirements, heat dissipation, type of bearing used etc. The task must be defined and clarified (and a specification drawn), the suitable application of a helical gear or spur gear has to be decided and analysed. The problem of vibration, noise etc has to be taken into account when deciding on the type of gearing (e.g., helical gears are normally quieter than spur gears). The gear arrangement required or necessary can be limited by the size of gearbox required. An iterative process is conducted until a suitable design emerges.

Having discussed the design process and models, the contribution and use of computers is discussed in the next section.

2.8 Computer-aided Engineering (CAE)

Advances in computer technology have had a remarkable effect in the field of Engineering. Computers are now cheaper, smaller and faster and this has lead to their increasingly rapid adoption by industry and commerce. CAE is still one of the fastest growth areas in industry and

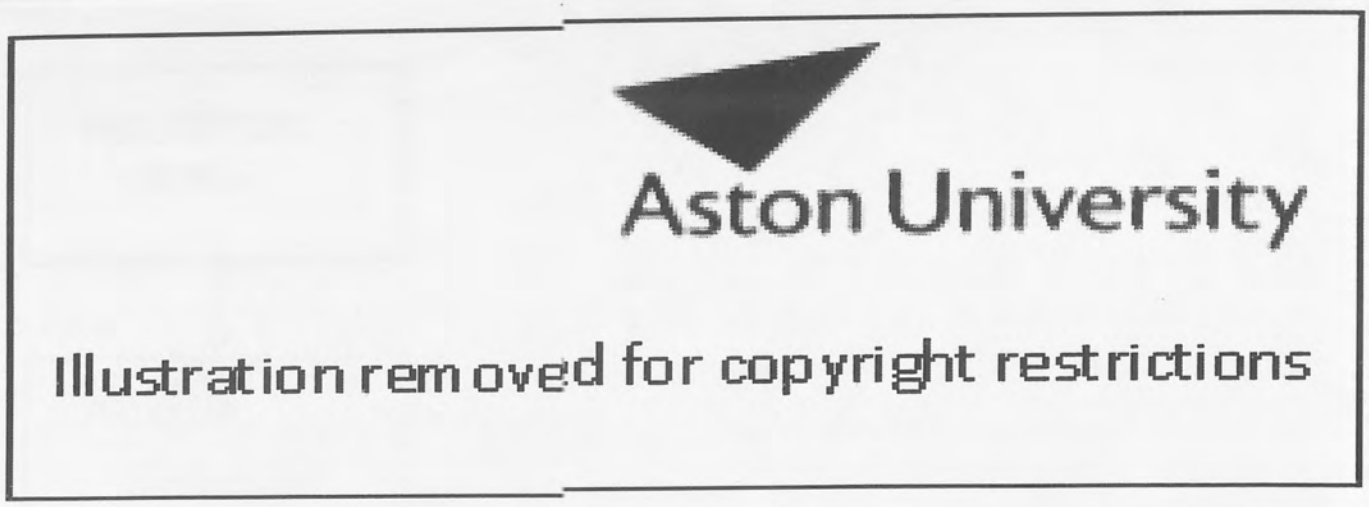
has a direct effect on industrial operations [21]. There is no doubt that many of todays methods of design, manufacture and assembly will continue to undergo radical changes as the whole philosophy and potential of CAE is realised. It is accepted that, in most cases, CAE methods enables the design process to be faster and cheaper and inceases the probability of products to be designed and manufactured correctly the first time round. The technology of CAE is concerned with the application of computers to the design and manufacturing of engineering components [22]. Reed [23] gave a product cycle block diagram, presented here in Fig 2.2, modified for the application of computers to the subject area. This product cycle incorporates the entire process from product need to manufacturing. It implies that the engineering design process is cyclic in nature and is a continuous process until there is no further need for the product. The aspect of CAE addressed in this thesis is computer-aid for engineering design (CAD) and Design Engineering.

CAD embraces all activities described in Fig 2.3. Therefore with this definition, CAD can represent objects in 2-, 2(1/2)- or 3-dimensional object form, embrace specific analysis programs developed for stress-strain, heat transfer, dynamic behaviour, mass properties, calculations etc. CAD software can be divided into two main areas, a) graphics software and b) design analysis software. It is the latter area that this thesis is concerned with.

Aston University

Illustration removed for copyright restrictions

Fig 2.2 Product cycle revised for CAD/CAM according to Reed [23].



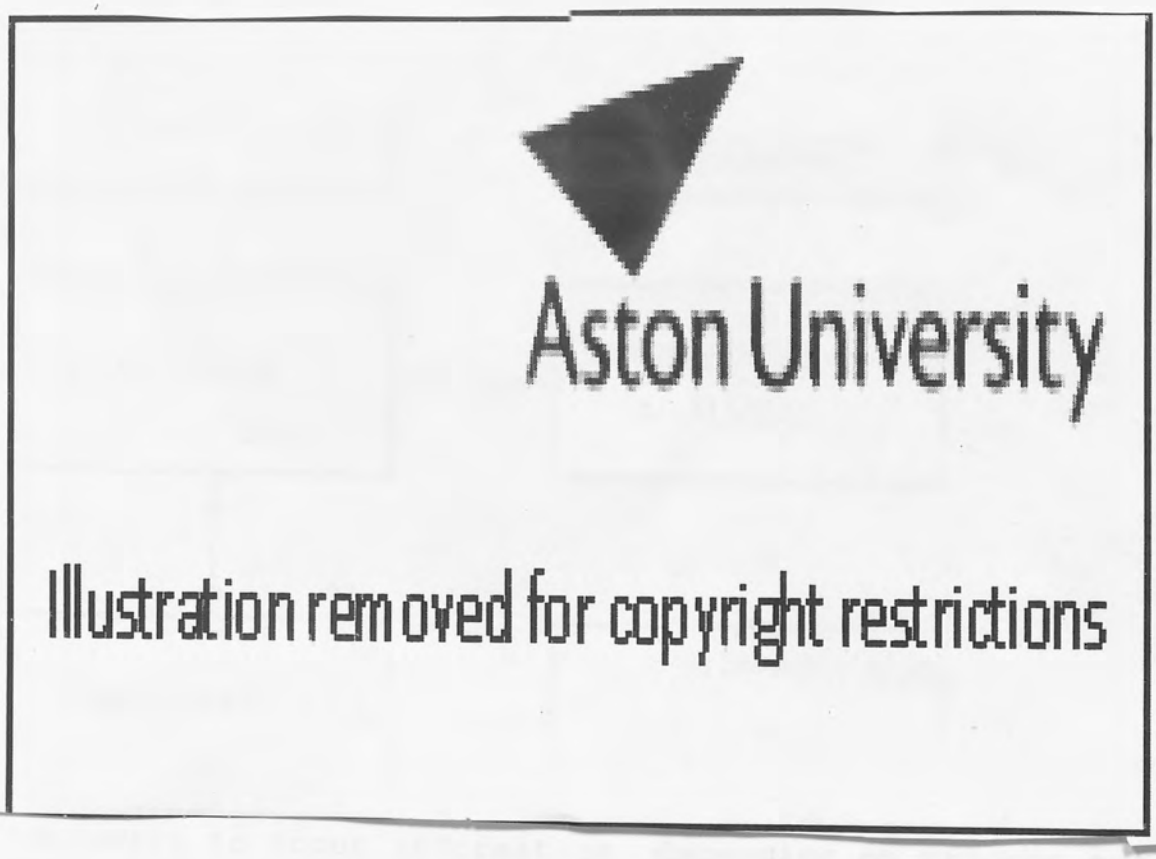


Fig 2.3 Application of computers to design process according to Reed [23].

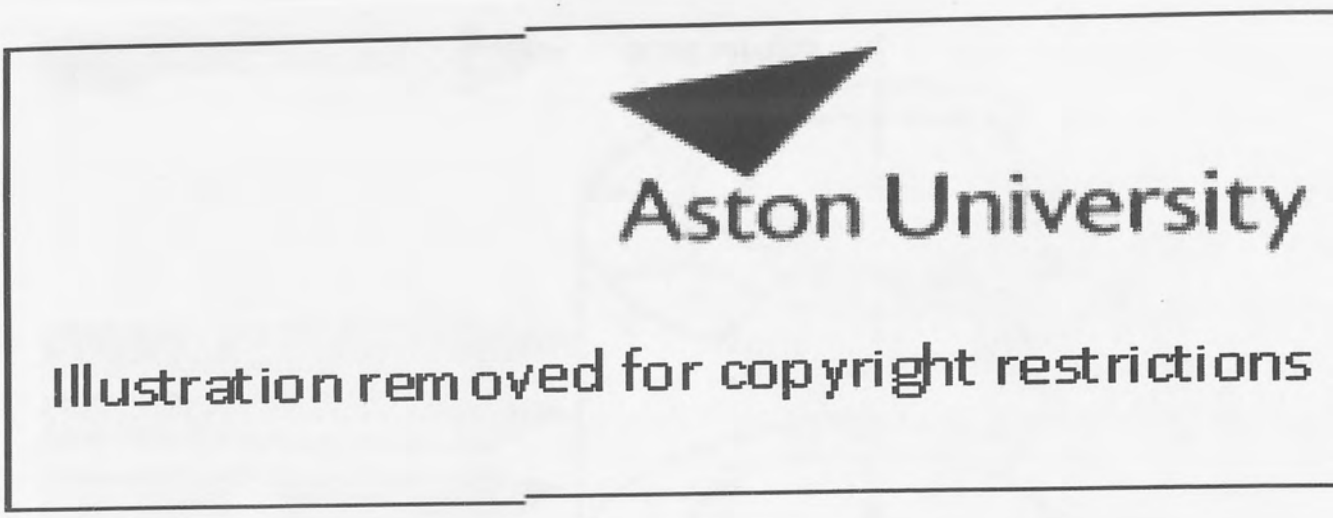
2.8.1 Design analysis software.

Every user has specific design calculations peculiar to their needs and it is known that many are suitable for direct interactive use with CAD systems. In most systems, an interface is provided to various high level languages such as Fortran. With the growth in capacity of both the main frame and the mini/micro computers, considerable scope will exist for more extensive analysis routines to be used and hence more complex problems to be handled. This in turn will generate complexities in the use of software and understanding of these tools and so, in effect, the software has to be more sophisticated and robust and also allow the user (that is, the designer) to understand the process and methods being adopted. To this end the extension of CAD systems to incorporate the values of specialised systems is needed [23]. These specialised systems are described in some detail in the next chapter.

2.8.2 Computer applications to design.

Fig 2.4 shows a view of the design activities as conceived by Sabin, [24] as proceeding through a series of identifiable core phases bounded by the product specification, commencing with the identification of a market need. To carry out these phases in practice, it is necessary to input information, depending on product area. The information obtained from various sources is not in the form that a computer can use, so techniques are needed to aid its manipulation into forms relevant to the need of the

product. It should be noted that inputting information is a dynamic activity that is being represented in a static model here. The representation of information inputs and techniques means that any one of them can be used if and when necessary and does not mean that they are inputted at discrete points. As has been stated earlier, design is a complex and dynamic activity involving iterations and a high degree of interaction between various elements. These interactions and inter-relationships are examined critically in later chapters as they are a "grey area" which, as shown in the literature review, other researchers have largely avoided when trying to apply the type of specialized programs mentioned earlier to engineering design. Smith [25] viewed design activity as one of information management. This is partly correct but this definition does not adequately present a true picture of design activity. The heuristic nature of some of the information and the iterative nature of the process cannot be appreciated when viewed in this respect. If Fig 2.4 is not viewed from the



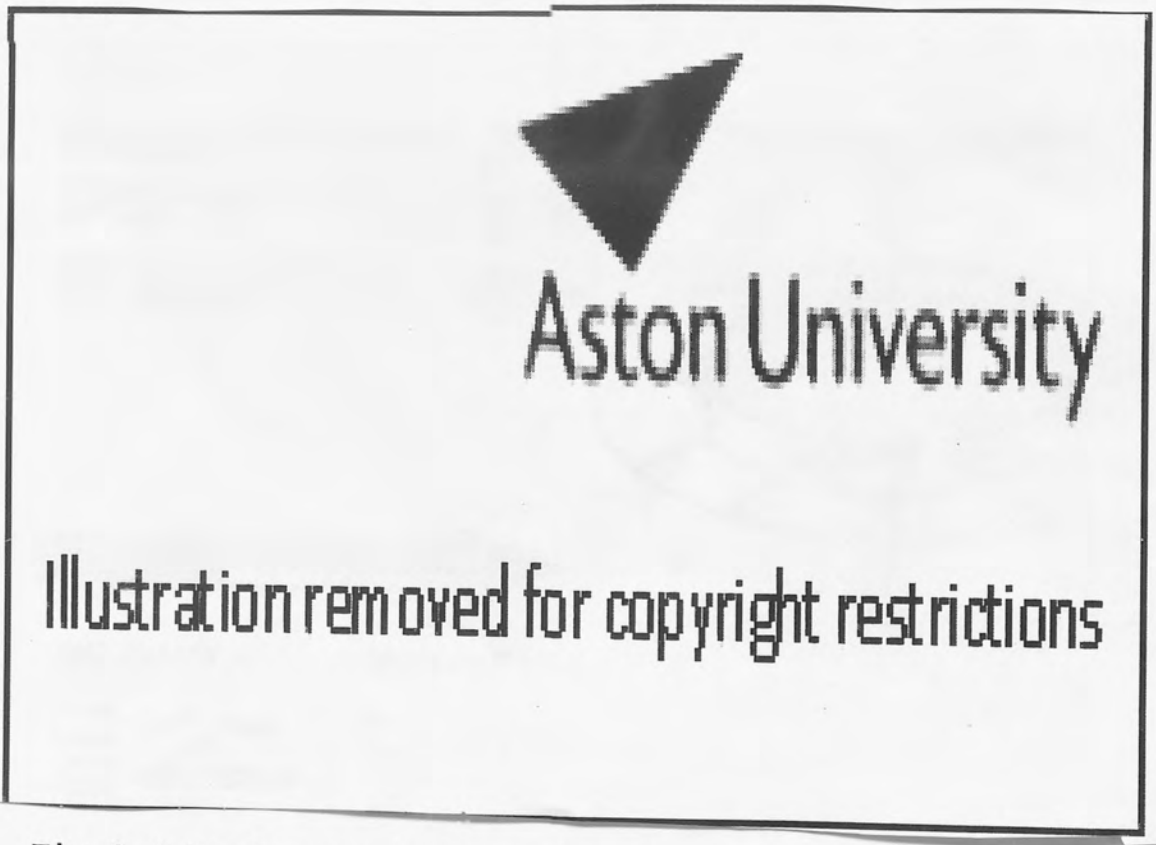


Fig 2.4 Design activity model showing information transfer according to Sabin [24].

Aston University

Illustration removed for copyright restrictions

Fig 2.5 Likelihood of computer useage according to Sabin [24].

point of view of computer application (Fig 2.5), then information transfer can be considered in two main divisions. Firstly, there is information which has been "captured" in the sense that it has been gathered, sifted and formed into a useful body of knowledge. Such information is in the form of published papers in the technical literature, codes of practice, etc. In addition, designers also need information which is not captured, and often necessitates a considerable amount of initiative to obtain it.

Now, for a computer to be considered useful, it must be programmed in such a manner as to become a tool to replace a skill normally exercised by a human. At present, existing computer design analysis software can only be used if the input data and procedures are completely specified. This disadvantageous aspect can be overcome by the use of specialized programs such as Expert Systems as will be demonstrated in Chapters 6 and 7. At the present time the actual input and the interpretation of the output remains the preserve of the designer. This state of affairs may not continue for long as more and more design software incorporates Expert System features. Computer programs are not creative in bringing together elements in new combination, nor in rearranging a set of components in a more suitable form, but with the advent of Artificial Intelligence techniques this inability can be said to be only temporary. In essence, if computers are to be used to support design decision-making, a clearly structured

understanding of the process is necessary in order to allow a well-defined division of labour between man and machine and also, to provide computer aids in a manner that is relatively general to be used for most types of problems. This division of labour now exists and was made clear by Nowacki [26].

Currently, CAD systems are applied to the area of design analysis and evaluation but not to problem definition and clarification areas. Extending CAD to include these areas will require the full exploitations of Expert Systems concepts. Warman [27] considered that the extension of Artificial Intelligence in general to computer-aided design is a natural and expected development. Simmons [28] is of the opinion that Artificial Intelligence techniques will make significant contributions to engineering design.

Having discussed the design activity, in the next chapter, the current state of Expert Systems techniques is discussed with a view to identifying their potential for application to component design. The application of Expert Systems to Engineering design is considered as an extension of CAD requiring improvement in design software.

CHAPTER THREE

EXPERT SYSTEM METHODOLOGIES

CHAPTER THREE

EXPERT SYSTEM METHODOLOGIES

3.1 Introduction.

It is important first to define what Expert Systems are in order that their potential application to Mechanical Engineering design can be appreciated; earlier researchers in the field tended to define the concept from the point of view of their domain of interest. Also early Artificial Intelligence language development (for example, List Processing Language; LISP) was appropriate to natural language applications.

Branchman et al [29] defined an Expert System as a computer program that has EXPERT rules, avoids blind search, performs well, reasons by manipulating symbols, grasps fundamental domain principles, deals with difficult problems in a complex domain and can reason about its own knowledge.

From the above definition, it is evident that his area of interest involves the use of probabilities and hence the search process may often require blind searches.

Other Artificial Intelligence researchers have a more simplistic view of an Expert System. They defined an Expert System as a computer program that should be able to mimic a

human expert in decision making tasks. This definition is somewhat ambitious in that human thinking and behaviour is very complex and not well understood. In that sense, what is expected of the system cannot be well defined. Waterman [30] defines Expert Systems as computer programs that have the ability to use knowledge, can exploit Heuristic concepts, incorporate inferential processes and can effectively manipulate large knowledge bases.

However, the committee of the British Computer Society specialist group on Expert Systems defined such systems as "The embodiment within a computer of a knowledge based component from an expert skill in such a form that the system can offer intelligent decisions about a processing function". It is the opinion of the author that the definition given by Waterman [30] should be expanded to include the requirement that the system be able to explain, in a logical manner, its reasoning process during and after the consultation. The explanations may be in either graphical, numerical or grammatical forms.

The next section describes the classes of Expert Systems before discussing the attributes expected of such systems. The systems are then compared with conventional programs (that is, formal logic programs).
The types of knowledge representation techniques required (section 3.3) and the knowledge representation systems available (section 3.4) to the author at the time when this project began are presented at the later part of the

chapter.

3.2 Classes of Expert Systems

Expert Systems are knowledge based systems. They use facts, rules and relations (the knowledge base) to infer or deduce solutions. Knowledge, according to the Oxford Illustrated dictionary, is defined as "familiarity gained by experience" and can also be defined as being " a person's range of information; or theoretical or practical understanding of a subject area". The process of acquiring expert knowledge from human experts or other sources (such as technical papers) is termed as Knowledge Engineering. Most Knowledge Engineering applications fall into distinct types: prediction, diagnostic and selection [31],[32].

Prediction systems deduce likely consequences from given situations. This group includes weather forecasting, traffic prediction, military forecasting, mineral exploration etc. A prediction system typically employs a dynamic model incorporating parameters, the values of which are fitted to a given situation. The system may use probability estimates to generate a large number of possible options.

Diagnostic systems use observations in an elucidation of behaviour. In this group are medical, electronic, mechanical and diagnosis software. Such systems relate observed symptoms to underlying causes and provide guidance

in identifying actions to overcome some defect in a machine, or in formulating treatment in a medical case.

Selection systems use specified requirements to deduce the type of object that will match the necessary requirements. Material selection falls into this grouping.
Beerel [32] proposed a fourth group concerned with design and optimisation tasks. It is only recently that research has been done in this area as discussed in section 4.2. The work presented in this thesis falls into this grouping.

In the following sections of this chapter, a background to Expert System concepts and their applications is presented in preparation for the work described in chapters 6 and 7. It was considered essential that the various knowledge representation techniques, shells and languages should be appreciated and the manner in which the concepts have been applied be understood.

3.2.1 Attributes of Expert Systems

Having described in the last section what Expert Systems are intended to be, we now consider the attributes they should possess particularly in applications to mechanical engineering design.

A distinguishing feature of most Expert Systems is that they have a domain specific knowledge base separated from a

problem solving methodology, the latter being embodied in a so-called inference Engine. This structure allows for the addition of new knowledge and the removal of obsolete facts if and when necessary. However, there are some exceptions were the knowledge base cannot be clearly separated from the inference engine. An example is an Expert System written in a language called Prolog (which is described in section 3.4.3). The inference engine is built into the language.

The knowledge base comprises the "working" knowledge and the "permanent" knowledge. The working knowledge is gathered by the Knowledge Acquisition Module which is responsible for the "elicitation of information" (by asking questions); this knowledge is temporary unless a conscious decision is taken to convert it to permanent status. The permanent knowledge is encoded into the system by the knowledge engineer; this comprises static and dynamic knowledge. The static knowledge consists of rules, equations, conditions, codes etc., whilst the dynamic knowledge is basically a temporary databank.

The dynamic knowledge should be expandable, (for example, knowledge of different kinds of material available for selection), modifiable, and allow the latching on of different knowledge modules special to specific areas. It is essentially a database.

The inference engine controls the way in which the system goes about solving a problem; it is part of the system's

knowledge. The main function of the inference engine is to plot the solution path using the information available from the knowledge base, avoiding contradictions in the reasoning and explanatory processes and to be able to predict the effect of adding new information to the knowledge base. It should be user "transparent", that is, the user should readily be able to understand the strategy. Thus, in the course of its problem solving process, the system should be able to explain WHY information is needed to complete the line of reasoning and HOW a conclusion was arrived at.

One of the most important attributes of these systems is the provision of Help facilities at convenient points in the solution paths. The "Help" given is usually in the form of more information as to what is expected from the user by the system in order to continue the consultation. Not all Expert Systems have this attribute; induction systems such as Expert Ease or TIMM do not to have such facilities.

An Expert System should be able to reason under conditions of uncertain and insufficient information and should be capable of probabilistic reasoning, commonly using Bayes' theorem [33]. This attribute is normally incorporated into systems such as Mycin, Prospector or Micro Expert used for solving prediction and diagnostic type problems. Mycin was developed by Shortcliffe, to diagnose blood infections and meningitis in patients and recommend drugs for the treatment of the disease; it was the first recognised rule

based system [34]. It uses so-called production rules (section 3.3.3). Prospector was developed to predict the location of minerals, by the update of the weighting attached to each item of information given at each stage of the process. The method used in this system is considered by Naylor [33] as the best worked out method of any Expert System of its time (1983). Some researchers, for example, Konopasek and Jayaraman [35], consider that an Expert System should think the way a human expert does. This may be one of the goals of an Expert System developer but in practice it borders on the impossible since the manner in which human experts think is too complex to be fully understood.

Expert Systems should be capable of learning from experience. This attribute is most prominent in induction type systems because they are built to develop their rules by example and so update their rules automatically as they "gain experience" with various types of problems. For most other types of system, this attribute is not available.

Lastly but most importantly, the system should be useful. The purpose of the research into the development of Expert Systems is to encode expert knowledge and hence free human experts for more pressing needs and also make their expertise available to more people. If this purpose is not fulfilled, it is doubtful if the system could be accepted as an Expert System.

3.2.2 Comparison of Expert Systems and Conventional Programs

The usefulness and the attributes of Expert Systems are best appreciated if compared with the use of conventional programs such as Fortran or Basic in the solution of problems. A conventional program basically manipulates data but an Expert System should also be able to manipulate knowledge [30]. An Expert System puts more emphasis on symbolic manipulation rather than mathematical computations. The result of this approach is that knowledge representation, comprising the choice, form and interpretation of the symbols used becomes of paramount importance. Uzel and Button [36] pointed out that Expert Systems turn knowledge into some form of expert function rather than turning data into information. Expert Systems can accept a problem stated in an arbitrary manner and convert it to a form that lends itself to a fast and efficient solution. Expert Systems have depth; that means they can operate effectively in a narrow domain containing difficult and challenging problems.

One of the most significant differences between an Expert System and a conventional program is that the latter is designed to give a "unique" correct answer alway, whereas the former are designed to give reasonable answers which are not necessarily "unique" nor "correct".

Finally, when a conventional program designed for performing complex tasks makes a mistake, the error may be very difficult to trace since the strategy, heuristics and basic assumptions upon which the program is based are not necessary explicitly stated in the program code. Other technical facts that distinguish knowledge representation schemes from programming languages are discussed below.

In conventional programs, facts are only used during the execution of some procedure whereas a knowledge base consists of facts that have multiple uses including reasoning, data and information access or problem matching. In an Expert System new facts are deduced from old ones in accordance with some laid down rules. At times, specialized procedures can be used to derive facts in a fixed way by (what is called) deductive reasoning. There are however, other kinds of reasoning processes such as inductive and abductive ones. The storage of and access to information in a Knowledge Base for question-answer purposes is another significant difference between knowledge representation systems and logical programming languages. The knowledge base is examined in order to know the current state of the process and hence determine what sort of question is appropriate to elicit information or the type of answer which is appropriate to a given query. Using facts for problem matching is a knowledge based operation which has a variety of uses including classification, correction, decomposition etc. In Expert Systems, a knowledge base cannot be considered to be necessarily complete whereas in

logical programming languages, this is so. The knowledge possessed by a logical programming language is used for a specific "narrow" purpose. In Expert Systems the task to be performed varies considerably within the domain of interest and it is impossible for the knowledge to be considered complete.

Also, it must be noted that technological advances are being made all the time and hence the knowledge required for a system to perform to current expectation will continue to change for that specific domain of interest. Levesque [37] dealt with this aspect in detail and argued that a language that can refer to both the application domain and the state of the knowledge base is required to specify and question an incomplete body of knowledge. He further stated that a knowledge base should thus be viewed as an incomplete and approximate model of a world which can always be improved upon. This proposition leads to design methodologies for Knowledge Based Systems that are drastically different from those of logical programming.
In formal logic programming languages, the designer starts with a clear idea of the algorithms he wishes to realise and proceeds to construct a complete design, but in Expert Systems, the knowledge base is developed over a period of time (years in some cases) [38] using different knowledge acquisition techniques that range from interactive sessions with the expert to automatic generation of new facts. In the research reported in this thesis, the knowledge was mainly obtained from papers and text books.

The last major difference between a logical programming language and knowledge based systems is the attribute of self-knowledge (or meta-knowledge). A system knowing about the state of incompleteness of its knowledge can be considered to possess a type of self knowledge, as can the making of facts available for question-answer or the reasoning process itself. However, the most common self knowledge attribute is the knowledge enabling the system to answer elementary questions about its actions or the strategies it uses for problem solving.

We now proceed to consider the types of knowledge representation schemes.

3.3 KNOWLEDGE REPRESENTATION TECHNIQUES

3.3.1 Types of Knowledge Representation Techniques

The selection of a scheme for knowledge representation affects the development of the resultant Expert System and is one of the most important and difficult phases in the building of a complete system. This was pointed out clearly by Waterman and Hayes-Roth [34] in their report on the investigation of eight Expert System shells and languages. The result of this investigation concluded that the knowledge engineer should test his tools early by building a prototype and also avoid choosing a tool with more generality than is needed. By conducting this investigation

the importance of selecting the appropriate knowledge representation scheme was highlighted. This section presents the different types of knowledge representation schemes which have been developed.

In Knowledge Based Systems, knowledge is represented as a combination of data structures and interpretative procedures that, if used correctly in a program, will lead to knowledgeable behaviour. Research on knowledge representation in Expert Systems has involved the design of several classes of data structures for storing information on computer programs, as well as the development of procedures that allow intelligent manipulation of these data structures to make inferences [31].

Techniques and theories about knowledge representation have undergone rapid change and development in the last ten years. The most important consideration in examining and comparing knowledge representation schemes is the suitability and usefulness of the representation scheme for problem solving in the domain of interest. The goal of knowledge based systems can be described in terms of cognitive tasks like recognising objects, answering questions and manipulating robotics devices, but the actual use of the knowledge in these programs involves three stages:

a) acquiring more knowledge
b) retrieving facts from the knowledge relevant to the

problem at hand.

c) reasoning about these facts in search of a solution.

These three aspects will be discussed in more detail in Chapter 5 as they are of fundamental importance to the working of a developed system.

The schemes presented are procedural (section 3.3.2), declarative and frames (section 3.3.6). The other schemes are production rules (section 3.3.3), semantic nets (section 3.3.4) and logic (section 3.3.5), all of which are grouped under procedural or declarative schemes. It will be seen, in due course, that the present work utilized semantic nets and frame knowledge representation schemes.

3.3.2 Procedural representation.

In a Procedural Representation, knowledge about the domain is contained in procedures - small programs that know how to do specific things, how to proceed in well-specified situations - that is, it is a collection of processes.

Mylopoulos and Levesque [39] identified two possible classes for this representation scheme. The first involves an activation mechanism offered to the processes. In this case, the knowledge base is considered to be a global database of assertions. Associated with this global database are theorems (or Demons) which are triggered off whenever there is a modification of the database or a search through the database. A Production Knowledge Representation scheme comes under this classification and

is described in more detail in section 3.3.3.
The next class is characterized by the control mechanism available. This area will be discussed further in the next section but it is worth noting that LISP control formalism [39] belongs to this class; it was rejected as a possible method to use for this present work because it uses only a Backtracking strategy which is unsuitable for the nature of the problem domain involved. (See Appendix A2).
A major flaw in this representation system is that it is difficult to verify and change. Nevertheless, most knowledge based systems use a Procedural Representation at some level of their operation.

3.3.3 Production systems

The basic idea of these systems is that the database consists of what are called production rules in the form of a condition-action pair: IF this condition occurs THEN do that action [31].
The rules are considered in a predetermined order until one is found for which the condition requirement matches the state. This rule is then triggered and the action executed.
The utility of the formalism comes from the fact that the conditions under which the rules that are applicable are made explicit and, in principle at least, the interaction between rules, are minimised (one rule does not call another).

Production systems have been found useful as a mechanism

for controlling the interactions between statements of deductive and procedural knowledge.

The nature of the representation scheme makes it easy to understand and modify since the knowledge base is considered as a collection of loosely coupled rules. This scheme also has the advantage of allowing the specification of direct interaction between facts, thus eliminating the need for wasteful search.

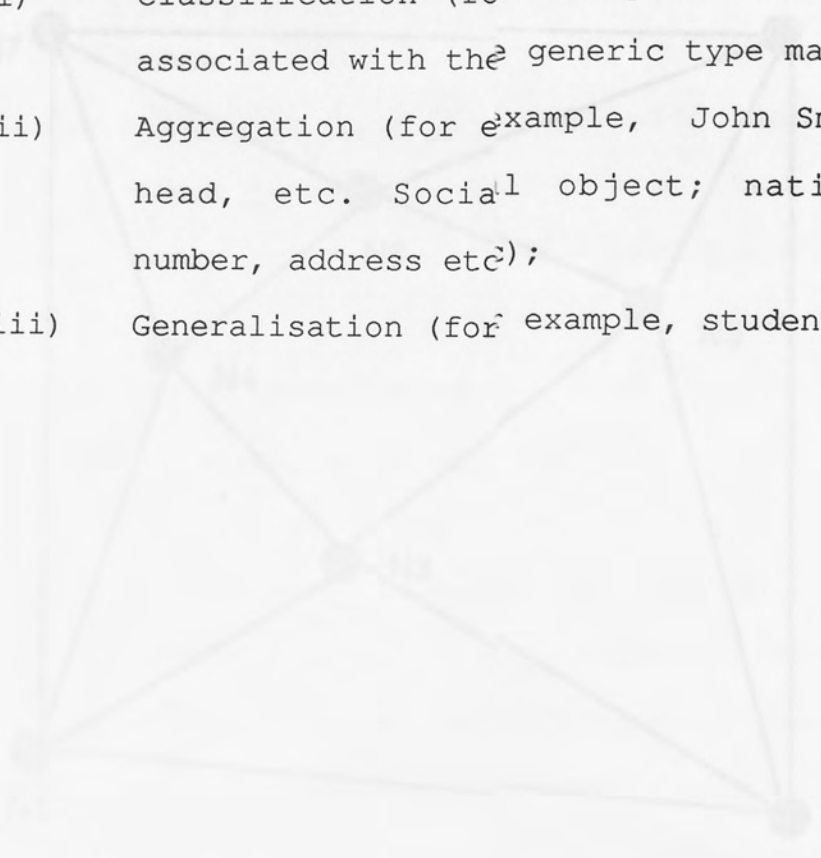
3.3.4 Semantic nets

Semantic nets consist of nodes, representing objects, concepts, states, or events and links between the nodes representing their inter-relations. For example, consider Fig 3.1, the equation $p = \pi d/z$ is represented by the node N1 and the equation $m = d/z$ is represented by the node N3. The relationship between these two equations (and therefore nodes) are the variables z and d. If a third equation ($p = \pi m$) is represented by the node N3, the relationship between the nodes N3 and N1 will be the variable p. The same interpretation applies to the other nodes. Hence, in this example, the network of nodes represent a cluster of knowledge provided by the equations. The knowledge base can be modified by the deletion or insertion of a node (or object) and the alteration of relationships. Mylopoulos and Levesque [39] criticised the early versions of semantic nets for not possessing a clear formalism. They also discussed semantic nets in the contexts of :

(i) Classification (for example, John Smith should be associated with the generic type man);

(ii) Aggregation (for example, John Smith: Part; arm, head, etc. Social object; national insurance number, address etc);

(iii) Generalisation (for example, student, person etc).



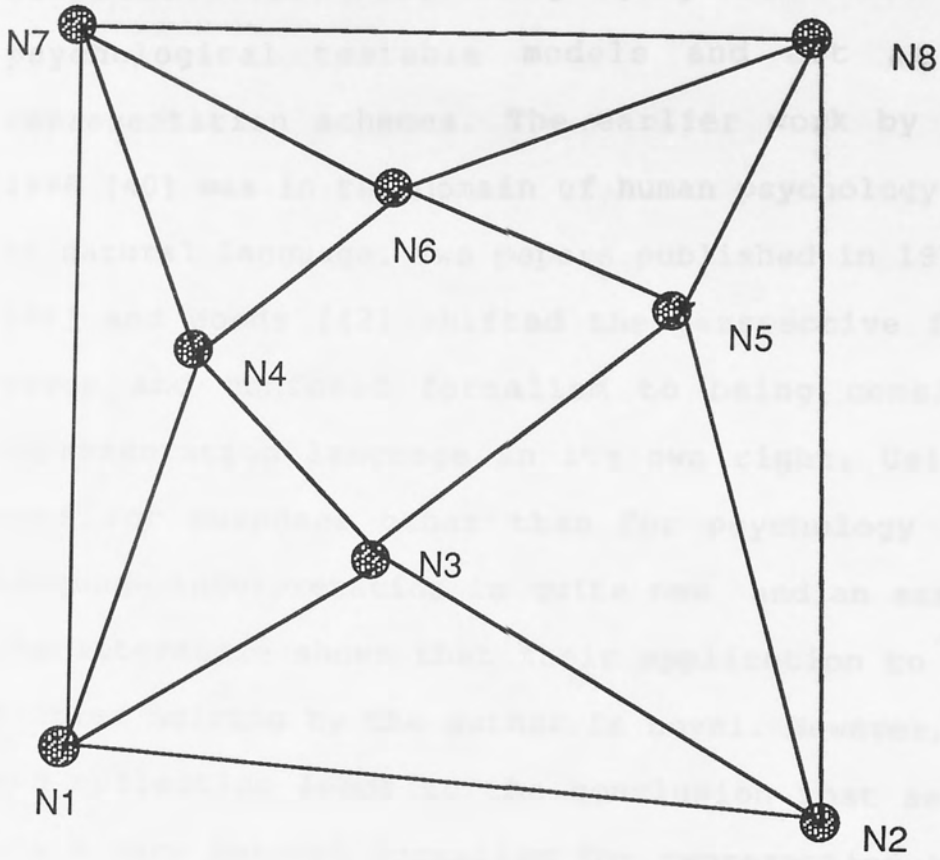


Fig 3.1 An example of semantic net

Semantic nets were originally intended mainly as psychological testable models and not as knowledge-representation schemes. The earlier work by Quillian in 1968 [40] was in the domain of human psychology and the use of natural language. Two papers published in 1975 by Minsky [41] and Woods [42] shifted the perspective from being a vague and confused formalism to being considered as a representation language in its own right. Using semantic nets for purposes other than for psychology and natural language interpretation is quite new and an examination of the literature shows that their application to engineering problem solving by the author is novel. However, some study and reflection leads to the conclusion that semantic nets are a very natural formalism for representing knowledge in the domain of engineering design.

The interpretation (semantics) of the net structure depends solely on the program that manipulates the network; there are no conventions about their meaning. To illustrate this, Fig 3.1. Instead of each node representing an equation, suppose each node represents a variable. Let node N1 represent the variable p and node N2 represent the variable z. Therefore, the relationship between p and z (hence nodes N1 and N2) is given by the equation $p = \pi d/z$. Also if the third node N3 represents variable d, the relationship between d and z (or nodes N2 and N3) will be equations $p = \pi d/z$ and $m = d/z$. Therefore, inferences drawn by the manipulations of the net are not assuredly valid, in the

sense that they are assured to be valid in a logic-based representation scheme. That is, the representation technique when applied to engineering design may not come up with the optimum solution all the time but most of the time. However, an important advantage of semantic network representation is the ease with which knowledge can be stored and accessed in question-answer situations [39]. This is because there can be links between any number of nodes. Some schemes such as production rules, are poor at this. Another important advantage is the freedom with which the knowledge base can be organised. Thus the nodes need not be arranged in a pre-determined order since the nodes call each other and direct the process.
The scheme can be represented graphically; this is valuable in applications to engineering design to ease the explanation and understanding of the reasoning process. The associations between concepts and formulae can easily be communicated.

A major disadvantage however, is the lack of a standard terminology. Isreal and Bachman [43] discussed in depth the deficiencies of semantic nets but, as with most authors, the discussion was in the context of natural language interpretations.

Semantic nets were chosen as one of the main knowledge representation schemes in this research. The reasons will be discussed more fully in Chapter 5.

3.3.5 Logic

The most important feature of logic and related formal systems is that deductions are guaranteed correct. No other representation scheme possesses this attribute. The interpretation of a set of logic statements (that is, the set of inferences or conclusions that can be drawn from those statements) is completely specified by the rules of inferences. Theoretically, the database can be kept logically consistent and all conclusions can be guaranteed correct. This scheme employs the notions of constants, variables, functions, logical connectives etc., in order to represent facts and logical formulae which partially describe the state of affairs. Modification to the knowledge base in this respect can be achieved by the insertion or deletion of formulae or equations. Therefore the formulae are the primitive units for manipulation.

The scheme allows facts to be represented once and independent of its uses. It has the advantage of having a well understood and clear formal interpretation of its meanings. It is however, not suitable for knowledge acquisition, and interpretation of beliefs etc.

A disadvantage of this scheme is the lack of organisational principles for the group of units that constitute the knowledge base. Procedural and heuristics knowledge cannot be represented. The language called Prolog was developed with the aim of integrating the virtues of logical and procedural representation.

3.3.6 Frames

The most recently introduced Artificial Intelligence knowledge representation scheme is the frame which is still in its early development stage [44]. Basically a frame is a database structure that includes declarative and procedural information in the form of predefined internal relations. An interesting feature of frame based processing is the ability of a frame to determine whether it is applicable to a given situation. The idea is that a likely frame is selected to aid in the process of understanding the current situation (dialogue, scene, problem) and this frame in turn tries to match itself to the data it discovers. If it finds that it is not applicable, it could transfer to a more appropriate frame. Therefore, a frame can be considered as a system which represents knowledge in a very narrow domain, it is a complex data structure for representing a stereotypical situation. Suppose in the given example in section 3.3.4 (semantic nets), equation node N3 is of further interest, that is, what assumptions were made in its derivation, when should the equation be used and within what range of values should the two variables be. A sub-program that specialises in this narrow domain can then be developed. This sub-program is called a frame. The frame can use any type of representation techniques that may be suited to the application. As a further example, consider Fig 3.2 which shows the relationships between four nodes labelled F,M,S, and D representing father, mother, son and daughter respectively. Suppose the behaviour of one member of the family is of interest, say the mother (node M).

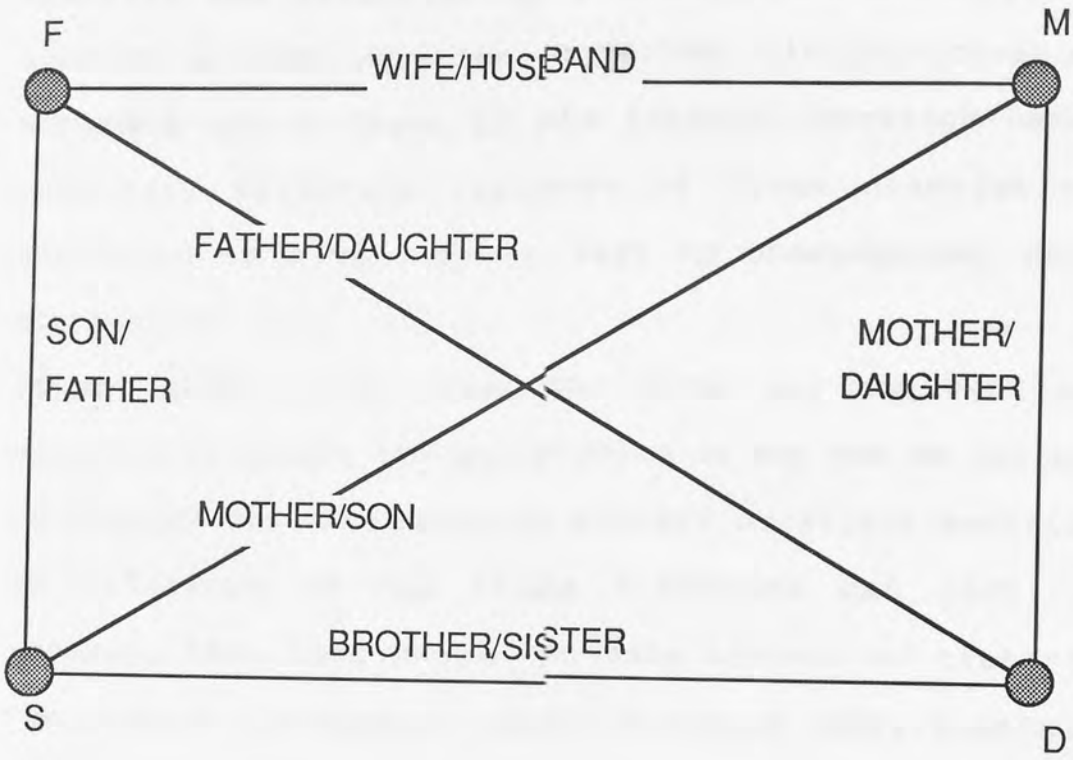


Fig 3.2 The relationships between family members.

A rule based sub-program can then be developed specialising in the actions of the mother under certain conditions. This sub-program within a main program represents a frame that specialises in this narrow domain.

Minsky [41] proposed the theory of frames; it provided a framework for developing representation schemes that combine ideas from procedural and declarative schemes.

Kuipers [45] stated some basic properties of frames. He

stated that frames provide some elaborate structures for creating and manipulating a description whose primitive element (a node, equation or variable in this case) can be expanded into a frame if the internal workings become of interest. Different features of frame description are justified in a variety of ways by observation, default, alternative etc.

If anomalies occur, then the frame may not be the most suitable to handle the sub-problem or may not be correct.

If changes in the reasoning process or slight modification in inference of the frame structure can rectify the anomaly, then this should be done instead of transferring the problem to another frame. Winograd [44], examined the essential features of declarative and procedural representation schemes and suggested ways in which frame representation can cater for the requirements of both types of schemes.

The part these representation schemes played in the consideration and selection of an appropriate tool for the implementation of a prototype system will now be discussed. Also, the types of Expert Systems tools (that is, shells and languages) available to the author at the time when the project was begun will also be discussed.

3.4 EXPERT SYSTEM TOOLS: (SHELLS AND LANGUAGES)

Expert System tools are programming systems that have been developed to simplify the work of building Expert Systems; they are devoid of knowledge about the particular domain of interest [46]. Five shells were available and assessed at the beginning of this project. Subsequently, a variety of others have been introduced to the market place, but it was not practicable to consider changing from one to another as they emerged, even if they were, perhaps, superior in some ways.

The assessment was carried out in the light of the desired applications to the solution of mechanical engineering design problems. The systems examined were Expert Ease, ES/P Advisor, Microexpert, Microsynics, TIMM, and Sage. Sage was the only system available on the mainframe at the time.

Amongst the reasons for the development of shells are [31]:

(i) minimising the time given for interactive transfer of knowledge into the knowledge base. It can be said that current shells in general have not been successful in accomplishing this task. Indeed, the restrictions in certain systems, such as SAGE, reduce the prospect of doing this [47]. However, this is a rapidly developing field and these remarks are becoming less valid.

(ii) the principal bottleneck in the transfer of expertise is considered to be the knowledge engineer and most shell developers believe that this intermediary should

be eliminated. Eliminating the role of a knowledge engineer cannot be achieved with the current range of shell packages, however.

In attempting to achieve the two stated objectives, Expert Systems Shell developers tend to approach the task from a rather generalised viewpoint. That is, the shell systems are developed to cater for as many domains as possible so that the special requirements for a given domain may not be well satisfied. If the development of shell systems is approached from a specialised angle and for a few specific domains, users will find the systems more useful and easier to use. The practice so far has been to develop systems on the basis of the knowledge representation technique adopted. This has the effect of restricting an Expert System builder to the use of one representation scheme instead of some combination of various techniques. It is the opinion of the author that human experts use different types of representation techniques without being aware of it.

3.4.1 Types of Expert System Shells.

The shell systems discussed in this section represent a variety of knowledge representation schemes. An assessment of these systems with some others gave a good indication of what representation scheme to use or what combination of shell attributes were suitable for the purposes of engineering design.

ES/P Advisor is an Expert System shell program that uses forward reasoning only (Appendix A2) and cannot handle uncertain knowledge [48]. ES/P Advisor is a special purpose advice system which operates on text only. Hence it has no direct engineering application. The system consists of a knowledge base which is written in a special purpose language called KRL (Knowledge Representation Language); the KRL program compiles the knowledge source file. A menu of the existing knowledge source files is listed for the user to choose from, syntax and logical checks are made on the source knowledge, and a Prolog file is produced. Numbers given to it may be integer or real, but integers cannot have negative values. Scientific notation is acceptable in the knowledge base but cannot be supported by the consultation module; the latter is written in Prolog and can have external Prolog routines to extend its power. The consultation shell can explain its line of reasoning; it has the ability to retract information already given and to accept information volunteered by the user before it is requested by some prompt statement. ES/P Advisor could be a useful tool for the automation of standards, procedures and regulations but it is inadequate for giving advice in solving numerically based engineering design problems.

Expert Ease creates its knowledge by the method of induction from examples [49]. It has a very elementary facility for representing static knowledge. The rules created are explicit, so that no uncertainty is allowed in the data. The system is forward reasoning and no

explanation facility is provided in the consultation module nor can one be designed for the shell. The lack of explanation facility makes the shell of little use for solving complex problems.

Values given to the system are restricted to logical (for text strings) or integer (for numerals). The integer value must be between -32767 and 32767. Expert Ease is based on the Pascal language; it is interactive and menu driven, and requires a fair number of examples to induce good rules. It is a good example of a system using the induction method to generate its rules, but it is not suitable for most engineering applications because it only supports integer values and places restrictions on their magnitudes.

TIMM also falls into the same category as Expert Ease but it does have the added capability of calling an external Fortran program. How fast the process can transfer from the external routine to the consultation module cannot be easily assessed especially with complex problems. This package was not available at the time the project began but would not have been adopted because it employs induction techniques. Such techniques, as earlier indicated, require examples to form rules. In a design process, most variable values are real (and hence, not integers or discrete). Practically, there are very large numbers of examples of variable combinations that can be given which will satisfy given design constraints. If the object or variable values are discrete, the system will be of limited use. This system is not useful for the purpose of engineering design

applications.

Micro Expert is basically a backward reasoning shell program which can handle uncertain knowledge by using weighting factors in a way similar to that used in the program called Prospector [50]. Production rules are used and although the system has explanation facilities the user interface is very clumsy; the methods used to block and amplify questions are rudimentary. The shell is based on the Pascal language and this gives it the added advantage of having no restrictions on numerical values. The system has the ability to chain another source file (that is, another model), it is not clear however, if the system can chain an external Pascal file or function. Pascal programming language was unavailable on the computer facility used and a practical assessement of this facility was not possible. The system will not be able to handle engineering problems (such as a spring or gear pair design). The modelling of such problems would be very difficult since mechanical design problems require a forward reasoning strategy for some processes, whereas Micro Expert only offers backward chaining. The shell is thus unsuitable as a tool for application to engineering design.

The Microsynics program is not considered by its designers to be a true shell but a system that can be used to connect separately developed Expert Systems [51]. It allows the consultation process to be conducted using a node network

to represent relationships. Microsynics illustrates the idea of a semantic network knowledge representation scheme and it is a good example of how textual dialogue information can be separated from associated tasks like arithmetic algorithms, peripheral handling etc.

The author has dubbed Microsynics a "pseudo language" since it does not have all the in-built facilities required for it to be considered as a true shell nor a fully fledged language. Microsynics is written in Basic language, partly uses production rules and allows dialogue to be conducted in the form of networks of nodes. The two case studies presented in this thesis used Microsynics as the development tool and the discussion below presents the basis on which this decision was taken.

Microsynics makes the modification and maintenance of a complete system simple. Programs written in Basic language can be readily interfaced to it so that it is a very useful tool for modelling numerically based engineering problems. The fact that the system is forward reasoning is an added advantage in its application to engineering, although it can also be modelled in a manner that gives a backward chaining strategy. This dual role increases its flexibility as a development aid. Some disadvantages of Microsynics are: it is incapable of handling uncertainty; whilst it has no in-built explanation facilities, this is not a serious drawback since by the nature of its structure, an explanation facility can be designed into the system to suit the particular problem; there is a limitation on the number of program lines possible dictated by the Basic

language environment in which it operates. The sub-functions are individual programs and are appended to the main program during consultations.

The simplicity of the system itself is an advantage although some Artificial Intelligence researchers argue that it may be too simple in structure to be considered as a true Expert System. The counter argument to this is that an Expert System building tool should be as simple as possible and easy to understand if it is to serve its purpose and justify its aim. Hart [52] stressed the fact that although Synics is conceptually relatively simple, its usefulness is considerable. The system consists of the consultation module and the compiler module.

The Microsynics structure consists of two sections, the dialogue and control section and the subfunction section. In the present work, the dialogue section was used to model the inference system because the user interface is rudimentary; the dialogue part was meant to be used as the user interface but it does not serve this function satisfactorily. The subfunction is used for calculations and it uses the Basic language for this purpose. The type of Basic language environment it uses is MSBASIC running on the Micro Soft Disk Operating System (MSDOS); it was adopted for the user interface because of the flexibility it provides. MSBasic can be understood easily and it is very flexible, attributes which make the package suitable as a development tool for the application of Expert Systems to mechanical engineering design and analysis. It allows design analysis to be broken down into subroutines for the

purpose of calculations and facilitates explanation for each calculation done. Another advantage of the package is that values of calculated or inputted parameters are considered global. Hence unlike most other packages, (SAGE and OPS5) the transfer of parameter values from one subfuction to another is not required. This facility reduces the amount of modelling and constraints that otherwise would have been imposed on the system developer. The control/dialogue part was used solely for the purpose of control as was mentioned earlier and this approach proved to be very powerful. The subfunction section was employed for the user interface but this had the disadvantage of requiring a significant computing effort.
The consultation module of the package was modified to allow it to record the analysis path taken during consultation. This allows the system developer easily to debug a system under development and renders the reasoning of the system transparent to the user. The knowledge contained in the system is thus explicit for the user to understand and decode. However, it is recommended that further modification is required to make the package more valuable for the fast development of Expert Systems in the field of mechanical component design and analysis. These modifications should be to:

a) Incorporate a standard screen control facility into the consultation module. This will standardise the user display and hence allow clarity and consistency in the presentation of information.

b) Incorporate an explanation facility or modify the dialogue section of the consultation module. This will reduce the memory required for system development and the computational efforts necessary.

c) The source code for the consultation module cannot be compiled because of the way it was structured. It "merges" on other files for execution and because of this, basic compilers cannot support this facility. This module should therefore be restructured in a form that can be compiled to improve the speed and response of the system.

When developing a system using this package, the system designer is not allowed to use single characters (that is, A to Z) to represent parameters or their values because these characters are used in both the run-time module and the compiler. System developers are therefore required to use two or more characters. The compiled file is suffixed "D".

There are now three versions of the package. The first version was the original as supplied by the Alvey programme. The other two consist of modifications carried out by the author on the original version as supplied by Alvey. Version 2.0 incorporates modifications to suit the requirements for mechanical engineering design. The third modified version runs in GWBASIC environment, this speeded up compilation of the knowledge base making more testing sessions practicable. It also allows for better screen control and user display. Figs 3.3 and 3.4 gives the block diagram of the third version of Microsynics. This version

was used for the gear pair design, it allows source files to be read from a data file instead of being input from the keyboard as was originally designed.

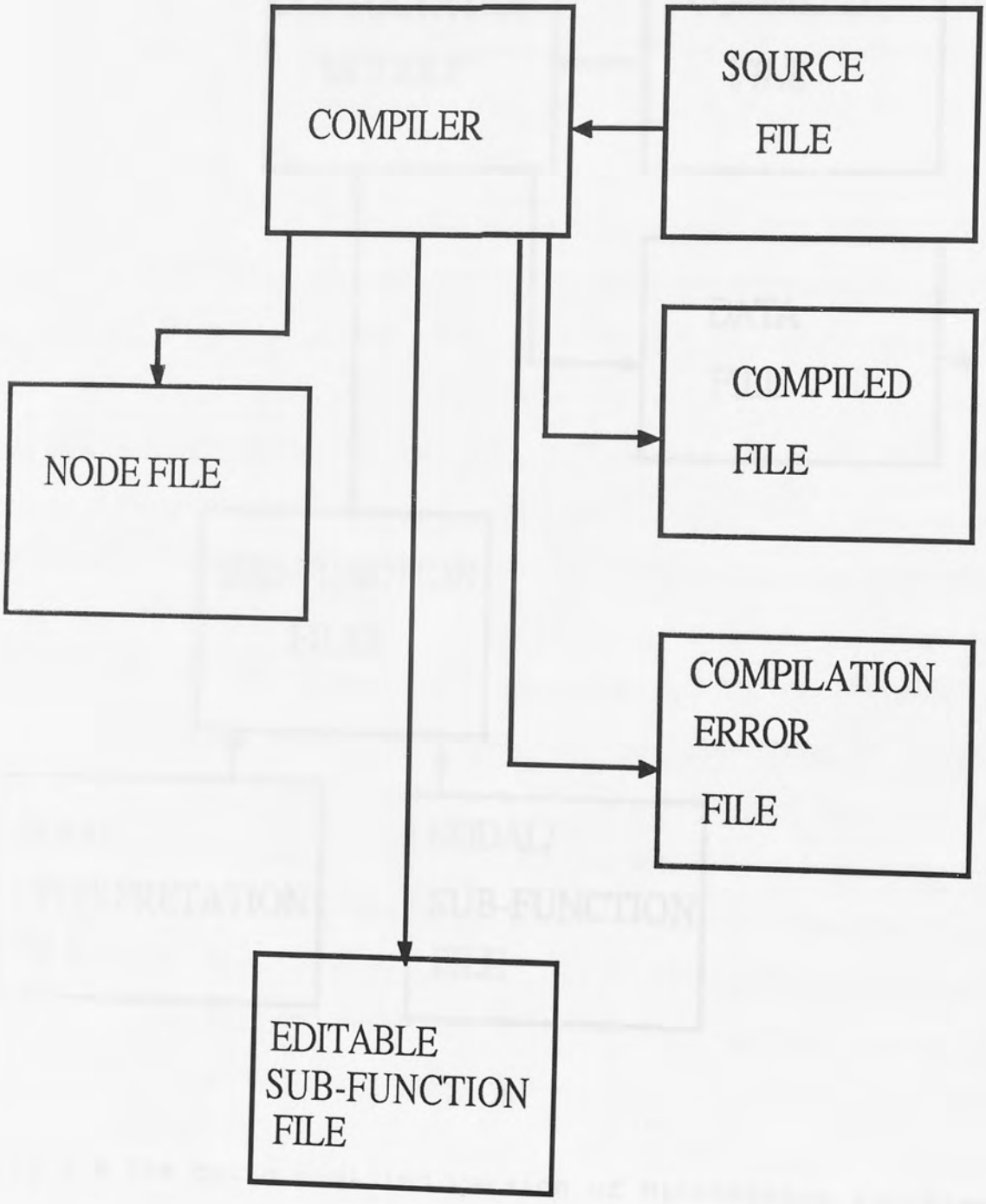


Fig 3.3 The third modified version of Microsynics compiler

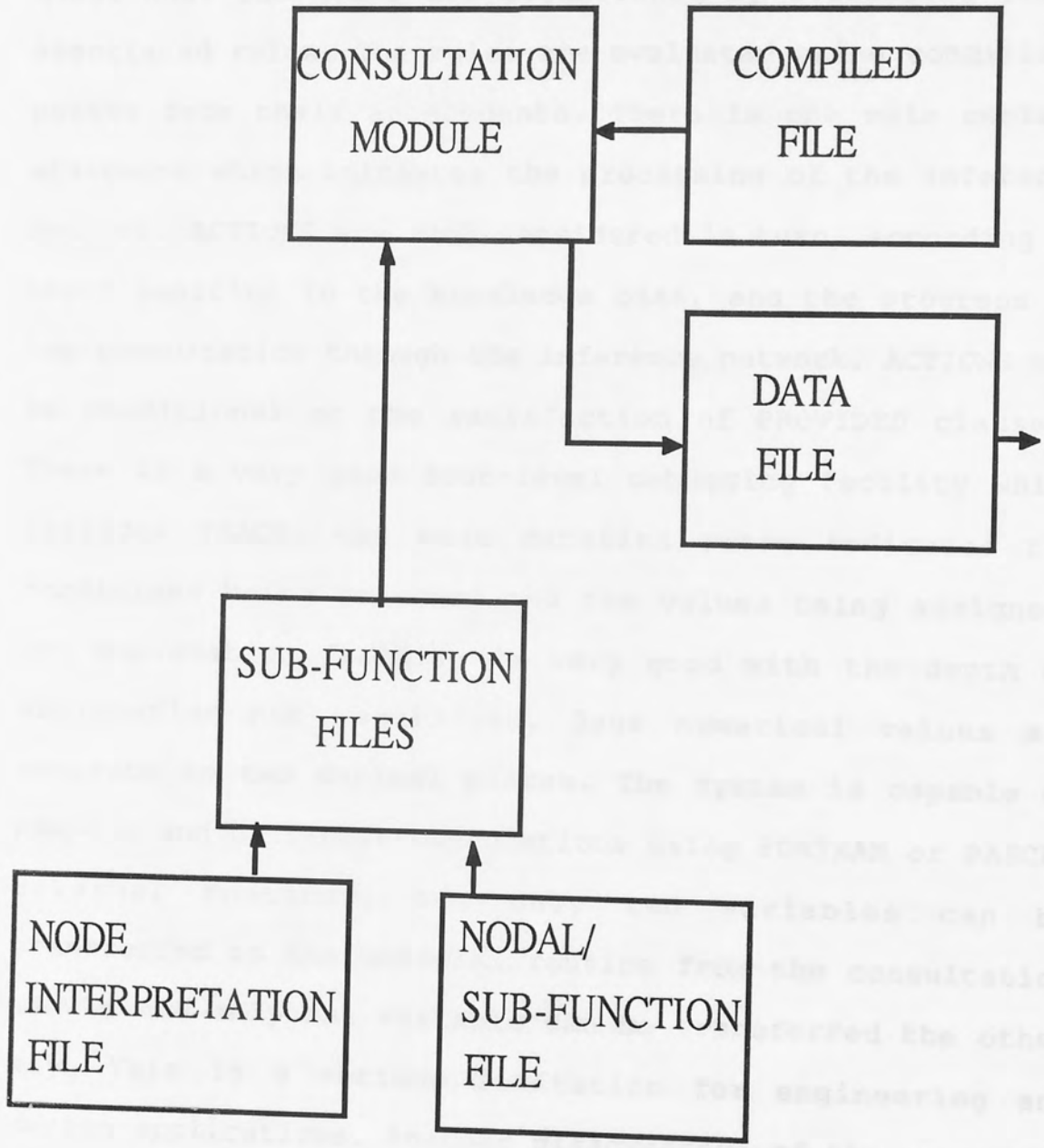


Fig 3.4 The third modified version of Microsynics run-time module

The final system examined was SAGE, a shell based on the Prospector inference network principle [53]. The main components of such a network are goals, rules and questions. The goals are established by evaluating their associated rules. The rules are evaluated using conditions passed from their antecedents. There is one main control statement which initiates the processing of the inference network. ACTIONS are each considered in turn, according to their position in the knowledge base, and the progress of the consultation through the inference network. ACTIONS may be conditional on the satisfaction of PROVIDED clauses. There is a very good four-level debugging facility which includes TRACE; the most detailed trace indicates the conditions being searched and the values being assigned. The explanation facility is very good with the depth of explanation not restricted. Sage numerical values are accurate to two decimal places. The system is capable of complex and difficult calculations using FORTRAN or PASCAL external routines, but only ten variables can be transferred to the external routine from the consultation module and only one variable can be transferred the other way. This is a serious limitation for engineering and design applications. Another disadvantage of the system is its inability to change the value given to an OBJECT. This is a serious deficiency for design applications, in which the values of variables may have to be changed frequently (see section 3.2.1). Sage is not suitable for engineering application, especially if iterative processes are required, because of its rigid structure and the inability

to change the value of an OBJECT and also because of the restriction in the transfer of data to and from the external routines [53],[54].

3.4.2 Tool Selection Process

To be useful as an aid to engineering design, the author concluded that an Expert System must possess the following attributes:

a) It should capable of forward and backward reasoning strategies.

b) Be capable of numerical manipulation or be able to call on specialised programs with relative ease to perform such tasks.

c) Have the capacity to call on other specialised systems and programs to perform such tasks as the management of data, processing of specialised functions, selection processing etc.

d) The variable and object values should be global for easy access. The restriction on storage or retrieval of such values must be minimal.

e) The alteration of variable and object values should be possible with relative ease since the nature of the problem is iterative. This is particularly important as any restriction to this process will result in an unacceptably slow performance.

f) The modelling of the problems being addressed (typically those in this thesis) should be subjected to as few restrictions on presentation, programming

structure etc. as possible.

g) Be readily modified to include more features or to remove unwanted features.

Having itemised the desired features of a tool for application to this domain, the use of an unstructured programming language was also evaluated. Initial experience was gained by developing a simple system, using the Basic language on a BBC computer, to design closed coil helical springs. This exercise showed that using an unstructured language for developing a prototype system would be both tedious and ineffective. The computation time required would be unacceptable and the representation of the knowledge in a form that could be readily utilised would not be made easy. Typical problems were the addition of heuristic knowledge, explanation of the reasoning process etc.

Having gained this experience, the various tools available for the research, and referred to earlier were examined. The balance of advantage lay with Microsynics which possesses many of the features which facilitate the building of an Expert System for application to this domain.

The assessement of languages and their capabilities are now examined in relation to the above shells. The next section examines two languages available at the time.

3.4.3 Expert System languages:

Rich [55] listed some features she considered necessary to ease the production of Artificial Intelligence systems. Amongst the list was the ability to produce an efficient code so that the performance of the system is acceptable. She was describing language attributes based on the broader subject of Artificial Intelligence. Though these attributes are considered relevant for Artificial Intelligence applications, they are inadequate for Expert Systems application to mechanical designs. One of the earliest Artificial Intelligence language was IPL (Information Processing Language) [55]; the addition of higher level constructs and knowledge structuring mechanisms produced one of the most famous AI languages called LISP (Fig 3.5).

Berk [56] described a good Artificial Intelligence language as one which is designed primarily for list processing and manipulation. He gave the example of a book. A book consists of a large collection of words or lists and the lists are well structured. It is for the manipulation of these structures that LISP is good. LISP, (LISt Processing language) was developed primarily to enable humans to communicate with computers naturally [57]; it was developed for natural language processing. A list is a very common data structure in non-numeric programming, it is an ordered sequence of elements that can have any length but the order in which the elements are placed is important. The elements of a list may be constants, variables and other lists. In

LISP, the only data structures available are constants and the lists; the latter are manipulated by splitting them up into a head (containing one element) and a tail (consisting of the other elements). The version, called Common Lisp, could only support numerical values equal to or greater than -32767 but less than +32767. This severe restriction

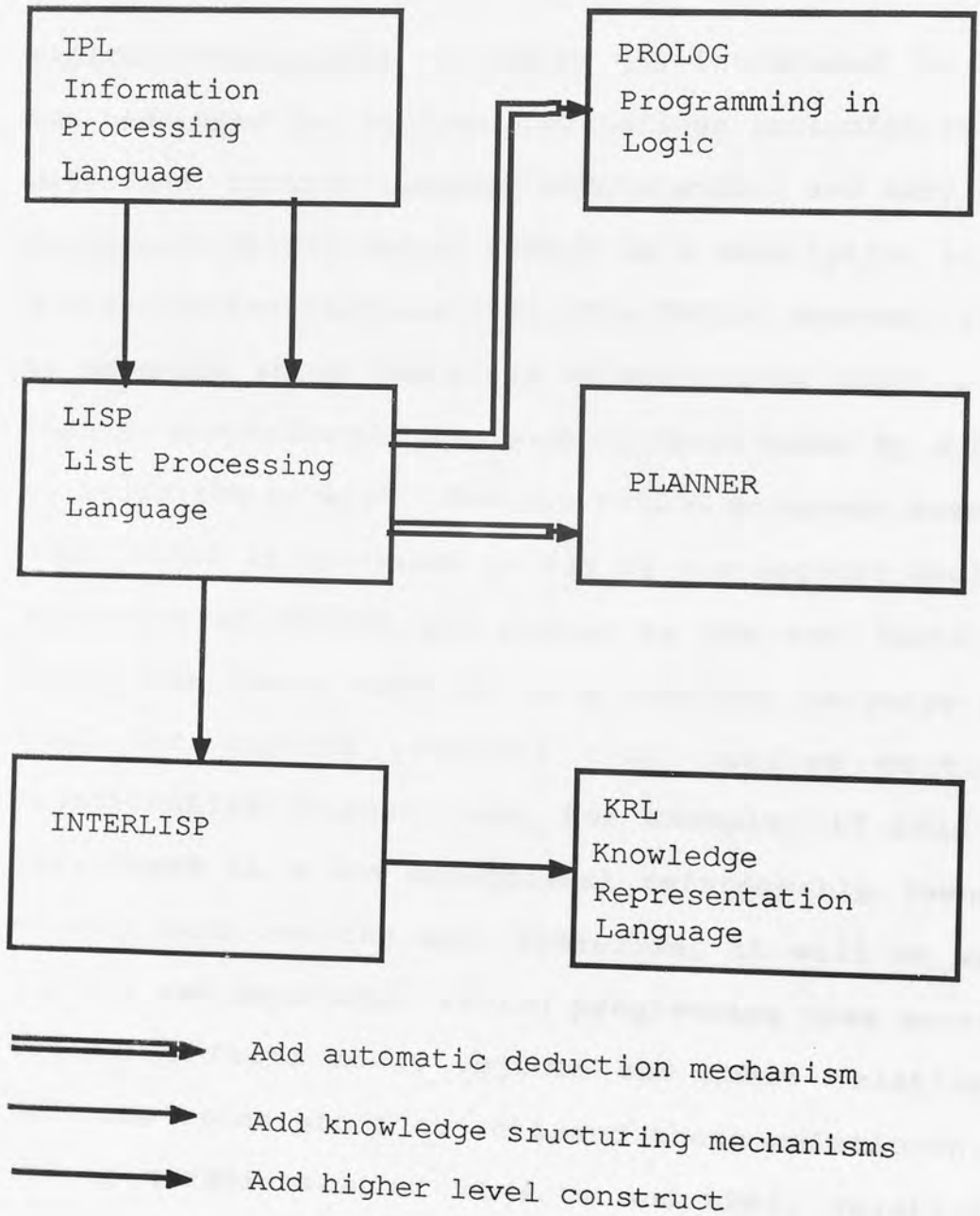


Fig 3.5 The evolution of Artificial Intelligence languages

disqualified the language for this project. The version of the language available to the author at the time could not call a formal logic language such as Fortran to manipulate or process values larger than that possible in the Lisp enviroment. The next step was to consider the other available language called PROLOG.

PROLOG (PROgramming in LOGic) was introduced in 1970 and has been used for various applications including relational databases, natural language understanding and many areas of Artificial Intelligence. PROLOG is a descriptive as well as a prescriptive language [58]. The PROLOG approach is rather to describe known facts and relationships about a problem than to prescribe the sequence of steps taken by a computer to solve the problem. The way PROLOG programs execute the computation is described partly by the logical declarative semantics of PROLOG and partly by the new facts it can infer from known ones. It is a computer language that is used for solving problems that involve objects and relationships between them. For example, if Lola owns a cat, there is a one directional relationship (ownership) between Lola and the cat. Therefore, it will be wrong to say the cat owns Lola. Prolog programming thus consists of declaring facts about objects and their relationships, defining rules about objects and their relationships and asking questions about objects and their relationships. PROLOG like LISP, has the facilities for list processing and recursion. The only difference is its added automatic deduction mechanism. Like LISP, it was designed with

natural language processing in mind. The version available also had restrictions on numerical values and could not transfer the process to a conventional programming language. The language was rejected for these reasons.

OPS5 (Official Production System: version 5) programming language was developed for building Expert Systems and for performing experiments in cognitive psychology [59]. Its first major application was by McDermott [60] for the implementation of the Expert System called R1 which configures digital computers. OPS5 is considered to be a "pure" production system in that it contains the production rules and architecture that are common to all production systems, and adds no special data structure. Its major advantage is its ability to call on external programs written in the hardware native-mode language. It can also call on system services that perform certain types of tasks more efficiently. (For example, Relational Database management programs). An OPS5 program consists basically of two parts; the working memory (or data storage area) and the production memory (which contains the production rules) [61]. The working memory and the production memory interact during program execution. The working memory determines the steps to be taken by the production memory, and the production memory changes the contents of the working memory as it executes as shown in Fig 3.6. Another advantage of this language is that the database is global. If this language had been available to the author at the time the project was initiated, it would have been the most

suitable of all those considered. The implementation of a prototype system was not carried out in the evaluation of this language as the author was already committed to the use of another. This normally gives a firm confirmation of a system suitability. It was however considered important

Fig 3.6 Interaction in an OPS5 program [61].

briefly to describe the language to show its relationships to the other contenders for the project.

Why the languages available were not considered adequate is explained in the next section.

3.4.4 Expert System language evaluation:

The languages evaluated were inadequate in the sense that the Common LISP and PROLOG available both on the BBC micro computer and the VAXA cannot handle very large numerical values as was explained in the last section. The version available at the time could not communicate with external programs which can handle the numerical process. Writing a subroutine in LISP to do this was not considered practicable at the time. Another inadequacy was the structure of the languages (that is, LISP and PROLOG). They do not lend themselves to iterative type problems. For the above reasons, the modelling of an engineering design problem using list processing functions is undesirable.

However, the advantage of using a language as opposed to using a Shell is that of flexibility. The system under development can be modelled without so many restrictions. Most of the shells discussed earlier had too many restrictions. At the current stage of Expert Systems development, it is the opinion of the author that researchers must have as much flexibility as desired.

The work of Expert Systems researchers in the engineering design domain is discussed in the next chapter.

CHAPTER FOUR

LITERATURE RELATING TO EXPERT SYSTEMS IN MECHANICAL ENGINEERING DESIGN.

CHAPTER FOUR

LITERATURE RELATING TO EXPERT SYSTEMS IN MECHANICAL ENGINEERING DESIGN.

4.1 Introduction:

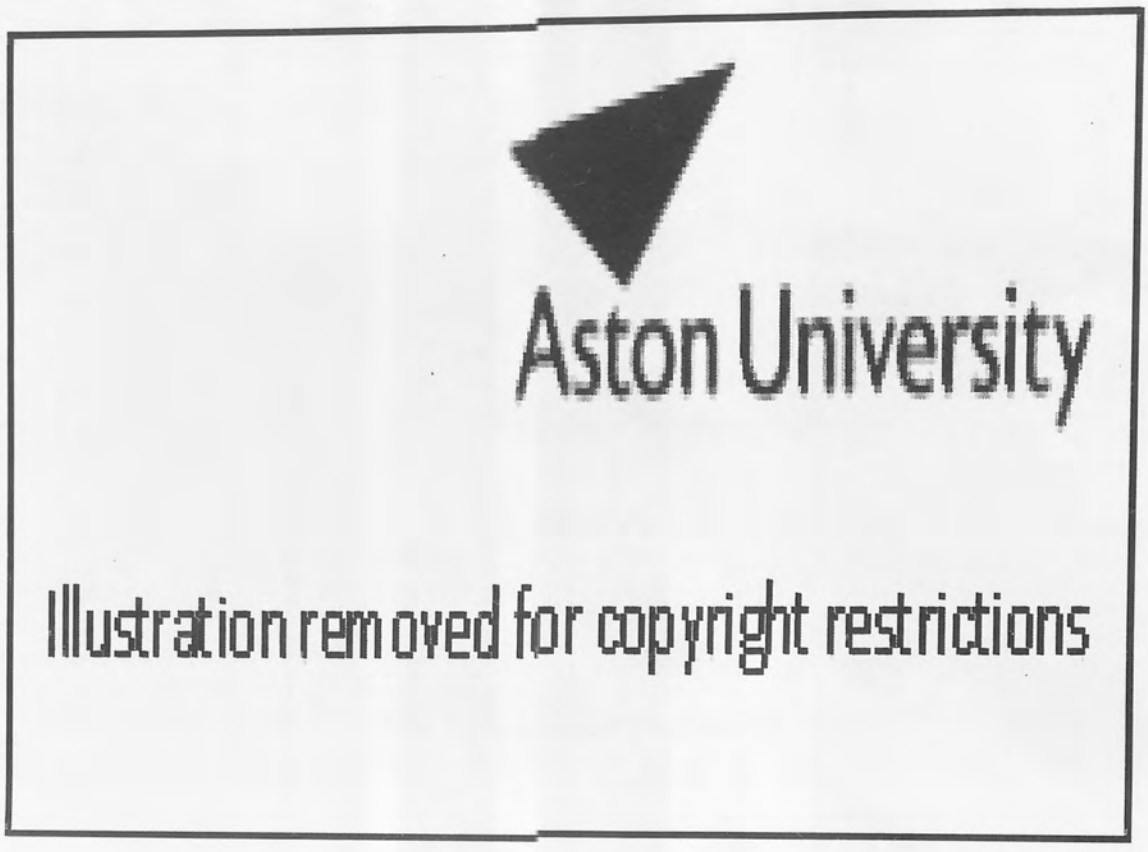
The literature reviewed in this chapter relates to the area of Expert Systems applications to mechanical design. The authors proposed various design models and sought to assess their merits by implementing prototype systems. A thorough literature search revealed that very little research has been published in this area and these contributions to knowledge are now discussed.

4.2 THE LITERATURE.

The earliest research paper on Expert Systems application to Mechanical design was published by Brown and Chandrasekaran [62] in 1983. They proposed a specific type of problem solving associated with expert design activities of the class 3 type. They presented an approach to mechanical design which they called Design Refinement involving a heirarchy of specialists solving the problem in a top-down fashion. Their paper was concerned with the conceptual approach and paid attention to the problem of knowledge structuring, an area which had received little

attention previously. A prototype system was developed for designing a small air-cylinder containing seventeen components. They concluded that only a subset of class 3 problems can be successfully solved using the proposed design refinement method. They were also of the opinion that for some design problems, there may be insufficient understanding of the problem solving processes. The effect of this is that difficulties arise when assessing the amount of knowledge that may be required for the proper functioning of a system. The approach is thought by the authors to reflect the formation of conceptual structures for problem solving. Brown and Chandrasekaran highlighted the current problems faced by Expert Systems developers in this domain as being the inadequacies of current Artificial Intelligence technology to capture the knowledge structure and problem solving techniques necessary. They also proposed possible design models, but they did not however, suggest possible knowledge representation techniques that could be considered suitable. They did not address the problem of how analytical design problems could best be handled (for example, component design).

Dixon and Simmons [63] used the architecture, proposed by Brown and Chandrasekaran, called the design-evaluate-redesign architecture as a basis for constructing an Expert System for the design of a standard V-belt drive. The architecture, though slightly modified, is based on an iterative design model consisting of four stages, Fig 4.1. It supports the design of components or simple systems and a hierachical system of design specialists was proposed for



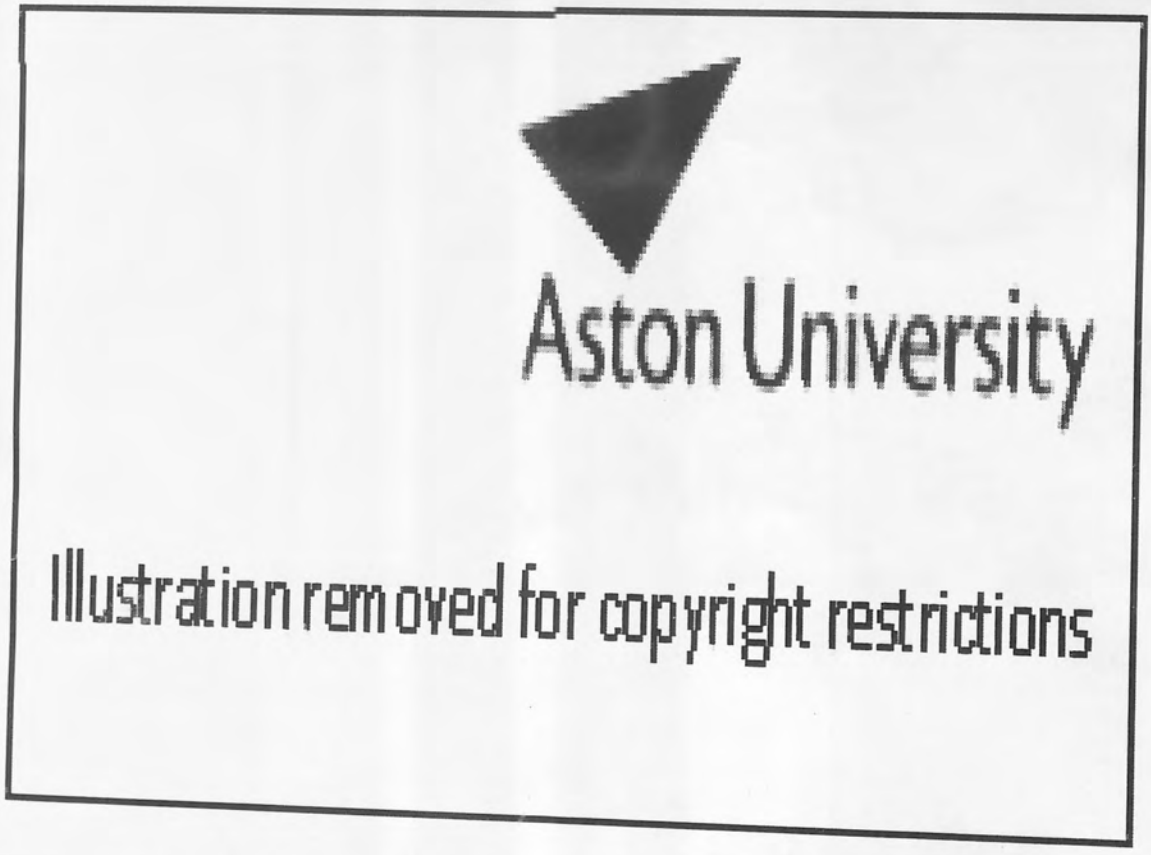


Fig 4.1 Iterative model of the design process (Dixon [63])

more complex tasks. The system developed, called Vexpert, was written in Lisp and applies production rules OPS5. The model was designed to accommodate the iterative nature of design. They concluded that their system was a competent designer by comparing the results from test cases to those obtained from human designers. The author believes that a very detailed knowledge of the relationships between design parameters will result in a process that requires less iteration. Furthermore, the use of standard algorithms, though not undesireable, does not provide adequate transparency of the knowledge and reasoning strategies in an Expert System approach. In this respect, the work of Dixon and Simmons was deficient.

Kulkani et al [64] developed a prototype Expert System called XENIF for the design of aluminium extruded rectangular heat fins. It addressed the problems of conflicting functional, economic and manufacturing subgoals. The system was based on the iterative redesign architecture used previously by Brown and Chandrasekaran [62] and Dixon and Simmons [63]. Kulkani et al believe that this strategy of iterative redesign is used by human experts, and is implicitly based on their knowledge of "dependencies" in the domain. Dependency was defined by these researchers [64] as "the relationship between design goals and a design variable which expresses qualitative and/or quantitative knowledge about how changes in that design variable will tend to alter the degree of satisfaction of the goal". One of the aims of their research was to identify the nature of dependencies in a

simple domain and to study the use of dependency knowledge in an Expert System that uses the iterative design strategy. Kulkani et al did not adequately address the knowledge representation structure necessary for this type of problem solving. The work presented in this thesis lay greater emphasis on the knowledge representation techniques needed in this application domain.

Howe et al [65], designed a domain-independent program for Mechanical design basically to examine the relative merits of depth versus generality strategies in Expert System developments. The system developed relied on so called "weak" methods [55] for its search process. In this case, the best first search method. They approached the problem from the generality angle as opposed to the direct specialisation used by Brown and Chandrasekaran [62], Dixon and Simmons [63] and Kulkani et al [64]. They, like the previous researchers, focussed on the iterative redesign of a component, single object or simple physical system. They compared results obtained from the system in two different domains with proven Expert Systems and human expert. In the designing of the V-Belt, the system was compared with Vexpert [63] and in the case of heat sink design, it was compared with Xenif [64]. The approach adopted in this work, and described later in Chapter 5 of this thesis, differs from the research discussed above in that the author approached the development of the system from a domain specific knowledge basis.

Korane [66] discussed the application of Expert Systems to

mechanical design stating that Expert Systems have failed to live up to expectations because they have not replaced hunan experts in such tasks as trouble-shooting, process control and engineering design. He gave two examples of Expert Systems that have been developed for mechanical design. The first system developed was by Noore Corporation of St Louis, (U.S.A.) for the design of joint flanges, which is a class 3 design task. Flange joint design has eight geometric variables and many material possibilities. Noore Corporation's designers always arrived at different flange joint configurations and an Expert System was employed to reduce these variations. The use of linear programming techniques proved unsuccessful because the design problem was not an optimization process. It basically involved the use of heuristic rules, guesses and engineering judgements based on previous experience. Amongst the problems encountered on the implementation using Expert Systems were disagreements amongst designers about proper procedure, lack of understanding about the fundamentals and some "grey" areas where engineering data simply were not available. However, the process of developing an Expert System for this task helped in the understanding of the actual problem and focused on instances of overdesign. The resulting Expert System it was claimed, improved the overall design, eliminated flange joint leakages, reduced the flange configuration time, reduced the time and production cost.

An important point made by Korane [66] was that the additional benefit that emerged only during the development

stage of the system was a better understanding of the fundamental elements in flange design. It would have been expected that design for commercial and production purposes would entail the utilization of designers with a thorough understanding of the fundamentals of the design domain. In practice, this is not so. In his discussion, the author gave no indication as to the type of Expert System used and what type of knowledge representation technique was employed. Since the design task described involved the extensive use of heuristics and judgement based on experience, it is the opinion of the author that a rule-based production system may have been employed.

The second example given was the design of O-ring glands. NL Industries uses O-ring seals to protect electronic sensing equipment in down-hole drilling. The operational environment is severe and pressures up to 140 MPa are applied. Individual designers use their own guidelines to select a suitable O-ring and to design glands. Hence, there are no standard guidelines or uniformity in the selection process. Although the design task was well defined, the development proved very difficult, mainly because of the amount of development time involved and acquiring the necessary knowledge base. Not all the information required for the program was available from the seal manufacturers. OPS5 was used as the knowledge representation tool. Another problem encountered was the inability of most Expert Systems tools to handle large numbers. Calculations took a long time and they suggested the use of conventional programs such as Fortran to handle the numerical aspect. This given example falls under selection and shows the

problems generally faced in applying Expert Systems techniques to engineering tasks, especially in terms of development time and tool suitability.

Dimitrov [67] dealt with the problem of knowledge representation for mechanical systems design. He divided the capabilities of knowledge representation systems into three levels, that is, the general level which provides the fundamental knowledge structuring and organisational facilities including the inference mechanisms, the world model level which reflects the general characteristics of the domain and is biased towards the types of problems to be solved, finally, the application level which reflects the concrete problems in the application area. Each level is further sub-divided. A knowledge representation system called INEX (INstrumental enviroment for building EXpert systems) was developed using LISP as the implementation language with the capability to use a logic programming language when required. It was claimed that the contribution of INEX to knowledge representation for mechanical engineering was an elaboration of the systems theory of a world model. For example, machines can be modelled best as systems performing certain functions on the basis of their component functions. This was illustrated by modelling a speed reducer (gearbox) consisting of three shafts and four gears. A suitable reducer is selected (e.g., simple reducer, worm reducer etc.) and its function is matched with that required. If successful, the reducer function is then elaborated. Dimitrov [67] concluded that the the system theory based on

world models provides a natural representation structuring framework in the domain of engineering design. The author added that the implicit organisation of the design process supported a declarative style of knowledge representation. This supports the result of the independent assessment, conducted by the present author, on knowledge representation formalisms suitable for mechanical design tasks as part of the research project. The success or failure of INEX was not mentioned in the paper. However, Dimitrov has initiated a research interest in an area neglected by previous researchers.

Since the completion of this project, many research papers have been published in the area. Based on the literature review, the next section of this chapter discusses the knowledge gaps identified by the author as existing in this domain of interest.

4.3 Knowledge gaps

The researches mentioned so far dealt with design problems using Expert System methodologies as an overall control structure, in contrast to using Expert System techniques for the analytical and numerical processing of the various component designs in the overall system.

The present work also addressed the difficult task of understanding the inter-relationship between design variables, with the hope of reducing the number of iterations required for a successful design and also to

give the user valuable and reliable advice as opposed to an educated "guess". But it focusses on mechanical component design rather than system design.

Another area neglected in the literature is the type of knowledge representation technique required. This thesis addresses this subject and, as a result, research was carried out to address the problem of a purpose built Expert System for mechanical design. This was another area neglected by previous workers.

The next chapter discusses the two concepts proposed by the author for the application of Expert Systems to this domain. These two concepts were developed because, as will be shown latter, design problems like the helical spring involves using design relationships and equations and less heuristic knowledge. In the case of gear design, the relationships between design parameters are not fully understood and are in most cases based on experience (or heuristic knowledge). Thus, design factors are used to compensate for this lack of knowledge. However, the second approach is considered as a more generalised approach than the first.

CHAPTER FIVE

DEVELOPING EXPERT SYSTEMS FOR MECHANICAL ENGINEERING DESIGN

CHAPTER FIVE

DEVELOPING EXPERT SYSTEMS FOR MECHANICAL ENGINEERING DESIGN

5.1 Introduction:

This chapter discusses the system structure developed for Expert System application to the domain of mechanical engineering component designs and contains four main modules. This methodology is a marriage between the requirements of the mechanical design process and Expert Systems methodologies. The chapter also discusses the development and concepts of two new knowledge groupings called the "Level Concept" (LC) and the "Frame Based Concept" (FBC). The reasons for the presentation of the two concepts will also be discussed.

5.2 Expert Systems Problem solving approach to Mechanical Design:

In the design of an Expert System, the problem scope (that is, what problems the system is designed to solve) must be well defined and the variables/parameters involved carefully examined and grouped. The grouping of variables and parameters is to ensure that a method of progressive substitution can be used enabling the design process to

progress "smoothly", a point that will be expanded upon in later sections. The problem solving approach described in this chapter is based on the premise that full understanding of the relationships between variables and equations is essential to the successful implementation of an Expert System. The assumptions made in the derivations of equations affect how the these equations are applied. Equations are regarded as Principal (or Primary) rules which cannot be violated.

To successfully apply the proposed concept to mechanical engineering design problems, a prototype system must be built and its performance evaluated to ascertain if it conforms with all the requirements. This also gives a fair indication of very practical issues such as disc storage requirements as will be shown later. Data and knowledge storage capacity and file management are essential for the successful application of this approach. The prototype system then passes through an iterative process of evaluation, analysis and optimization until a satisfactory system is obtained and demonstrated.

Since the area of interest is engineering design, the first step was to develop a system structure that conforms to the requirements. Hence, the need for the following three stages.

a) specification and clarification

b) design analysis and evaluation and

c) presentation of the results.

Also, for Expert Systems requirements, the system structure should basically consist of the following two sections (see Appendix A2).

1) An Inference engine, and

2) a Knowledge base.

The blending of these two sets of requirements led to the derivation of the four stages (or modules) used for the system structure. These four modules will now be discussed.

5.3 The system structure

The system developed was divided into four modules (Fig 5.1) for reasons mentioned in the last section. These four modules are labelled;

(a) Information Acquisition (question and answer)

(b) The Inference Engine (Control Module)

(c) The domain specific knowledge

i) Test module

ii) Calculation module

(d) The Explanation and Self-knowledge module (Reasons and Advice).

We now proceed to describe these modules in detail and the

relations/interactions between them.

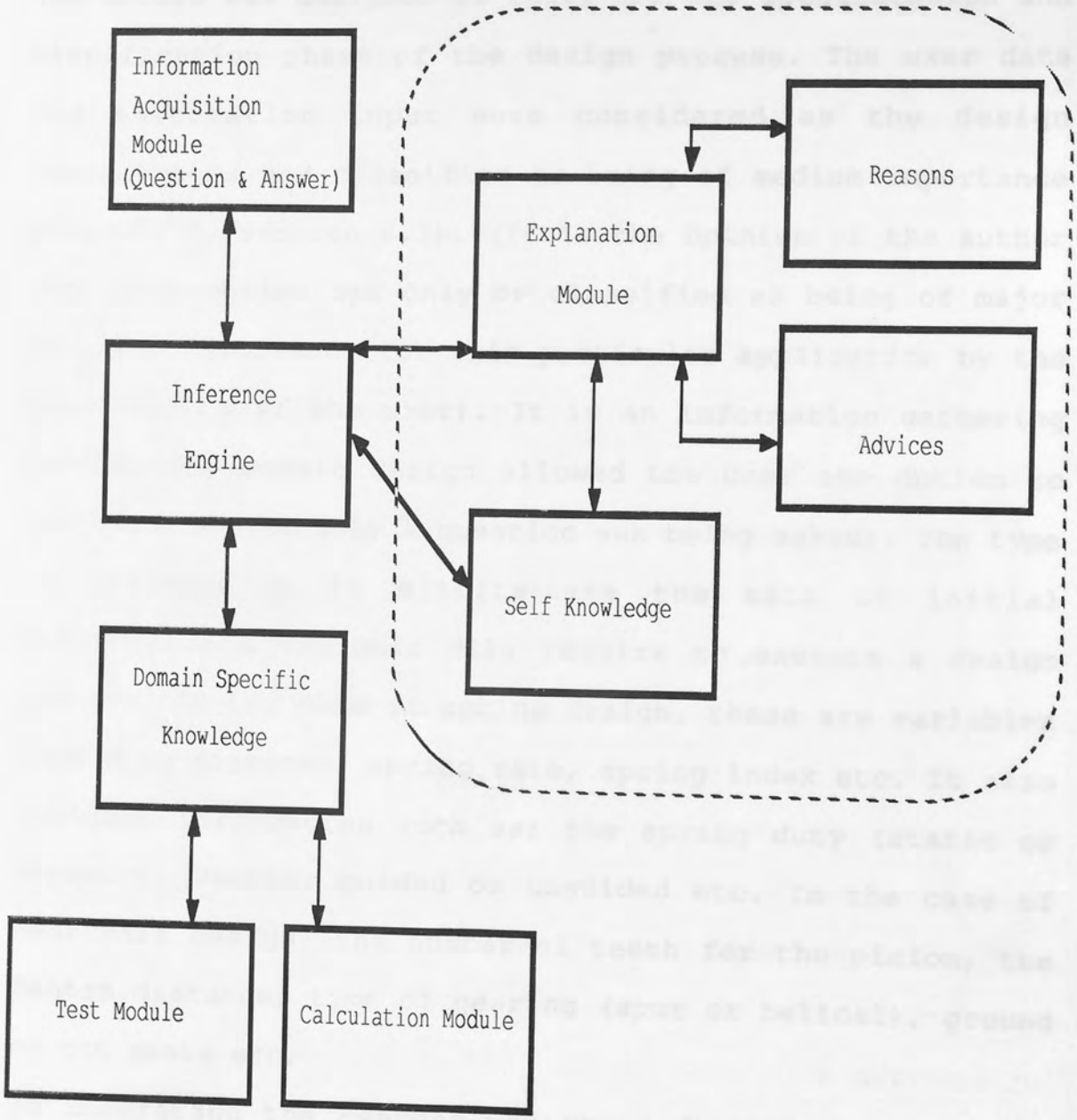


Fig 5.1 The basic system structure for Expert Systems application to mechanical design.

5.4 Information Acquisition module:

The module was designed to cater for the specification and clarification phase of the design process. The user data and information input were considered as the design requirements and classified as being of medium importance (Chapter 2, section 2.3). (It is the opinion of the author that information can only be classified as being of major or minor importance for this particular application by the requirements of the user). It is an information gathering module. The module design allowed the user the option to query the system (why a question was being asked). The type of information it elicits are the sets of initial information a designer will require to execute a design process. In the case of spring design, these are variables like wire diameter, spring rate, spring index etc. It also includes information such as; the spring duty (static or dynamic), whether guided or unguided etc. In the case of gear pair design, the number of teeth for the pinion, the centre distance, type of gearing (spur or helical), ground or cut gears etc.

To understand the reasons why the information module was structured in the manner discussed below, it is appropriate to refer to the initial structure used with the implementation on the BBC microcomputer using the Basic programming language (it is regarded as an unstructured language as far Expert Systems applications is concerned). In this first attempt, the module asked for the initial

information in a predetermined manner (Fig 5.2, see nomenclature for symbol meanings).

The predetermined method of information gathering was adopted because some information is usually asked for first, the remainder is only asked for if the "first choice" information cannot be obtained. (For example, a spring index will be asked for and if it is unknown, the Wahl factor may be asked for with the aim of calculating the spring index from the given value). This approach also means that all the variable values involved in the design of the component shall be asked for, making the information acquisition process slow. Some of the information asked for may be redundant (that is, can be obtained by calculation or deduction from the information already known). Another implication was that the approach was found to be rigid; it does not allow for flexibility in the modification and improvement of the module. This was apparent in cases where the variable values were changed and a new set of values needed to be asked for but not in the predetermined manner designed into the module. For example, if the spring rate is required, the module will go through the process of asking for all the other variables first.

The approach to the design of the later module structure (figs 5.3a and 5.3b) was based on the approach used in the previous case. It can be argued that, although a human expert commonly asks for some set of information ("first choice information") and obtains the others by deduction or

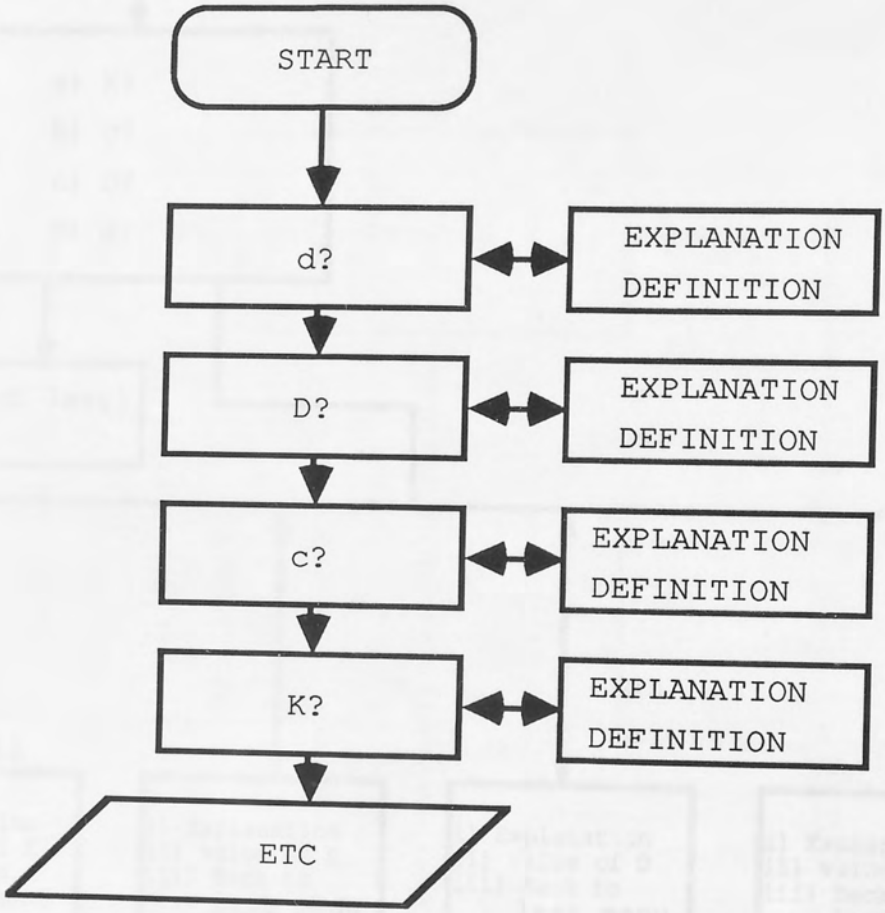


Fig 5.2 The old structure of the Information acquisition module.

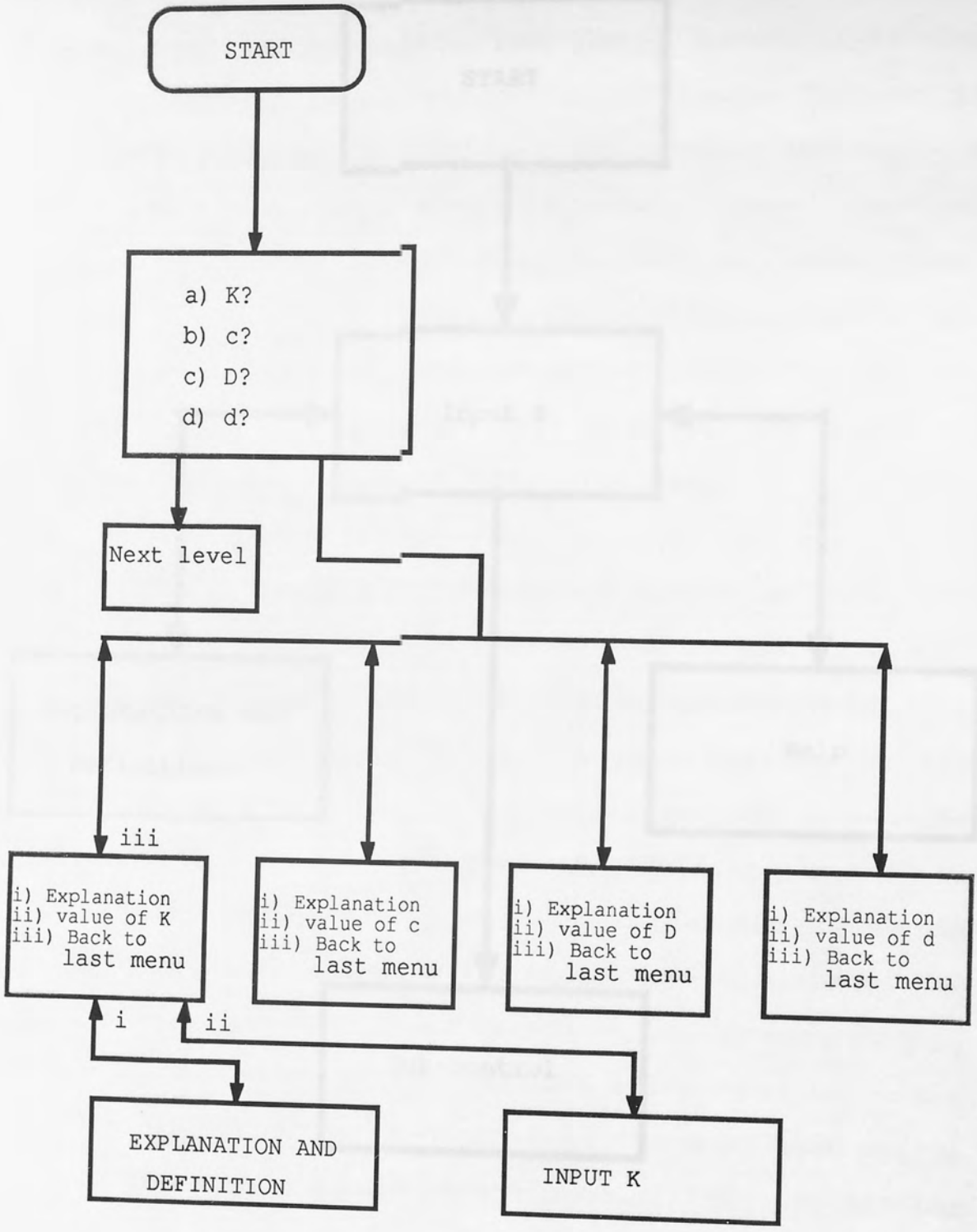


Fig 5.3a The structure of the Level concept approach to the Information acquisition module.

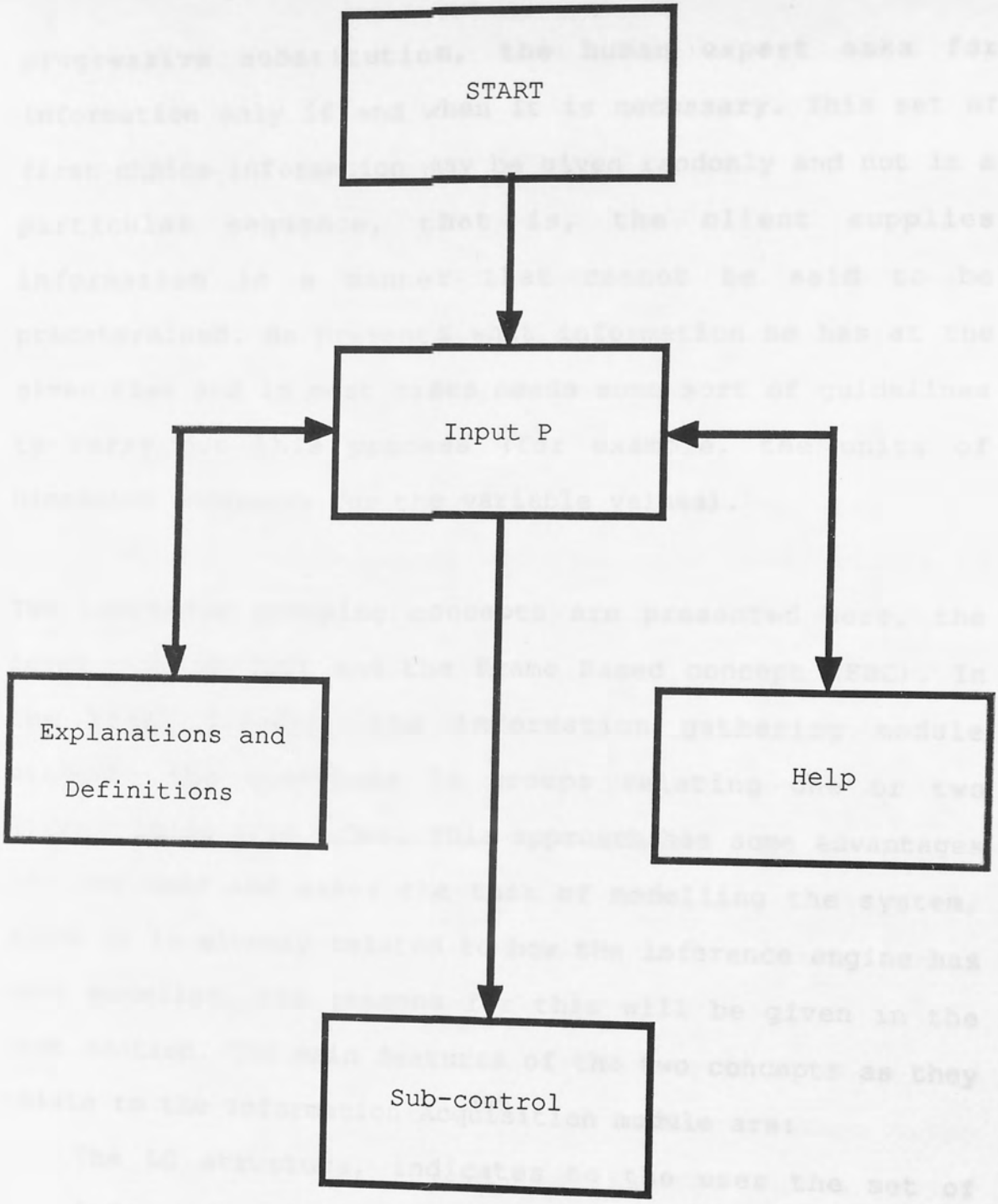


Fig 5.3b The structure of Frame Based Concept approach to the Information Acquisition Module.

progressive substitution, the human expert asks for information only if and when it is necessary. This set of first choice information may be given randomly and not in a particular sequence, that is, the client supplies information in a manner that cannot be said to be predetermined. He presents what information he has at the given time and in most cases needs some sort of guidelines to carry out this process (for example, the units of dimension necessary for the variable values).

Two knowledge grouping concepts are presented here, the Level concept (LC) and the Frame Based concept (FBC). In the Level concept, the information gathering module presents the questions in groups relating one or two PRIMARY rules (Fig 5.3a). This approach has some advantages for the user and eases the task of modelling the system, since it is closely related to how the inference engine has been modelled, the reasons for this will be given in the next section. The main features of the two concepts as they relate to the Information Acquisition module are:

The LC structure, indicates to the user the set of information required by the system. The user has the option of presenting it in any order and also of finding out what the question was all about without committment to answer the question. If there is insufficient information, the system can quickly detect this and advise the user before proceeding further. This cuts the consultation time and provides a close

guide for the user. It detects quickly if a rule is satisfied or not with the information at hand.

For example, the primary Rules :

$c = D/d$ and $K = (4c-1)/(4c-4)+0.615/c$

are associated with the variables c, D, d and K. and these correspond to the group of questions asked. The information received will determine the path the process will follow.

The FBC system determines which question it wishes to ask in an order determined by itself. The user still has the option to answer or refuse to answer the questions. In this system, the restriction on rules about which information is to be elicited is removed and hence all the rules are considered to be at the same level. This concept will be explained further in the next section. The improvement in the control module and the development and inclusion of a search module ensures that the speed at which the system detects insufficient information is not affected. Also, using the semantic network knowledge representation scheme allows access to different parts of the system. The feature is taken advantage of fully in the FBC structure. The principle is to have a system that is mainly controlled by the search module and an input that accepts both numerical and alphabetical characters (thereby identifying and correcting input errors) as opposed to either numerical or alphabetical (which will

cause the system to "crash"). This rationalisation also saves on memory.

5.5 The Inference Engine (Control Module):

An important attribute of an Expert System is its ability to explain its reasoning process and give explanations for its conclusions. This means that it must also have a degree of knowledge about itself. To do this, the system must at all times know the stage of its progress towards a goal. This task is handled by the Control module. It is the most important structure of an Expert System and the philosophy used for modelling it will affect the performance of the system. The type of knowledge representation scheme adopted will affect the capabilities of the module. The basic concept actually used and another equally effective approach to the modelling of this module will be described below. Semantic networks provide the ability to determine what has been done and what possible options are left open at any stage of the process. Most of the knowledge under consideration dealt with relationships that could be described in terms of algebraic equations, empirical functions and heuristics.

The ability to keep track of the solution process is enhanced using semantic nets. Within this network structure are inbuilt tree structures for each level. For example, in Fig 5.4, the states A, B, C, and D are considered as

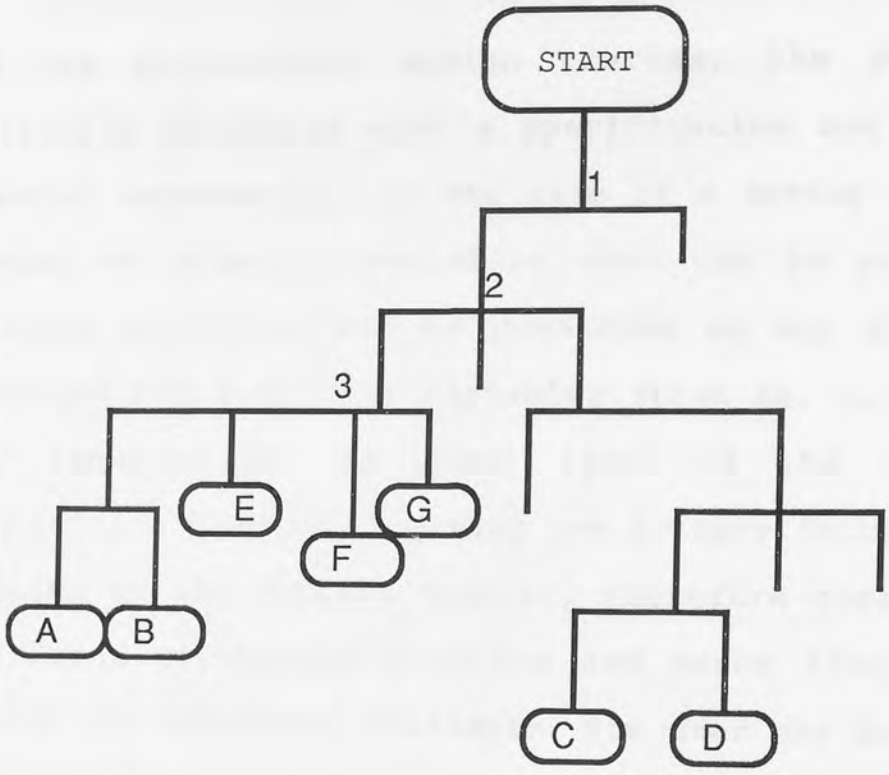


Fig 5.4 Possible solution paths

possible goals to a problem. At each node, depending on the information presented to the system, it makes a decision as to what route to follow on its way to a goal. If at Node 3, the system was to give advice as to the state of the problem and what path (or reasoning process) it has open to it, the system would be able to say that the goal options C and D are not accessible whereas there are five possible conclusions depending on further information being provided. With this type of structure the system is in control of the solution process and improves the ability to give credible and accurate information on the state of the process.

In the engineering design process, the designer is initially presented with a specification and some other general information. In the case of a spring design, the number of possible variables that can be presented is sixteen and these can be presented in any combination. Consider Fig 5.5, four variables (that is, c,d,D, and K) are required at the first level of the information acquisition process involving two Primary Rules (the same applies to the control module), therefore there are five different situations possible and hence five different routes (or branches) available. The user may have none of the variable values to offer (in which case branch 0 is taken) or may have one, two, three or all four of the variable values. At this point the system chooses the path depending on the number of values offered by the user. If two values were given the system would follow Branch 2. In Branch 2 there are then ${}^{n}C_{r}$ possible paths open to it (where n is the maximum number of variable values required and r is the number of variable values offered by the user). This method makes it possible to calculate the number of branches the control module can have and hence can estimate the size of the system and in turn be able to limit the problem scope to suit the computer in use and the time available for implementation. Fig 5.6 shows the number of paths the process can take when offered two variable values out of a maximum of four required by the system. This method also allows an iterative type design problem to

be implemented using Expert System concepts and the system will have the ability to explain what stage of

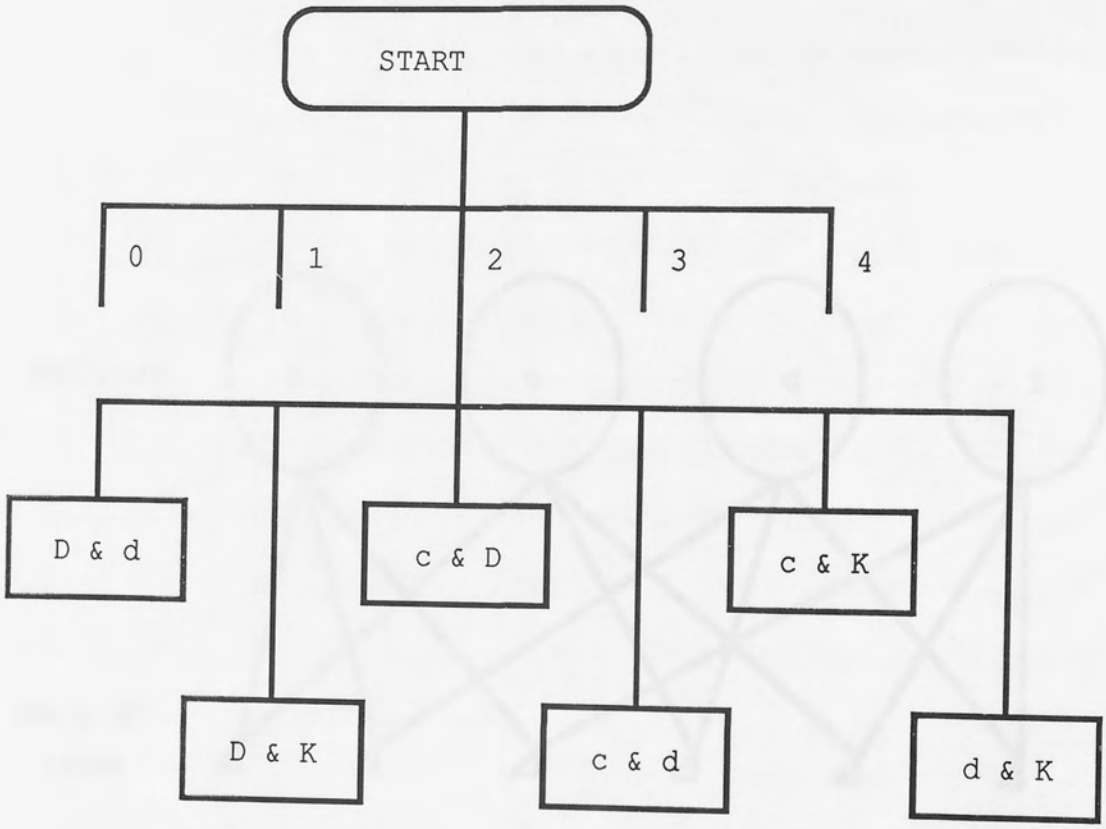
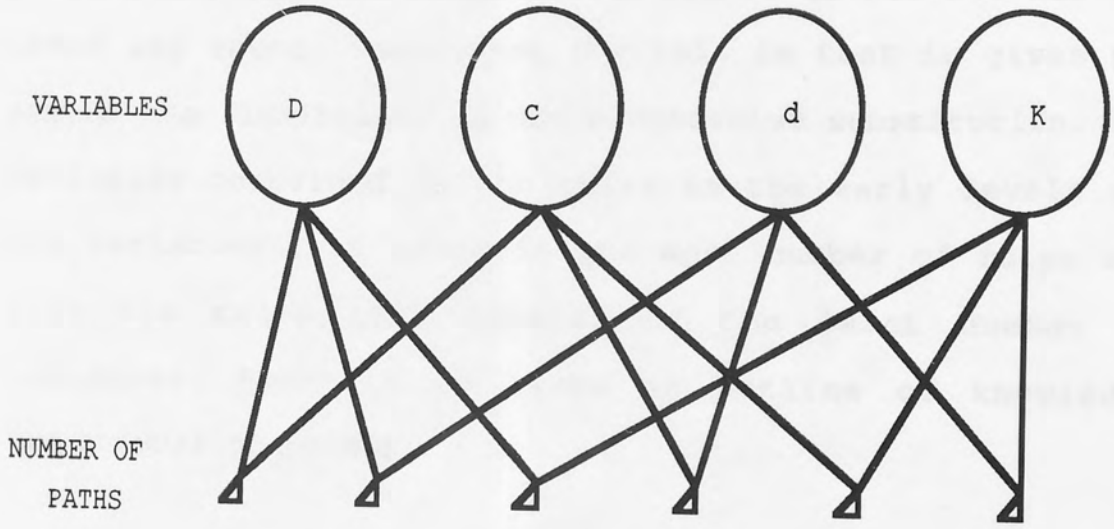


Fig. 5.5 The tree structure within the Inference engine



ie 4C2 = (4!)/(2!2!) = (4*3*2)/(2*2) = 6

Fig 5.6 Number of possible paths depending on the number of variables concern.

the solution process it is in, which path it has to follow and lastly enable Meta-knowledge to be designed into the system.

The LC approach to the Inference engine structure was based on the premise that all equations (or rules) concerned in the design of components can be divided in such a manner as to form some distinct levels. Each level consisting of two or more closely related principal (or primary) rules and the rules consist of related variables. In general, the variables in the lower levels (that is, Levels 1,2,...) can be in the rules contained in the higher levels but not the other way round. The reason for this is that it gives the system the flexibility to use progressive substitution. The variables contained in the rules in the early levels are the variables that occur in the most number of rules and also the rules that consist of the least number of variables. Appendix A3 gives an outline of knowledge (equations) groupings.

It is however, theoretically possible to have a situation were the number of rules associated with each variable is equal and the number of variables in each rule is equal (Fig 5.7). If this occurs, then a condition arises where there is a "perfect" network with each node having the same number of branches. This method may then fail in this type of condition. There will be a "congestion" of variables at the lower levels of the system.

Also, as the reasoning part of each level is developed further there is the tendency to have lower levels that give advice to the user with little or no understanding of

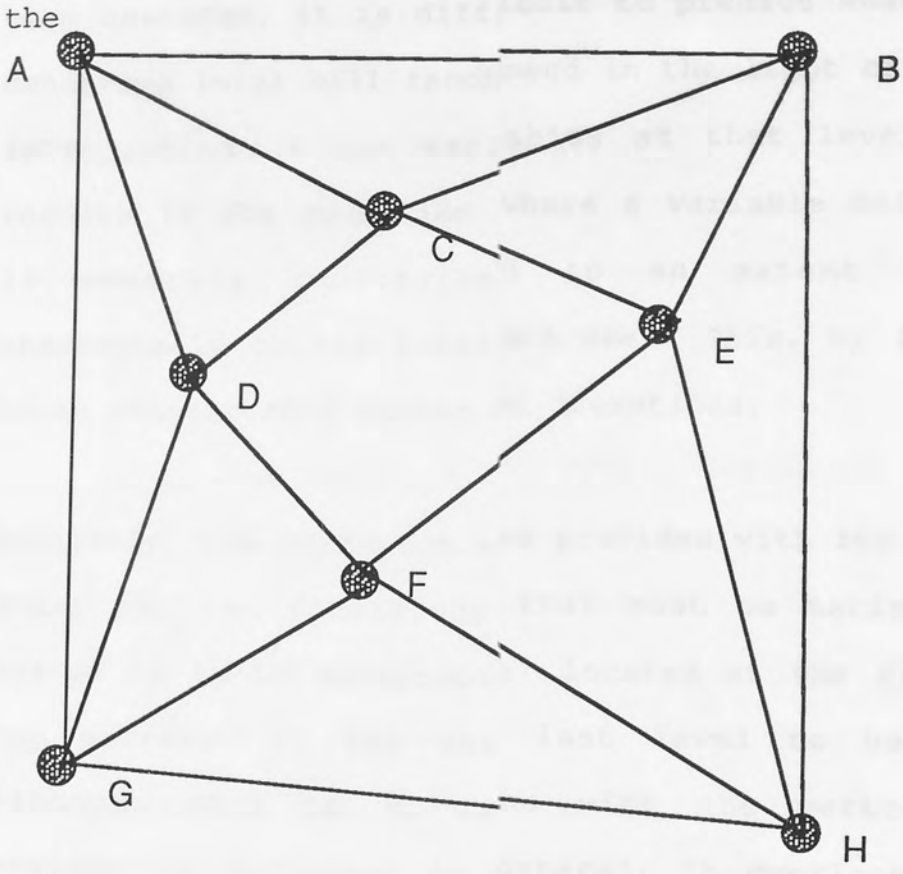


Fig 5.7 Nodes (equations) and their relationships (variables)

the consequences of these actions at higher levels. The reason for this is that since the higher levels have not

been executed, it is difficult to predict what actions the concerned level will recommend in the light of the possible introduction of new variables at that level. This also results in the condition where a variable being asked for is severely constrained to an extent that it is unacceptable to the intended user. This, by itself, leads to an unacceptable number of iterations.

Initially, the structure was provided with its tests levels (that is, the conditions that must be satisfied if the design is to be acceptable) located at the final part of the process. It was the last level to be processed. Although this is in line with the methods used by engineering designers in general, it overlooked the fact that this is done with a deep insight into what the final outcome will be and with experience into the particular type of design procedure. Having the tests at this part makes the possibility of iteration more likely.

To avoid the condition mentioned in the last paragraph of the last section (that is, a perfect network occuring), the concept of dividing the variables and rules (that is, equations) into levels was closely re-examined. The solution derived (that is, the FBC) was to convert these levels into frames (that is, programs that are knowledgeable in a specific but narrow domain). It should be realised that the levels could be considered as "Pseudo-frames" in this respect since they are specialist

sub-programs manipulating a set of two or more equations. However if each frame were to cater for more than one equation then the computation required and complexity in logical reasoning of the frames would represent a disadvantage of the method. It was then decided that each frame should cater for only one equation.

The principle of lower and higher levels (or frames as it can now be viewed) can be replaced by the concept of "equality" of levels. That means each level will no longer have to pass on the process to a higher level as was previously the case, instead the preceding or the subsequent levels should be decided by the inference engine. In effect, all levels are completely independent of other levels.

Placing the tests levels at the end of the process was also re-examined. It was located at the very beginning of the design process. The reason was that for a test to be satisfied, the various variables must be within a certain limit; and for that to happen, certain conditions or processes must be carried out. This view presumes that engineers, when manipulating data, are aware of the conditions that are required to be satisfied and do not just start looking for values that are indirectly related to the values needed to undertake the tests. The consequence of this is that the information searching process and the control adopt some degree of backtracking.

In engineering design, although the initial stage is known (that is, the data given at the onset of the design constitute the initial state), the final state or design values are unknown. But a successful design is that which satisfies some laid down rules (or primary conditions), for example, conditions for safety, stress requirements or some other factors. If the primary conditions cannot be satisfied, then the sub-condition needed to satisfy the main condition will be examined next and the sub-sub-conditions after that. The process is then repeated until a reasonable conclusion is arrived at. On the other hand, the examination and usage of variable values as the principal states leads mainly to forward reasoning approach. This approach was adopted for the inference mechanism in the LC method. The FBC system adopts a means-end-analysis strategy.

The result of these modifications was that the system has the inference engine sitting on what are termed E-frames and T-frames. The E-frame stands for Equation frames or Rule frames and the T-frame stands for Test frames. On top of these is the reasoning part responsible for assigning which tests should be undertaken to satisfy any type of design required. See Fig 5.8.

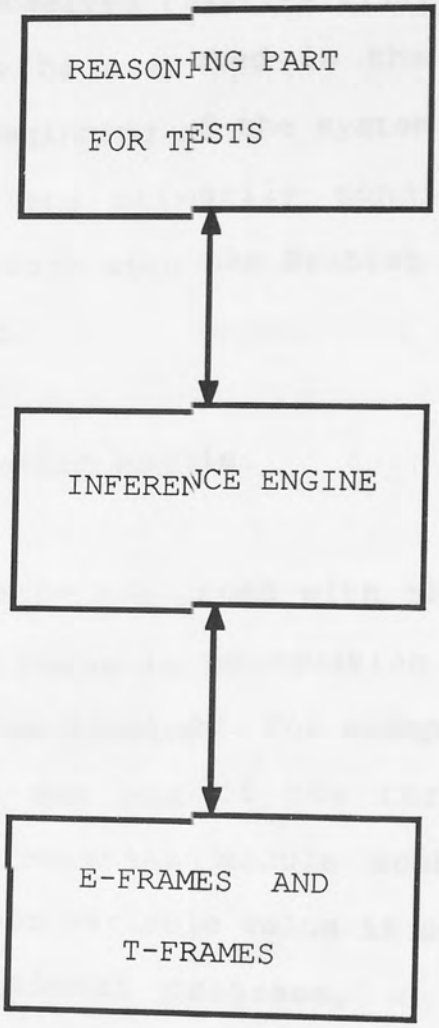


Fig 5.8 Block diagram of Frame based concept.

5.6 The Domain Specific Knowledge

5.6.1 The Test Module:

This section contains the Domain Specific Knowledge of tests to be satisfied (for example, buckling test, test for strength) before a component is recommended for use. The knowledge was represented as Production-Rules and ensured

that the codes and standards of the British Standards Institute were observed [19],[20],[68]. This section is the last section to be executed in the LC approach but is located at the beginning of the system structure in the FBC approach. They are primarily conditions that must be satisfied to conform with the British Standard requirement for the component.

5.6.2 The Calculation Module.

This module is only concerned with the calculation of an unknown variable value in an equation given the values of the other variables involved. For example, the equation u = z2/z1, may have any one of the three variable values unknown. The calculation module enables the system to calculate whichever variable value is unknown. On the other hand, in conventional programs, a specific dependent variable value (for example, u) can be calculated in accordance with some equation but the other two (say z1 and z2) must be given. If either of the other two is not known, then the process comes to a halt. This is not the case using this Expert System problem solving approach. If all variable values but one is known, the unknown variable value can be calculated. In the FBC approach, the module resides in the E-frames but is considered as the last module in the LC system.

5.7 The Explanation and Self Knowledge Module (Reasoning and Advice Module):

Most Expert Systems have what is called an explanation facility. This is knowledge for explaining how the system arrived at its answers [35]. Most of the explanation involves displaying the inference chains and explaining the rationale behind each rule used in the chain. Explanation is usually underestimated [69]. Explanation is important because:

i) System development is faster since it is easier to debug.
ii) The assumptions underlying the systems operation are made explicit rather than being implicit.
iii) It is easier to predict the effect and test the effect of a change on the system operation.

Although explanation is an important feature of Expert Systems, it is an aspect that is usually badly done [70]. The explanation facilities of the two case studies presented in this thesis were in two sections:

i) The first section resides in the information acquisition module (Figs 5.3a and 5.3b) and is concerned with the definitions and explanations of parameters and questions (for example, are the spring ends to be ground or plain?). It defines the

parameters being asked for to enable the user to understand what the parameter is and secondly, a question is re-asked in another form and the reason for asking is given. This method leaves the user in no doubt as to what the system requires.

ii) Rule Violation: When a primary rule is violated, the user is informed of this and an explanation given as to how the rule has been violated and what should be done.

This module contains all definitions, explanations of results and advice. It contains Self-Knowledge (that is, second order knowledge or Meta-knowledge: knowledge about its own knowledge). It is involved with every section of the system architecture. For this reason, the module is the most difficult to update and "tune". The explanation module must be updated with the continuous process of updating the knowledge base. If not, the performance and reasoning process of the system will not be transparent to the user. The explanation given at any time must also be consistent with past explanations and advice. The definition of a variable, equation, concepts etc., is part of the effort to provide an efficient communication with the user. This process contributes to the user understanding of the design paths.

The two concepts described in this chapter were applied to the two case studies to be discussed in the next two

chapters. The experience gained in the implementation of the first case study (that is, HELCOM) lead to the development of the Frame based concept applied to the second case study (that is, GEAREX). However, the Level based concept applied to spring design can also be applied to gear pair design (Appendix 3). The Frame based concept can also be applied to spring design as will be shown in Chapter 7. The next chapter gives the structure of the HELCOM program for the design of Helical compression springs.

CHAPTER SIX

CASE STUDY (1) - HELICAL COMPRESSION SPRING DESIGN

CHAPTER SIX

CASE STUDY (1) - HELICAL COMPRESSION SPRING DESIGN

6.1 Introduction.

The design of helical compression springs was chosen as a vehicle for embodying Expert Systems concepts in engineering design. As a result, the HELCOM program embodying the Microsynics knowledge representation system was developed and tested in a number of spring design tasks.

The design process embodies the expertise embodied in the guidelines laid down by the British Standards Institute on the design and specification of coil springs (Helical compression springs BS 1726 Part 1: 1964)[68] and the Spring Research Association [71]. It allows for the design of springs having wire diameters in the range 0.1 mm to 13.2 mm, the number of active (working) coils to be equal to or greater than four and using patented cold drawn carbon steel to BS5216:1975, [19]. Table 6.1 summarises some information on grades 1 to 5 (that is, the maximum allowable shear stress). The control module was written in Microsynics "native" language, the calculation module and user display were written in the Basic programming language.

This chapter describes in detail, the structure of HELCOM and its application to a number of test cases. To give a background to these test cases, the theory underlining the design of helical compression springs shall now be discussed in the next section.

TABLE 6.1: Grade of steel, duty and wire diameter range, [19][72].

6.2 The theory of Helical spring design

The type of helical compression springs under consideration are those made out of circular cross section wire formed into a cylindrical helix as shown in fig 6.1 below.
The maximum induced shear stress is given by the equation,

$$\sigma i = (K*8*F*c)/(\pi*d^2) \qquad (6.2.1)$$

Where K represents the Wahl factor. This "stress concentration" factor takes into account the effects of the wire curvature and transverse shear stress and is defined by Shigley [2], Hall et al [4] and Quayle [72].

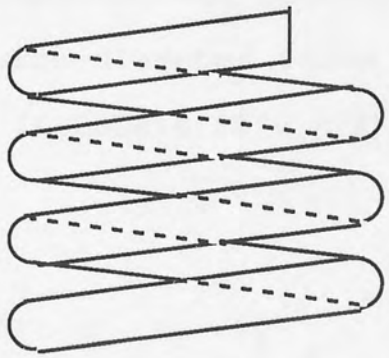


Fig 6.1 Helical spring

$$K = [(4c-1)/(4c-4)]+0.615/c \qquad (6.2.2)$$

The sub-factor (4c-1)/(4c-4) corrects for curvature of the spring wire and 0.615/c accounts for the transverse shear stress. The variable c, is the spring index. It is the ratio of the mean coil diameter, D and the spring wire diameter, d (Fig 6.2). That is,

$$c = D/d \qquad (6.2.3)$$

A spring having an index less than 2.9 is difficult to

coil, giving this parameter a minimum value for acceptability; an upper limit of 10.3 is given because it is difficult to obtain an accurate coil diameter for a spring with an index over this value. The combined effect of the limits on the index and the available wire diameter imposes a limit on the coil diameter. The variable, F (in equation 6.2.1) is the axial compressive force. The application of the full Wahl factor (eqn. 6.2.2) produces a conservative design [4]. The full Wahl factor was used in the program developed. The maximum allowable shear stress for preferred wire diameter sizes for patented cold drawn steel are given in BS5216:1976 [19].

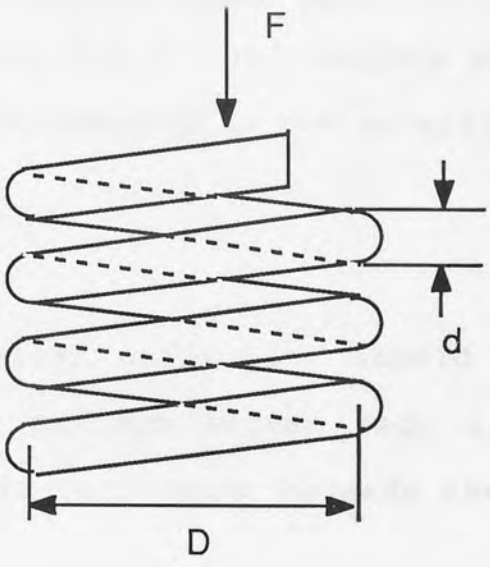


Fig 6.2 Helical spring showing the coil and wire diameters

The maximum deflection for a given load value, F is given

by the equation [4][73]:

$$\delta = (8FD^3n)/(d^4G) \qquad (6.2.4)$$

The variable, G is the modulus of rigidity.
The variable, n is the active number of coils.
The actual deflection for the specified spring rate, k and applied load, F is given by the equation;

$$\delta = F/k \qquad (6.2.5)$$

The spring rate is normally quoted in specifications. The actual deflection as defined by equation 6.2.5, should be less than the deflection under maximum allowable stress (as defined by equation 6.2.4). The maximum physical deflection however, can be represented by the equation [71];

$$\delta = 0.85(Fl-Lc) \qquad (6.2.6)$$

The maximum physical deflection should be less than the deflection under maximum stress (eqn 6.2.4). The spring will break if this deflection exceeds the deflection under maximum stress.
The variable, Fl is the free length. It is the length of the spring when not subjected to any load. For springs with closed and ground ends [71](Fig 6.3);

$$Fl = 0.41*n*D+1.5*d \qquad (6.2.7)$$

The variable, Lc is the solid length and represents the length of the spring when fully compressed.
Table 6.2 gives the relationships between the number of active coils, the solid length, and the spring end conditions. The spring end conditions are shown in Fig 6.3.

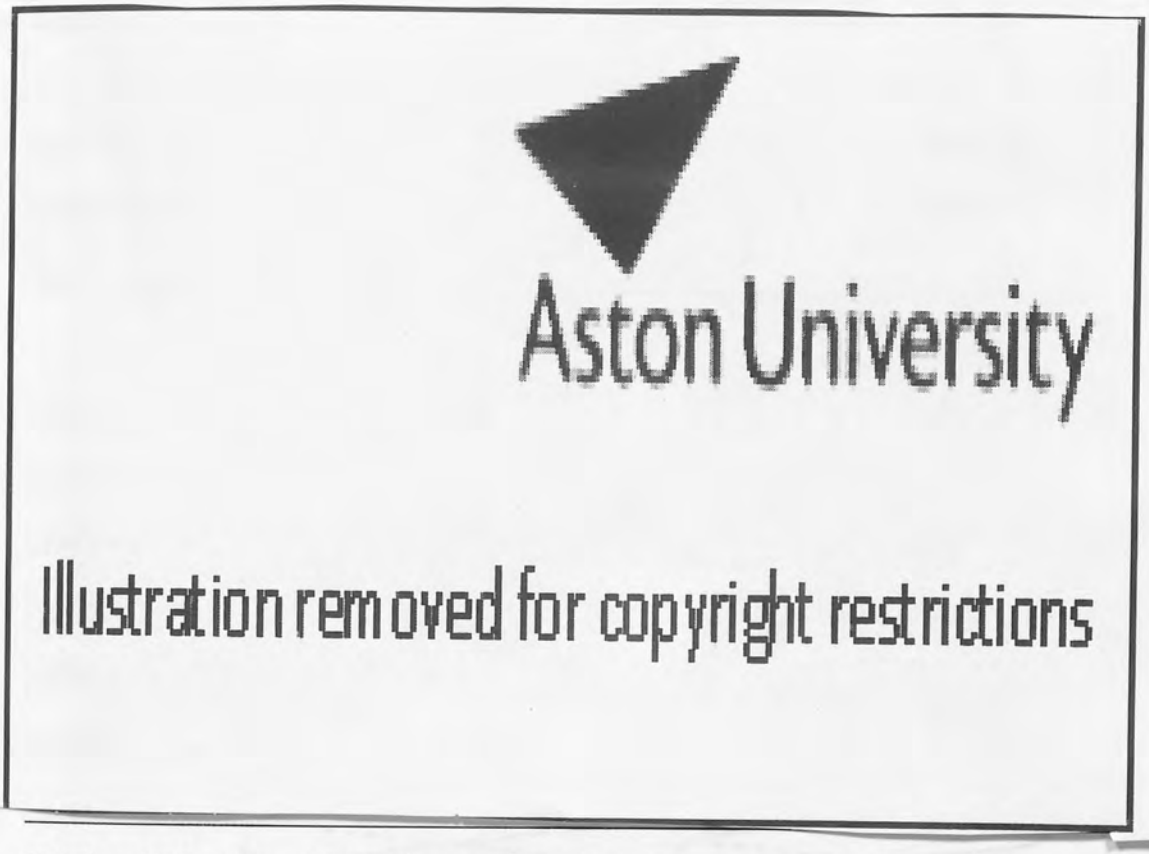


Table 6.2 The relationships between the number of active coils, the solid length and the spring end conditions [2],[71].

This relationships can be written in form of equations as;

$$N = n+EH \qquad (6.2.8)$$

$$Lc = (n+EH1)d \qquad (6.2.9)$$

Where EH = 0 or 2 depending on the spring end conditions (Table 6.1, column 2) and EH1 = 0,1,2 or 3 depending also on the spring end conditions (Table 6.2, column 3). EH and EH1 shall be referred to as constants in this thesis. Other geometric variables to consider in the design of helical springs are:

The operating length, OPL, is the length of the spring when no further deflection is possible due to factors such as the spring housing or the application of the maximum load.

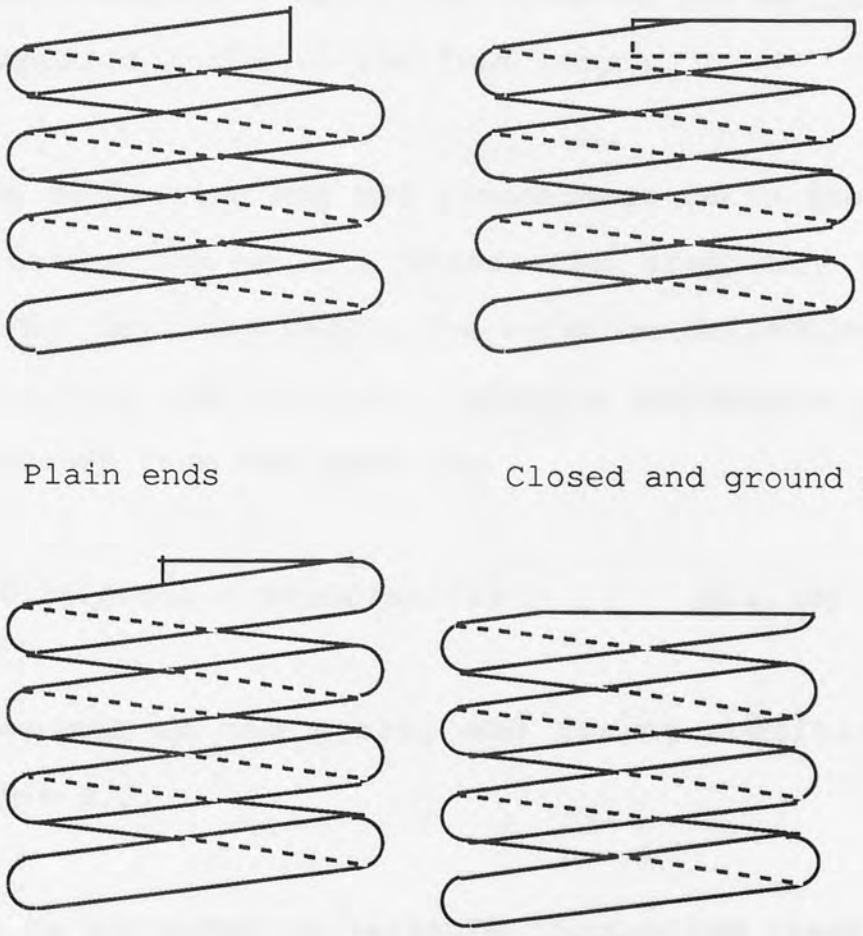


Fig 6.3 Types of spring ends

The length in position, LP is the length of the spring as fitted for operation. This may include the length on application of an initial load or without. The length in position is the free length if no initial load is applied.
The slenderness ratio SL, is the ratio of the free length to the mean diameter.
The relative deflection, RD is the ratio of the deflection due to the applied load F to the free length.

The relative deflection and the slenderness ratio are used in the determination of the transverse stability under static loading (buckling test). The relative deflection, RD must be less than the critical relative deflection which can be calculated from the equation;

$$RD = 0.881[1 \pm (1-6.89(CC/SL)^2)] \qquad (6.2.10)$$

where CC depends on the spring end fixing condition as shown in Table 6.3.

If a spring is subjected to variable/fluctuating loads, the modified Soderberg equation is applied to test for fatigue [4]:

$$[\sigma y/(\sigma m-\sigma y+(2*\sigma v*\sigma y/\sigma rel))] \geq 1 \qquad (6.2.11)$$

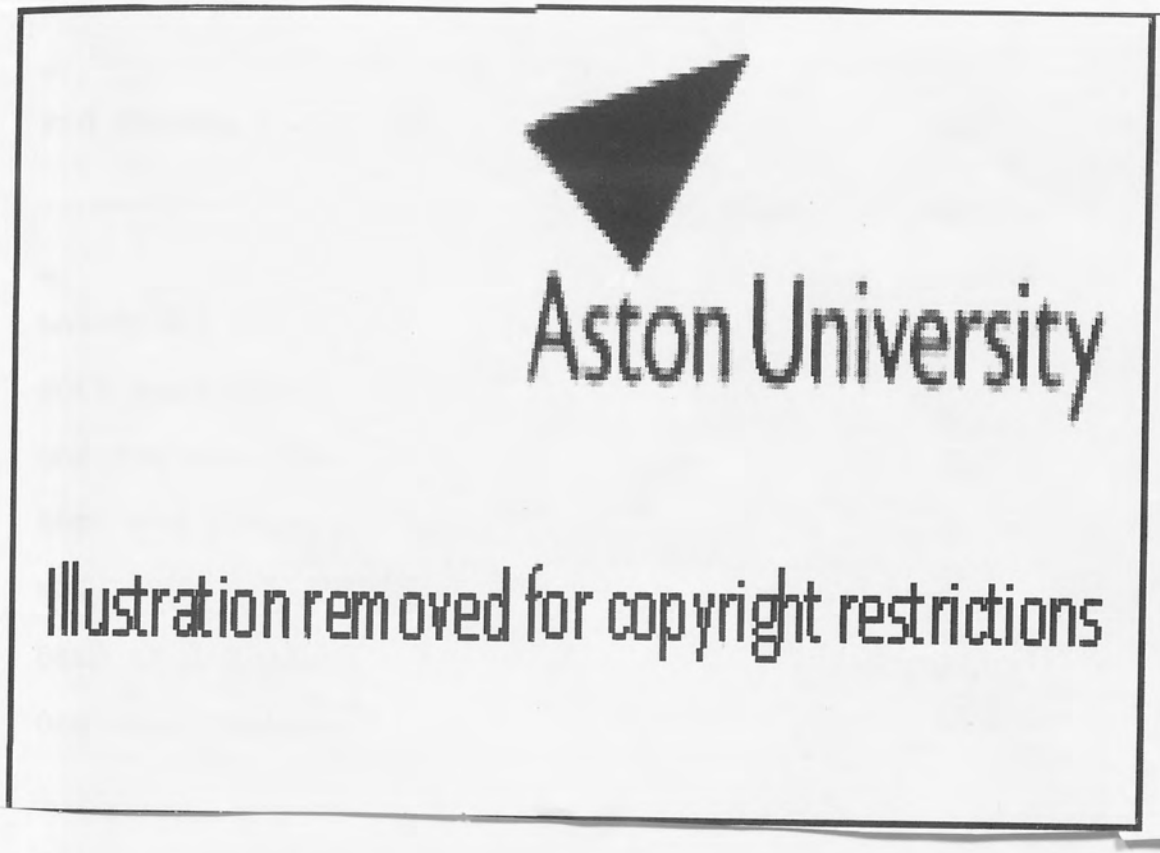


Table 6.3 Values of constant CC. [68]

Where σy is the yield stress of the spring material.

σm is the mean stress applied which is given by the equation;

$$\sigma m = K*8*Fm*c/(\pi d^2) \qquad (6.2.12)$$

Where Fm is the mean force as defined by the equation;

$$Fm = (Fmax+Fmin)/2. \qquad (6.2.13)$$

Fmax is the maximum applied force and Fmin is the minimum applied force.

σv is the variable stress which is given by the equation;

$$\sigma v = K*8*Fv*c/(\pi d^2) \qquad (6.2.14)$$

Where Fv is the variable force as defined by the equation;

$$Fv = (Fmax-Fmin)/2 \qquad (6.2.15)$$

σrel is the endurance strength (N/mm^2) for released loading which can be approximated from the equation [4][74];

$$\sigma rel = 530/(d/1000)^{0.2} \qquad (6.2.16)$$

Also, for variable loading, a surging test is recommended to ensure that the frequency of loading is not too close to the natural frequency. The natural frequency, fn (in cycles per minute) is estimated from the equation;

$$fn = 21500*d/(n*D^2) \qquad (6.2.17)$$

Having discussed the basic equation involved in the design of helical springs, the design procedure is now discussed.

6.3. Spring Design Procedure

The design of a new helical compression spring involves a consideration of the following factors.

* Space within which the spring must fit.
* The length as fitted for duty (with or without an initial load).
* The length of the spring when no further deflection is possible due to spring housing or maximum load.
* The spring rate and
* The environmental conditions such as temperature and corrosive atmosphere.

The designer uses the factors above to select a material and specify suitable values for the wire size, the number of turns, the mean coil diameter and free length, the type of ends and the spring rate needed to satisfy the force-working deflection requirements. The problem is not as simple as inserting numbers into procedures that will produce a complete set of results because of the interdependence of the variables.

There are usually only a limited number of wire sizes that can be used and restrictions on the minimum number of turns allowable. BS1726:part1:1964 [68] gives the basic guide to

compression spring design. The purpose of the standard is to clarify the dimensional specification of a spring by the designer; it also gives guidance on the allocation of sizes by reference to features which are of direct importance to the functioning of the particular assembly, bearing in mind that a spring can be completely geometrically specified by the four dimensions of wire diameter, free length, coil diameter and number of coils. BS5126:1975 [19] on the other hand gives the wire diameter sizes (in mm) for each grade of patented cold drawn carbon steel. It also gives a guide as to what sort of duty the spring may be applied.

There are many ways of creating a conventional spring design program; it usually comprises two or three display frames in response to which the user is required to give answers or data values perhaps to particular questions and the program will not proceed to the next stage. A conventional program proceeds through its analysis without consultation with the user and at the end of the process gives results without explanation. The process used by a conventional program is what the current author termed as "monopath" in comparison with the "multipath" (or network) structure used by Expert Systems. In a monopath approach, if a piece of information cannot be obtained by the program, the path is effectly cutoff (that is, the program cannot proceed). There is no conflict resolution strategy in such systems since the primary method is that of progressive substitution and hence one execution path.

In the program discussed later, the deflection under maximum allowable load (eqn 6.2.4), the actual deflection (eqn 6.2.5) and the maximum physical deflection (eqn 6.2.6) are compared and presented to the user for decisions. The deflection under maximum allowable load must be greater than or equal to the maximum physical deflection. If not, the stress in the spring wire exceeds the maximum allowable value and it is thus liable to break. When this condition is satisfied, the buckling, fatigue and surging test procedures are then executed.

Having discussed the design procedure, the knowledge based program architecture developed to satisfy the design equations and conditions is now discussed.

6.4 Architecture of Program (HELCOM)

Fig 6.4 gives the general block diagram of the program developed; it is called HELCOM. At the start of the program, the first screen display gives the option to:

a) be introduced to the program,

b) be introduced to Microsynics,

c) recall old data file for modification and

d) start the consultation.

The next process is the execution of the information acquisition phase. It entails the determination of constant values like EH, EH1, as will be described later (block labelled "CONSTANTS") and parameter values like d, D, etc.(block labelled "PARAMETER" and "PARAMETERS"). These

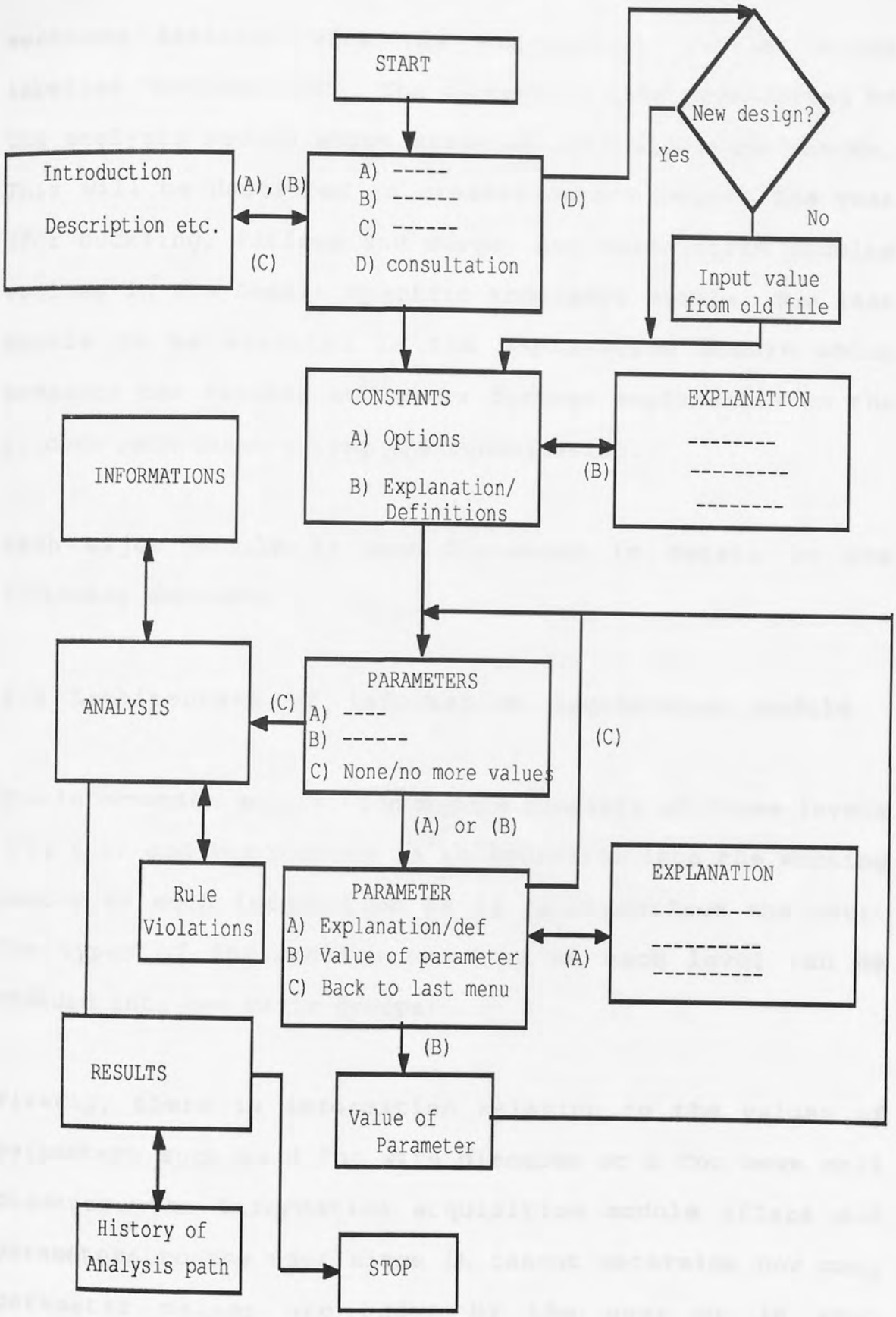


Fig 6.4 General architecture of HELCOM

sections interact with the explanation module (block labelled "EXPLANATION"). The process is then transferred to the analysis module which executes rule violation checks. This will be described in greater detail below. The test (for buckling, fatigue and surge) and calculation modules resides in the Domain Specific knowledge module. The last module to be executed is the explanation module which presents the results and gives further explanation on the process path taken during the consultation.

Each major module is now discussed in detail in the following sections.

6.5 Architecture of information acquisition module

The information acquisition module consists of three levels (Fig 6.5) and its purpose is to introduce into the working memory as much information as is required from the user. The types of information required at each level can be divided into two major groups:

Firstly, there is information relating to the values of parameters such as d for wire diameter or D for mean coil diameter. The information acquisition module offers all parameters to the user since it cannot determine how many parameter values are known by the user or in what combination(s) the known parameters appear. There are seventeen parameters present and theoretically, the user may have the value of none or all available.

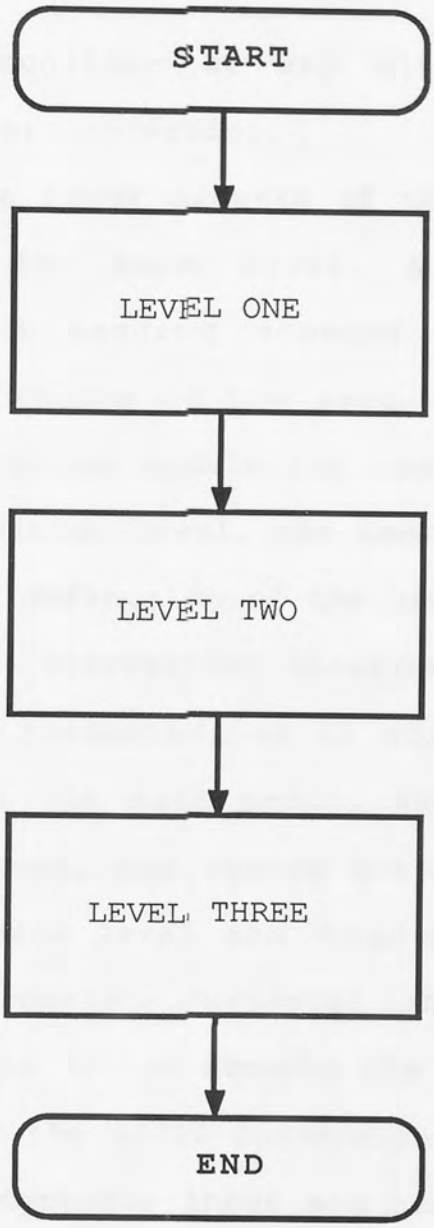


Fig 6.5 The number of levels in the program.

In the case where none is given, the user is informed that the design process cannot continue due to lack of information. There are seventeen different possibilities initially in trying to determine how many parameter values are given. For this reason, the information acquisition module was divided into the three levels mentioned; this reduces the number of combinations and alternatives the

system has to consider at any given time giving the advantage of faster processing.

Fig 6.6 gives the block diagram of the information module for parameters for each level. A parameter list is displayed with a heading showing the current level. Depending on the choice of the user, the system then goes into a sub-information module for the information chosen. At the sub-information level, the user has three options : a) looking at the definition of the parameter concerned and all other relevant information concerned with it; b) giving the value of the parameter, or c) going back to the main information level (or main menu). When the value of the parameter is entered, the system automatically returns to the main information level and displays the entered value alongside the appropriate parameter labelled. When the user offers a value, the system accepts the value as a character string and checks the ASCII conversion for the input to be sure it is an acceptable input and hence can be converted to a numerical value if necessary without error. The main reason for the main and sub-structure of the information module for the parameters is to allow the user to look up the definition of the parameters if he so wishes without being committed to offering a value. It is also easier to expand and modify the module with this format.

The questions relating to the determination of parameter values are divided into three levels and this has a direct bearing on the manner in which the analysis is carried out.

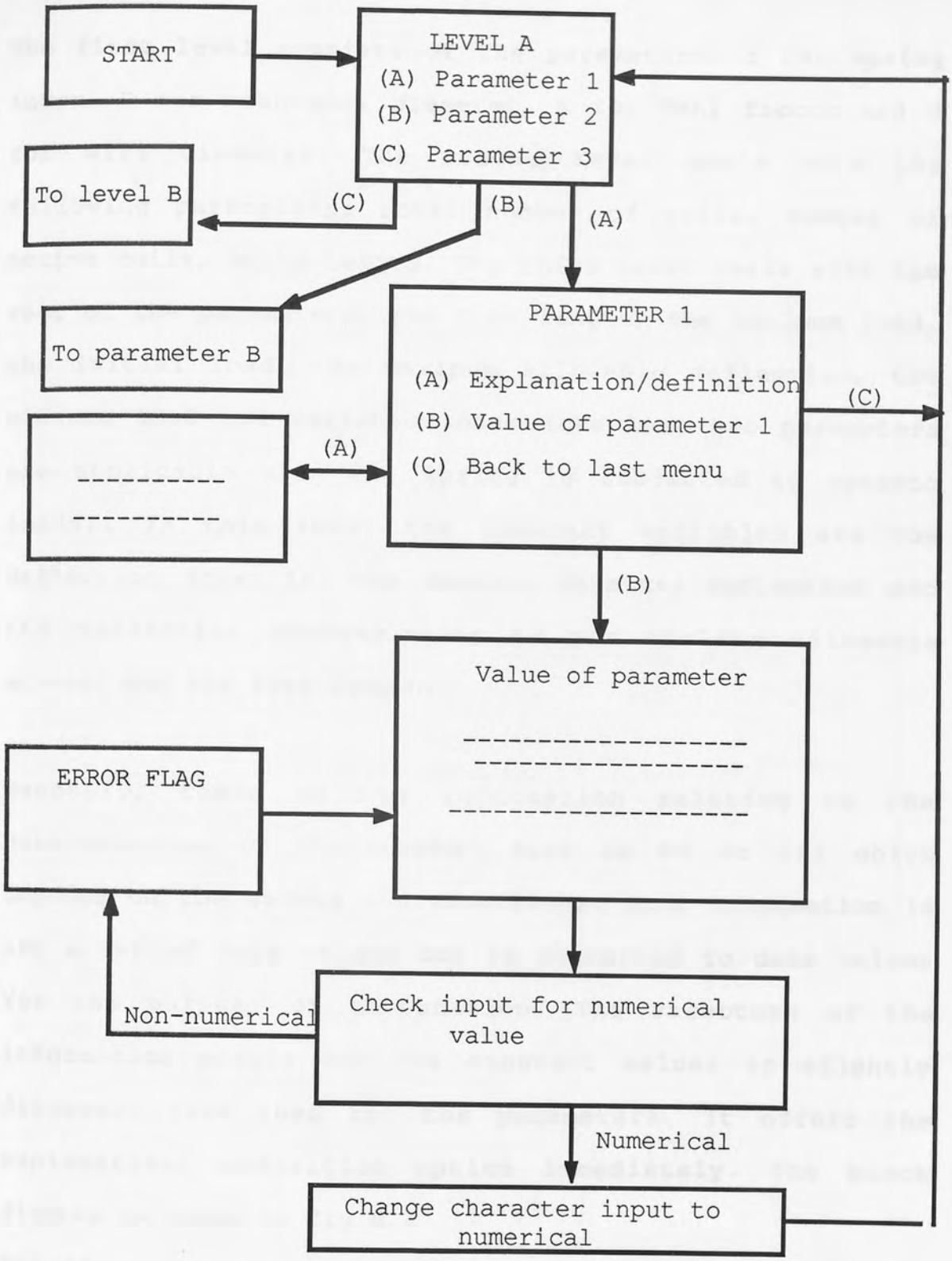


Fig 6.6 The architecture of the Information acquisition module (parameter).

The first level consists of the parameters; c for spring index, D for mean coil diameter, K for Wahl factor and d for wire diameter. The second level deals with the following parameters; total number of coils, number of active coils, solid length. The third level deals with the rest of the parameters; the free length, the maximum load, the initial load, the maximum allowable deflection, the minimum load and variable loads (the last two parameters are applicable when the spring is subjected to dynamic loads). In this level the dominant variables are the deflection (that is, the maximum physical deflection and the deflection corresponding to the maximum allowable stress) and the free length.

Secondly, there is the information relating to the determination of "constants" such as EH or EH1 which depends on the spring end conditions. Such information is not a set of data values but is converted to data values for the purpose of computation. The structure of the information module for the constant values is slightly different from that for the parameters. It offers the explanation/ definition option immediately. The block diagram is shown in Fig 6.7.

The first four questions relate to the determination of constants and the selection of the grade of steel that should be recommended; this in turn has a bearing on the wire diameter sizes available. The constants EH and EH1 which relate to the spring end finished conditions, ground, plain, closed etc., are determined in this section; the

selection of the end condition being from a menu. The constant CC relates to the spring's lateral stability and hence to buckling.

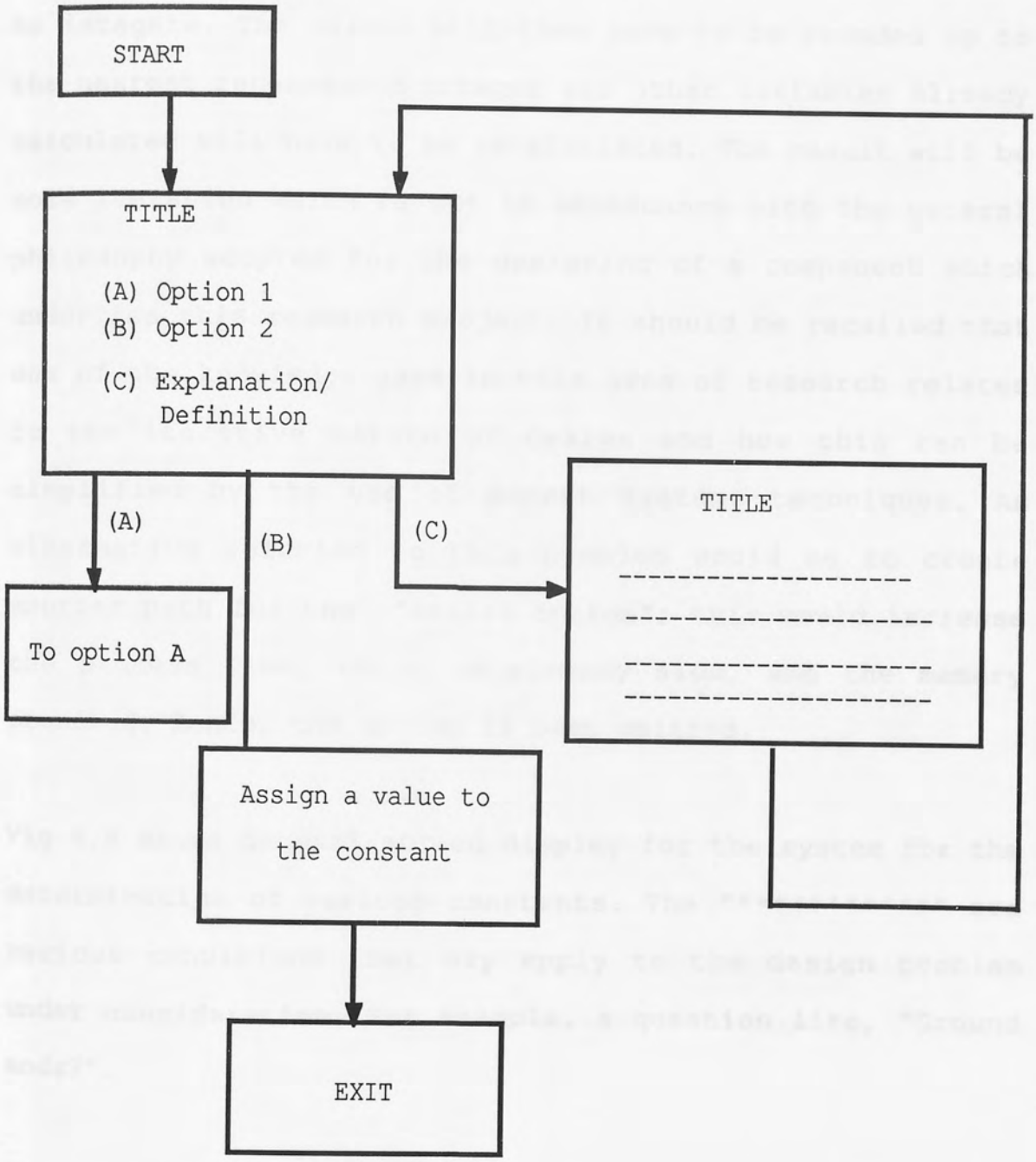


Fig 6.7 Architecture of Information acquisition module (constant).

The questions asked in this section do not give the option of selecting the "unknown" (or do not know). The reason for this was that if the values of EH and EH1 are to be determined by calculation, the values calculated will not be integers. The values will then have to be rounded up to the nearest recommended integer and other variables already calculated will have to be recalculated. The result will be more iteration which is not in accordance with the general philosophy adopted for the designing of a component which underlies this research project. It should be recalled that one of the knowledge gaps in this area of research relates to the iterative nature of design and how this can be simplified by the use of Expert Systems techniques. An alternative solution to this problem would be to create another path for the "denied option"; this would increase the process time, which is already slow, and the memory required. Hence, the option is best omitted.

Fig 6.8 shows general screen display for the system for the determination of various constants. The "**********" are various conditions that may apply to the design problem under consideration. For example, a question like, "Ground ends?".

The user screen display for the information module is structured in such a manner as to give the user information about the state of the system and which "node" and subfunction it is displaying. This helps in the debugging and location of syntax errors. The structure of the user

TITLE

FMDL

B$MOD*

A) **********

B) **********

C) **********

D) **********

E) NONE/NO MORE VALUE

(Other options N,H,X or Z)

Select (A,B,C,D or E)

Fig 6.8 A typical screen display for the determination of the various constants influencing a spring design.

interface is shown in Fig 6.9. It is desirable to display the Node names and Function names to let the user know that processing is taking place and provide some assurance. The node names, identify the eight program "batches" or listings

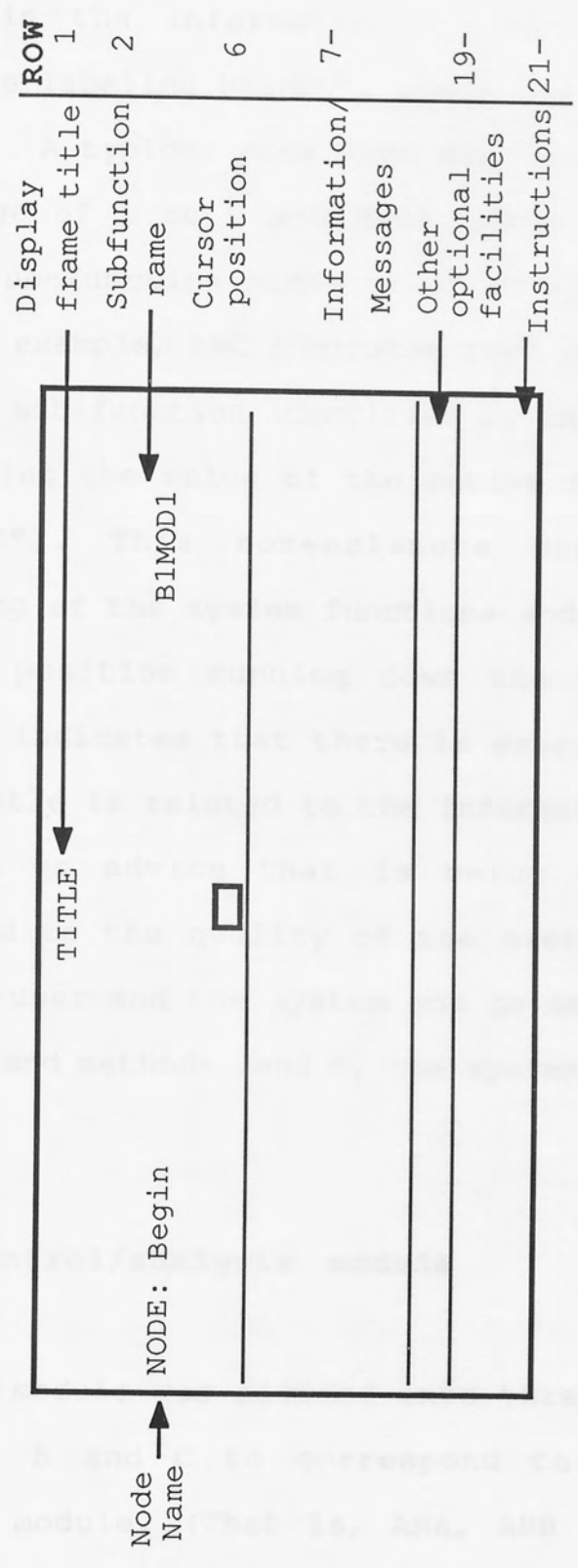


Fig 6.9 User screen display

contained in the information acquisition program. The listings are labelled HELMOD*, where * has a value ranging from 1 to 8. A typical node name may be F$MDL*. Where $ is in the range of 1 to 6 and MDL* means batch number (Fig 6.8). The sub-function names start with the variable code symbol (for example, BWC indicates that the process control is within a sub-function identified by the first letter "B" and processing the value of the active coil with variable symbol "WC"). This nomenclature contributes to an understanding of the system functions and processing paths. The curser position running down the left side of the screen also indicates that there is execution in progress. The frame title is related to the information sought or the information or advice that is being given. All these features add to the quality of the overall communication between the user and the system and to an understanding of the process and methods used by the system.

6.6 The control/analysis module

The control module was divided into three levels labelled analysis A, B and C to correspond to the information acquisition module. (That is, ANA, ANB and ANC). As was mentioned earlier, each level of the information acquisition module relates to a group (or set) of primary rules (see table 6.4) and the manner in which the analysis is carried out.

LEVEL	RULE No	PRIMARY RULES
ONE	A1	$c = D/d$
	A2	$K = (4c-1)/(4c-4) + 0.615/c$
TWO	B1	$Lc = (n + EH1)d$
	B2	$N = n + EH$
THREE	C1	$\delta = (8FD^3n)/(d^4G)$
	C2	$F = (\sigma\pi d^2)/(8cK)$
	C3	$\delta = 0.85(Fl - Lc)$
	C4	$Fl = 0.41*n*D + EH*d$
	C5	$k = F/\delta$

Table 6.4 Levels and associated primary rules

The basic approach to the analysis was to obtain a tree like path (within the general network) for each level of the analysis. Each path is distinct and thus allows the system to explain and recall the path to the solution; in this way, the reasoning process of the system is made explicit.

The steps used for each level were :

i) The determination of the number of parameters given. Out of the seventeen parameters involved in the total design process only eleven are related through the primary rules and it is these that are examined at the three levels. If the analysis was not divided into three stages, then the number of branches at the first node (which determines the number of parameters given) would be 12 (that is, from zero number of parameters available to eleven parameters). But with each level having a smaller number of parameters (see Table 6.5), the branching is less and each level, in effect, constitute a module and is expandable as more knowledge is encoded into the system. Figure 5.5 illustrates the branches in level 1. This level has the following parameters, c, K, d and D. There are x+1 branches at this level (where x = number of parameters associated with the primary rules). Hence there are five branches in this case.

ii) The determination of the specific parameters known. Still considering Fig 5.5, the branch labelled 2 indicates that there are two parameters given (that is, D and d, D and K, c and D, c and d, c and K and lastly d and K). If this branch were to be followed, then the number of branches going out would be $4C_2$ = 6. (that is, 4!/2!2! = 4.3/2 = 6). If the analysis

search path had not been segregated into levels, the number of branches at the first node would have been $11C_2 = 55$. (that is, 11!/9!2! = 55).

Equations	Parameters	Parameters from previous level	Level
c = D/d K=Kc+Ks Kc=(4c-1)/(4c-4) Ks = 0.615/c	c,d,D,K	-	1
N = n + EH	N,n,EH1	d	2
Lc = (n + EH1)d	Lc,EH		
$\delta = (8FD^3n)/(d^4G)$	δ,P,S,FL	D,d,n,c K,LC,EH	3
$k = P/\delta$ $F = (\sigma\pi d^2)/(8cK)$ $\delta = 0.85(Fl - Lc)$ FL = 0.4nD + EH.d			

Table 6.5: Table showing equation (primary rules), parameters and levels

iii) The Execution of the conflict resolution strategies when applicable.

This phase of the analysis involves the substitution

of values into relevant equations to ensure that there is no violation of the rules. This action usually applies to equations where all parameters involved are given. In some cases, all but one parameter values are given for a specific rule and, by substitution, the unknown parameter value is calculated and checked for violation of the concerned primary rule. However, since there are several equations involved at each level, there is a priority level attached to each rule and also to each parameter value. This depends on the state of the search process and on the stage of the analysis. For example, in level one;

$c = D/d$ - Rule A1

$K = ((4c-1)/(4c-4)) + 0.615/c$ - Rule A2

If all parameter values but one (say c) are given, then c can be calculated from both rules and, unless the values are the same, one rule will be violated if the value calculated from the other is accepted. In this situation, a decision has to be made as to which value must be accepted and why. Because the number of parameters associated with rule A1 is higher, and they carry higher priority values partly because of the restriction placed on them, the value of parameter c calculated from that rule is accepted and the value of K is discarded and recalculated. The user is, however, notified of this.

iv) The correction for violations of principal and minor rules and offering alternative values for unacceptable parameter values.

v) The checking of calculated parameters to ensure that they do not constitute a violation of the rules.

The primary rules at each level are in a manner that allows for a progressive substitution strategy to be carried out if and when required. If Table 6.5 is examined closely, it will be observed that new parameters are introduced at each level and the parameters from previous levels (if applicable) will have been calculated or determined by then. The values of all parameters at a given level are determined before the analysis proceeds to the next level.

6.7 The domain specific knowledge

6.7.1 Tests:

Possible designs which emerge from the analysis module must be tested against buckling, fatigue (if subjected to variable load), and surging criteria. If a proposed design fails the buckling or fatigue tests, the user is notified immediately and given suggestions as to how modifications can be made. The user is strongly advised to reject the design. If the surging test is failed, the user is not advised to reject the design but will however, be advised

on what actions to take to modify the design. The buckling test is for springs that are perfectly straight and with strictly straight and parallel faces [68]. BS17216:part 1; stated that this test is seldom or never satisfied, hence the user can only be advised on what to do to rectify a design that fails the test.

6.7.2 The calculation module

This module calculates the unknown parameters. It manipulates equations to obtain unknown parameter values. This module is invoked only after the analysis module has determined the known and unknown parameter values.
The analysis module is divided into three sections as has already been mentioned. After the analysis has identified the number of parameter values offered in a particular level and in what combination, the calculation module ensures that all unknown parameters associated with the level are calculated. For example, in level 1, there are four parameters associated with this level. They are c,d,D and K. The two primary rules are A1: c = D/d and A2: K = [{4c-1)/(4c-4)]+0.615/c. Therefore the calculation module can only be called if any two of the parameter values associated with rule A1 are known. The third can be calculated. Also, if any one of the parameter values associated with rule A1 is known and any one parameter value associated with rule A2 is known, then the calculation is called. If no parameter value is offerd or only one parameter value is offered (that is, c,d,D or K)

or all parameter values are offered, then the calculation module will not be called.

For level 2; The two primary rules associated with this level are B1: Lc = (n + EH1)*d and B2: N = n + EH.
The parameter values associated with this rules are Lc, n, N, EH, EH1 and d. But, EH, EH1 and d will have been determined earlier in level 1. Therefore, only N, n, and Lc will be analysed by the analysis/control module for this level. After the analysis/control module has determined the number of parameter values offered by the user and in what combinations, the calculation module will then be called except if Lc, n and N were all offered or these parameters were not offered by the user.

In level 3, the load at maximum allowable stress is calculated from rule C2. For springs with closed and ground ends, the maximum free length is calculated from rule C4. The parameters that are analysed at this level are the free length Fl, the spring rate k, the load F and the deflection δ.

If none of these parameter values is known, then the load at maximum allowable stress is used to calculate the deflection at maximum allowable stress from rule C1. This value is then used to calculate the free length from rule C3 and compared with the value calculated from rule C4 only if the spring ends are closed and ground. The spring rate is calculated from rule C5.

If only the load is offered, its value is compared with the calculated load at maximum stress. The lower of the two values is used for the calculation of the deflection using rule C1. Fl and k are calculated from rules C3 and C5 respectively.

If only the free length is offered, it is used to calculate the maximum physical deflection by rule C3. The value is compared with the deflection calculated using the maximum allowable load from rule C1. The maximum physical deflection should be the smaller value. If it is not, the user is warned that the spring may fail before attaining its maximum physical deflection. The lower of the two values is used to calculate the spring rate.

If the spring rate only is offered, the actual deflection is calculated from rule C5 and is compared with the deflection at maximum stress (rule C1). The actual deflection must be lower than the deflection at maximum stress. The actual deflection is used to calculate the free length (rule C3). If the spring end is closed and ground, the maximum physical deflection is calculated (rule C3) and compared with the actual deflection and the deflection at maximum stress.

If the deflection only is offered, the spring rate is calculated useing rule C5, and the free length is calculated using rule C3. The given deflection must be less

than the deflection at maximum stress.(rule C1).

If Fl and F are known, F must be less than the maximum allowable load (rule C2). The lesser of the two values is used for the deflection calculation using rule C1. The maximum physical deflection is calculated using rule C3 which must be less than the deflection calculation using rule C4, for closed and ground end springs only. The lesser of the deflections calculations is used for the spring rate calculation.

If Fl and k are the only known parameters, the maximum physical deflection is calculated (rule C3), the deflection at maximum stress is calculated (rule C1) and the actual deflection is calculated (rule C5). These deflections are then compared.

If the free length and deflection are the two known parameters, the maximum physical deflection and the deflection at maximum stress are calculated (rules C3 and C1 respectively). These are compared with the given deflection. The lowest of the three is used for the spring rate calculations.

If the load and the spring rates are known, the given load must be less than the load at maximum stress. The lower of the two is used for the deflection calculations (rules C1 and C5). The free length is calculated from the lower of the deflections.(rule C3).

If the load and deflection are given, the same procedure as in the previous paragraph is followed. But the spring rate is calculated from the lower of the load and deflection values.

If the spring rate and the deflection are given, then the load is calculated from rule C5. This load should be less than the load at maximum stress. The given deflection should be less than the deflection at maximum stress. The lower of the two deflections is used for the free length calculation.

For any three known parameter values, the general rules are:

1) if the deflection is given, it must be lower than the deflection at maximum stress and the maximum physical deflection.

2) If the load is given, it must be less than the load at maximum stress.

3) If the spring rate is given, than it must be used to calculate other parameters. It has the highest priority amongst the parameters.

4) If the free length is given, the maximum physical deflection must be calculated and used for further calculations or comparisons. The given free length must be less than that calculated using rule C4 where applicable.

6.8 The explanation and self-knowledge module

The results module and the history-of-analysis modules reside in the explanation module. These modules are now described.

6.8.1 Results

The results displayed are recommended values for each parameter. They are given to two decimal places but the calculations are done to four decimal places. The result can be stored if required and recalled for modification if necessary. If the latter situation arises, the values recalled are regarded as "given" values (that is, values input by the user). Some parameters, such as the maximum physical deflection, are not re-inputted because they will be recalculated automatically as part of the design process.

The Results module has the facility to recall the history of the analysis. This gives a detailed step-by-step account of the process and hence enables the user to understand the logic and reasons for the total process.

6.8.2 History of Analysis.

This module aids the explanation module in making the process transparent to the user. The facility is only available at the end of the consultation and it is basically a log of each process executed by the system.

Since each process is controlled by a node and sub-function, the description of the process can be obtained from the log of the nodes and subfunctions used during the consultation. This log file gives the nodes/subfunctions used in the order of execution. A file containing the description of these nodes/sub-functions will then be list printed for the user's information.

6.9 Example problems and discussion of results

This section gives four examples of consultations using Helcom. The computer screen displays will be shown in the sequence they occur for the first two examples. The two examples were chosen to show some of the features of the system, including the sort of information and reasoning it adopted. The reasoning methods can be modified to suit the user. If the system is to be used for educational purposes, the explanation of the reasoning strategies will have to be more detailed than if used in a practical situation where the result is of importance not the means by which it was arrived at.

6.9.1 Spring Design example problem (example 1)

(i) Suppose a spring, pivoted at one end, fixed on the other and not laterally guided, was required. The ends are squared and it is required that:

spring rate = 11 N/mm

wire diameter = 6 mm

minimum deflection = 20 mm

spring index = 6

Starting with trial values for the following parameters;

Mean coil diameter = 18 mm

Number of active coils = 24

Free length = 196 mm

Load = 180 N

Consultation begins with an introduction to the system and proceeds to the design process itself. The user has the option whether or not to offer any of the parameter values presented to at each level. Fig 6.10a shows a typical screen display when seeking information about parameter values. Option "E" is selected if the user wishes not to offer any value or has offered and wishes to proceed on to the next level. In this example, the user has parameter values for spring index, mean coil diameter and wire diameter. First selecting option "A", the spring index sub-module is entered and the screen display is shown in Fig 6.10b. The first option on this screen (that is, "A") gives the definition and relevant explanation about the spring index. When option "B" is selected, the system is then ready to accept a value for the spring index. Option "C" returns the process to the main level and the display on Fig 6.10a reappears. Fig 6.10c shows the screen display for option "B". The "*" shown is the cursor position and

the "+" gives the position where the entered value is echoed for user confirmation. If the value is confirmed as an acceptable entry, then the process goes back to the main screen display shown in Fig 6.10a. If the value is not confirmed as acceptable entry, then the value is asked for again. The screen displays for the other two levels are similar.

LEVEL 1

NODE F1MDL4 B10MOD4

A) SPRING INDEX

B) SPRING COIL DIAMETER

C) WAHL FACTOR

D) SPRING WIRE DIAMETER

E) NONE/NO MORE VALUES

(Other options N,H,X or Z) *

Select (A,B,C,D, or E)

Fig 6.10a Example 1 (level 1): Information acquisition screen display (sub-information level)

SPRING INDEX

NODE A2 B16MOD4

A) EXPLANATION/DEFINITION

B) VALUE OF SPRING INDEX

C) BACK TO LAST MENU

(Other options N,H,X or Z) *

Select (A,B or C)

Fig 6.10b Example 1 (level 1): Information acquisition screen display

VALUE OF SPRING INDEX

NODE A4 BSI

*

Please enter the value of the

Spring index

(Note : The maximum acceptable value is 10.3 and)

(The minimum acceptable value is 2.9)

+

Type HELP and press RETURN key for assistance

Press return key after entering value

Fig 6.10c Example 1 (level 1): Information acquisition screen display (Parameter value input)

The analysis phase of the design process starts by re-checking parameter values for rule violations and informing the user accordingly. The reasoning for this is that, at this stage of the process, the system has not analysed how many parameters were offered and in what combination. If all the parameter values associated with a primary rule (for example, c = D/d) were offered, as in this case, then a rule violation check is necessary. The information displayed when a rule is violated, as in this case, is shown in Fig 6.11a. Options are then offered by the system. Fig 6.11b shows a screen display giving the two options offered by the system. The value of the wire diameter is kept on both options. This parameter has priority over the other three parameters concerned with this level. This is because it is connected with the shear stress values (BS5216:1975 [10]) and is thus restricted. Option "A" recalculates the spring index and option "B" recalculates the coil diameter. Option "B" is selected. The second level contains no violation and hence the system does not need to display any information but proceeds to the third level. At the beginning of the analysis of this level, the system informs the user of its intentions and a brief description of its reasoning strategies because of the complexity of the decisions which may be required in arriving at a suitable design. Assuming the user wishes to have this information, Appendix 4 gives the sequence of information screen displays. The rule C5, $k = F/\delta$, is violated as shown in Fig 6.12.

RULES A1 VIOLATED

NODE ANAA BANAA1

Rules A1 have been violated

Rule A1: c = D/d

Where

c = Spring Index

D = Mean coil Diameter

d = Wire diameter

(Other options N,H,X or Z)

Press any key to continue *

Fig 6.11a Example 1 (level 1) Rule violation Information screen display (First scroll), showing rule A1 violated.

RULES A1 VIOLATED

NODE ANAA BANAA1

A)	Spring Index	(calculated)	= 3.00
	Coil diameter	(given)	= 18.00 mm
	Wire diameter	(given)	= 6.00 mm
B)	Spring Index	(given)	= 6.00
	Coil diameter	(calculated)	= 36.00 mm
	Wire diameter	(given)	= 6.00 mm

(Other options N,H,X or Z)

Select (A or B) *

Fig 6.11b Example 1 (level 1) Rule violation information screen display (Second Scroll)

RULES C5 VIOLATED

NODE ANCH BANCH1

Rule C5 has been violated

Rule C5: k = F/DFL

Where

k = Spring rate

DFL = deflection under load F

F = load

The spring rate calculated from known variables does not correspond to the given spring rate.

(Other options N,H,X or Z)

Press any key to continue *

Fig 6.12 Example 1 (level 1) Rule C5 violation information screen display.

Screen display information giving reasons why the rule is violated is shown in Fig 6.13a. Figs 6.13b and 6.13c shows screen display informing the user what the intentions of the system are. The spring rate (if given by the user) has priority over any other parameter in this level. This is because it is usually quoted in specifications (BS1726 part1:1964 [52]) and its value is therefore retained. Fig 6.13d shows the screen display informing the user that the condition for retaining the entered load has been satisfied; that is, it should not exceed the maximum load. Figs 6.13e and 6.13f shows the screen display informing the user that the deflection has been calculated from two different equations. The results of these calculations are displayed on the screen illustrated in Fig 6.13g. Since the applied load is static, only the buckling test is carried out. If the test is failed there will be information to that effect. In this case, the test is passed so the results are presented as shown in Figs 6.14a and 6.14b. These results will be compared with a hand calculation in the next section. The system tries at all stages to check and cross check the information it has to be sure it does not come to an unreasonable decision, answer or results that make no sense and have no interpretation. The system is both advisory and decision making so that it is useful as a teaching tool and a serious design tool.

A hand calculation is now given for the example;

$c = D/d = 6 \neq 18/6$ hence rule A1 was violated.

INFORMATION

NODE ANCV BANCV1

The entered spring rate is different from the calculated spring rate from Rule C5. The entered spring rate should be the same as that calculated from the given maximum load and the given maximum deflection.

The given spring rate = 11 N/mm
The calculated spring rate = 9 N/mm
The given maximum load = 180 N/mm
The given deflection = 20 mm

(Other options N,H,X or Z)

Press any key to continue *

Fig 6.13a Example 1 (level 3) Information screen display (first scroll)

INFORMATION

NODE ANCV BANCV1

The given spring rate will be retained

ALSO The maximum load will be retained

IF It does not exceed the load at maximum stress

OTHERWISE The value of the maximum deflection will be retained

PROVIDED It does not exceed the deflection at maximum load.

(Other options N,H,X or Z)

Press any key to continue *

Fig 6.13b Example 1 (level 3) Information screen display (second scroll)

INFORMATION

NODE ANCV BANCV1

AND The value of the maximum deflection does not exceed the value of the deflection at maximum stress.

BUT IF The two conditions stated above are not satisfied.

THEN The load at maximum stress will be considered as the maximum recommended load.

(Other options N,H,X or Z)

Press any key to continue *

Fig 6.13c Example 1 (level 3) Information screen display (third scroll)

INFORMATION

NODE ANC421Q

BANC421

The given load will be retained as it does not exceed the load at maximum stress. The given deflection however, will be discarded.

The entered value of the maximum load = 180 N

The load at maximum permissible stress = 2.73 kN

(Other options N,H,X or Z)

Press any key to continue *

Fig 6.13d Example 1 (level 3) Information screen display

INFORMATION

NODE ANCB BANCB1

---------- Deflection calculation ----------

The deflection at maximum allowable stress has been calculated using Rule C1.

Rule C1 : $DFL = 8*F*D^3*n/d^4*G$

where F = load at maximum stress

D = Mean coil diameter

n = number of active coils

d = spring wire diameter

G = Modulus of rigidity

DFL = deflection at maximum stress.

(Other options N,H,X or Z)

Press any key to continue *

Fig 6.13e Example 1 (level 3) Information screen display (Physical deflection calculation)

INFORMATION

NODE ANCC BANCC1

---------- Deflection calculation ----------

The maximum physical deflection was calculated using Rule 8.

Rule C3 : DFL = 0.85*(Fl-Lc)

where Fl = Free length

Lc = Solid length

DFL = deflection at maximum stress.

(Other options N,H,X or Z)

Press any key to continue *

Fig 6.13f Example 1 (level 3) Information screen display (Deflection at maximum stress)

INFORMATION

NODE ANCB BANCB1

---------- Deflection calculation ----------

The maximum physical deflection is less than the deflection at maximum stress, hence the load that will be applied to attain the maximum physical deflection will be the recommended maximum load and will be calculated from Rule C3.

The value of the deflection at maximum stress = 235.77 mm
The value of the maximum physical deflection = 28.90 mm
The maximum recommended load = 334.49 N

(Other options N,H,X or Z)

Press any key to continue *

Fig 6.13g Example 1 (level 3) Information screen display (deflection advice)

RESULT

NODE RESULT1 BRES1

Recommended freelength	= 196.00 mm
Solid length	= 162.00 mm
Spring Rate	= 11.00 N/mm
Spring wire diameter	= 6.00 mm
Mean Coil diameter	= 36.00 mm
Spring index	= 6.00
Wahl factor	= 1.25
Total number of coils	= 26
Active number of coils	= 24
Maximum load	= 334.49 N

Press a key to continue *

Fig 6.14a Example 1 (result module) Information screen display (first Scroll)

RESULT

NODE RESULT1 BRES1

Maximum deflection	= 28.9 mm
Deflection at maximum stress	= 235.77 mm
Maximum physical deflection	= 28.9 mm
Maximum shear stress	= 1.45 kN/mm^2
Load at maximum stress	= 2.73 kN
Shear modulus	= 80.0 kN/mm^2

Press a key to continue *

Fig 6.14b Example 1 (result module) Information screen display (second Scroll)

For option A, c = 18/6 = 3 (D = 18, d = 6)

For option B, D = 6*6 = 36 (d = 6, c = 6)

Since option B was chosen, d = 6, c = 6, D = 36.

and the value of the Wahl factor will be;

K = [(4c-1)/(4c-4)+0.615/c]

= [(4*6-1)/(4*6-4)+0.615/6] = 1.25.

The solid length was calculated from Lc = (n + EH1)*d;

EH1 = 3 and EH = 2 (from table 6.2), since the ends are squared. Therefore, Lc = (24+3)*6 = 162.(rule B1)

The total number of coils = 24 + 2 = 26 (rule B2).

The load at maximum stress (which is 1450 N/mm^2 [BS5216]) is given by rule C2;

F = (1450*3.142*6*6)/(8*6*1.2525) = 2.73 kN.

Using rule C1, the deflection at maximum stress is

δ = (8*2727.7301*36*36*36*24)/(6*6*6*6*80000)

= 235.68 mm.

The maximum physical deflection is 0.85*(Fl-Lc) = 28.90 mm.

The maximum deflection is now used to calculate the maximum recommended load using rule C1. That is,

F = (d*d*d*d*80000*28.9)/(8*24*36*36*36) = 334.49 N.

Rule C5 was violated since the calculated rate is not equal to the given rate; that is k = F/d = 180/20 = 9 N/mm.

The buckling test was passed. (the buckling load is given by the equation [68]: $0.881[1\pm(1-6.89(CC/(Fl/D))^2)]^{0.5}$ = 1.2277 or 0.2124. (CC = 0.5; Table 6.3)). The slenderness ratio of the spring is 28.9/196 (δ/Fl) = 0.1474 which is lower than 0.2124, hence the spring is stable.

It can be observed from this example that the results obtained by the system are accurate to at least one decimal place to the result obtained by hand calculations.

6.9.2 Spring Design Example problem (Example 2).

Suppose a spring similar to that in example 1 is now subjected to dynamic load. The spring is pivoted at one end, fixed on the other and laterally guided. The spring is to fit into a spigot of diameter 12 mm and have open ends. Other specifications are that;

wire diameter	= 6 mm
minimum deflection	= 10 mm
Minimum load	= 100 N

Starting with trial values for the following parameters;

Mean coil diameter	= 18 mm
Number of active coils	= 24
Free length	= 160 mm
spring index	= 6
Wahl factor	= 1.3
solid length	= 144 mm

From the specification, the constant EH value will be 0 (because the ends are plain/open) and the constant EH1 value will be 1 (from table 6.2). Since the values of all parameters involved in level 1 are known, the primary aim at the beginning of the analysis will be to check for

primary rule violation. The two primary rules concerned are Rule A1: c = D/d and Rule A2: K=(4c-4)/(4c-1)+0.615/c. At this point the system informs the user of the violation. The Figure 6.15a shows the information screen display. Fig 6.15b informs the user of the priority levels of the equations involved with the level. Rule A1 has priority over rule A2. The next figure (Fig 6.15c) shows the two options open to the user. It can be seen that in both the options, the wire diameter value is retained and either the mean diameter calculated with the spring index value retained or the spring index value retained and the mean diameter recalculated. Option B is selected.

The consultation then proceeds to level 2. At this level there are two primary rules involved.
They are Lc = (n + EH1)d and N = n + EH. The parameter values d, EH1 and EH have previously been determined. The parameter values N and Lc are given. The program proceeds in the standard manner of checking for rule violation. The messages shown on Fig 6.16a are displayed for the users information as to the state of the process. It informs the user that there is no priority of one rule over the other. Fig 6.16b reminds the user of the rules and the meaning of each parameter code. Two options are again presented to the user for decision. Since EH = 0, then N = n. Therefore, in option A, the number of active coils is retained and the solid length calculated. In option B, the solid length is retained and the number of active coils calculated. Option B is selected.

RULES A1 AND A2 VIOLATED

NODE ANA441

BANA441

Rules A1 and A2 have been violated

Rule A1: c = D/d

Rule A2: K = (4c-4)/(4c-1)+0.615/c

Where K = Wahl factor

c = Spring index

D = Mean coil diameter

d = Wire diameter

(Other options N,H,X or Z)

Press any key to continue

Fig 6.15a Example 2 (level 1) Rule violation information screen display (First scroll)

RULES A1 AND A2 VIOLATED

NODE ANA441

BANA441

Rule A1 is given priority over rule A2.

(i.e. The program seeks not to violate rule A1 at the expense of violating rule A2.)

The program will discard the value of the Wahl factor given and calculates it from Rule A2.

(Other options N,H,X or Z)

Press any key to continue

Fig 6.15b Example 2 (level 1) Rule violation information display screen (Second scroll).

RULES A1 AND A2 VIOLATED

NODE ANA441

BANA441

A) Spring Index (calculated) = 3.00

Coil diameter (given) = 18.00 mm

Wire diameter (given) = 6.00 mm

B) Spring Index (given) = 6.00

Coil diameter(calculated) = 36.00 mm

Wire diameter (given) = 6.00 mm

(Other options N,H,X or Z)

Select (A or B)

Fig 6.15c Example 2 (level 1) Rule violation information screen display (Third Scroll)

MESSAGE

NODE ANBB BANBB1

IF The number of active coils are calculated from Rule B1
Then Rule B2 will be violated

And

IF The number of active coils are calculated from rule B2
Then rule B1 will be violated

(No priority on rules)

(Other options N,H,X or Z)

Press any key to continue

Fig 6.16a Example 2 (level 2) message screen display on potential rule violation (First scroll)

MESSAGE

NODE ANBB BANBB1

Rule B1: N = n + EH

Rule B2: Lc = (n + EH1)d

Where N = Total number of coils

n = Active number of coils

Lc = Solid length

EH = A constant whose value depends on the spring end condition.

EH1 = A constant whose value depends on whether the spring end is open or closed

(Other options N,H,X or Z)

Press any key to continue

Fig 6.16b Example 2 (level 2) message screen display on potential rule violation (Second scroll).

MESSAGE

NODE ANBB BANBB1

A) Total number of coils (Calculated) = 24.00

Number of active coils (calculated) = 24.00

Solid length (given) = 150.00 mm

B) Total number of coils (Calculated) = 23.00

Number of active coils (calculated) = 23.00

Solid length (given) = 144.00 mm

(Other options N,H,X or Z)

Select (A or B)

Fig 6.16c Example 2 (level 2) message screen display on potential rule violation (Third Scroll)

The third and final level of the process is executed without any rule violation. The same screen displays as in example 1 are displayed, that is, Figs 6.13d to 6.13g. However, the maximum physical deflection is 13.6 mm which was then used to calculate the maximum recommended load which is 164.1 N. The test module is then executed for advice on buckling (test for transverse stability), surge and fatigue. The buckling and the fatigue tests were passed hence, the process is passed on to the result module. The result module is the final part of the consultation. The results for this example are presented in the frame shown on Figs 6.17(a),(b),(c). All details are given and an option for further explanation of the analysis is presented to the user. This option is called the "history of analysis". It simply gives in sequential order, the consultation process as best as it can in order of execution. A final frame thanking the user for the consultation is shown in Fig 6.18.

Calculations will now be given for this example. The calculated Wahl factor (rule A2) using the given spring index will be; K = [(4c-1)/(4c-4)+0.615/c] = 1.25.
The given Wahl factor is 1.3, hence the rule was violated. Also, as in the case of example 1, the calculated spring index (D = 18 and d = 6) is 3. The calculated coil diameter (d = 6 and c = 6) is 36 mm. Option B was selected, therefore D = 36 mm, d = 6 mm and c = 6. In level 2, the calculated solid length (rule B1) is (24 + 1)*6 = 150 mm. This is not the same as the given solid length (144 mm)

hence there is a rule violation and options were given. The calculated active number of coils is 23. This option was chosen. Therefore, Lc = 144 mm, N = 23 and n = 23. Only two parameters are known in level three. The force at maximum stress is 2.73 kN

RESULT

NODE RESULT1 BRES1

Recommended freelength	= 160.00 mm
Solid length	= 144.00 mm
Spring Rate	= 12.10 N/mm
Spring wire diameter	= 6.00 mm
Mean Coil diameter	= 36.00 mm
Spring index	= 6.00
Wahl factor	= 1.25
Total number of coils	= 23
Active number of coils	= 23
Maximum load	= 164.30 N

Press a key to continue

Fig 6.17a Example 2 (result module) Information screen display (first Scroll)

RESULT

NODE RESULT1 BRES1

Maximum deflection	= 13.6 mm
Deflection at maximum stress	= 235.77 mm
Maximum physical deflection	= 13.60 mm
Maximum shear stress	= 1.45 kN/ mm^2
Load at maximum stress	= 2.73 kN
Shear modulus	= 80.00 kN/mm^2

Press a key to continue

Fig 6.17b Example 2 (result module) Information screen display (second Scroll)

RESULT

NODE RESULT1 BRES1

Spring end	not ground
Spring end finish condition	open (plain)
Spring has	one end fixed (Spring guided)

Do you wish to store result?

Press Y for yes

Press R for History of analysis

Fig 6.17c Example 2 (result module) Information screen display (third Scroll)

HELICAL COMPRESSION SPRING DESIGN

END OF CONSULTATION

Thank you for the consultation

(please wait)

Fig 6.18 Last screen display of consultation

(using rule C2) as in example 1 and the deflection at maximum stress is 235.68 mm (rule C1). The maximum physical deflection is 0.85*(160-144) = 13.6 mm. The maximum recommended load is (13.6*6*6*6*80000)/(8*36*36*36*23) = 164.25 N. The spring rate calculated is 164.2512/13.6 = 12.08 N/mm.

The buckling test is passed because the critical relative deflection is $0.881[1\pm(1-6.89(CC/(Fl/D))^2)]^{0.5}$ = 1.23 or 0.26. (CC = 0.5, for one end pivoted and the other fixed.). The actual relative deflection is 0.085.

The frequency of loading must be higher or lower than 4.33 which is the natural frequency calculated.

The fatigue test using equation 6.11 is passed. That is, $[\sigma y/(\sigma m-\sigma v+(2*\sigma v\sigma y/\sigma rel))]\geq 1$. Now, $\sigma m = Fm*8*c*K/(\pi*d^2)$ where Fm = (164.2512+100)/2 = 132.13 N.

Therefore, σm = 70.24 N/mm^2.

$\sigma v = Fv*8*c*K/(\pi*d^2)$ where Fv = (164.2512-100)/2 = 32.13 N.

Therefore, σv = 17.08 N/mm^2. $\sigma rel = 530/d^{0.2}$. that is,

σrel = 370.38 N/mm^2. σy = 1.45 kN/mm^2.

$[\sigma y/(\sigma m-\sigma v+(2*\sigma v\sigma y/\sigma rel))]$ = 7.76.(test passed).

6.9.3 Spring design example problem 3 and 4.

The results of two other examples are tabulated in table 6.6, The figures in brackets (that is, ()) are hand calculated values to compare. The figure with "*" are given values. The rest are calculated by Helcom. For example 3, the spring is pivoted at one end and fixed on the other. It is laterally guided and the ends are closed and ground. The given parameter values produced actual deflection greater

than the maximum physical deflection or the deflection at maximum stress. To rectify this situation, if the number of active coils is increased from 11 to 21, the deflection at maximum stress is then 41 mm. Also, the maximum recommended free length (from rule C4) was 109.32 mm. This in turn produces a physical deflection of 59.8 mm. Hence, the increase in the active coil number allows the design to satisfy the deflection requirements. Rule C4 is used in this case because the spring ends are closed and ground. The spring can fail if compressed beyond 41 mm. The buckling test was passed.

In the fourth case, the load is dynamic. Both ends of the spring are open. The spring is also guided. The deflection at maximum stress is taken as the maximum physical deflection since the free length was not given and rule C4 cannot be used because the spring ends are open. The load at maximum stress is used to calculate the actual deflection from rule C5. This deflection is less than the deflection at maximum stress, hence the deflection criterion is satisfied. The buckling test is passed but the fatigue test is failed. The minimum stress will have to be increased in order to increase the mean stress. Increasing the mean stress gives the design better fatigue properties.

It can be observed that Helcom values are the same as hand calculated values.

Parameters	Example 3	Example 4
Spring wire diameter (mm)	3.00*	4.00*
Mean Coil diameter (mm)	12.00*	28(28)
Spring index	4(4)	7*
Wahl factor	1.4(1.4)	1.21(1.21)
Total number of coils	13(13)	18(18)
Active number of coils	11*	18*
Solid length (mm)	39(39)	76(76)
Spring Rate (N/mm)	5.00*	15.00*
Load (N)	200*	1313
Load at maximum stress (N)	915.1(915.1)	1313(1313)
Minimum load	-	100*
Deflection at maximum stress	21.51(21.47)	202.66(202.66)
Maximum physical deflection	17.92	202.66
Actual deflection (mm)	40(40)	87.53
Maximum shear stress (N/mm^2)	1450	1770
Recommended freelength (mm)	60.12(60.12)	178
End fixing condition	One pivoted(G)	Ends fixed(G)
End finished condition	square/ground	open
Duty	Static	Dynamic
Buckling	Passed	Passed
Surging	-	6
Fatigue	-	Failed

Table 6.6 Results for Examples 3 and 4.

CHAPTER SEVEN

CASE STUDY (2) : GEAR-PAIR DESIGN

CHAPTER SEVEN

CASE STUDY (2) : GEAR-PAIR DESIGN

7.1 Introduction.

Pensulo [75] developed an integrated design package for a simple gearbox adopting relational database methodologies as shown schematically in Fig 7.1. Generally, it designs the gear pair and shaft, draws the shaft and selects the bearings. It was not developed using Expert Systems methodologies.

The work described in this chapter was intended to complement the work carried out on spring design and to be a generalisation of that work, it was directed towards using Expert Systems concept to the design of gear pairs. It was realised that the application of the concept to the gearbox design was not practical within the limits of this project. However, the work carried out by the author can later be extended to a more general system such as a gearbox. To this end, the work carried out by the author would be valuable.

The work carried out on gearpairs will now be described. Two types of gears where addressed, they are single reduction spur and helical gears. As a background to the work described here, the theory of gear design will first be described in the next section.

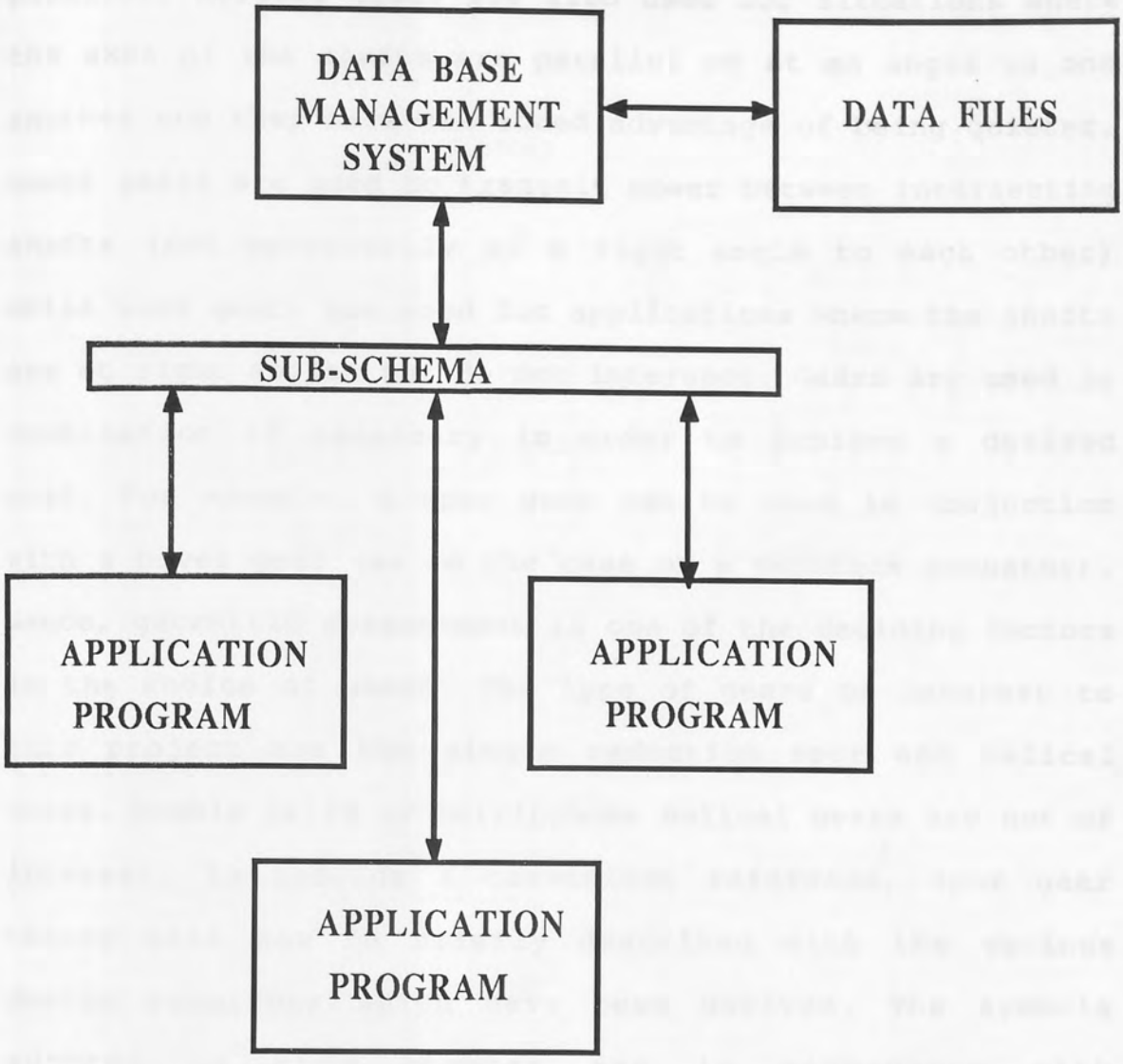


Fig 7.1 Block diagram for integrated gearbox design software

7.2 The introduction to gear design theory.

Gears are designed to transmit constant power without slip. There are five basic types of gears [76]. They are the spur, Helical, Spiral, Bevel and Worm gears. Each of these gear types has its own usefulness, for example, spur gears are used for applications where the axes of the shafts are

parallel. Helical gears are also used for situations where the axes of the shafts are parallel or at an angle to one another and they have the added advantage of being quieter. Bevel gears are used to transmit power between intersecting shafts (not necessarily at a right angle to each other) while worm gears are used for applications where the shafts are at right angles but do not intersect. Gears are used in combination if necessary in order to achieve a desired goal. For example, a spur gear can be used in conjuction with a bevel gear (as in the case of a penstock actuator). Hence, geometric arrangement is one of the deciding factors in the choice of gears. The type of gears of interest to this project are the single reduction spur and helical gears. Double helix or herringbone helical gears are not of interest. To provide a convenient reference, spur gear theory will now be briefly described with the various design equations which have been derived. The symbols adopted in this chapter are in accordance with BS2519:Part1:1976 [77] and BS2519:Part2:1976 [78].

7.2.1 Spur gears

These are gears with teeth parallel to the axis of the shaft (Fig 7.2). The tooth cross-section is uniform with constant shape and size. They are used in items such as watches, vehicle gearboxes, rolling mills, actuators etc. Houghton [76] likened a spur gear to a cylinder blank which has a series of equally spaced grooves around its perimeter so that the projections on one blank may mesh in the grooves of the second.

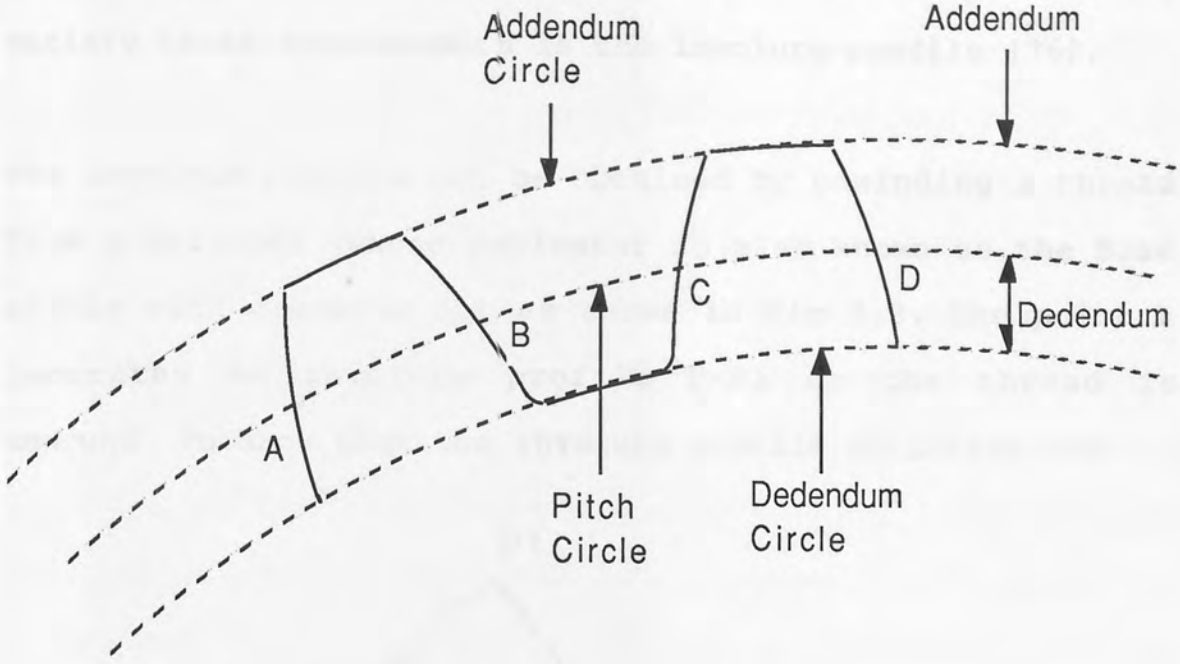


Fig 7.2 Spur gear teeth and nomenclature

These projections or teeth are designed such that the gears are always in mesh, the revolution made by each is regular and in inverse ratio to the number of teeth. The shape of each tooth is extremely important because it should have a profile that allows the gears to run with uniform motion. The tooth thickness must be large enough to withstand the induced stresses and the tooth profile should be such that it allows contact only at the designed gear faces and not in areas that will allow a tooth to "dig" into (or undercut) its mating tooth. The gear teeth must also mesh

at a correct centre distance. Over the years, experience has shown that the best type of gear tooth profile to satisfy these requirements is the involute profile [76].

The involute profile can be obtained by unwinding a thread from a cylinder (whose perimeter is also known as the Base circle with diameter db) as shown in Fig 7.3. The point P generates an involute profile P-P1 as the thread is unwound. To show that the involute profile satisfies the

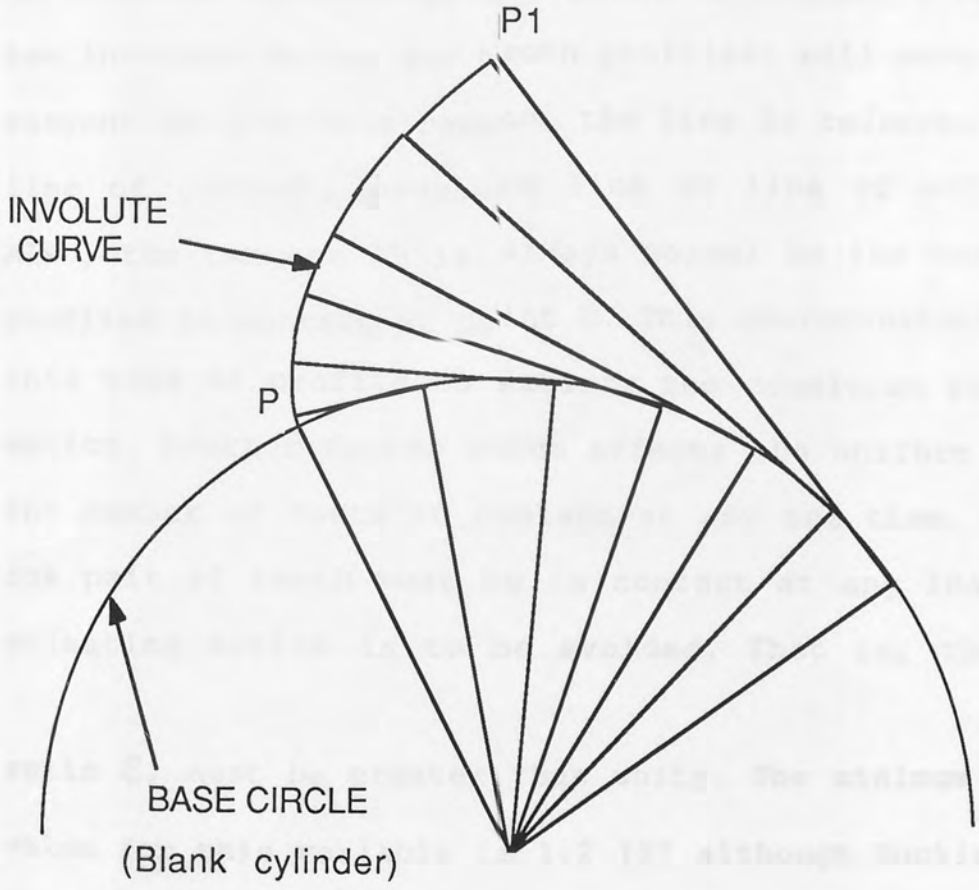


Fig 7.3. Generating an involute curve

condition for uniform motion, consider Fig 7.4. Two circles BC1 and BC2 have centres at C1 and C2 respectively. The radii of the circles are C1A and C2B respectively. The line AB is tangent to both circles. Consider that the point C on the line AB is chosen and the line AC is part of an unwinding thread from base circle BC1 generating an involute curve T1-T2 and the line BC corresponds to an identical thread unwinding from the base circle BC2 thus generating the involute curve C-T3. As the circles rotate in opposite directions, the point of contact C between the two involute curves (or tooth profiles) will move along the tangent AB. For this reason, the line is referred to as the line of contact, pressure line or line of action [79]. Also, the tangent AB is always normal to the two involute profiles in contact at point C. This characteristic enables this type of profile to satisfy the condition for uniform motion. Another factor which affects the uniform motion is the number of teeth in contact at any one time. More than one pair of teeth must be in contact at any instant if a pulsating motion is to be avoided. That is, the contact

ratio ε, must be greater than unity. The minimum advisable value for this variable is 1.2 [2] although Buckingham [79] considered that this value to be 1.4. Contact ratios will be discussed further later in the chapter.

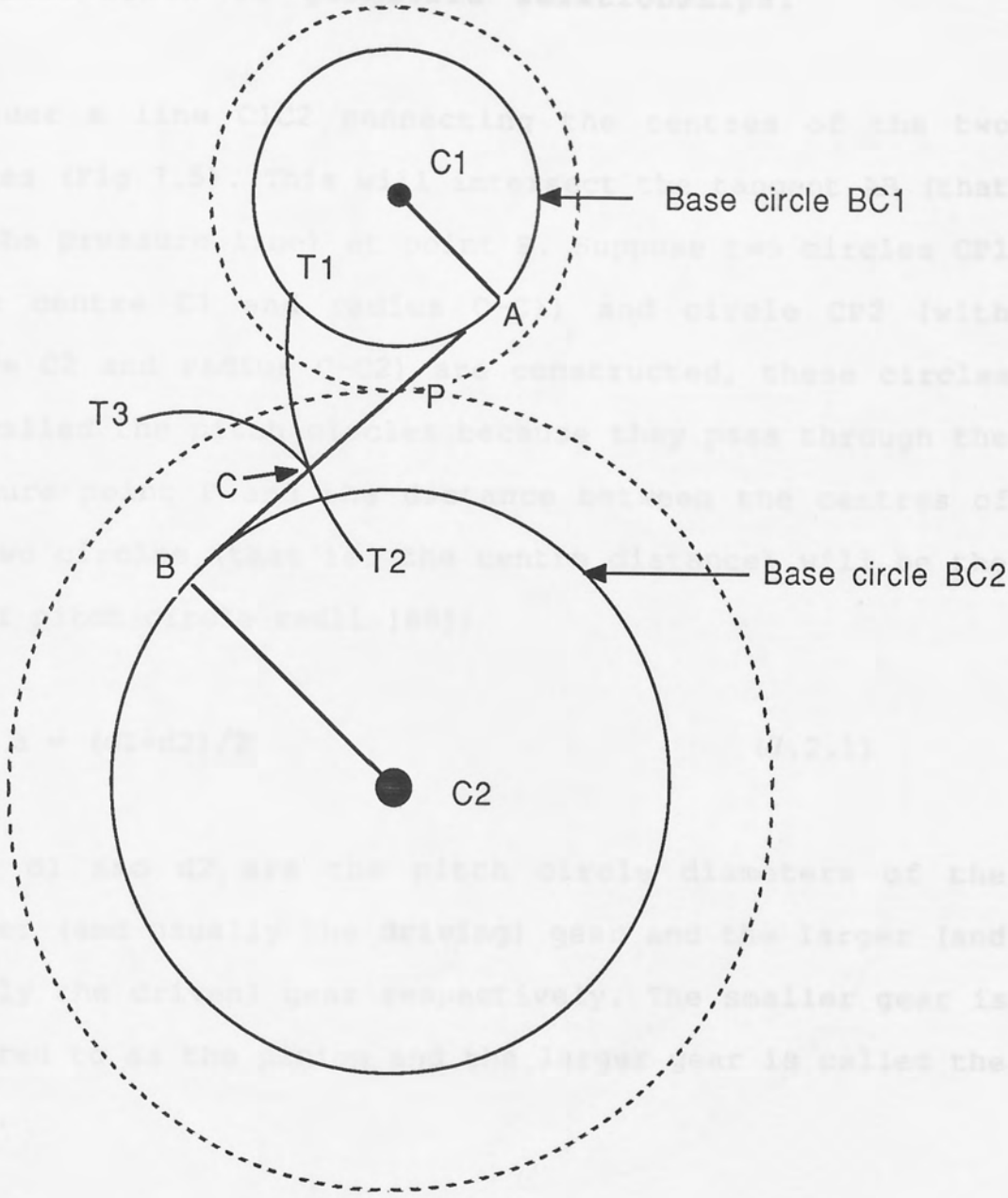


Fig 7.4 Action of involute profiles.

The derivation of geometric relationships:

Consider a line C1C2 connecting the centres of the two circles (Fig 7.5). This will intersect the tangent AB (that is, the pressure line) at point P. Suppose two circles CP1 (with centre C1 and radius C-C1) and circle CP2 (with centre C2 and radius C-C2) are constructed, these circles are called the pitch circles because they pass through the pressure point P and the distance between the centres of the two circles (that is, the centre distance) will be the sum of pitch circle radii [80];

$$a = (d1+d2)/2 \qquad (7.2.1)$$

where d1 and d2 are the pitch circle diameters of the smaller (and usually the driving) gear and the larger (and usually the driven) gear respectively. The smaller gear is referred to as the pinion and the larger gear is called the wheel.

Considering Fig 7.5, the angle that the line EF makes with the pressure line AB (that is, angles EPB or APF) is called the pressure angle. In the past, gears with a pressure angle of 14.5 degrees were commonly used, however, at present the most commonly used pressure angle is 20 degrees and in rare cases a 25 degree pressure angle is adopted when the number of teeth on the pinion must be a minimum. Again considering Fig 7.5, suppose the pinion rotates in a clockwise manner with a rotation speed of n1 revolutions

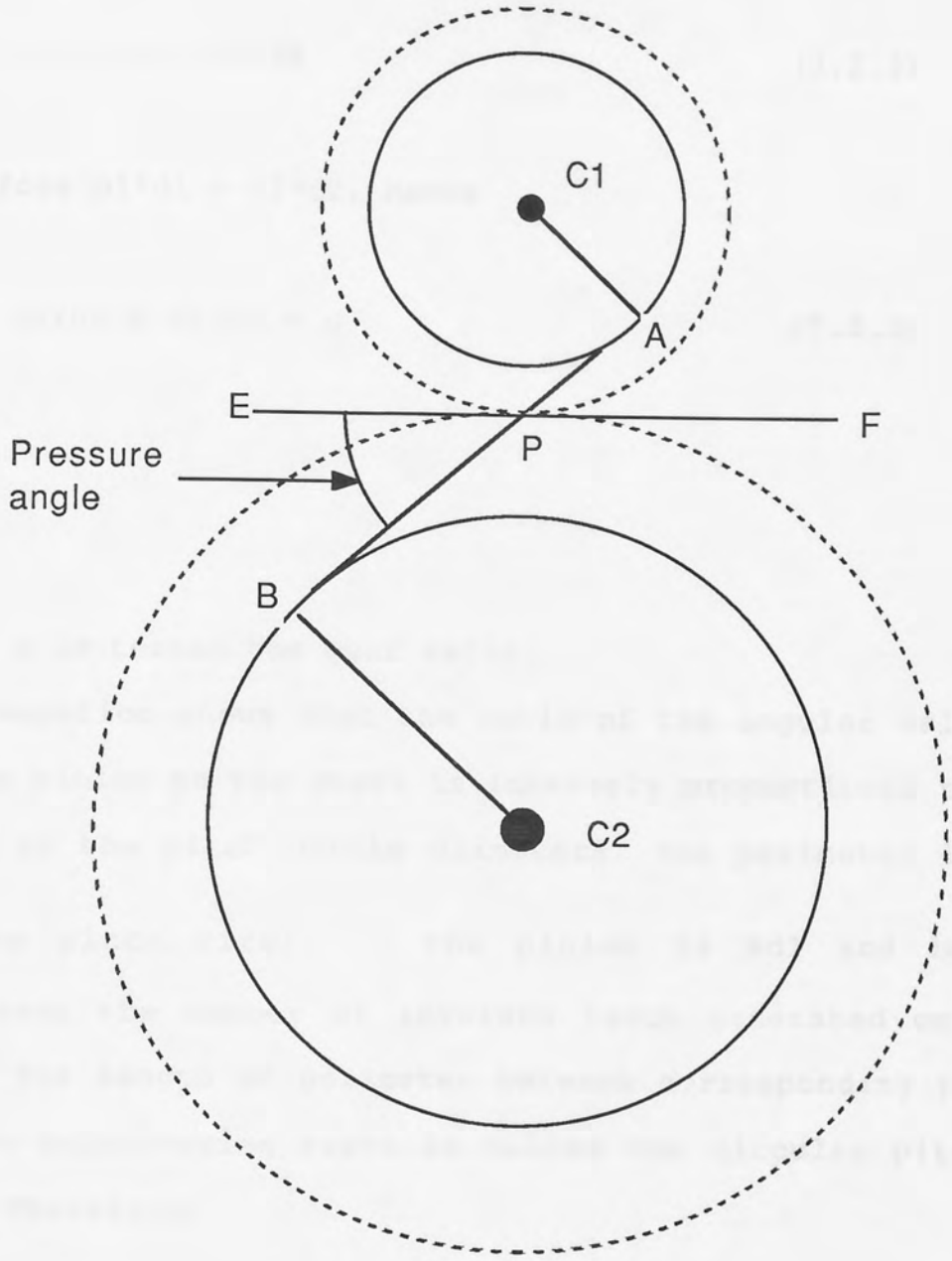


Fig 7.5. Pressure angle

per minute, the wheel will rotate in an anticlockwise manner with a rotational speed of n2 revolutions per minute. Now, since the tangent AB is perpendicular to the gear teeth faces, the instantaneous velocity (in m/s) at point P along the pressure line AB or the pitch line velocity is given by the equations [20];

$$v = n1*d1/19098 \qquad (7.2.2)$$

Therefore $n1*d1 = n2*d2$, hence

$$n1/n2 = d2/d1 = u \qquad (7.2.3)$$

Where u is termed the gear ratio.

This equation shows that the ratio of the angular velocity of the pinion to the wheel is inversely proportional to the ratio of the pitch circle diameters. The perimeter length of the pitch circle of the pinion is $\pi d1$ and let z1 represent the number of involute teeth generated on this gear. The length of perimeter between corresponding points on two neighbouring teeth is called the circular pitch, p [80]. Therefore;

$$p = \pi d1/z1 \qquad (7.2.4)$$

and the circular pitch is related to the base circular pitch pb by the equation

$$pb = p*\cos\alpha \qquad (7.2.5)$$

where α is the pressure angle (Fig 7.5). The pitch circle diameter is related to the base circle diameter by the equation

$$db = d*\cos\alpha \qquad (7.2.6)$$

The ratio d1/z1 (from equation 7.2.4) is called the module, m [2]. That is;

$$m = d1/z1 \qquad (7.2.7)$$

Hence;

$$p = \pi m \qquad (7.2.8)$$

Meshing gears must have the same module and this variable is used as a method of standardisation partly to reduce tooling requirements. The standard module values are tabulated in Table 7.1. The first choice modules are commonly used with readily available gear cutting tools while non-standard modules will require the production of special tooling.

Module values can be used to determine tooth geometric parameters such as the addendum which is the value of the distance between the addendum circle and the pitch circle (Fig 7.2). The British Standard BS436:part2:1970 [81] gave the value of the addendum as;

Choice	Module
First	1, 1.25, 1.5, 2, 2.5, 3, 4, 5, 6, 8, 10, 12, 16, 20, 25, 32, 40, 50.
Second	1.125, 1.375, 1.75, 2.25, 2.75, 3.5, 4.5, 5.5, 7, 9, 11, 14, 18, 22, 28, 36, 45.

Table 7.1 Lists of preferred module values. [81]

$$ha = m \tag{7.2.9}$$

Where ha represent the addendum. The dedendum, hf which is the distance between the pitch circle and the dedendum circle, should be greater than or equal to 1.25 of the module but less than 1.40 of the module. That is;

$$1.40m > hf \geq 1.25m \tag{7.2.10}$$

The root clearance, c between the addendum circle of a gear and the dedendum circle of the mating gear should be between 0.25 and 0.40 of the module. That is,

$$0.40m > c \geq 0.25m \tag{7.2.11}$$

The value of the sum of the addendum and the dedendum of a tooth is called the tooth (whole) depth. Therefore,

$$h = ha + hf \tag{7.2.12}$$

where h represents the tooth depth. The minimum working depth is given as the tooth depth minus the root clearance. The root radius should be between 0.25 and 0.39 of the module.

$$0.39m > rr \geq 0.25m \tag{7.2.13}$$

The tooth thickness at the pitch circle is 1.5708 of the module. That is,

$$t = 1.5708m \tag{7.2.14}$$

Also the outside diameter, do is given by the equation

$$do = 2*ha + d \tag{7.2.15}$$

Quayle [72], Avollone and Baumeister III [80] and Oberg et al [82] recommended that the addendum be modified in order to obtain full involute action, avoid undercutting and to obtain a better zone factor for contact stress (which shall be described later in the section). The stated conditions for a gear pair to be modified are: for $z1 \geq 10$, $z2 > 10$, if $z1 + z2 \geq 60$ then $k = 0.4(1 - z1/z2)$.
Provided that $k > kmin$ where: k is the addendum correction

factor for both pinion and wheel, and kmin = 0.02(30 - z1). If z1 + z2 < 60 then, k for pinion = kmin and k = 0.02(30 - z2) for wheel. Therefore:

ha = m(1 + k) for pinion

ha = m(1 - k) for wheel

hf = m(1.25 - k) for pinion, and

hf = m(1.25 + k) for wheel.

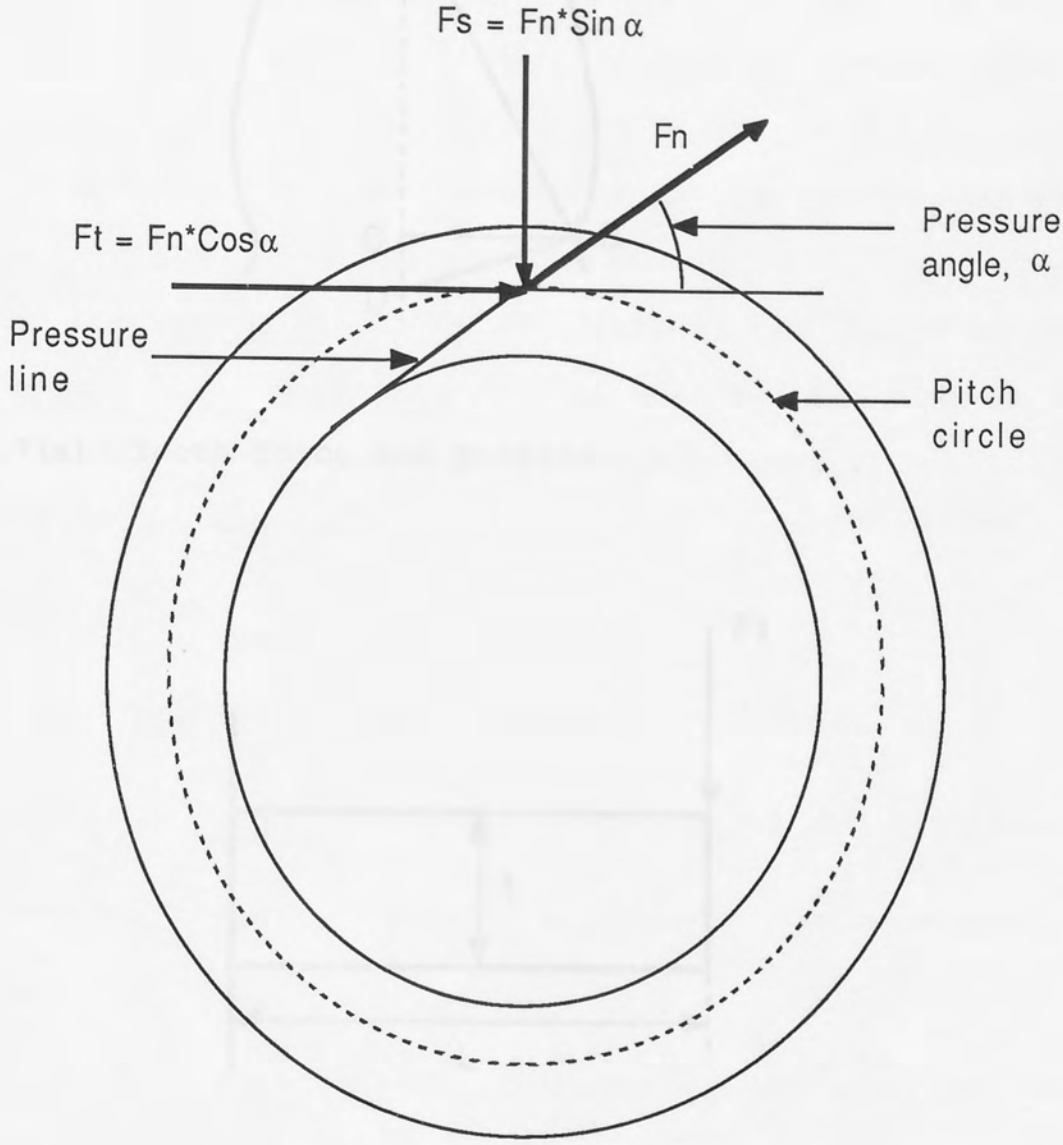
Tooth forces and stresses

Having described the geometry of the spur gear tooth, Fig 7.6 shows the forces acting. Fn represents the force acting along the direction of the pressure line. This force can be resolved into two components, the perpendicular component Fs, which acts to separate the two mating gears. The other component is the tangential (transmitting) force, Ft. The power transmitted is based on this force. That is,

$$P = Ft*v \qquad (7.2.16)$$

where P is the power transmitted.

A formula for stresses induced in the teeth due to gear action was derived by Lewis in 1892, [2][79][83]. He compared a gear tooth of face width, b, to a cantilever as shown in Fig 7.7. Considering Fig 7.7(a), the point B represents the root of the tooth and A represents the point



Fn = Force normal to tooth face
and along pressure line.

Ft = Tangential force

Fs = Separating force

α = Pressure angle

Fig 7.6 Gear tooth forces

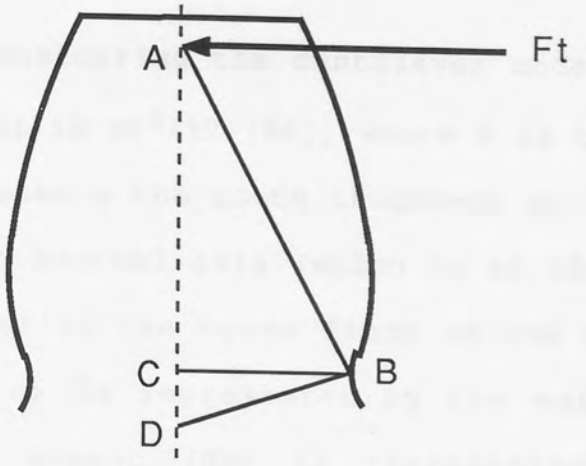


Fig 7.7(a) Tooth force and profile

Fig 7.7(b) Cantilever model for tooth stress action [2]

through which the maximum tangential force acts. The triangles ABD and ABC are right angled triangles with the right angles at B and C respectively. Therefore, BC/CD = AC/BC. Rearranging, CD = BC*BC/AC. But BC represents half the tooth thickness at the root. So, CD = $t^2/(4*AC)$. The distance AC will be equivalent to the length L, in the cantilever model as shown in Fig 7.7(b). Therefore, CD =

$t^2/(4*L)$. Now, considering the cantilever model, the second moment of area (I) is $bt^3/12$ [84], where b is the tooth face width and t represents the tooth thickness at its root. The distance from the neutral axis (which is at the centre axis of the cantilever) to the outer fibre of the cantilever is t/2. This distance is represented by the variable y. The maximum bending moment (BM) is represented by Ft*L (a cantilever with concentrated load at the free end), where L is the length of the beam. The maximum bending stress will be described by the equation

$\sigma f = BM*y/I$

Where σf = induced bending stress.

Therefore $\sigma f = (6*Ft*L)/(b*t^2)$. Rearranging the equation,

$\sigma f = 6*Ft/(4*b*[t^2/4L])$ and $CD = t^2/(4*L)$ (as previously derived).

Therefore, $\sigma f = 6*Ft/(4*b*CD)$. Rearranging the equation and multiplying the numerator and the denominator by the circular pitch p;

$\sigma f = Ft*p/(p*b*2*CD/3) = Ft/(p*b*[(2/3)*CD/p])$.

Suppose (2/3)*CD/p is represented by the variable Yf, then the stress equation will become,

$$\sigma f = Ft/(p*b*Yf) \qquad (7.2.17)$$

This equation is called the Lewis stress equation

[2][83][87] and the variable Yf is the Lewis form or strength factor. This factor is usually tabulated for various numbers of teeth [2][79][88]. Buckingham [88] tabulated the values of the form factor against number of teeth for the full depth, 20^{o} pressure angle on the gear as shown in Table 7.2.

Table 7.2 Lewis form factor by Buckingham [88]

Baath [4] derived an equation for the calculation of the allowable tooth stress corrected for pitch line velocity effects and taking into account the gear material. The

endurance stress is multiplied by the velocity factor, vf.

$$\sigma_{FP} = \sigma_{FO} * vf \qquad (7.2.18)$$

where

$$vf = (x/(x+(v^r))) \qquad (7.2.19)$$

and

σ_{FP} = Allowable or permissible stress (N/mm^2)

σ_{FO} = Endurance stress in bending of the material corrected for stress concentration (N/mm^2). It is estimated to be approximately one third of the ultimate tensile stress [4].

x = 3 for pitch line velocities less than 10 m/s

= 6 for pitch line velocity of between 10 m/s and 20 m/s

= 5.6 for pitch line velocities greater than 20 m/s

v = Pitch line velocity

r = 0.5 for pitch line velocity greater than 20 m/s

r = 1 for pitch line velocity less than 20 m/s.

The developed program (Gearex) calculates the allowable stress and uses this value as the allowable induced stress on the gear teeth, that is, $\sigma_F = \sigma_{FP}$. Using equation 7.2.17, the face width is calculated. This value is the minimum face width allowable. A wider face width may be used as this produces a lower induced bending stress on the gear teeth. However, it is advisable that the calculated face

width should be greater than the circular pitch of the gears, hence a value lower than the circular pitch is not acceptable by Gearex. The upper limit for the face width value is four times the value of the circular pitch [2].

Buckingham examined induced loads on gear teeth due to dynamic effects like vibrations etc. (as opposed to static/bending load researched by Lewis). This additional load on a gear tooth is due to factors such as gear misalignment,tooth deflection in operation resulting in velocity changes giving rise to local accelerations, variable forces and hence stress fluctuations. He gave the dynamic load (Fd) as [83]:

$$Fd = [(21*v*(b*CC*+Ft))/(21*v+(b*CC+Ft)^{0.5})]+Ft \quad (7.2.20)$$

Where CC = a constant which depends on the tooth form, material and the accuracy with which the tooth was cut. Table 7.3 gives the permitted tooth error for precision gears (class 1), carefully cut commercial gears (class 2) and commercial gears (class 3).

Table 7.4 gives the maximum permissible tooth error for various values of pitch line velocity.

Buckingham also gave an equation for the allowable load for wear. The wear load Fw, was given as [2][4][83]:

$$Fw = d1*b*KK*QQ. \quad (7.2.21)$$

Circular pitch	Class 1	Class 2	Class 3
80	0.122	0.061	0.031
40	0.102	0.051	0.026
27	0.081	0.041	0.021
20	0.066	0.033	0.018
16	0.056	0.028	0.014
13	0.051	0.025	0.013

Table 7.3. Maximum tooth error in millimetre [88]

Pitch line velocity (mm/s)	Error (mm)	Pitch line velocity (mm/s)	Error (mm)
1270	0.094	12700	0.030
2540	0.081	13970	0.025
3810	0.071	15240	0.023
5080	0.061	16510	0.020
6350	0.053	17780	0.018
7620	0.048	20320	0.015
8890	0.043	22860	0.015
10160	0.038	25400	0.013
11430	0.033		

Table 7.4 Maximum tooth error versus pitch line velocity [88]

Where KK = load stress factor.

$$KK = \sigma e^2 (\sin\alpha)(1/E1+1/E2)/1.4 \qquad (7.2.22)$$

where σe = Surface endurance limit for the material.

E1 = Youngs modulus for pinion material.

E2 = Youngs modulus for wheel material.

α = Pressure angle

$$QQ = 2*z2/(z1+z2) \qquad (7.2.23)$$

The wear load should be less than the dynamic load which should be in turn less than the allowable endurance load, Fo, where

$$Fo = \sigma FO*b*Yf*p \qquad (7.2.24)$$

These equations in their various forms are tabulated in Table 7.5 for geometric relationships and Table 7.6 for force and stress relationships.

The British Standard BS436:Part3:1986 section 1 [20] gives the design procedure for spur gears using; a) stress or b) power to determine the load carrying capacity. In case (a), the permissible (or allowable) stress is calculated from the material endurance limits corrected by the stress modifying factors. These modifying factors will be discussed in detailed at a later stage. The allowable stress must be greater than the induced (or actual) stress which is calculated from the tangential force (corrected by the load modifying factors) and gear geometry.

Equation number	Equation
7.2.1.	$a = (d1+d2)/2$
7.2.2.	$v = n1*d1/19098$
7.2.3.	$n1/n2 = d2/d1 = u$
7.2.4.	$p = \pi d1/z1$
7.2.5.	$pb = p*\cos\alpha$
7.2.6.	$db = d*\cos\alpha$
7.2.7.	$m = d1/z1$
7.2.8.	$p = \pi*m$
7.2.9.	$ha = m$
7.2.10.	$1.4m > hf \geq 1.25m$
7.2.11.	$0.4m > c \geq 0.25m$
7.2.12.	$h = ha + hf$
7.2.13.	$0.39m > rr \geq 0.25m$
7.2.14.	$t = 1.5708m$
7.2.15.	$do = 2*ha + d$

Table 7.5 Geometric relationships

Equation Number	Equation
7.2.16.	$P = Ft*v$
7.2.17.	$\sigma f = Ft/(p*b*Yf)$
7.2.18.	$\sigma FP = \sigma FO*vf$
7.2.19.	$vf = x/(x+v^r)$
7.2.20.	$Fd = [(21*v*(b*CC+Ft))/(21*v+(b*CC+Ft)^{0.5})]+Ft$
7.2.21.	$Fw = d1*b*KK*QQ$
7.2.22.	$KK = \sigma e^2(Sin\alpha)(1/E1+1/E2)/1.4$
7.2.23.	$QQ = 2*z1/(z1+z2)$
7.2.24.	$Fo = \sigma FO*b*Yf*p$

Table 7.6 Stresses and forces relationships

In the second case, the power capacity of the gear is calculated from the allowable stress which should be greater than the power to be transmitted. BS436 gave the induced stress as

$$\sigma F = (Ft/(b*m))*Yf*Ys*Ka*Kv*K F\alpha*K F\beta. \qquad (7.2.25)$$

The factors $Y_f, Y_s, K_A, K_v, K_{f\alpha}, K_{f\beta}$ are called the load modifying factors.

The Form factor, Yf takes into account the effects of the tooth geometry. The calculation of this factor as given by BS436 [20] is complex and requires a knowledge of the hob geometry.

The stress correction factor, Ys takes into account the effects of increased stress in the locality of a fillet and the bending moment arm on the bending stress.

The application factor Ka accounts for the load fluctuations from the mean loads caused by external sources. This fluctuation depends on the nature of the power source, the driven load and system characteristics. It can be determined from Table 7.7.

Load characteristic of prime mover	Load of driven machine			
	Uniform	Moderate shock	Medium shock	Heavy shock
Uniform	1.00	1.25	1.50	1.75
Light shock	1.10	1.35	1.60	1.85
Moderate shock	1.25	1.50	1.75	2.00
Heavy shock	1.50	1.75	2.00	2.25

Table 7.7 Values of application factor [20]

Examples of load characteristics of the prime mover is given in Table 7.8.

Load characteristics of prime mover	Prime mover
Uniform	Electric motor
Light shock	Steam turbine, Gas turbine
Moderate shock	Multi-cylinder combustion engine
Heavy shock	Single cylinder combustion engine

Table 7.8 Examples of prime mover with different working characteristics [20]

Examples of driven machines with uniform load characteristics are: generators, uniformly loaded belt or platform conveyors, worm conveyors, light elevators, packaging machines, feed gears,centrifugal pumps etc. Examples of moderate shock load are industrial and mine ventilators, rolling mills, multi-cylinder piston pumps, heavy elevators, main drives of machine tools etc. Wood working mills, single cylinder piston pumps, lifting gears etc. can be placed under equipments that induce medium shock. Heavy shocks are induced by loads such as rotary drilling apparatus, braking drums, cold strip rolling mills, brick moulding press etc.

The dynamic factor Kv takes into account the gear tooth accuracy. It is assumed to be unity.

The load factor for bending stress $K_{F\alpha}$ and $K_{F\beta}$ accounts for the uneven distribution of bending moment across the facewidth caused by uneven loading across the facewidth. Theoretically, these factors can be assumed to be unity. That is, the load is assumed to be evenly distributed across the facewidth.

BS436 [20] gave the permissible stress σ_{FP} as:

$$\sigma_{FP}=[2*\sigma_{FO}*(\sigma_B-\sigma_R)*Y_N*Y_R*Y_X*Y_M*Y\delta]/[(\sigma_B+\sigma_{FO}*Y_N*Y_R*Y_x)] \quad (7.2.26)$$

where;

σ_B is the ultimate tensile stress for the material

σ_R is the residual stress which depends on the grinding operation. It can be determined from Table 7.9.

Operation	Algebraic change to σ_R (MN/m^2)
Very light grinding carefully controlled	0
Light full-form grinding	+300(tensile)
Heavy abrasive grinding	+600 (tensile)

Table 7.9. Residual stresses due to grinding [20].

The material quality factor for bending stress, Y_M takes into account the effects of better quality material. For surface hardened steel, $Y_M = 1$; $Y_M = 0.3$ for cast iron.

The size factor, Y_X accounts for the effects of size on the allowable bending stress. If $m \leq 5$ then $Y_X = 1$. Generally, if $5 < m < 30$, then $Y_X = 1.05-0.01m$. If $m \geq 30$ then $Y_X = 0.75$.

The surface condition factor for bending, Y_R accounts for the effects of surface flaws and tooth root fillet on the endurance limit and hence the bending stress. For commonly used industrial materials such as cast iron,
$Y_R = 4.924-3.9(6*Ra+1)^{0.01}$,where Ra is the roughness average.

The life factor for bending stress Y_N takes into account the increase in allowable stress if the number of stress cycles is less than the endurance life. This value can theoretically be assumed as unity. In effect, the gear designs are for an infinite number of stress cycles.

The sensitivity factor for bending, $Y\delta$ accounts for the effects and sensitivity of the material to notches such as root fillet.

British Standard accounted for the wear effects on gears by using the following equation;

Permissible contact stress:

$$\sigma_{HP} = \sigma_{Hlim} * Z_L * Z_V * Z_R * Z_M * Z_W * Z_X * Z_N \qquad (7.2.27)$$

Actual/induced stress:

$$\sigma_H = Z_H * Z_E * Z_\varepsilon (F_t * (u+1) * K_a * K_V * K_{H\alpha} * K_{H\beta} / (u * b * d_1))^{0.5} \qquad (7.2.28)$$

The factors Z_L, Z_V and Z_R are called the lubricant, roughness and speed factors respectively. They take into account the type of lubricant used for gear lubrication and hence the viscosity of the lubricant, the effect of surface roughness of the gear mating faces on lubrication and the effect of the pitch line velocity on the lubricant film thickness.

The material quality factor for contact stress Z_M takes into account the effects of better quality material. For surface hardened steel, $Z_M = 1$; and $Z_M = 0.7$ for cast iron.

The work hardening factor for contact stress Z_W takes into account the effect of work hardening on contact stress. This in turn depends on the Vickers hardness of the material and the roughness average. Unity is recommended by the British Standard for all cases except the meshing of a through hardened steel wheel with a surface hardened pinion.

The size factor Z_x, which takes account of the effect of manufacturing process and material size on the material quality is recommended by BS436 to be taken as unity.

The life factor for contact stress Z_N takes into account the increase in permissible stress if the number of stress cycles is less than the endurance life. This value can theoretically be assumed to be unity. In effect, the gear design is for infinite number of stress cycles.

The zone factor for contact stress Z_H is a geometric factor that accounts for the influence of tooth flank curvature on the contact stress.

The effect of the contact ratio is accounted for by the factor $Z\varepsilon$. This can be calculated from the equation;

$$Z\varepsilon = ((4-\varepsilon\alpha)/3)^{0.5} \qquad (7.2.29)$$

Where $\varepsilon\alpha$ is the transverse contact ratio. The transverse contact ratio can be calculated from the equation:

$$\varepsilon\alpha = ((do1^2-db1^2)^{0.5}+(do2^2-db2^2)^{0.5} -2*a*\sin\alpha)/(2*\pi*m*\cos\alpha) \qquad (7.2.30)$$

This parameter is very important in spur gear design. The higher the value, the smoother the power transmission.

The factor Z_E accounts for specific material properties like Poisson's ratio and Young's modulus. The Table 7.10 gives the value of some material properties.

Gear material	Z_E
Steel/steel	189
Steel/SG cast iron	181
SG cast iron/SG cast iron	174
Grey cast iron/Grey cast iron	146

Table 7.10 Values of Z_E [20].

The factors $K_{H\alpha}$ and $K_{H\beta}$ account for non-uniform distribution of surface load due to various factors such as deflections, tooth modifications etc. Gearex adopted the contact stress calculation methods by BS436[20] to determine wear/contact stress requirements. However, Z_H, $K_{H\alpha}$, $K_{H\beta}$, and K_v are considered to be unity.

Table 7.11 lists all the equations for bending, contact stresses and contact ratio relationships according to BS436[20].

It can be seen from the theory of gear design that each of the factors described can in future be of interest to the knowledge engineer/user. Each factor is dealt with by a frame and each frame can thus be developed into large specialised programs as more accurate ways of determining the factors are found. The next section discusses the additional factors to be considered when designing Helical gears.

Equation Number	Equation
7.2.25.	$\sigma F = (Ft/(b*m))*Yf*Ys*KA*Kv*KF\alpha*KF\beta.$
7.2.26.	$\sigma FP = [2*\sigma FO*(\sigma B-\sigma R)*YN*YR*YX*YM*Y\delta]/[(\sigma B+\sigma FO*YN*YR*Yx)]$
7.2.27.	$\sigma HP = \sigma Hlim*ZL*Zv*ZR*ZM*ZW*ZX*ZN$
7.2.28.	$\sigma H = ZH*ZE*Z\varepsilon(Ft*(u+1)*Ka*Kv*KH\alpha*KH\beta / (u*b*d))^{0.5}$
7.2.29.	$Z\varepsilon = ((4-\varepsilon\alpha)/3)^{0.5}$ (Spur gears only)
7.2.30.	$\varepsilon\alpha=((do1^2-db1^2)^{0.5}+(do2^2-db2^2)^{0.5}-2*a*\sin\alpha)/(2*\pi*m*\cos\alpha)$

Table 7.11 Stress relationships [20]

7.2.2 Helical Gears

Helical gears differ from spur gears in that the teeth are inclined at an angle to the axis of the shaft. This angle is referred to as the helix angle (Fig 7.8). Shigley [2] described the generation of an helicoid as the line generated by a point on the angular edge of an unwinding paper cut in the shape of a parallelogram wrapped round a cylinder. Fig 7.9. Helical gear engagement starts from a point contact and proceeds to become a line contact as the engagement advances. This gradual engagement gives the gear the advantage of being quieter than spur gears. Also because of the nature of the engagement and gear geometry, the contact ratio for helical gears is of less importance. However, the contact ratio for a helical gear is the sum of the transverse contact ratio and the overlap ratio. The latter is given by,

$$\varepsilon\beta = b*\sin\beta/(mn*\pi) \qquad (7.2.30a)$$

to equation 7.2.30. The Total contact ratio is $(\varepsilon\alpha + \varepsilon\beta)$. Considering Fig 7.8, The parameter pn is called the normal circular pitch and is related to the circular pitch p, by the equation

$$pn = p*\cos\beta \qquad (7.2.31)$$

Where β is the helix angle. Because of the teeth inclination at an angle, the pressure angle in the

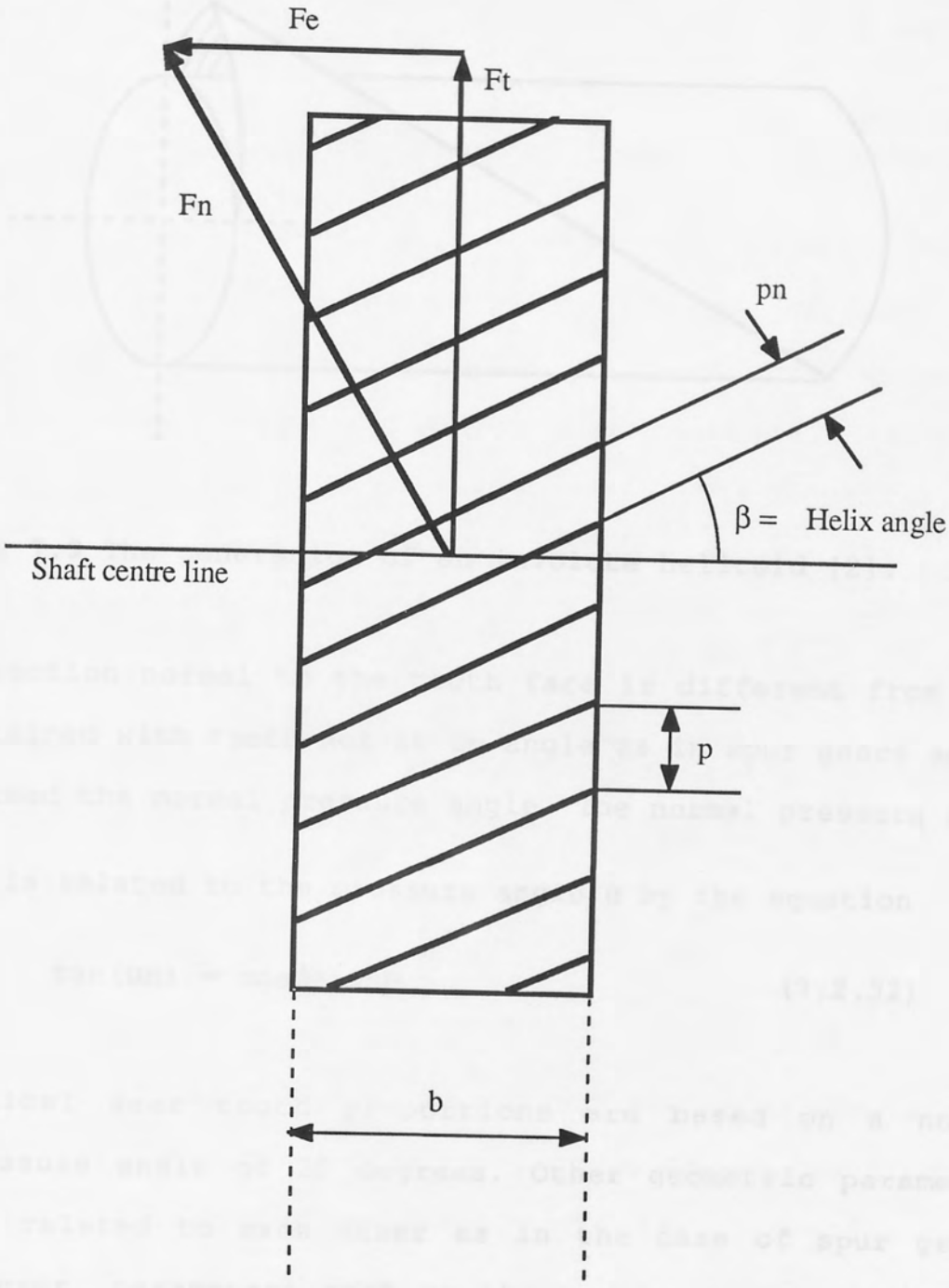


Fig 7.8 Helical gear showing helix angle [4]

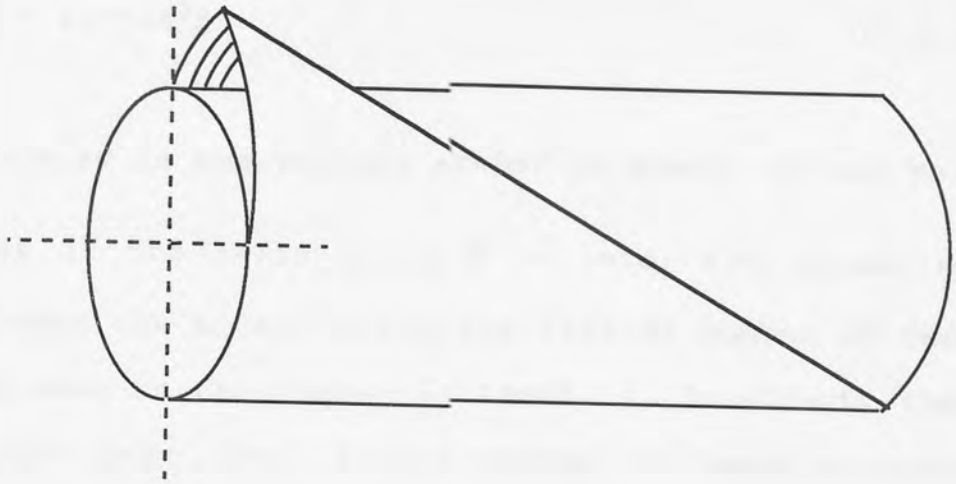


Fig 7.9 The generation of an involute helicoid [2].

direction normal to the tooth face is different from that obtained with teeth not at an angle as in spur gears and is termed the normal pressure angle. The normal pressure angle αn is related to the pressure angle α by the equation

$$\tan(\alpha n) = \cos\beta * \tan\alpha \qquad (7.2.32)$$

Helical gear tooth proportions are based on a normal pressure angle of 20 degrees. Other geometric parameters are related to each other as in the case of spur gears. However, parameters such as the module m, are related to the module measured in the direction normal to the gear tooth by the equation [89],

$$mn = m * \cos\beta \qquad (7.2.33)$$

where mn is termed the normal module. Also the virtual number of teeth, z [89] is given by:

$$z = zv*\cos^3\beta \qquad (7.2.34)$$

Where zv is the virtual number of teeth. It can be observed that if the helix angle β is zero, the normal module mn becomes the module m and the virtual number of teeth zv is the same as the number of teeth, z. In effect, the gear is a spur gear. The virtual number of teeth is used in the calculation for stength using the Lewis equation. The minimum face width is given by

$$b = p/\tan\beta \qquad (7.2.35)$$

and the upper limit for the face width is 6*p. The minimum number of teeth to avoid undercutting is given by the equation

$$zmin = \cos^3\beta(2/\sin^2\alpha) \qquad (7.2.36)$$

As shown in Fig 7.8, the gear forces acting on the teeth of a helical gear includes an axial force Fe, in the direction of the shaft axis. This means that a thrust bearing should be used on shafts carrying a single reduction helical gear.

$Fe = F*\cos\alpha n*\sin\beta$, this can be given in terms of the tangential force as

$$Fe = Ft*\tan\beta \qquad (7.2.37)$$

The helix angle of a gear should be restricted to no more than 45 degrees because of the larger axial force that will

be produced for greater angles. The seperating force is related to the resultant force by the equations Fs = F*Cosαn*Cosβ, this can be given in terms of the tangential force by the equation

$$Fs = Ft*\tan\alpha \qquad (7.2.38)$$

The tangential force is related to the resultant force by the equation

$$Ft = F*\sin\alpha n \qquad (7.2.39)$$

When using the British Standard 436 [20] to calculate the induced bending stress, an additional factor has to be taken into consideration. That is, the helix angle factor, $Y\beta$. Hence; $\sigma_F = (Ft/(b*m))*Yf*Ys*Y\beta*Ka*Kv*Kf\alpha*Kf\beta$

The helix angle factor $Y\beta$ takes into account the effects of the lower stress values due to the inclined line of contact on helical gears at the root of the tooth. The factor is assumed to be unity for spur gears and is calculated from the equation,

$$Y\beta = 1-\varepsilon\beta(\beta/120) \qquad (7.2.40)$$

This equaution applies to helical gears only. For spur gears $Y\beta = 1$.

Another factor which differs in calculation from that of

spur gears is the contact ratio factor $Z\varepsilon$. For a gear with overlap ratio less than unity,

$$Z\varepsilon = [((4-\varepsilon\alpha)*(1-\varepsilon\beta)/3)+\varepsilon\beta/\varepsilon\alpha]^{0.5} \qquad (7.2.41)$$

For gears with an overlap ratio equal to or greater than unity,

$$Z\varepsilon = (1/\varepsilon\alpha)^{0.5} \qquad (7.2.42)$$

Table 7.12 gives addition relationships relating to helical gears.

The next section will now discuss the system developed to cater for the complex nature of this type of design activity. It particularly embodies the characteristics of designs where additional factors may be added as the operating conditions of a component is better understood as in the case of gear design.

Equation Number	Equation
7.2.30a.	$\varepsilon\beta = b*\sin\beta/(mn*\pi)$
7.2.31.	$pn = p*\cos\beta$
7.2.32.	$\tan(\alpha n) = \cos\beta*\tan\alpha$
7.2.33.	$mn = m*\cos\beta$
7.2.34.	$z = zv*\cos^3\beta$
7.2.35.	$b = p/\tan\beta$
7.2.36.	$zmin = \cos^3\beta(2/\sin^2\alpha)$
7.2.37.	$Fe = Ft*\tan\beta$
7.2.38.	$Fs = Ft*\tan\alpha$
7.2.39.	$Ft = F*\sin\alpha n$
7.2.40.	$Y\beta = 1-\varepsilon\beta(\beta/120)$
7.2.41.	$z\varepsilon = [((4-\varepsilon\alpha)*(1-\varepsilon\beta)/3)+\varepsilon\beta/\varepsilon\alpha]^{0.5}$
7.2.42.	$z\varepsilon = (1/\varepsilon\alpha)^{0.5}$

Table 7.12 Additional relationships for gear design

Equation Number	Equation
7.2.43.	dr = d - 2*hf
7.2.44.	T = P*1000*60/(2πn)
7.2.45.	Yf is function of z (table 7.2)
7.2.46	z2 = z1*u

Table 7.12 Additional relationships for gear design (cont)

7.3 System Architecture.

Having considered the nature of a complex design such as gear pair, the architecture presented here (Fig 7.10) was developed to cater for such designs. At the onset of the consultation, the solution process is controlled by the main inference engine (block labelled "main inference engine"). Within the main inference engine reside the sub-inference engines and the control module. The consultation starts with the main inference engine transferring the process to the domain specific knowledge module through the sub-inference engine.

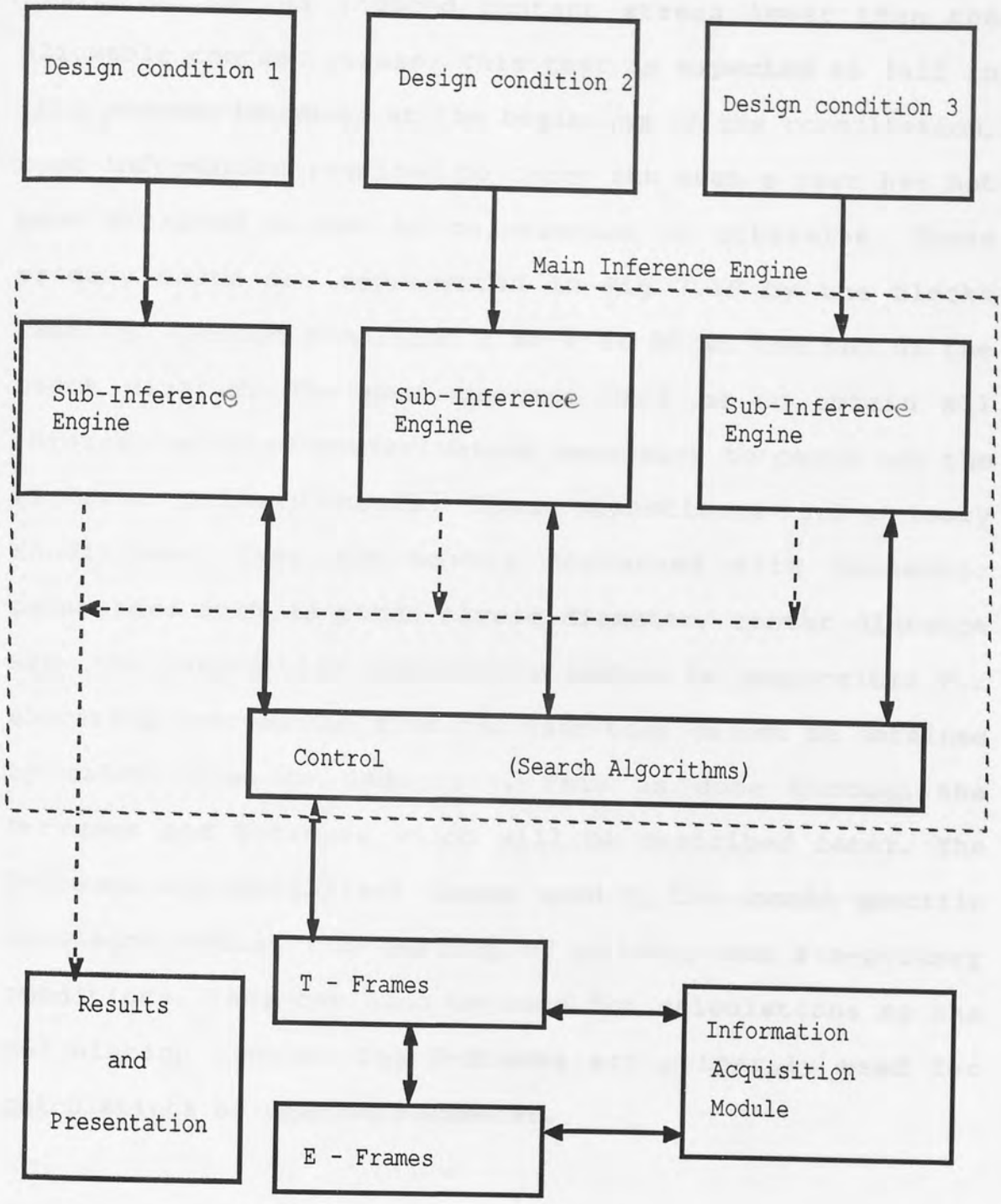


Fig 7.10 The general system structure.

The domain specific module will test for the primary conditions the proposed design is required to satisfy. For example, is the facewidth of the gear within acceptable

limits or is the induced contact stress lower than the allowable contact stress. This test is expected to fail in this process because, at the beginning of the consultation, most information required to carry out such a test has not been obtained either by calculation or otherwise. These primary tests are represented in Fig 7.10 by the blocks labelled "Design condition 1 or 2 or 3" at the top of the block diagram. The next process will be to obtain all information or parameter values necessary to carry out the required primary tests. These constitute sub-primary conditions. They are mainly concerned with geometric parameters such as pitch circle diameter, center distance etc. The information acquisition module is responsible for eliciting information from the user that cannot be obtained by calculation or deduction. This is done through the T-frames and E-frames which will be described later. The T-frames are specialised frames used by the domain specific knowledge module for testing of primary and sub-primary conditions. They can also be used for calculations by the calculation module. The E-frames are primarily used for calculations of unknown parameter.

Each main section of the programm will now be elaborated in the next sections.

7.4 The Information acquisition module:

This module is responsible for information gathering for parameter values such as the pitch circle diameter, module

etc. It is also responsible for obtaining information which allows the system to deduce required information such as the application factor Ka. For example, questions like "the type of driving (such as combustion engine) or driven machines (such as generators or elevators).

Part of the module resides in the E-frame or T-frame. Since there is no knowledge grouping involved, the module has less functions that its counterpart in the Level based concept (LBC) system. It however checks that inputs are of the correct type. For example numerals or character strings. The user screen display is part of this module.

7.4.1 Gearex: User screen display

The user screen display of Gearex is divided into five windows as shown in Fig 7.11. The top window displays a permanent title of the component being designed. In this case "Spur and Helical Gear Design". The next window down displays all explanations, advice and information on the progress of the process under execution. The third window (mid screen) shows the Node name and subfunction names. Unlike Helcom, the nodes and subfunction names are shown as each stage of the solution process is executed. It gives an indication that there is execution in progress and, as the user gets familiar with the program, the user can have an indication of what the system is doing. The process itself is made extremely transparent as the user becomes familiar with the system. The fourth window displays all instructions and options open to the user at any point.

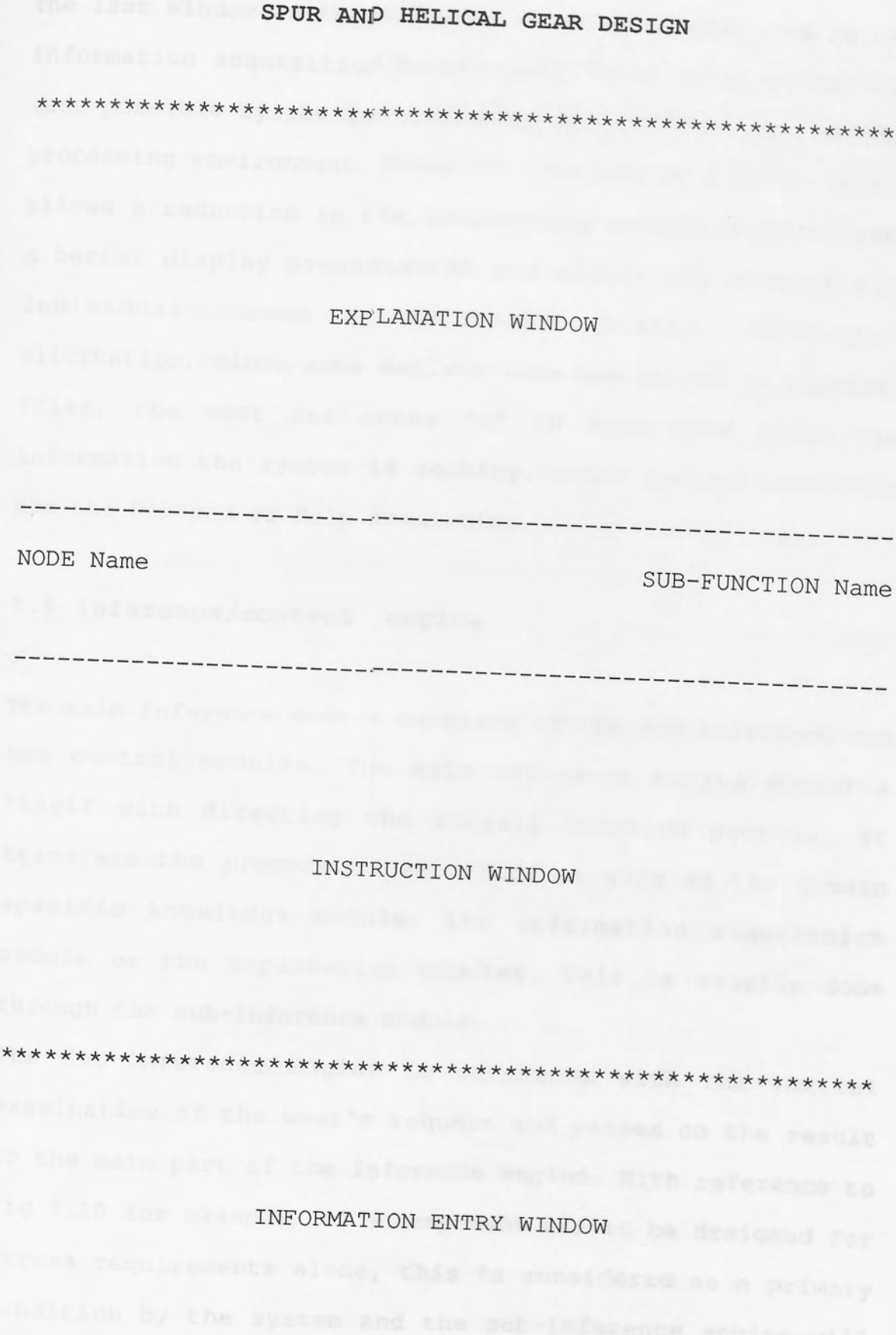


Fig 7.11 User screen display

The last window is the information entry window used by the information acquisition module only. This split screen was made possible by the conversion of Microsynics into another processing environment (GWBasic). The use of a split screen allows a reduction in the programming necessary to achieve a better display presentation and allows the scrolling of individual screen as necessary. During information elicitation, since some explanations are stored in separate files, the user can enter "m" to know more about the information the system is seeking. Other options available are the Notepad or Help facilities.

7.5 Inference/control engine

The main inference engine consists of the sub-inference and the control modules. The main inference engine concerns itself with directing the overall solution process. It transfers the process to main modules such as the domain specific knowledge module, the information acquisition module or the explanation modules. This is usually done through the sub-inference module.

The Sub-inference Engine is concerned with the initial examination of the user's request and passes on the result to the main part of the inference engine. With reference to Fig 7.10 for example, if a component was to be designed for stress requirements alone, this is considered as a primary condition by the system and the sub-inference engine will call the T-Frames (domain specific knowledge) associated with this requirement for initial evaluation of information

and conditions. If at the onset, there is enough information, then the sub-inference engine transfers it to the main inference engine. This then either calculates for all other relevant information, if necessary, or transfers the information to the result and presentation module for presentation. If there is not enough information, then the sub-inference engine will have to transfer the process to the main inference section for execution. The process is transferred back to the sub-inference only when a decision has been taken by the main frame as to the possibility of having a satisfactory design. The sub-inference conducts a re-test (through the Domain specific knowledge module) to confirm the decision before the results are either presented or the user informed.

The control part of the inference engine controls how the search for information proceeds and decides which E-Frame(s) should be called on. It searches for the variable associated with primary rules and vice versa. The search is mainly concerned with geometric parameters. A typical process will be described after the functions of frames have been described.

7.5.1 Variable status and conditions.

The information on each parameter helps the inference engine in conducting the overall solution process. To help in knowing the state of each parameter (that is, if the parameter value is known or not and, if it is, then is it given or calculated). Each variable has a status flag assigned to it. The status flag informs the inference

engine of the nature of the variable and whether a value has been assigned to it or not and how the value was assigned. This enables the inference engine to keep track of what occurs in other parts of the system and especially in the frames. The table shown below (Table 7.13) gives the eight states of a variable.

STATUS No	FL**%	ASKED	GIVEN	CALCULATED
1	000	0	0	0
2	100	1	0	0
3	010	0	1	0
4	001	0	0	1
5	110	1	1	0
6	101	1	0	1
7	011	0	1	1
8	111	1	1	1

Table 7.13: Status Flags

An example of flag FL**% is FLPC%, which is the status flag assigned to the variable called Circular pitch (with variable code p). Each variable has a code name given to it.

State 1 above is the initial state. When the consultation starts, all variables assume this state. The variable values have not been asked for by the system nor has it

been volunteered by the user nor has it been calculated. This state also occurs when the system does not have enough information to calculate the variable value and the control has not instructed the necessary frames to ask for the variable value.

State 2, means that the variable value has been asked for from the user but was not given and the system has not calculated it due to insufficient information.

State 3, indicates that the variable value has been given or obtained. The variable has not been asked for and was not calculated either because of insufficient information or the frame(s) concerned has not done so yet and will no longer try to calculate it since the value is now known. This state indicates that an old design is undergoing modification and the relevant design information and variable value have been obtained from a databank.

State 4, occurs when the variable value was obtained through calculations. It also indicates that the variable value was not asked for by the system. The system had sufficient information to calculate the variable value without needing to ask the user for it.

State 5, means that the variable value was asked for and given by the user. The system was not able to calculate the variable value due to lack of necessary information to carry out the task.

State 6, This state indicates that the system was not able to calculate the variable value due to lack of information and had to ask the user to provide this value but the user did not give the required value. The system, however was able to gather enough information at a later stage to enable it to calculate the variable value.

State 7, this state should not occur. It is used as a monitor for any logical errors in the reasoning process in the frames associated with the variable concerned. This status is called the Null status. The state should not occur for the following reasons: if the variable value was obtained from a databank for an old design, then it is a waste of time for the system to endeavour to calculate the variable value and on the other hand, if the variable value was initially calculated, then the system should not seek to obtain the same value from the databank of an old design. Only if a frame or frames concerned with the variable under consideration passes on the wrong state of affairs to the control will it be possible for the state to occur. This will lead to the system and other frames taking the wrong decision based on the information they recieve. To avoid this, each frame checks for the occurrence of this condition and it serves as a constant check on previously executed frames.

State 8 is a more severe occurrence of state 7. It, however, indicates that the variable value was either

calculated initially or obtained from the databank of an old design. This was somehow not acknowledged by the frame responsible and hence was not noted by the control during later processes or asked the same frame or another frame to ask for the variable value.

In all the states mentioned above, the variable value has been asked for once, the system does not ask for it again and tries to obtain the value by other means.
The same variable value will only be asked for again if and only if, the user, on the advice of the system, decides to discard either the calculated value or the value obtained from the databank of an old design, or the value given by that user when asked.
One of the advantages of assigning a status flag to each variable is that as the system gets bigger and more complex and more knowledge is added on, logical errors are likely to occur and can be traced by the combination of the process history and the current state and status of the variables.

7.6 Domain specific knowledge module

The Domain specific knowledge module consists of the section responsible for equation manipulation such as the E-frames, heuristic knowledge (some of which are production rules) and the section responsible for testing such as T-frames.

7.6.1 E-Frames

There are two types of E-Frames denoted by "E2" and "E3". The E3-Frames are those that consist of equations that have three variables to manipulate and the E2-Frame have two variables and a constant. Equations involving more than three variables are handled by a combination of E2 and E3-frame networks. For example, a five variable equation will consist of one E2-frame and one E3-frame. The advantage of this architecture is that it accommodates instances were a parameter cannot easily be classified as a variable or constant. Thus, if it is considered a constant, an E2-frame will handle the process and if it is considered as a variable (probably in later updated version) then an E3-frame will handle the process. An example is the Lewis equation : $Ft = \sigma * \pi^2 * b * m^2 * Yf$.

The factor Yf can be obtained from Table 7.2, but this table was compiled from experimental data. It is dependent on the number of gear teeth. On the other hand, an equation may in the future be developed to give an accurate value for this factor which may be dependent not only on the number of gear teeth but on other factors. If that occurs, the factor will be considered as a parameter. Either approach is convenient with the method developed. These frames are designed in such a way that they are totally independent of each other and of the preceding or subsequent frames. Each frame consists of an equation which it manipulates and any information which has been requested

concerning the particular equation and variables associated with it. It will only do what it is instructed to do by the inference part of the system and will carry out the task independent of the control once the process has been transfered to it. All decisions are made in light of the information passed on to it by the sub-inference engine or by the control. It also bases its decision on the information it gathers itself.

A frame is normally in one of two states, the Off state or the On state. These states are dictated by the control module. In the Off state (that is, the user communication flag is zero) the frame will not allow communication with the user through the information acquisition module, it will not elicit any information from the user, it will only manipulate an equation to derive values or information needed or try to deduce information. If it fails to obtain the information it needs, it passes the process back to the control module. However, in this state it will ask the user for confirmation of decisions made.

In the On state, (that is, the user communication flag is 1), the frame will allow communication with the user, it elicits the information it requires from the user. The frame will allow information to be requested from the user if there is insufficient information to carry out its task. All information requested is recorded and the control informed. The control is not informed about the violation or non-violation of primary rules as this does not concern

the control. This type of sub-problem (the violation of a primary rule) is dealt with by the frame and until the rule is no longer violated the process is not transferred back to the control. The general block diagram for a frame is shown in Figs 7.12 and 7.13. These figures give the process

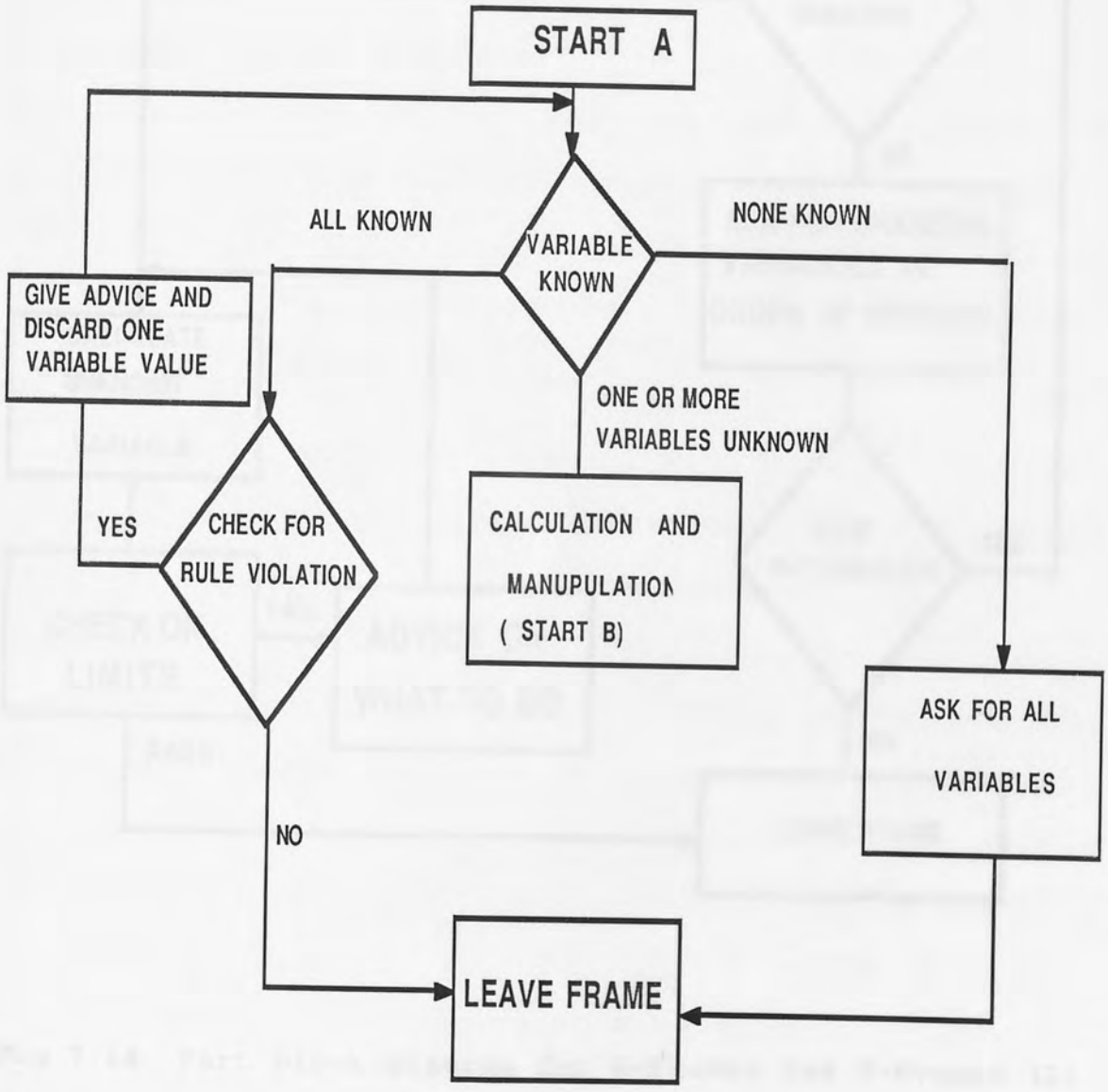


Fig 7.12 Part block diagram for E-Frames and T-Frames

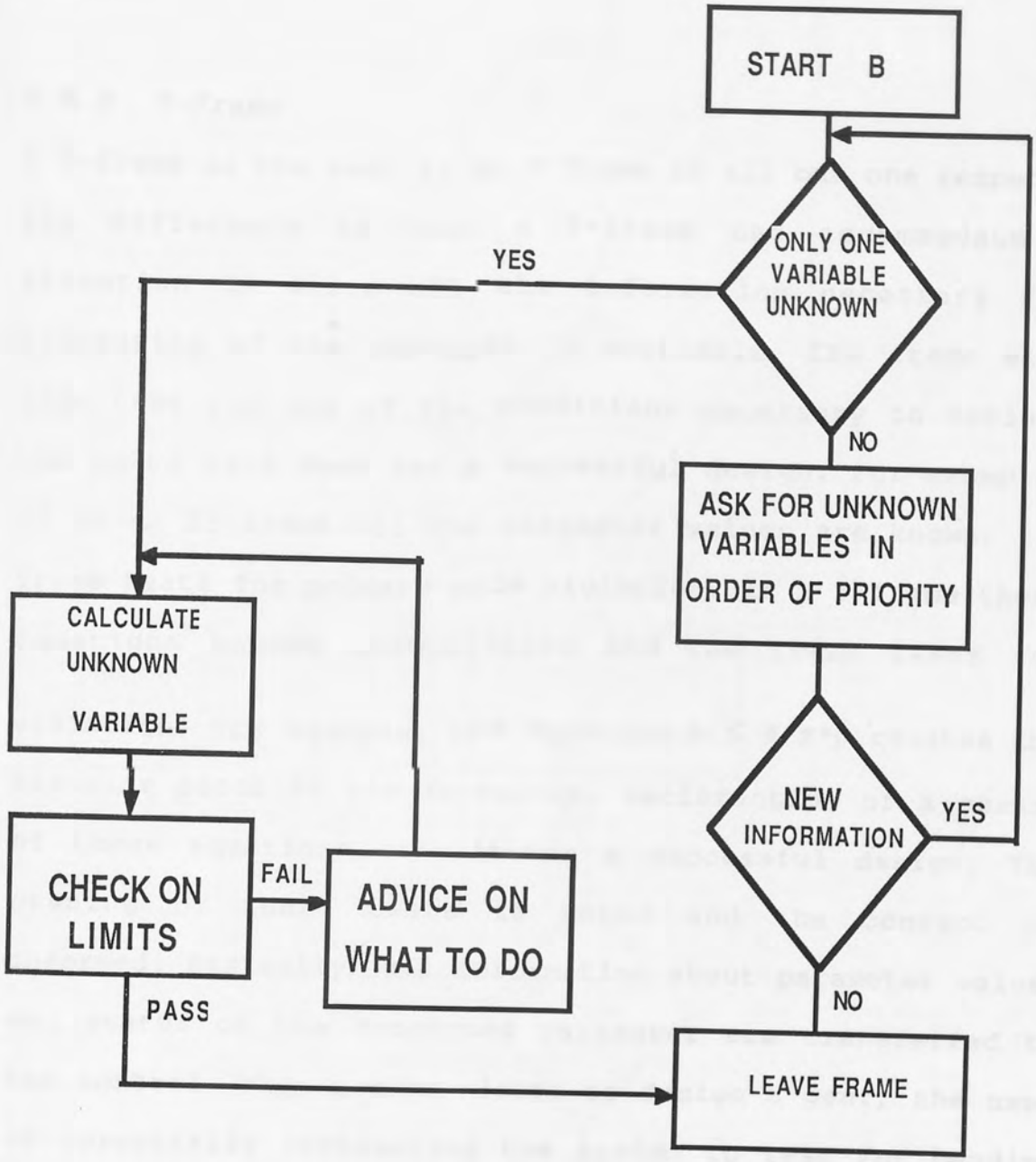


Fig 7.13 Part block diagram for E-Frames and T-Frames (2)

of information determination concerning a specific equation in an E-frame. Fig 7.14 shows a general variable process flow diagram.

Another type of frame will now be discussed in the next section.

7.6.2 T-Frame

A T-frame is the same as an E-frame in all but one respect. The difference is that a T-frame can accommodate a situation in which all the information necessary for processing of the sub-task is available. The frame will also test for one of the conditions necessary to satisfy the rules laid down for a successful design. For example, if in an E3-frame all the parameter values are known, the frame tests for primary rule violation. In a T-frame these equations become inequalities and the frame tests for violation. For example, the equation $b \leq 4*\pi*p$ relates the circular pitch to the facewidth. Satisfaction of a series of these equations constitutes a successful design. The passing of these tests is noted and the control is informed. Basically, all information about parameter values and status of the concerned parameter are transferred to the control. When a user wishes to design a gear, the user is essentially instructing the system to test for bending and contact stresses. The success of the design depends on the satisfaction of these tests. Other tests on geometric dimensions must also be passed; for example $b \geq \pi m$.

Suppose in a typical design search process the gear ratio, power, pinion speed and pinion number of teeth are the parameter values given to the process is as follows.

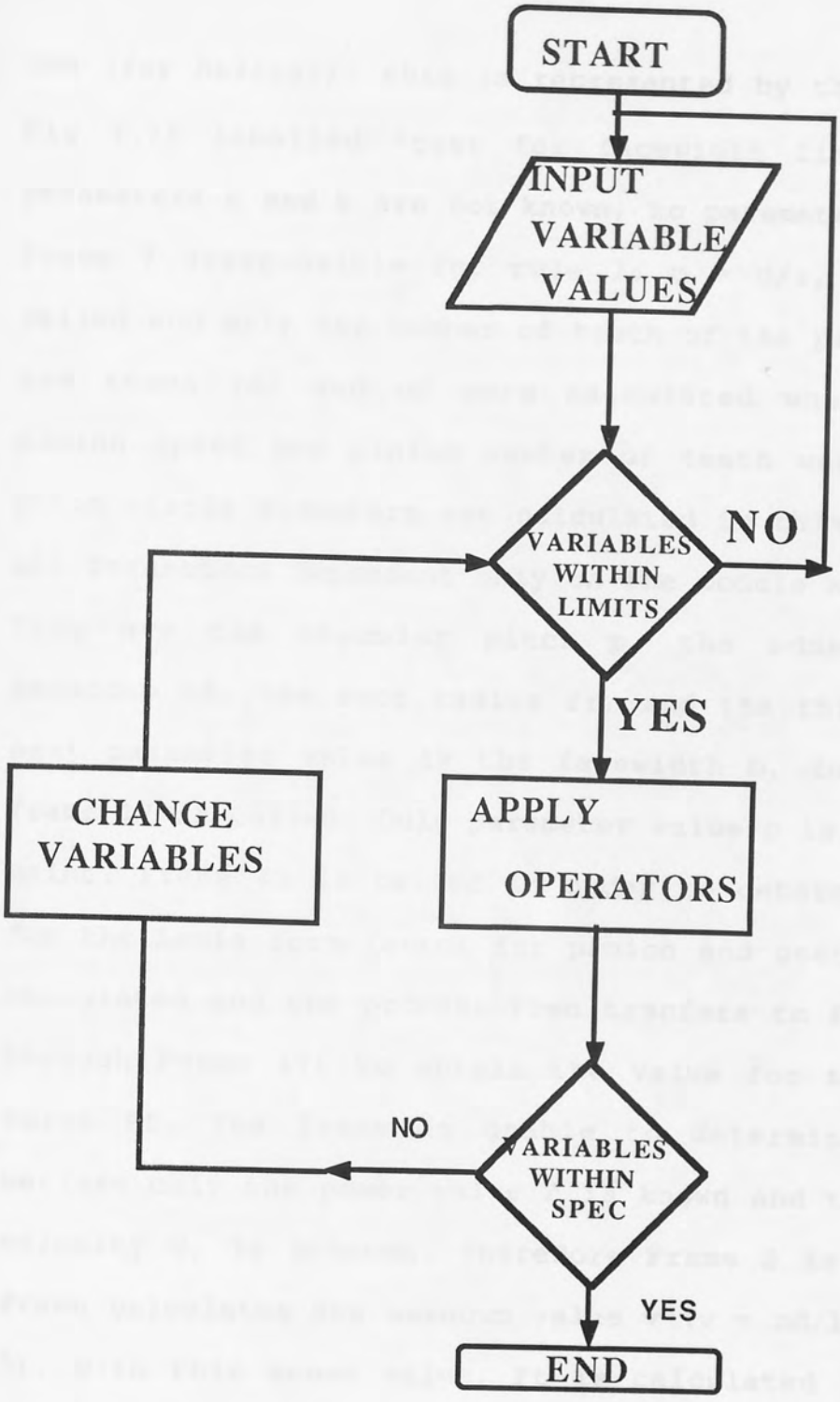


Fig 7.14 The Variable process flow diagram.

The main condition test is carried out as described above using the equation $\pi m \leq b \leq 4\pi m$ (for spur) or $\pi m/\tan\beta \leq b \leq$

$6\pi m$ (for helical); this is represented by the top block in Fig 7.15 labelled "test for facewidth limits". The two parameters m and b are not known, so parameter m is sought. Frame 7 (responsible for rule 7: m = d/z, Table 7.5) is called and only the number of teeth of the pinion and wheel are known (z2 and n2 were calculated when gear ratio, pinion speed and pinion number of teeth were given). The pitch circle diameters are calculated in this module. Also, all parameters dependent only on the module are calculated. They are the circular pitch p, the addendum ha, the dedendum hf, the root radius rr, and the thickness t. The next parameter value is the facewidth b, for both gears. Frame 17 is called. Only parameter value p is known at this point. Frame 45 is called in order to determine the value for the Lewis form factor for pinion and gear, Yf. This is calculated and the process then tranfers to Frame 16 (back through Frame 17) to obtain the value for the tangential force Ft. The frame is unable to determine this value because only the power value P is known and the pitch line velocity v, is unknown. Therefore Frame 2 is called. This frame calculates the unknown value v (v = nd/19098 equation 2). With this known value, Ft is calculated from Frame 16 and the process tranfers to Frame 18 through 17 to determine the induced stress. The allowable stress is taken to be equal to the induced stress. Therefore, with the

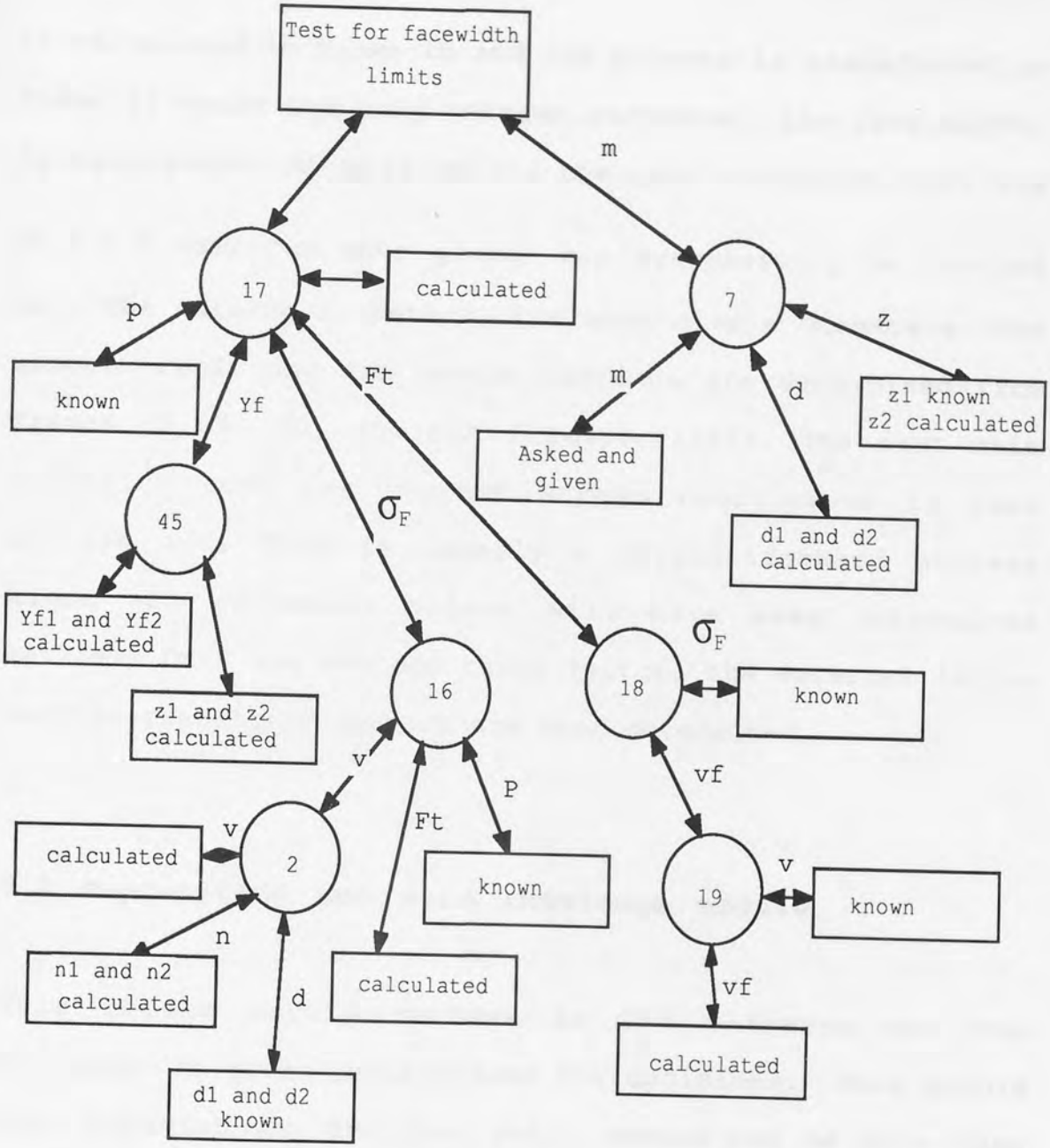


Fig 7.15. A typical process path for facewidth requirements

parameter value for the permissible bending stress known (from material selection) and the velocity factor unknown, Frame 19 is called. This frame determines the value for the velocity factor which is dependent on the pitch line velocity only (equation 7.2.19). Hence the induced stress

is calculated in Frame 18 and the process is transferred to Frame 17 where the only unknown parameter, the face width, is calculated. At this point, the main condition test (ie $\pi m \leq b \leq 4\pi m$, for spur gears) can successfully be carried out. The outside diameters, the base circle diameters, the contact ratio and the centre distance are determined from frames 15, 6, 30, 30a and 1 respectively. The next main condition test for contact stress requirement is then carried out. This is usually a straightforward process since all parameter values will have been determined earlier. Only the contact ratio factor, the material factor and the elasticity factors are then determined.

7.7 Explanation and Self knowledge module.

This module mainly resides in the E-frames and the T-frames. It gives explanations for decisions. This module also explains why the gear ratio should not be more than 10, or why the gear material should be changed instead of the module. This module is also responsible for informing the user that further progress cannot be made due to insufficient information.

7.8 The model for helical spring using FBC.

If the helical compression spring design process were to be modelled using FBC, the main condition requirements will be

based on the buckling test and the deflection requirements for static loading. For dynamic loading, surging and fatigue tests are added requirements. The deflection requirements will be that the given (or actual) deflection δa must be lower than the physical one and that under maximum stress ($\delta max2$ and $\delta max1$ respectively). The network process diagram is shown in Fig 7.16. The circles represent frames and the top block is the initial test for deflection requirements. At the start of the search process, it is assumed that the deflection value required for the test are unknown. If this is the case, assuming that the actual deflection δa is the first parameter value to be sought, the frame C5 is called. If the spring rate is not given when requested, the actual deflection cannot be calculated. If given, the value of the load F will then be the parameter to be deteremined. To obtain this value, the frame C2 is called after checking if a load value has been given. If it has, the normal routine of checking if this load is less than the load at maximum stress is conducted. The parameters associated with this frame are requested from the user if enough information is not available. If still insufficient information is given by the user, than frames containing each parameter are called in turn. For example, frame A1 is called in an attempt to obtain the value for c. No frame is called twice in the cause of determining a parameter value with no information offered. If new information is obtained, all frames previously called can

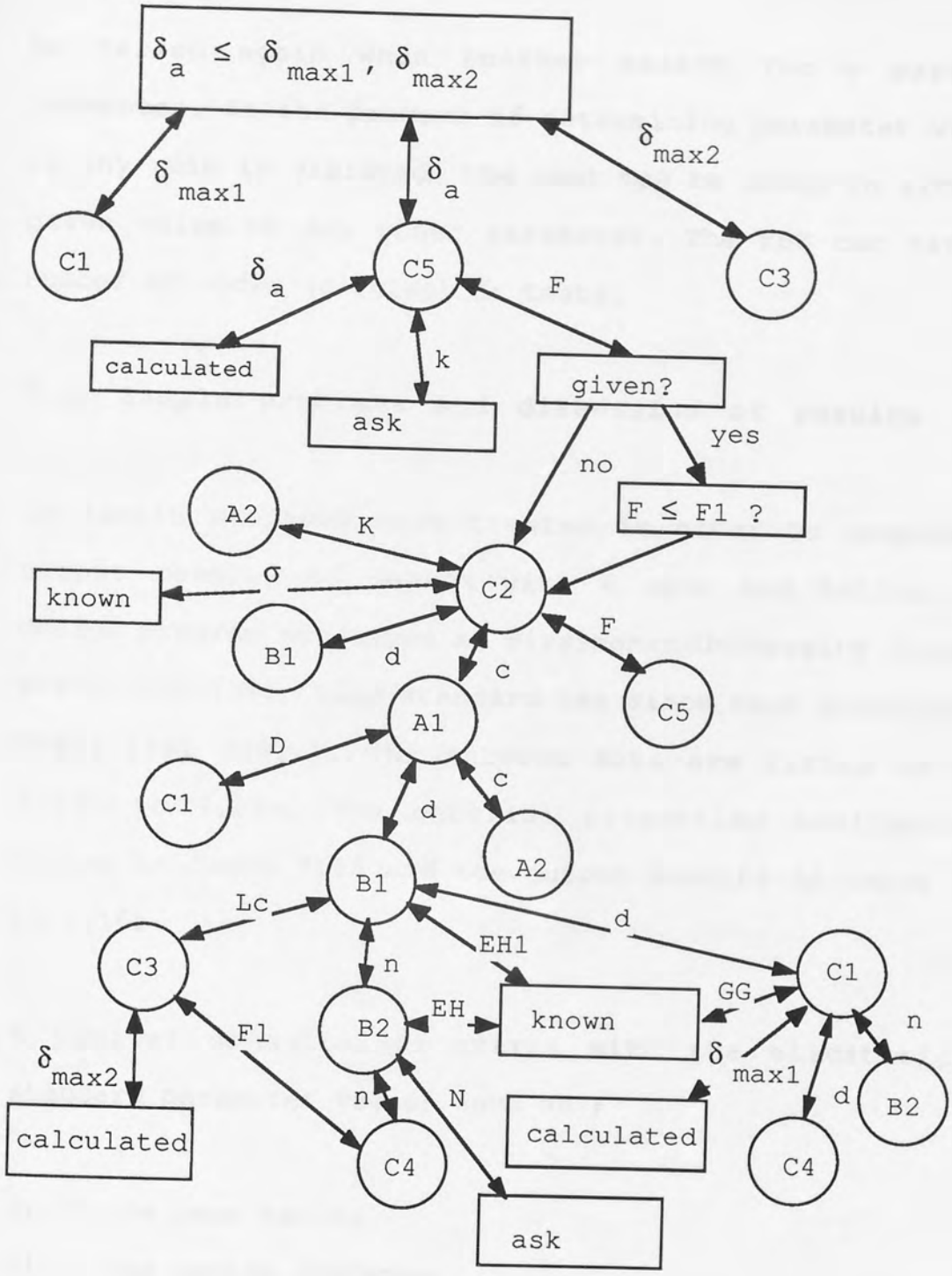


Fig 7.16 Network for helical spring deflection requirements.

be called again when another search for a parameter commences. In the process of determining parameter values, if any rule is violated, the user may be asked to alter the given value or any other parameter. The FBC can have any number of nodes (circles) or tests.

7.9 Sample problems and discussion of results.

18 sample problems were treated in order to compare the output results of Gearex with a spur and helical gear design program developed at Birmingham University (based on BS436:1940 [90], this standard has since been superceded by BS436:1986 [20],). The entered data are listed in table 7.14a to 7.14e. The material properties available are listed in table 7.15 and the output results in table 7.16a to 7.16i.

A typical consultation starts with the elicitation of standard parameter values such as ;

i) the gear ratio,
ii) the centre distance,
iii) the power transmitted (kW),
iv) the pinion speed (rev/min),
v) the type of gear material for pinion and wheel.

The value of the pinion teeth is assumed to be the minimum number of teeth and the user is informed of this fact and has the option to reject this value and enter a choice.

Parameter	Values			
Design number	1	2	3	4
Gear ratio	4	6	6	8
Minimum teeth	16	16	16	17
power (kW)	80	80	80	80
Pinion speed	2000	2000	2000	2000
Surface finish	ground	ground	ground	ground
Pinion material	1	1	2	3
Wheel material	1	1	2	3
Gear type	spur	spur	spur	spur

Table 7.14a Design Specification

Parameter	Values			
Design number	5	6	7	8
Gear ratio	8	10	5	7
Minimum teeth	17	17	16	17
power (kW)	100	100	100	100
Pinion speed	4000	4000	4000	4000
Surface finish	ground	ground	ground	ground
Pinion material	4	5	4	9
Wheel material	4	5	2	9
Gear type	spur	spur	spur	spur

Table 7.14b Design Specification

Parameter	Values			
Design number	9	10	11	12
Gear ratio	7	7	4	4
Minimum teeth	17	17	16	16
power (kW)	200	200	80	80
Pinion speed	10000	10000	2000	2000
Surface finish	ground	ground	ground	ground
Pinion material	1	1	1	1
Wheel material	1	1	1	1
Gear type	spur	spur	helical	helical

Table 7.14c Design Specification

Parameter	Values			
Design number	13	14	15	16
Gear ratio	4	4	4	4
Minimum teeth	16	16	16	16
power (kW)	80	80	80	100
Pinion speed	2000	2000	2000	6000
Surface finish	ground	ground	ground	ground
Pinion material	1	1	1	1
Wheel material	1	1	1	1
Gear type	helical	helical	helical	helical

Table 7.14d Design Specification

Parameter	Values	
Design number	17	18
Gear ratio	4	4
Minimum teeth	16	16
power (kW)	100	200
Pinion speed	6000	20000
Surface finish	ground	ground
Pinion material	1	1
Wheel material	1	1
Gear type	helical	helical

Table 7.14e Design Specification

Number	Material	U.T.S	SS	BS
1	Cast Iron	247.1	9.31	52.4
2	Phosphor Bronze	185.3	4.9	49.6
3	Mild steel	494.2	9.64	117.2
4	Carbon steel (n)	540.59	9.64	131
5	Carbon steel (HT)	617.8	13.79	168.9
6	Carbon steel (SH)	540.59	19.31	117.2
7	Alloy steel	849.4	20.68	231
8	Alloy steel (SH)	849.4	35.16	182.7
9	Mild steel (CH)	494.2	63.43	275.79

U.T.S = Ultimate tensile stress (N/mm^2)

SS = Surface stress (N/mm^2)

BS = Bending stress (N/mm^2)

n = Normalised

HT = Hardened, tempered

SH = Surface hardened

CH = Case hardened

Table 7.15 Material properties as given by Birmingham university gear design program for spur and helical gears (BUGDP).

Parameter	BUGDP1	GRX1	BUGDP2	GRX2
z1	16	16	16	16
z2	64	64	96	96
m (mm)	10	10	10	10
a (mm)	400	400	560	560
b (mm)	90.97	112.7	85.92	112.8
WG	P	P	P	P
d1 (mm)	160	160	160	160
d2 (mm)	640	640	960	960
do1 (mm)	186	186	186.67	186.67
do2 (mm)	654	654	973.33	973.33
ha1 (mm)	13	10	13.33	10
ha2 (mm)	7	10	6.67	10
dr1 (mm)	137.2	135	137.87	135
dr2 (mm)	605.2	615	924.53	935
db1 (mm)	150.35	150.35	150.35	150.35
db2 (mm)	601.4	601.4	902.11	902.11
T (N m)	381.97	381.97	381.97	381.97
ε	1.57	1.64	1.58	1.67
v (m/s)	16.76	16.76	16.76	16.76
Ft (kN)	4.77	4.77	4.77	4.77
Fr (kN)	1.74	1.74	1.74	1.74
Conclusions	FWR	FWR	FWR	FWR

Table 7.16a Design results

Parameter	BUGDP3	GRX3	BUGDP4	GRX4
z1	16	16	17	17
z2	96	96	136	136
m (mm)	12	12	10	10
a (mm)	721.6	721.6	882.2	765
b (mm)	117.58	38.06	71.33	21.7
WG	P	P	P	P
d1 (mm)	192	192	170	170
d2 (mm)	1152	1152	1360	1360
do1 (mm)	224	224	197	190
do2 (mm)	1168	1168	1373	1380
ha1 (mm)	16	12	13.35	10
ha2 (mm)	8	12	6.5	10
dr1 (mm)	165.44	162	148.2	145
dr2 (mm)	1109.44	1122	1324.2	1335
db1 (mm)	180.42	180.42	159.75	159.75
db2 (mm)	1082.53	1082.53	1277.98	1277.98
T (N m)	381.97	381.97	381.97	381.97
ε	1.58	1.67	1.59	1.7
v (m/s)	20.1	20.1	17.8	17.8
Ft (kN)	3.98	3.98	4.49	4.49
Fr (kN)	1.45	1.45	1.64	1.64
Conclusions	FWR	FWR	FWR	FWR

Table 7.16b Design results

Parameter	BUGDP5	GRX5	BUGDP6	GRX6
z1	17	17	17	17
z2	136	136	170	170
m (mm)	8	8	8	8
a (mm)	612	612	748	748
b (mm)	10.38	21.7	7.08	16.82
WG	P	P	P	P
d1 (mm)	136	136	136	136
d2 (mm)	1088	1088	1360	1360
do1 (mm)	157.6	152	157.6	152
do2 (mm)	1098.4	1104	1370.24	1376
ha1 (mm)	10.8	8	10.88	8
ha2 (mm)	5.2	8	5.12	8
dr1 (mm)	118.56	116	118.72	116
dr2 (mm)	1059.36	1068	1331.2	1340
db1 (mm)	127.8	127.8	127.8	127.79
db2 (mm)	1022.39	1022.39	1277.98	1277.92
T (N m)	238.73	238.63	238.73	238.64
ε	1.59	1.7	1.59	1.71
v (m/s)	28.48	28.48	28.48	28.48
Ft (kN)	3.5	3.51	3.5	3.51
Fr (kN)	1.28	1.28	1.28	1.28
Conclusions	FWR	-	FWR	-

Table 7.16c Design results

Parameter	BUGDP7	GRX7	BUGDP8	GRX8
z1	16	17	17	17
z2	80	85	119	119
m (mm)	10	10	5	5
a (mm)	480	510	615.8	340
b (mm)	91.86	27.77	32.22	53.56
WG	W	W	P	P
d1 (mm)	160	170	85	85
d2 (mm)	800	185	595	595
do1 (mm)	186.4	190	98.43	95
do2 (mm)	813.6	870	601.57	605
ha1 (mm)	13.2	10	6.71	5
ha2 (mm)	6.8	10	3.29	5
dr1 (mm)	137.6	145	74.03	72.5
dr2 (mm)	764.8	825	577.17	582.5
db1 (mm)	150.35	159.74	79.81	79.87
db2 (mm)	751.75	798.7	559.12	559.09
T (N m)	238.73	238.64	238.73	238.64
ε	1.57	1.67	1.59	1.69
v (m/s)	33.51	35.61	17.8	17.81
Ft (kN)	2.98	2.81	5.62	5.62
Fr (kN)	1.09	1.02	2.04	2.05
Conclusions	FWR	-	FWR	FWR

Table 7.16d Design results

Parameter	BUGDP9	GRX9	BUGDP10	GRX10
z1	17	17	17	17
z2	119	119	119	119
m (mm)	10	10	8	8
a (mm)	680	680	544	544
b (mm)	85.54	38.15	127.82	55.67
WG	P	P	P	P
d1 (mm)	170	170	136	136
d2 (mm)	1190	1190	952	952
do1 (mm)	196.86	190	157.49	152
do2 (mm)	1203.14	1210	962.51	968
ha1 (mm)	13.43	10	10.74	8
ha2 (mm)	6.57	10	5.26	8
dr1 (mm)	148.06	145	118.45	116
dr2 (mm)	1154.34	1165	923.47	932
db1 (mm)	159.75	159.74	127.8	127.79
db2 (mm)	1118.23	1118.18	894.59	894.54
T (N m)	190.99	190.91	190.99	190.91
ε	1.59	1.69	1.59	1.69
v (m/s)	89.01	89.01	71.21	71.21
Ft (kN)	2.3	2.25	2.81	2.81
Fr (kN)	0.82	0.82	1.02	1.02
Conclusions	FWR	FWR	FWR	FWR

Table 7.16e Design results

Parameter	BUGDP11	GRX11	BUGDP12	GRX12
z1	16	16	16	16
z2	64	64	64	64
m (mm)	6	6	8	8
a (mm)	277.14	277.16	340.54	340.55
b (mm)	141.6	213.2	107.79	149.75
WG	P	P	P	P
d1 (mm)	110.86	110.86	136.22	136.22
d2 (mm)	443.42	443.46	544.87	544.88
do1 (mm)	126.46	122.86	157.02	152.22
do2 (mm)	451.82	455.46	556.07	560.89
ha1 (mm)	7.8	6	10.4	8
ha2 (mm)	4.2	6	5.6	8
dr1 (mm)	97.18	95.86	117.98	116.22
dr2 (mm)	422.54	428.46	517.03	560.88
db1 (mm)	102.2	102.2	127.02	127.02
db2 (mm)	408.78	408.79	508.09	508.07
T (N m)	381.97	381.82	381.97	381.82
ε	2.18	7	1.81	3.55
v (m/s)	11.61	11.61	14.26	14.26
Ft (kN)	6.89	6.89	5.61	5.61
Fr (kN)	2.9	2.51	2.17	2.04
Fa (kN)	3.98	3.98	2.04	2.04
Conclusions	FWR	FWR	FWR	FWR

Table 7.16f Design results

Parameter	BUGDP13	GRX13	BUGDP14	GRX14
z1	16	16	16	16
z2	64	64	64	64
m (mm)	8	8	8	8
a (mm)	353.09	353.11	390.67	390.71
b (mm)	96.7	148.18	75.8	144.07
WG	P	P	P	P
d1 (mm)	141.24	141.24	156.27	156.29
d2 (mm)	564.94	564.97	625.07	625.14
do1 (mm)	162.04	157.24	177.07	172.29
do2 (mm)	567.14	508.98	636.27	641.14
ha1 (mm)	10.4	8	10.4	8
ha2 (mm)	5.6	8	5.6	8
dr1 (mm)	123	121.24	138.03	136.28
dr2 (mm)	537.1	544.98	597.23	605.14
db1 (mm)	131.06	131.06	142.8	142.81
db2 (mm)	524.25	524.23	571.22	571.23
T (N m)	381.97	381.82	381.97	381.82
ε	1.96	3.93	2.49	4.54
v (m/s)	14.79	14.79	16.36	16.37
Ft (kN)	5.41	5.41	4.89	4.89
Fr (kN)	2.17	1.97	2.17	1.78
Fa (kN)	2.52	2.52	3.42	3.42
Conclusions	FWR	FWR	FWR	FWR

Table 7.16g Design results

Parameter	BUGDP15	GRX15	BUGDP16	GRX16
z1	16	16	16	16
z2	64	64	64	64
m (mm)	8	8	8	8
a (mm)	417.76	417.83	353.09	353.11
b (mm)	81.05	141.56	57.23	39.01
WG	P	P	P	P
d1 (mm)	167.1	167.13	141.24	141.24
d2 (mm)	668.42	668.53	524.94	564.98
do1 (mm)	187.9	183.13	162.04	157.24
do2 (mm)	679.62	684.53	576.14	580.98
ha1 (mm)	10.4	8	10.4	8
ha2 (mm)	5.6	8	5.6	8
dr1 (mm)	148.86	147.13	123	121.24
dr2 (mm)	640.58	648.53	537.1	544.98
db1 (mm)	150.93	150.94	131.06	131.06
db2 (mm)	603.73	603.76	524.25	524.24
T (N m)	381.97	381.82	159.15	159.09
ε	2.93	4.75	1.96	2.09
v (m/s)	17.5	17.5	44.37	44.37
Ft (kN)	4.57	4.57	2.25	2.25
Fr (kN)	2.17	1.66	0.91	0.82
Fa (kN)	3.84	3.84	1.05	1.05
Conclusions	FWR	FWR	FWR	FWR

Table 7.16h Design results

Parameter	BUGDP17	GRX17	BUGDP18	GRX18
z1	16	16	16	16
z2	64	64	64	64
m (mm)	6	6	6	6
a (mm)	317.22	264.83	317.22	264.83
b (mm)	96.05	64.307	113.08	54.75
WG	P	P	P	P
d1 (mm)	105.93	105.93	105.93	105.93
d2 (mm)	423.71	423.73	423.71	423.73
do1 (mm)	121.53	117.93	121.53	117.93
do2 (mm)	432.11	435.73	432.11	435.73
ha1 (mm)	7.8	6	7.8	6
ha2 (mm)	4.2	6	4.2	6
dr1 (mm)	92.25	90.93	92.25	90.93
dr2 (mm)	402.83	408.73	402.83	408.73
db1 (mm)	98.3	98.29	98.3	98.29
db2 (mm)	393.19	393.18	393.19	393.18
T (N m)	159.15	159.09	95.49	95.45
ε	1.96	2.88	1.96	2.66
v (m/s)	33.28	33.28	110.93	110.93
Ft (kN)	3	3	1.8	1.8
Fr (kN)	1.21	1.094	0.72	0.66
Fa (kN)	1.4	1.4	0.84	0.84
Conclusions	FWR		FWR	

Table 7.16i Design results

The minimum number of teeth is calculated from equation (hence rule) 7.2.36. For spur gears, this value is 17 and for helical gear it ranges from 6 (for 45 degrees helix angle) to 16 (for 10 degrees helix angle).

Search procedure

The main design analysis starts with the test of the main condition by T-frame. That is, $4\pi m \geq b \geq \pi m$ for spur gears and $6\pi m \geq b \geq \pi m/\tan\beta$ for helical gears. Since it is the beginning of a consultation, the values of m and b are unknown therefore the search process commences to obtain the values for these parameters. The Frame 7 is called in an attempt to calculate the value of m. Since d is not known, this is not successful and the control is informed. The control passes the process back again to Frame 7 with the instruction to elicit the information from the user. A value of m is offered, say 10 for sample problem 1 (table 7.12a, column 1). The values for d1 and d2 are then calculated.

With this new information, the the T-frame tests for the main condition. This test will also be unsuccessful since the value of b has not yet been determined. This process of testing is repeated every time new information is obtained by the system until the required information is obtained. If the value of m is not offered, all frames responsible for rules associated with the parameter m will be called. However, only rule 7 is involved here. The other rules

(8,9,10,11,13 and 14) will not be called unless m is known. With m known, p, ha, hf, c, rr and t are determined. The system then attempts to seek a value for b. Frame 17 is the only frame that will be called. Rules 3, 4, 20 to 25, and 40 are not installed in the current system. Rule 3 and 4 were not installed because they were considered surplus to requirements; the values can be determined from other rules. Rules 20 to 23 were not installed because they are used for the determination of wear and dynamic loading conditions and they can also be determined from 26 and 27. Rules 24 and 25 determine the bending stress requirements, which is also provided by 17 to 19. However, these rules can be installed if desired. With the process in Frame 17, Yf is obtained from rule 45 for pinion and wheel. The tangential force is the next parameter to be determined. For this Frames 16, and 39 are called. Ft is not obtained because of lack of other information. The system then proceeds to obtain the value for v. Only Frame 2 is associated with this parameter (16 has already been used unsuccessfully). With this rule, only v is unknown therefore the value is calculated (16.76 m/s).

Having determined the value for v, Ft (4.77 kN) can be calculated from Frame 16. When the process backtracks to Frame 17, the induced stress is the only parameter not known and will therefore be sought. The induced stress should not exceed the allowable stress corrected for velocity effects (equation 7.2.17), therefore by taking the induced stress to be of same value as the allowable stress,

the calculated facewidth (from equation 7.2.18) will be the minimum value acceptable to satisfy the stress criteria. Frame 18 is therefore called. The bending stress is known (52.4 N/mm^2) and the velocity factor is next determined from frame 19. With this parameter now known, the facewidth for the pinion and wheel is obtained and the main test carried out (i.e., $4\pi m \geq b \geq \pi m$ for spur gears and $6\pi m \geq m \geq \pi m/\tan\beta$ for helical gears). The value of b1 and b2 determined are 114.54 mm and 82.56 mm respectively. The minimum and maximum recommended values are 31.43 mm and 125.71mm respectively. Since the calculated facewidth for the pinion and wheel (i.e., b1 = 114.54 mm and b2 = 82.56 mm) are the minimum acceptable values, the pinion facewidth is recommended for both gears. Therefore, the gears should have a facewidth of 114.56 mm.

The next test is for the contact stress requirements. The test for this is more straightforward since most of the parameter values have been determined earlier. From frame 27, only the material factor Zm is considered. The others, $Z_L, Z_v, Z_R, Z_x, Z_w, Z_N$ are assumed to be unity (see section 7.2, section on "tooth forces and stresses"), that is, they have no effect on the contact stress. From rule 28, the factors considered are the elasticity factor ZE, the application factor Ka and the contact ratio factor $Z\varepsilon$. The actual contact stress calculated from frame 28 was then compared to the allowable contact stress calculated from frame 27.

It was found to be satisfactory.

Discussion of results

The minimum number of teeth to avoid undercutting as calculated by Gearex is 17 using the equation 36.
The minimum number of teeth acceptable using the Birmingham University gear design program (BUGDP) is 16. To compare the results from Gearex with BUGDP, the calculation for the minimum teeth to avoid undercutting was bypassed so as to design spur gears with 16 teeth (the first three sample results tabulated in Table 7.16a). The column headed BUGDP# shows the Birmingham University gear design program output results. The "#" represents the sample number. For example, BUGDP1 is sample number one. The columns headed by GRX# shows the Gearex output results. Again, the "#" represents sample number. The first three samples in Table 7.16a show that the results from the two programs are similar with the following exceptions;

a) The addendum value for the pinion and wheel for BUGDP results are different. The Gearex results shows that the addendum values for the pinion and wheel (i.e., equation (7.2.9), ha = m) are the same in accordance to BS436:part2 [81]. The BUGDP calculations for addendum is based on addendum modification calculations [72],[80],[82] described earlier in section 7.2. For example, consider the result for example problem 1, it satisfies the condition stipulated (by [72],[80],[82]) for addendum modification

(section 7.2 : The derivation of geometric relationships). The conditions are that:

$z1 \geq 10$, $z2 \geq 10$ and $z1 + z2 \geq 60$.

For example 1, $z1 = 16$, $z2 = 64$. Therefore, the addendum modification factor, k will have a value of 0.3.

Hence, $ha1 = (1 + k)m = 13$ mm, $ha2 = (1 - k)m = 7$ mm. These are the values obtained from BUGDP results for the pinion and wheel addenda. BS436:part2 [81] stated that gear teeth are modified in practice. However, the standard did not expanciate on how or for what reasons gear teeth should be modified. For this reason, unmodified addendum was adopted for use in Gearex and was considered acceptable.

b) The root diameter values given by Gearex for the pinion are less than the values given by BUGDP. The Gearex values ranges from 97.7% to 98.8% of the BUGDP values. Root diameter dr, is calculated from the equation 7.2.43; dr = d - 2*hf and equation 7.2.7; d = m*z1. The dedendum hf, is calculated from the equation hf = 1.25m for Gearex and hf:pinion = m(1.25 - k) or hf:wheel = m(1.25 + k) for BUGDP. For the pinion, the value of hf calculated will be lower than the value obtained from Gearex. This in turn has the effect of increasing the value of the root diameter obtained from BUGDP. The exception to the higher root diameter values of BUGDP is example result 7 where the result is 94.9% lower. The reason for this is that BUGDP gave a minimum pinion number of teeth value as 16 with wheel number of teeth as 80. Gearex gave a minimum pinion teeth value of 17 with a wheel teeth value of 85. In all

the other examples, the number of teeth on the pinion and wheel for Gearex and BUGDP were the same. The relatively higher value for the pinion teeth (i.e. 17) as calculated by Gearex compared with the value 16 obtained from BUGDP in example 7, gave a calculated value for the pitch circle diameter (from equation 7.2.7) that was high enough to offset the higher values for root diameters obtained from BUGDP. The root diameter values given by BUGDP for the wheel is less than the values given by Gearex for the same reasons explained above. The value of the root diameter obtained from BUGDP ranged from 92.2% to 99.3% of the value obtained from Gearex.

c) The value of the total contact ratio is higher for the Gearex results than for BUGDP results. For the spur gear results, (that is, samples 1 to 10), BUGDP values ranges from 93% to 96% lower. This is due to the difference in the values of the outside diameters (caused by addendum modification) for both the pinion and the wheel used in the calculation of the contact ratio (equation 7.2.30). Although the value of the outside diameter for the pinion obtained from BUGDP is higher, the value of this parameter for the wheel is significantly lower, hence giving a lower value of contact ratio. For helical gear values, (that is, samples 11 to 18), the BUGDP values ranges from 26% to 65% lower. The values for contact ratio given by BUGDP is the transverse contact ratio. It does not consider the overlap ratio (equation 7.2.30a). For spur gears, the total contact ratio is the same as the transverse contact ratio. For

helical gears, the total contact ratio is the sum of the transverse contact ratio and the overlap ratio [20]. For this reason, the values of the contact ratio given by Gearex was considered acceptable. Also, factors such as the elasticity factor, application factor etc, taken into consideration when calculating the bending stress or/and contact stress may make a significant difference to the value of the facewidth calculated. The value of the facewidth will, in turn, affect the value of the overlap ratio in the case of helical gear calculations. It is difficult to assess which factors BUGDP took into account for the calculation of the face width because it was based on British Standard 436:1940 [90].

The gear designs generally produced by Gearex are conservative compared to BUGDP. Based on the facewidths, the larger the face width the more conservative the design. The reason for this difference stems from the reasons given in the last paragraph which was that the factors taken into account when calculating the face width is difficult to determine for BUGDP. However, another reason is that the criteria that determines a satisfactory face width are the bending/contact stress requirements and the geometric requirements as given by following equations,

$\pi m \leq b \leq 4\pi m$ for spur gears

$\pi m/\tan\beta \leq b \leq 6\pi m$ for helical gears.

Where b represents the facewidth. This criterion gives a relatively wide range for acceptable face width values.

Gearex often recommends the higher face width values within the range. This will decrease gear face wear. 10 out of the 18 listed Gearex sample results are conservative compared to BUGDP. The examples 5,6,7,11 and 16 (table 7.16c and 7.16d) design results were not considered satisfactory by Gearex. All 18 examples were considered satisfactory by BUGDP. Of the 5 non-satisfactory Gearex results, 3 were conservative compared to the results by BUGDP.

The example 5 (table 7.16c) design failed to satisfy the geometric criteria mentioned above because the facewidth value was lower than the minimum acceptable limit. It was 21.7 mm. The minimum acceptable facwidth is 25.13 mm. Gearex recommended that a material of lower bending stress capacity should be considered (for example, mild steel). The reason for this recommendation was that a lower facewidth indicates that the material allowable surface stress value is higher than required or/and the allowable bending stress value was also higher than required. That is, the material quality was too high. Hence, a material of lesser quality (also cheaper) should be considered. Mild steel was then tried and found to be unsatisfactory. The use of cast iron produced a satisfactory result. The facewidth was 54.21 mm. Alternatively, a the minimum facewidth can be used without having to change the material. That is, 25.13 mm could be used. This will produce a lower induced stress. For a facewidth value of 21.7 mm, the induced stress was the same value as the maximum allowable stress. The flaw in this approach is that

a material of higher quality than required was being recommended and hence if cost was to be considered in the future (perhaps as an extension to the knowledge base), the design will not be cost effective nor would it have been properly optimized. The design produced by BUGDP was considered unsatisfactory because of the relatively smaller facewidth which did not satisfy the geometric criteria.

The sample number 6 (table 7.16c) failed the geometric criteria for the same reason as sample 5. The facewidth was too low. The minimum recommended face width was 25.14 mm and the calculated face width was 16.82 mm. The recommendation by Gearex was to use a material with lower bending stress or use a face width of 25.14 mm if the user wishes to avoid changes to the values of other geometric parameters. Alternatively, a lower value for the module is recommended (which will change the values of the geometric parameters such as the pitch circle diameter). The use of phosphor bronze (material number 2, table 7.15) was found to be adequate. It produced a facewidth of 27.94 mm with minimum recommended facewidth being 25.14 mm and a maximum recommended value of 100.57 mm. If the module was reduced to 6 mm, using the same material (number 5) the result was found to be satisfactory also. It is tabulated in table 7.17, headed GRX6A. Again, the design produced by BUDGP had a face width value that did not satisfy the geometric criteria and hence is considered unsatisfactory.

Parameter	GRX6A	GRX11A
z1	17	16
z2	170	64
m (mm)	6	8
a (mm)	935	369.55
b (mm)	27.94	146.28
WG	P	P
d1 (mm)	102	147.82
d2 (mm)	1020	591.28
do1 (mm)	114	163.82
do2 (mm)	1032	607.28
ha1 (mm)	10	8
ha2 (mm)	10	8
dr1 (mm)	87	127.82
dr2 (mm)	1005	571.28
db1 (mm)	95.84	136.26
db2 (mm)	958.44	545.05
T (N m)	238.64	381.82
ε	1.71	4.26
v (m/s)	21.36	15.48
Ft (kN)	4.68	5.67
Fr (kN)	1.704	1.88
Fa (kN)	0	2.99
Conclusions	FWR	FWR

Table 7.17 Design results (iterated)

For sample 7, the wheel is the weaker gear. The calculated face width for the wheel was 27.77 mm and for the pinion was 14.68 mm. The minimum acceptable facewidth is 31.43 mm. The use of a poorer quality material for the both gears was recommended or the use of a lower module value. But, the wheel material is the weakest available material in the database. Therefore, it was recommended to use a facewidth of 31.43 mm and retain the wheel material. The material for the pinion can be changed to a weaker material such as cast iron or phosphor bronze. This made the pinion the weaker gear with reasonable face width of 38.38 mm. The design obtained from BUGDP had wider facewidth and hence more conservative than that produced by Gearex.

Sample 11 was not considered satisfactory because the face width calculated was higher than the maximum acceptable value of 113.14mm. The calculated value was 213.2 mm. The recommendation was that the module should be increased or a material with higher bending stress used. The use of a such material (that is, mild steel) produced a face width of 95.32 mm which was considered satisfactory. The use of a higher module (m = 8) also produced a satisfactory result which is tabulated in table 7.17.

The sample 16 was also considered unsatisfactory for similar reasons to samples 5 and 6. The calculated value of 39.01 mm was lower than the minimum recommended of 53.89 mm. Hence the use of 53.89 mm was recommended. This will produce a lower induced stress.

All the examples were satisfactory for contact stress requirements. For example, the induced contact stress in sample 16 for the pinion and wheel were 67.25 N/mm^2 and 33.62 N/mm^2 respectively. The maximum allowable contact stress was 172.97 N/mm^2.

CHAPTER EIGHT

RESEARCH ACHIEVEMENTS, CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER WORK.

CHAPTER EIGHT

RESEARCH ACHIEVEMENTS, CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER WORK.

8.1 RESEARCH ACHIEVEMENTS

Through the development of the necessary techniques and their embodiment in a software package, the overall achievement of the research was the verification of the fact that Expert Systems may be used for mechanical design and not simply building a suitable control into an existing formal logic program. The breaking down of any design problem into its primitive processes such as the manipulation of equations in an expert way is possible as demonstrated by the concepts developed in this thesis. Also, the use and the interpretation of relationships (such as equations or/and heuristic knowledge) between primitive variables such as gear module or pitch circle diameter as part of the knowledge manipulative process was demonstrated.

The type of knowledge representation technique best suited to class 3 mechanical design problems was adequately addressed by the modification, adaptation and merging together of the requirements of engineering design process with those of Expert Systems (Chapter 5) resulting in a four stage process, which is:

a) The information acquisition module

(Question and answer)

b) The inference engine (Control module)

c) The domain specific knowledge

i) test module

ii) calculation module

d) The explanation and self knowledge module (reasons and advice)

Within this framework, the concept of grouping equations together into levels to cater for the process of progressive substitution (Level based concept) was developed. The intercommunication between these levels allowed the author to merge formal logic concepts with Expert System semantic nets concepts. This approach was extended into the frame based method as discussed in Chapter 7. This latter method can be used for any type of design involving any number of equations. The only limitation is the computer memory capacity and speed.

The project was able to show the need for a purpose-built shell for this class of design task. Component designs are undertaken in most mechanical engineering design problems and to this end, this research has contributed to the use of Expert Systems in these areas and serves as a guide for the direction of future researches into this area. A system cannot be successful without the various components that constitute the system being successfully designed. The research was necessary and important in that it bridges the gap left in this area.

The two concepts developed (Level based concept and Frame based concept) allow for the addition of knowledge on a continuous basis. Unlike a conventional program, where the development of a system is completed with the achievement of the desired unique result to every problem given to the program, the system development is never completed. It is an ongoing process. However, new techniques introduced such as the use of status flags to keep track of how variable values or/and knowledge (table 7.3) were obtained. This helps in the explanation of the system status to the user especially when recommending the change to a variable value and why. Other techniques such as the use of "transmission" and "connection" values (see appendix A3) for each and every node enabled the system to be modelled in an efficient manner.

A drawback in the use of Expert Systems for design problems is that until sufficient knowledge is encoded, the system will not be useful. The process of encoding this knowledge is long and slow. The demonstration of the system capabilities is what could reasonably be expected but the full use of the system will require the encoding of temporary "unavailable" knowledge over a period of time. For example, assuming some factors to be unity for the Gear pair problems.

8.2 CONCLUSIONS

An Expert System knowledge representation method is usually suitable for a well-defined class or domain type of problem. For example, the representation techniques used in this research cannot be said to be suitable for all design problems but for design problems of a certain type. It is not suitable for use for example if the finite element technique is required. (Taig [91] and Kissil and Kamel [92] researched the use of Expert Systems for finite element applications. Taig [91] concluded that the system developed, called FEASA, was too cumbersome for general use but showed that the application of Expert Systems to finite element analysis pre- and post-processing is possible. Kissil and Kamel [92] concluded that the use of Expert Systems as a tool to aid structural analysts to construct finite element meshes rapidly and with reasonable accuracy is promising).

The work described in this thesis also confirms the "specialisation versus generality" argument in that a system that is an expert and has the required depth and speciality cannot attain the type of generality expected by current users. The system described in this thesis was developed and implemented was from a "speciality angle". That is, it was initially developed to design mechanical components such as helical springs and then gradually generalised to embody class 3 design problems. The Level based concept is best suited to component designs although it can be applied to system design as demonstrated in chapter 6. The Frame based

concept is best suited to component or mechanical system design (a system in this case refers to a collection of components inter-related in such a manner as to have a common collective function). They cannot be used as the sole implementation method for a design system containing several design classes and domains encompassing, for example, selection, prediction or diagnostic. The Level and Frame based concepts can however be used in conjuction with other design systems in any mode, whether in a hierachical structure or network structure.

The results obtained from the two case studies conducted demonstrated that these concepts can be applied to a practical situation. For example, in the first case study using the Level based concept on spring design produced results that compared favourably with hand calculations. Also, the results obtained in the second case study using the Frame based concept on gear design also produced very good results when compared with a dedicated gear design program developed by Birmingham University.

The use of a rudimentary shell enabled the author to develop the methods presented in this thesis with greater flexibility than the "fully fledged" Expert Systems shell available at the time. This would not have been possible with the use of any of the "fully fledged" shells available at the time because of the limitations embodied in them. The chosen shell has been used to the limits of its capability.

The thesis has positively demonstrated the potential of semantic nets for application to engineering design problems and they are likely to gain wider acceptance if shell programs are specially developed for this purpose. This concept is still at its infancy in that the traditional methods of equation and formula manipulation using conventional methods is entrenched in various computer applications. The understanding of this usage is lacking in most cases. In this area the use of Expert Systems as demonstrated in the research, will bring in this understanding and with time a wider acceptance of the method. Also, this study has shown that Semantic Nets provide a very natural formalism for representing knowledge in the domain of engineering design.

Visual display aids (Graphics) to assist in the explanation of the reasoning process and to describe components and show their dimensions are extremely desirable and almost essential. The need for such facilities was explained in Chapter 3. It is obvious that the clarity that will be obtained in terms of understanding the reasoning process of the system will be enormous. Brady, M [93], Descotte and Latombe [94] and Lim [95], have made progress in this direction by integrating a 3D graphics system with an Intelligent Knowledge Based System. Other Artificial Intelligence scientists have revealed the importance of combining the desirable attributes of Expert Systems technology within those of Computer Aided Design. To this end the use of semantic nets knowledge representation

formalism will be of benefit in the sense that nodes (which can represent knowledge grouping, equations or variables) can be displayed graphically with their interconnections (e.g., Fig 7.15). This will make Expert Systems applied to mechanical engineering design more transparent than it would otherwise have been if semantic nets formalism was not adopted. Again, the inherent virtues of semantics nets can be exploited.

The continued evalution of the tools and hardware necessary for the implementation of Expert Systems into a new domain of interest as was conducted in this research is important. Presentation and improved screen control capability is possible by using the right combination of hardware and software. For a fully working system of this nature, a powerful workstation (such as Apollo Sun or Silicon graphics) with various other user environments is necessary if an efficient and effective system is to be produced. The author is of the opinion that by conducting this research, this point will be well appreciated by future researchers and guide against the possibility of using hardware that provides less options than may be required during future developments of the system.

Expert System developments need to be conducted in an environment where experts are available. One of the major problems in this type of research is that the knowledge engineer also plays the role of the expert. This is very difficult because it is widely accepted that an expert may

have in depth knowledge of his domain of interest but may not be able to give detailed explanations for his expertise in a form that can be encoded into a computer. Having an expert to continue to assess the system development and give critical appraisal of the system is necessary.

With the wide use of computers for engineering purposes and the general increase in the use of Expert Systems techniques in this domain, the division of labour between man and machine will become more distinct than already exists.

8.3 Recommendations for further work.

From a very narrow perspective, improvement to the package can be achieved. Microsynics was converted from Basic programming language environment to a lower level language like C. The package, which consists of two section (the compiler and the run-time module), could be used as a platform for the development of a purpose built shell. This has already been done to some extent in this project but more work is required. Although the Microsynics operating environment was changed from MSBasic to GWBasic (having its own compiler) for the purpose of achieving faster execution time, it was only possible to compile the Microsynics language "compiler" but not the Run-time module. This is because the Run-time module has to chain-merge other programs to save on memory; this module will have to be written in a language that can be compiled. Ideally, it

should be written in assembler. This will achieve the fastest processing time and hence aid in the quick development of an expert design system.

The knowledge representation system itself must be developed further to allow the compilation of the source code with the ability to link on to it any further addition of the source code. Currently, any change to the source code (Nodal parts) will require the complete compilation of the entire program before execution. This slows down development and the removal of this disadvantage will enhance the speedy experimentation of various methods and concepts as well as development strategies.

The use of graphics to illustrate consequences of decisions is necessary. This should be adequately researched for this type of design system. Using plotted graphs or diagramatical illustrations in various ways will contribute greatly to this area of Expert System development.

The use of semantic nets knowledge representation formalism as a basis for the development of a shell program with the inherited virtues of this scheme should be conducted incorporating all the attributes required of an Expert System mentioned in chapter 3.

Viewed from a broader perspective, the frame based concept should be adopted in other areas of Expert Systems applications such as Industrial or financial management

decision making tasks. Constraints (such as manpower shortages, cashflow problems, departmental policies, etc) can be represented as nodes which can be critically examined and modified as industrial or financial situations changes. The links between these constraints are explicit and can therefore be understood, and the problems and implications of decision taken, appreciated. A typical company consists of departments (such as works, personnel, engineering, maintenance). The implications of inter- and intra-departmental policy decisions can be represented by nodal links and the nodes representing the various departments. Since each node can be represented by a frame if there is sufficient interest in its internal "workings", the sectional policies within a department can be examined if required.

Other areas of application could be to assist in military battlefield decisions. Various military brigades, platoons etc., can be represented as nodes. The interactions, communications and assistance required by one military section to another can be explicitly defined by the nodal links and any complications that might arise will be easily noticed.

The construction industry can utilise the versatility of these concepts to aid in the organisational arrangements between various functions on a site. It can assist in the determination of possible difficulties that might arise between various contractors on a construction site. Various

consultants, contractors and workers on a building site have different needs at different times. This can be problematic. The concepts described in this thesis can be very useful to such industries.

This simple but very powerful knowledge representation concept can be extended to most Expert Systems application areas within the engineering discipline. The concepts discussed in this thesis will be a useful aid for decision making tasks in most areas of interest. Hence, although the FBC was developed for mechanical engineering design applications, the basic concept and versatility of this scheme allows its use in wider areas of applications.

REFERENCES

REFERENCES

1. **FINKELSTEIN, L. AND FINKELSTEIN, A.C.W.**
   "Review of design methodology"
   IEE proc., 1983, Vol.130, Part A, pp.213-222.

2. **SHIGLEY,J.E.**
   Mechanical Engineering Design, First metric edition,
   McGraw-Hill, 1986.

3. **PAHL,G. AND BEITZ,W.**
   Engineering Design,
   The Design Council, 1984.

4. **HALL, HOLOWENKO AND LAUGHLIN**
   Machine design,
   Schaum series, (SI) Metric edition,
   McGraw-Hill, 1980.

5. **BEAKLEY, G.C. and LEACH, H.W.**
   Engineering: An Introduction to a Creative Profession.
   Collier-Macmillan ltd, London, 1967.

6. **RZEVSKI, G.**

"On the design of a design methodology"

Design,science,method: proc. of the 1980 Design Research Society.

Eds., Jacques, R and Powell, J.

Guildford, 1981.

7. **DUDERSTADT, J.J., KNOLL, G.F. and SPRINGER,G.S.**

Principles of engineering.

John Wiley and sons Inc., New York, 1982.

8. **EDER, W.E.**

"Definition and methodologies"

The Design Method, Butterworths, London, 1966,

Eds Gregory, S.A.

9. **CROSS, N., NAUGHTON, Jand WALKER, D.**

"Design method and scientific method"

Design,science,method: proc. of the 1980 Design Research Society.

Eds., Jacques, R and Powell, J.

Guildford, 1981.

10. **GASPARSKI, W.**

"The humanistic view of design"

Design,science,method: proc. of the 1980 Design Research Society.

Eds., Jacques, R and Powell, J.

Guildford, 1981.

11. **DILNOT, C.**

"Transcending science and anti-science in the philosophy of design methods"

Design,science,method: proc. of the 1980 Design Research Society.

Eds., Jacques, R and Powell, J.

Guildford, 1981.

12. **WOODSON, T.T.**

Introduction to engineering design.

McGraw-Hill Inc., 1966.

13. **SIMON, H.A.**

A student introduction to engineering design.

Pergamon Press Inc., 1975.

14. **ASIMOW, M.**

Introduction to design.

Prentice-Hall Inc., Englewood Cliffs, N.J., 1962.

15. **FOX, R.L.**

Optimization methods for engineering design,

Addison-Wesley publishing company, 1971.

16. **FRENCH, M.J.**

Conceptual design for engineers.

The Design Council,

Springer-Verlag, 1985.

17. **FRENCH, M.J.**
Engineering design: The conceptual stage.,
Heinemann education books, 1971.

18. **HUBKA, V.**
Principles of engineering design.,
Butterworth Scientific, 1980.

19. **BRITISH STANDARDS INSTITUTION.**
British Standard specification for patented cold drawn carbon steel wire for mechanical springs.
BS5216:1975.

20. **BRITISH STANDARDS INSTITUTION.**
Spur and helical gears.
Part 3. Method for calculation of contact and root bending stress limitations for metallic involute gears.
BS436:1986.

21. **SHELDON, D.F.**
"The present state of computer-aided Engineering industry and education".
In CADCAM: Training and education through the 80's
Eds Arthur, P., Proc. of the CAD ED '84 Conf.
Kogan Page Ltd, 1985, pp. 9-36.

22. **BARR, P.C., KRIMPER, R.L., LAZEAR, M.R., and STAMMEN, C.**
CAD:Principles and applications.
Prentice-Hall, 1985.

23. **REED, J.**
"CADCAM technology - five years ahead"
In CADCAM: Training and education through the 80's
Eds Arthur, P. Proc. of the CAD ED '84 Conf.
Kogan Page Ltd, 1985, pp. 37-41.

24. **SABIN, M.A.**
"CAE - A longer view"
In CADCAM: Training and education through the 80's
Eds Arthur, P. Proc. of the CAD ED '84 Conf.
Kogan Page Ltd, 1985, pp. 42-50.

25. **SMITH, D.G.**
"CAD in the context of engineering business - a necessary education perspective?".
In CADCAM: Training and education through the 80's
Eds Arthur, P. Proc. of the CAD ED '84 Conf.
Kogan Page Ltd, 1985, pp. 115-125.

26. **NOWACKI, H.**

"Modelling of design decisions for CAD"
Lecture notes in computer science (89).,
Eds Goos, G and Hartmanis, J.,
Computer-aided design modelling, systems engineering,
CAD systems.,
Crest advanced course.,
Darmstadt, 8-12 sept., 1980.,
Eds Encarnacao., Spring-verlag, Berlin, 1980, pp.
137-172.

27. **WARMAN, E.A.**

"Computer-aided design: an intersection of ideas"
Artificial intelligence and pattern recognition in
computer-aided design., North-Holland, 1978.,
Eds Latombe, J.C.

28. **SIMMONS, M.K.**

"Artificial intelligence for engineering design"
Computer-aided engineering journal,
April 1984, pp75-83.

29. **BRANCHMAN,R.J. ET AL**

"What are expert systems ?"
Building Expert systems,
Eds. Hayes-Roth, F., Waterman, D. and Lenat, D.
Vol.1, Addison-Wesley, 1983.

30. **WATERMAN,D.**

"How do Expert systems differ from conventional programs"

Expert Systems

Jan., 1986, Vol 3, No1 1

31. **BARR,A. AND FEIGENBAUM,E.A.**

The Handbook of Artificial Intelligence

Vol.1, Pitman, 1982.

32. **BEEREL, A.C.**

Expert Systems: Strategic implications and applications.

Ellis Harwood Ltd, 1987.

33. **NAYLOR, C.**

Building your own Expert systems.

Wilmslow Sigma Technical, 1983.

34. **WATERMAN, D.A. AND HAYES-ROTH, F.**

An investigation of tools for building expert systems.

Rand, 1980.

35. **KONOPASEK;M. AND JAYARAMAN;S.**

"Expert systems for personnal computers:

The TK! Solver approach"

Byte, May 1984 pp.137-150.

36. **UZEL, A.R. and BUTTON, B.L.**
"Guidelines for Expert Systems applications"
Chartered Mechanical Engineer, Feb., 1987, pp40-45.

37. **LEVESQUE, H.J.**
The logic of incomplete knowledge
On conceptual modelling
Eds. Brodie, M.L., Mylopoulos, J. and Schmidt, J.W.
Springer-Verlag, 1984.

38. **RYCHENER, M.D.**
"Expert Systems for Engineering design"
Expert Systems, Jan., 1985, Vol.2., No.1., pp30-44.

39. **MYLOPOULOS, J. AND LEVESQUE, H.J.**
An overview of Knowledge Representation
On conceptual modelling
Eds. Brodie, M.L., Mylopoulos, J. and Schmidt, J.W.
Springer-Verlag, 1984.

40. **QUILLIAN, M.R.**
"Semantic memory"
Semantic information processing
Eds Minsky, M.
MIT Press, 1968, pp. 227-270.

41. **MINSKY, M.**

"A framework for representing knowledge"

The psychology of computer vision

Eds Winston, P.

McGraw-Hill, New York, 1975., pp. 221-277.

42. **WOODS, W.A.**

"What is a link: Foundation for semantic networks"

Representation and Understanding

Eds Bobrow, D and Collins, A

Acedemic Press, New-york, 1975, pp. 35-82.

43. **ISREAL, D.J. AND BACHMAN, R.J.**

Some remarks on the semantics of Representation language.

On Conceptual Modelling

Eds. Brodie, M.L., Mylopoulos, J. and Schmidt, J.W.

Springer-Verlag, 1984.

44. **WINOGRAD, T.**

"Frame representation and the procedural/declarative controversy"

Representation and Understanding: Studies in cognitive science.

Eds Bobrow, D and Collins, A.

Acedemic Press, Inc., New York, 1975.

pp 185-210.

45. **KUIPERS, B.J.**

"A frame for frames: Representing knowledge for recognition"

Representation and Understanding: Studies in cognitive science.

Eds Bobrow, D and Collins, A.

Acedemic Press, Inc., New York, 1975.

pp 151-184.

46. **ALLWOOD,R.J; STEWART,D.J; HINDE,C AND NEGUS,B**

Evaluation of Expert systems shells for the construction industry application.

Department of Civil Engineering,

Loughborough University of Technology, 1985.

47. **WILLIAMSON-TAYLOR, H.O.**

"Application of Expert systems methodologies to engineering designs",

First annual report,

Department of Mechanical engineering

The University of Aston in Birmingham, Oct 1986.

48. **EXPERT SYSTEMS INTERNATIONAL LTD.**

ES/P Advisor: User guide and reference manual,

Issue 1, June 1984.

49. **INTELLIGENT TERMINALS LTD.**

Expert Ease user manual, 1984.

50. **COX, P.R., and BROUGHTON, R.K.**
Micro Expert user manual, Version 4.0.

51. **ALVEY IKBS.**
Microsynics reference manual, 1985.

52. **HART, A.**
"The role of induction in knowledge elicitation"
Expert System, Jan., 1985, Vol2, No1,pp24-28.

53. **SPL INTERNATIONAL**
Sage user manual (for sage version 1.3),
Knowledge Engineering group SAG03.,
Systems Programing Ltd., 1983.

54. **SPL INTERNATIONAL**
Sage Expert system language specification (for sage version 1.3),
Knowledge Engineering group SAG02.,
Systems Programing ltd., 1983.

55. **RICH, E.**
Artificial intelligence,
McGraw-Hill, 1983.

56. **BERK, A.A.**
LISP: The language of artificial intelligence.,
Collins, 1985.

57. **BERK, A.K.**
Micro-prolog and artificial intelligence.,
Collins, 1985.

58. **CLOCKSIN, W.F. and MELLISH.**
Programming in Prolog.,
second edition, Springer-Verlag, Berlin, 1984.

59. **DIGITAL**
The artificial intelligence education series,
VAX OPS5: student guide,
Education series, 1985.

60. **McDERMOTT, J.**
"R1: A rule based configurer of computer systems"
Artificial Intelligence, Vol. 19, No. 1, 1982,
pp.39-88.

61. **BROWNSTON, L., FARREL, R., KANT, E., MARTIN, N.**
Programming expert systems in OPS5.,
An introduction to rule-based programming.
Addison-Wesley publishing company, 1985.

62. **BROWN, D.C. AND CHANDRASEKARAN, B.**
" An approach to Expert system for mechanical design"
IEEE 1983, PP. 173-180.

63. **DIXON, J.R. AND SIMMONS, M.K.**
"Expert systems for Engineering design: Standard V-Belt drive design as an example of the design-evaluate-redesign architechture".
Proceeding of the 1984 ASME Computers in Engineering Conference, Las Vegas, NV, August 12-15, 1984. pp.332-337.

64. **KULKANI, V.M., DIXON, J.R., SUNDERLAND, J.E. AND SIMMONS, M.K.**
"Expert systems for design: The design of heat fins as an example of conflicting subgoals and the use of dependencies"
Proceeding of ASME Computers in Engineering Conference, Boston, MA., August 4-8, 1985, pp.145-150.

65. **HOWE, A. E., COHEN, P.R., DIXON, J.R. AND SIMMONS, M.K.**
"DOMINIC: A domain independent program for mechanical engineering design."
Artificial Intelligence, 1986, Vol. 1, No 1. pp. 23-28.

66. **KORANE, K.J.**
"Applying Expert Systems to Mechanical Design."
Machine design, Dec., Vol. 11., 1986, pp121-124.

67. **DIMITROV, I.I.**

"Knowledge representation for mechanical systems design"

In knowldege based Expert Systems in engineering: planning and design.

Eds. Sriram, D. and Adey, R.A., Computational Mechanics Publications, 1987, pp367-376.

68. **BRITISH STANDARD INSTITUTION**

British standard guide to design and specification of coil springs (Helical compression springs).,

BS 1726 PART 1 : 1964

69. **KASSATLY, A and BROWN, D.C.**

"Explanation for routing design problem-solving"

In knowledge based Expert Systems in engineering: planning and design., Eds Sriram, D. and Adey, R.A.

Computational mechanics publications, 1987, pp225-239.

70. **GUILFOYLE, C.**

"Expert explanation"

Expert System User

April 1986, pp.25-27.

71. **SPRING RESEARCH ASSOCIATION**

The Spring Research Association manual: design data sheet No1.

72. **QUAYLE, J.P.**
Kempe's Engineering Year Book,
92nd edition,
Morgan-Grampian book pub. co. Ltd., 1987.

73. **STEPHENS, R.C.**
Strength of Materials: Theory and Examples
Edward arnold. 1978.

74. **BURR,A.H.**
Mechanical analysis and design,
Elsevier, 1981.

75. **PENSULO, E.**
Integrating computer-aided engineering functions: The management of information.
PhD Thesis, The Department of Mechanical Engineering, The University of Aston in Birmingham. 1987.

76. **HOUGHTON, P.S.**
GEARS: Spurs, Helical, Bevel, Internal, Epicyclic and Worm., Technical Press, London, 1970.

77. **BRITISH STANDARDS INSTITUTE**
Glossary for Gears: Part 1. Geometric definitions
[ISO title: Glossary of gears - geometrical definitions].
BS2519:Part 1:1976 ISO/R 1122-1969.

78. **BRITISH STANDARD INSTITUTE**
Glossary for Gears: Part 2. Notation
[ISO title: International gear notation - symbols for geometrical data].
BS2519:Part 2:1976 ISO 701-1976.

79. **BUCKINGHAM, E.**
Spur gears: Design, operation, and production.
McGraw-Hill, 1st ed., New York, 1928.

80. **AVOLLONE, E.A. and BAUMEISTER III, T.**
Mark's Standard Handbook for Mechanical Engineer,
9th edition,
McGraw-Hill, 1987.

81. **BRITISH STANDARDS INSTITUTE**
Specification for spur and helical gears,
Part 2.Basic rack form, modules and accuracy,
(1 to 50 metric module).
BS436:Part2:1970; metric units,

82. **OBERG, E., JONES, F.D. and HORTON, H.L.**
Machinery's handbook, 22nd edition, Industrial press Inc., New York, 1984.

83. **STOKES, A.**
High Performance Gear Design, The Machinery Publishing Company., Ltd. London, 1970.

84. **MUNDAY, A.J. and FARRAR, R.A.**
An Engineering data book.
Macmillan Press Ltd, London, 1979.

85. **BENHAM, P.P. AND WARNOCK, F.V.**
Mechanics of Solids and Structures
Pitmans Publishing Ltd., London, 1979.

86. **STEPHENS, R.C.**
Strength of Materials: theory and examples.
Edward Arnold, 1970.

87. **MERRIT, H.E.**
Gears, 3rd ed., Pitman, London, 1962.

88. **BUCKINGHAM, E.**
Manual of gear design :Spur and internal gears
Section two,
Industrial Press, New York, 1935.

89. **BUCKINGHAM, E.**
Manual of gear design: helical and spiral gears,
Section three,
Industrial press, New-york, 1964.

90. **BRITISH STANDARDS INSTITUTION.**
Spur and helical gears.
Method for calculation of contact and root bending stress limitations for involute gears. BS436:1940.

91. **TAIG, I.C.**

"Expert aids to finite element systems application"
In Applications of Artificial intelligence in engineering problems, 1st Int. conf., Southampton, U.K., April 1986, Vol 2., pp 759-770.,
Eds Sriram, D and Adey, R., Springer-Verlag.

92. **KISSIL, A. and KAMEL, H.A.**

"An Expert System finite element modeller"
In Applications of Artificial intelligence in engineering problems, 1st Int. conf., Southampton, U.K., April 1986, Vol 2., pp 1179-1186.,
Eds Sriram, D and Adey, R., Springer-Verlag.

93. **BRADY, M.**

"AI in Robotics"
AI, 1985, Vol.26, pp.79-121.

94. **DESCOTTE, Y. AND LATOMBE, J.**

"Making compromises amongst antagonistic constraints in a planner"
AI, Nov., 1985, Vol. 27, pp. 183-217.

95. **LIM, B.S.**

"An IKBS for integrating component design to tool engineering"
Expert Systems: The International Journal for Knowledge Engineering., Vol. 4., 1987.

APPENDICES

APPENDIX A1

Design Classifications

Class 1 Designs: This design task is rarely undertaken. This type of design activity applies to innovative design and it involves access to a wide-ranging knowledge structures, and searches in a very large space of design alternatives.

Class 2 Designs: This design task is closer to routine (or well established design activity) but many of the establised patterns may be broken. Some aspects of the design may require substantial innovation. The design task is undertaken in the expectation that the invested time and effort will pay off by identifying a potentially large market. Then the new elements of design can be "routinised".

Class 3 Designs: This is the most routine of the design tasks. It follows well established design alternatives which are reasonably understood, but nevertheless still require a skilled human expert to perform the design task. The function and solution principles remain unchanged.

APPENDIX A2

The major parts of an Expert System

Expert Systems consist of two fundamental parts.

a) The Knowledge base

b) The Inference engine.

THE KNOWLEDGE BASE:

The knowledge base consists of the knowledge about the domain of interest elicited from the human expert. The knowledge is usually stored in the form of rules (e.g. production rules). These rules are then used by the inference engine to conduct a search for a solution to the problem at hand. The knowledge base of most Expert Systems is usually large.

THE INFERENCE ENGINE:

This part of the Expert System decides how a search for a possible solution to a problem should be conducted and what rules should be used.

There are three basic methods of making use of available information to arrive to an acceptable solution to a search problem. They are :

a) The forward reasoning method

b) The backward reasoning method

c) Means End Analysis.

Forward Reasoning method:

This method (also called the DATA DRIVEN method) is best used in problems where the number of combinations of goal situations is either extremely large or is unknown and the information available is comparatively small. This method allows the system to proceed, using the current information available to it (prompted with more if required),to the goal situation. This method is suitable for most engineering design applications.

Backward Reasoning method:

This method (also known as the GOAL DRIVEN method) is the opposite of forward reasoning method. The number of goal situations should be relatively small and the data available to it extremely large. It starts with a goal situation and looks for factors (or information) that will confirm the goal hypothesis. If the information denies the goal hypothesis, then the next goal situation is considered and the process is repeated. The method has proved successful when implemented on diagnostic problems (for example, MYCIN).

Means End Analysis:

This method applies both forward and backward reasoning to arrive at a goal. The strategy allows the major part of the problem to be solved using one method and solving the minor problems that arise in combining the major problems

together in the other method. The method centres around detection of the differences between the current state and the goal state. Means end analysis can be applied to most engineering problems.

APPENDIX A3

THE OUTLINE OF KNOWLEDGE GROUPING

The factors to consider when grouping knowledge in the form of equations using the Level concept are;

A3.1 The number of variables in an equation.
It is considered favourable if the number of variables in an equation is low. Equations with the lower numbers of variables are most likely to be grouped together. The reason is that, the number of variable values that will be elicited by the system from the user will be lower and the chances of using the progressive substitution method for other variable values is higher.

A3.2 The number of connections to an equation node.
Consider Fig A3.1, the number of connections from node G1 is six. The equation nodes with high connection are likely to be placed at higher levels. These equations tend to have a high number of variables. However, equations with high (connection)/(variable number) ratio should be placed in

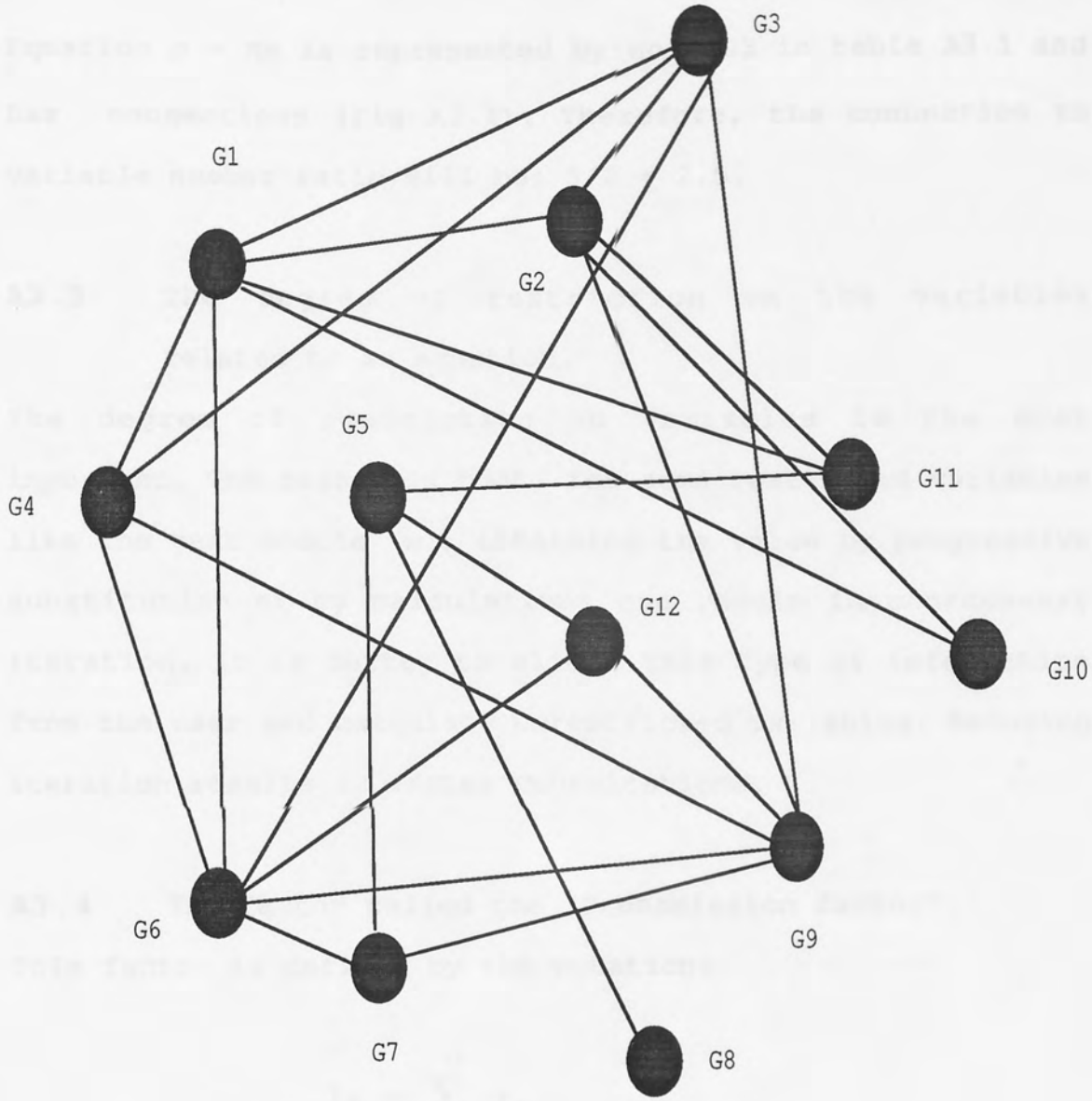


Fig A3.1 Equation nodes and connections for gear pair design.

lower levels. For example p = πm has a ratio of 2.5.

Equation p = πm is represented by node G3 in table A3.1 and has connections (Fig A3.1). Therefore, the connection to variable number ratio will be: 5/2 = 2.5.

A3.3 The degree of restriction on the variables related to an equation.

The degree of restriction on variables is the most important. The reason is that, for some restricted variables like the gear module (m), obtaining the value by progressive substitution or by calculations can result in unnecessary iteration. It is better to elicit this type of information from the user and calculate unrestricted variables. Reducing iteration results in faster consultations.

A3.4 The factor called the "Transmission factor".

This factor is defined by the equation:

$$T_k = \sum_{i=1}^{n} t_i$$

Where i is the connection number

t_i is the number of parameters in connection i

n is the number of connections for node k

k is the node number and

T_k is the total transmission for node k

NODE	EQUATION
G1	$p = \pi d/z$
G2	$m = d/z$
G3	$p = \pi.m$
G4	$b = K.p$
G5	$\sigma_{FP} = \sigma_{Fo}(x/(x+v^r))$
G6	$Ft = \sigma f.b.p.Yf$
G7	$P = Ft.v$
G8	$\sigma_{Fo} = UTS/3$
G9	$Ft = \sigma f.K.Yf.m.\pi^2$
G10	$u = z2/z1$
G11	$v = d1*n1/19098$
G12	$\sigma f \leq \sigma_{FP}$

Table A3.1 Equation nodes and corresponding equations (gear pair design)

This factor is the summation of the number of variables making up all the connections for a node.

For example, equation node G6 (Fig A3.1) has six connections. Therefore n = 6 and k = 6. The connections are G1-G6, G3-G6, G4-G6, G6-G7, G6-G9 and G6-G12 (see Table A3.2). The parameter making connection G1-G6 is p. (i = 1). Hence, the transmission factor for that connection is 1. The transmission factor t_i for the other connections (i = 2,3,4,5 and 6) are 1, 2, 1, 3 and 1 respectively. The summation of these figures is 9. Therefore, the transmission factor T_k, for node G6 is 9. Hence, $T_6 = 9$. Equation node G9 also has six connections and eight transmissions. Therefore, k = 9, n = 6 and $T_9 = 8$.

Fig A3.2 gives a network of equation nodes and connections for spring design. The Tables A3.2 to A3.3 gives the number of transmissions for each connection and the equation nodes involved. The Tables A3.1 to A3.4 gives the equations to each Equation node. Table A3.5 lists the spring equations and their corresponding equation nodes.

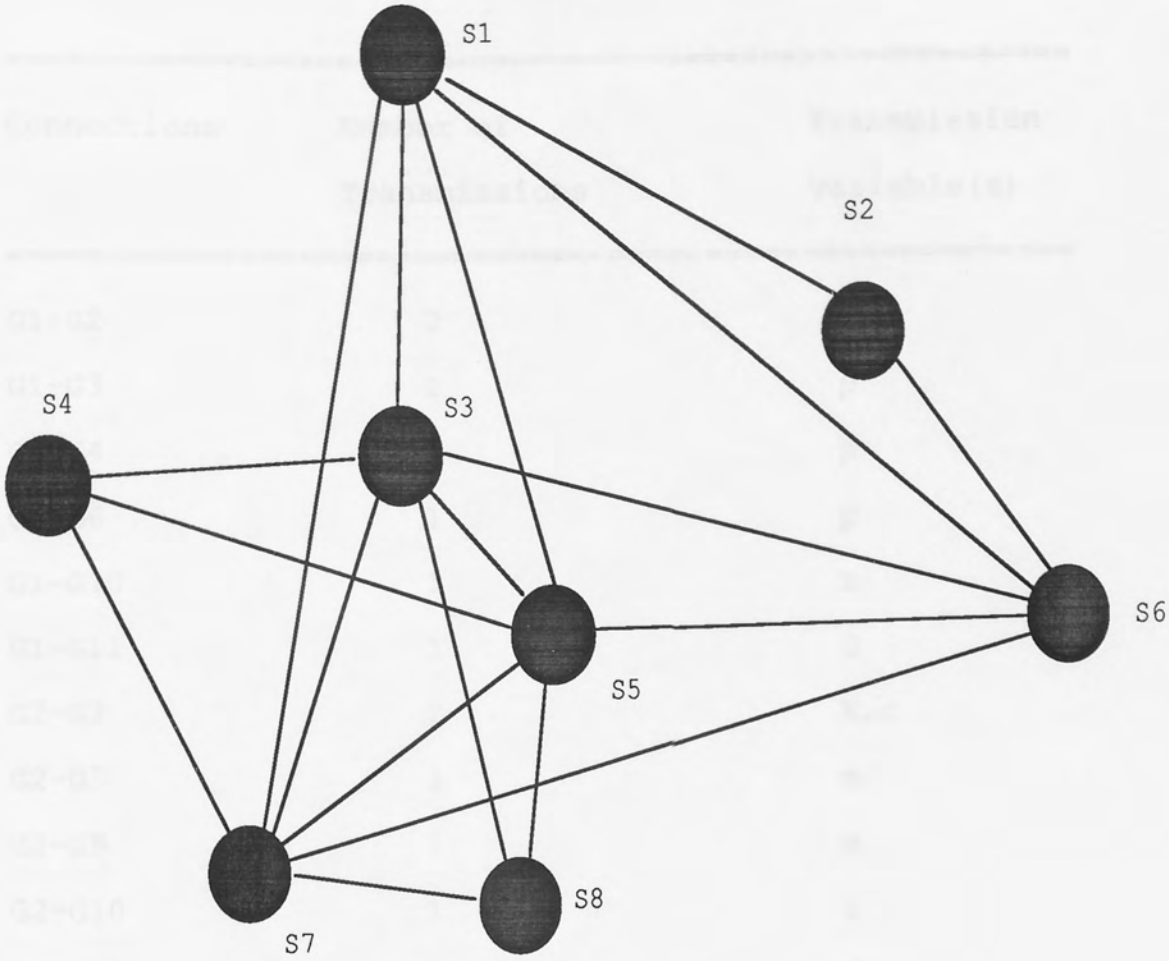


Fig A3.2 Equation nodes and connections for spring design.

Connections	Number of Transmissions	Transmission variable(s)
G1-G2	2	z,d
G1-G3	1	p
G1-G4	1	p
G1-G6	1	p
G1-G10	1	z
G1-G11	1	d
G2-G3	2	K,c
G2-G3	1	m
G2-G9	1	m
G2-G10	1	z
G2-G11	1	d
G3-G4	1	p
G3-G6	1	p
G3-G9	1	m
G4-G6	2	p,b
G4-G9	1	K
G5-G7	1	v

Table A3.2 Connections, transmissions and variables for Fig A3.1 (gear pair design).

Connections	Number of Transmissions	Transmission variable(s)
G5-G8	1	σ_{FO}
G5-G11	1	v
G5-G12	1	σ_{FP}
G6-G7	1	Ft
G6-G9	3	Ft, σf, Yf
G6-G12	1	σf
G7-G9	1	Ft
G9-G12	1	σf

Table A3.3 Connections, transmissions and variables for Fig A3.1 (gear pair design).

Connections	Number of Transmissions	Transmission variable(s)
S1-S2	1	c
S1-S3	1	d
S1-S5	2	d,D
S1-S6	2	c,d
S1-S7	1	d
S2-S6	2	K,c
S3-S4	1	n
S3-S5	1	n
S3-S6	1	d
S3-S7	3	EH,n,d
S4-S5	1	n
S4-S7	1	n
S5-S6	2	F,d
S5-S7	2	d,n
S5-S8	1	δ
S6-S7	1	d
S7-S8	1	Fl

Table A3.4 Connections, transmissions and variables for Fig A3.2 (spring design).

NODE	EQUATION
S1	$c = D/d$
S2	$K = (4c-1)/(4c-4)+(0.615/c)$
S3	$Lc = (n-EH1)d$
S4	$N = n+EH$
S5	$\delta = (8.F.D^3.n)/(d^4.G)$
S6	$P = (Q.\pi.d^2)/(8.c.K)$
S7	$Fl = 0.41n+EH.d$
S8	$\delta = 0.85(Fl-Lc)$

Table A3.5 Equation nodes and corresponding equations (spring design)

APPENDIX A4

INFORMATION FRAMES ON PROCEDURE TO BE FOLLOWED

(SAMPLE1; LEVEL 4)

INFORMATION

NODE ANC4 BANCQ4

---------- ANALYSIS C ----------

The variables involved in this stage of the analysis are all known.

They are : The free length
The maximum load
The maximum deflection
The spring rate

(Do you wish to see the procedure to be followed)

(Answer Y/N)

Fig A4.1 Example 1 (level 1) Rule violation Information screen display.

INFORMATION

NODE ANC4Q BANCQ4A

The following procedure will be used in the analysis

a) The value of the free length will be checked against the calculated maximum value of the freelength allowable.

ONLY If the spring end is closed and ground.

(Other options N,H,X or Z)

Press any key to continue

Fig A4.2 Example 1: Information screen display on procedure to be followed (first scroll).

INFORMATION

NODE ANC4Q BANCQ4A

BUT IF The value of the entered free length exceed the calculated maximum permissible free length

THEN The user will be asked to re-enter the value of the free length

(Other options N,H,X or Z)

Press any key to continue

Fig A4.3 Example 1: Information screen display on procedure to be followed (second scroll).

INFORMATION

NODE ANC4Q BANCQ4A

b) The value of the entered spring rate will be checked against the value of the calculated spring rate from the known value of the maximum load and maximum deflection.

IF The given value of the spring rate differs from the calculated value of spring rate

THEN The value of the given spring rate is retained.

(Other options N,H,X or Z)

Press any key to continue

Fig A4.4 Example 1: Information screen display on procedure to be followed (third scroll).

INFORMATION

NODE ANC4Q BANCQ4A

AND EITHER The value of the given maximum load is retained

PROVIDED This value does not exceed the value of the load at maximum stress calculated from Rule 6.

OR The value of the given maximum deflection is retained.

(Other options N,H,X or Z)

Press any key to continue

Fig A4.5 Example 1: Information screen display on procedure to be followed (fourth scroll).

INFORMATION

NODE ANC4Q BANCQ4A

PROVIDED That this value does not exceeed the value of the deflection at maximum stress or the value of the maximum physical deflection.

OTHERWISE The value of the given spring rate is retained but the load at maximum stress is considered as the maximum load.

(Other options N,H,X or Z)

Press any key to continue

Fig A4.6 Example 1: Information display on procedure to be followed (fifth scroll).