KNOWLEDGE-BASED
AUTOMATIC TOLERANCE ANALYSIS SYSTEM

by

Shamsuddin Baharin
B.Sc.(Distinction), M.A., CMfgT

Submitted to the University of Warwick
in accordance with the requirements for the degree of
Doctor of Philosophy

Rover Advanced Technology Centre
Department of Engineering, University of Warwick
Coventry CV4 7AL

August 1994

SUMMARY

Tolerance measure is an important part of engineering, however, to date the system of applying this important technology has been left to the assessment of the engineer using appropriate guidelines. This work offers a major departure from the trial and error or random number generation techniques that have been used previously by using a knowledge-based system to ensure the intelligent optimisation within the manufacturing system. A system to optimise manufacturing tolerance allocation to a part known as Knowledge-based Automatic Tolerance Analysis (KATA) has been developed. KATA is a knowledge-based system shell built within AutoCAD. It has the ability for geometry creation in CAD and the capability to optimise the tolerance heuristically as an expert system. Besides the worst-case tolerancing equation to optimise the tolerance allocation, KATA's algorithm is supported by actual production information such as machine capability, types of cutting tools, materials, process capabilities etc. KATA's prototype is currently able to analyse a cylindrical shape workpiece and a simple prismatic part. Analyses of tolerance include dimensional tolerance and geometrical tolerance. KATA is also able to do angular cuts such as tapers and chamfers. The investigation has also led to the significant development of the single tolerance reference technique. This method departs from the common practice of multiple tolerance referencing technique to optimise tolerance allocation. Utilisation of this new technique has eradicated the error of tolerance stackup. Three tests have been undertaken, two of which are cylindrical parts meant to test dimensional tolerance and an angular cut. The third is a simple prismatic part to experiment with the geometrical tolerance analysis.

The ability to optimise tolerance allocation is based on real production data and not imaginary or random number generation and has improved the accuracy of the expected result after manufacturing. Any failure caused by machining parameters is cautioned at an early stage before an actual production run has commenced. Thus, the manufacturer is assured that the product manufactured will be within the required tolerance limits. Being the central database for all production capability information enables KATA to opt for several approaches and techniques of processing. Hence, giving the user flexibility of selecting the process plan best suited for any required situation.

DEDICATION

To my father

Baharin Ma'amor

"...... every Lord of knowledge there is one more knowing" (Yusuf: 76).

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Dr. Gordon F. Smith, for his support and guidance during my studies and the preparation of this thesis. I would also like to thank Graeme Hoyle and Richard Bobrowski who have always been ready to offer technical direction. I would like to thank Ivan Chan for his assistance on LINDO and interactive programming without which the development would not have been successful.

I am grateful to the Director of Advanced Technology Centre, Dr. Peter Davies who have enabled me to carry out this research. I am also grateful to John Slater of Rover Group for providing the part and the data for me to work on.

Acknowledgements are due to the National University of Malaysia and Malaysian Department of Public Services for their support and sponsorship of my studies in the United Kingdom.

Finally, I would like to thank my wife, Zaini, and my children, Sumayyah, Nawal, Nurhusna and last but not least Usamah, for their love, support and amazing resources of patience over the years.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

DECLARATION

I declare that all the work described in this thesis was undertaken by myself unless otherwise acknowledged in the text and that none of the work has previously been submitted for any academic degree. All sources of quoted information have been acknowledged by means of references.

Shamsuddin Baharin

CHAPTER 1

INTRODUCTION

In the middle ages, the aristocracies were the prime users of manufactured goods. During this period, product quantities were low and repetitive machinery was scarce. Museum exhibits demonstrate that the skills attained by craftsmanship were of the highest degree [1]. A product was evaluated based on its artistic value and uniqueness, and thus, product variation was never an issue. The idea of limiting variations during production started as early as AD 1215.

Magna Carta stated [2],

> *"let there be one measure of wine throughout our kingdom and one measure of ale and one measure of corn .... Let it be the same with weights as with measures."*

Subsequently, during the industrial revolution, the development and use of power driven machinery prompted the evolution of batch and mass production methods.

In 1798 Eli Whitney received a contract from the government to mass produce muskets by using parts that were all `sufficiently identical' to be interchangeable. Not knowing of Le Blanc's project in Europe, Whitney wrote; *"this establishment was commenced and has been carried on upon a plan which is unknown in Europe, and the great leading object of which is to substitute correct and effective operations of machinery for that skill of the artist which is acquired only by long practice and experience"* [3]. On the other hand, Thomas Jefferson, then American minister to France in a letter written to the American Congress about Le Blanc's work wrote; *"an improvement is made here in the construction of muskets, ... It consists in the making every part of them so exactly alike, that what belongs to any one, may be used for every musket in the magazine"* [3].

Phrases such as *'sufficiently identical'* and *'exactly alike'*, implies that there must already exist standards of controlling manufactured part variations. Conformance of these manufactured parts to these standards is what tolerance control is all about.

Tolerance is a phenomenon that has considerable impact on the cost and quality of a product. A classic example about the enigma of tolerance is that, as a design variable for performance, the designer wants the tolerance to be as close to zero as possible. However, generally, the tighter the tolerance about the nominal dimension, the more difficult for it to be processed. Consequently, as a manufacturing variable the engineer wants it to be larger due to the constraints of process and plant capabilities. Rohan on reporting the enigma due to poor tolerance distribution says [4], *"the designers of Elco were using unrealistically tight tolerances in their designs in the hope that the shop floor would come close. As a consequence, the cost of bad quality was running at 3% of sales, or more than the net income of the year. The company was crediting up to US$20,000 per month to some customer for off-grade fasteners."*

It is significantly important not only to think of extremes of clearances that will be satisfactory in practice but also the accuracy available from existing machinery. Production economics should be considered as a vital determinant for the success of the company [5]. A simple ascription of a tolerance to a feature becomes a major determinant for the selection of manufacturing processes, cost of making it and quality of the product.

In general there are three levels of requirements within the definition of a product that are relevant to dimension and tolerance. Those requirements are: (a) specifications; (b) assembly; and (c) processes.

Specifications and assembly relate to either numerical or descriptive standards established for dimension and tolerance characteristics or attributes of a part. Standards are to ensure that the part will meet the acceptable level of functional requirements.

Modern standardisation began in 1901 when the first meeting of the Engineering Standards Committee was held. In 1918 the committee's name was changed to the British Engineering Standards Association. The present name, British Standards Institution was adopted when a Royal Charter and a supplementary charter were granted in 1929 and 1931 respectively. A consolidated Royal Charter was later issued in 1981 [6]. The broad aims of standardisation can be summarized as [7,8]:

(a) to provide a media of communication to all interested parties;

(b) to economize in human effort, materials and energy in the production and exchange of parts;

(c) to protect consumer interests through adequate and consistent quality of goods and services;

(d) to promote quality of life in safety, health and environment; and

(e) to promote trade by removing barriers caused by differences in national practices.

Dimensions and tolerances play an important role in the engineering drawing. Through dimensions and tolerances the designer can relate the design details to departments or individuals in a manufacturing plant.

In Britain the formal approaches to presenting engineering information on drawings started in September 1927, in the publication of the British Standards for Engineering Drawing Practice, BS 308 [9,10,11]. At present these standards continue to be the definitive statements on the specification of workpieces. BS 308 in its three parts describes the practices, conventions and symbols to be used in engineering drawings to assist and guide the user "... *to present the information in the most economic manner without impairing clarity or completeness.*" Ten types of drawings which cover various level of details from assembly arrangements and layout to detail drawings are listed.

The importance of drawing details can be described as:

> "*... a single object and includes all the necessary information required (e.g. the form, dimensions, tolerances, materials, finishes, treatment, etc.) to define completely the object.*" (BS 308 Part 1:§2.3, p 4);

> "*... a product is designed so that it can only be assembled one way, the right way, then it cannot be assembled the wrong way.*" [12];

> "*... a detailed drawing is not just an instruction to manufacture; it is a financial authority stating tolerances, materials and manufacturing methods that will demand the inevitable and irrevocable expenditure.*" [13]

In another catalogue of standards, BS 4500 Part 1 [14], §1.1 covers the specification for limits and fits. It is a system of tolerances and deviations suitable for plain workpieces. BS 4500 §1.2, gives commonly used tolerance grades and deviations in table form. Thus, it can be said that standards are a set of rules for the manufacturer. For example, a domestic problem pointed out by Sir Joseph Whitworth in 1880, that candlebutts and candlesticks came in so many sizes that often did not match will no longer be a problem to the consumer [13].

Processes, on the other hand, deal with the aspect of manufacturing capabilities and production economy. Commonly, these process requirements are categorized into:

(a) *Process Selection* -- several alternative mean of manufacture can be selected depending on the capabilities of the plant. Types of processes and their related tolerance grades are illustrated in table 1.1 and 1.2 and can be acquired from any machining handbook[15].

Machining Operation	Tolerance Grades									
	4	5	6	7	8	9	10	11	12	13
Lapping and Honing										
Cylindrical Grinding										
Surface Grinding										
Diamond Turning										
Diamond Boring										
Broaching										
Reaming										
Turning										
Boring										
Milling										
Planing/ Shaping										
Drilling										

Table 1-1: Relation of Machining Processes to Tolerance Grades

Nominal Size (inch)	Grade									
	4	5	6	7	8	9	10	11	12	13
Over To	Tolerance in thousandth of an inch*									
0.00 - 0.12	0.12	0.15	0.25	0.4	0.6	1	1.6	2.5	4	6
0.12 - 0.24	0.15	0.2	0.3	0.5	0.7	1.2	1.8	3	5	7
0.24 - 0.40	0.15	0.25	0.4	0.6	0.9	1.4	2.2	3.5	6	9
0.40 - 0.71	0.2	0.3	0.4	0.7	1	1.6	2.8	4	7	10
0.71 - 1.19	0.25	0.4	0.5	0.8	1.2	2	3.5	5	8	12
1.19 - 1.97	0.3	0.4	0.6	1	1.6	2.5	4	6	10	16
1.97 - 3.15	0.3	0.5	0.7	1.2	1.8	3	4.5	7	12	18
3.15 - 4.73	0.4	0.6	0.9	1.4	2.2	3.5	5	9	14	22
4.73 - 7.09	0.5	0.7	1	1.6	2.5	4	6	10	16	25
7.09 - 9.85	0.6	0.8	1.2	1.8	2.8	4.5	7	12	18	28
9.85 - 12.41	0.6	0.9	1.2	2	3	5	8	12	20	30
12.41 - 15.75	0.7	1	1.4	2.2	3.5	6	9	14	22	35
15.75 - 19.69	0.8	1	1.6	2.5	4	6	10	16	25	40
19.69 - 30.09	0.9	1.2	2	3	5	8	12	20	30	50
30.09 - 41.49	1	1.6	2.5	4	6	10	16	25	40	60
41.49 - 56.19	1.2	2	3	5	8	12	20	30	50	80
56.19 - 76.39	1.6	2.5	4	6	10	16	25	40	60	100
76.39 - 100.90	2	3	5	8	12	20	30	50	80	125
100.90- 131.90	2.5	4	6	10	16	25	40	60	100	160
131.90- 171.90	3	5	8	12	20	30	50	80	125	200
171.90- 200.00	4	6	10	16	25	40	60	100	160	250

* All tolerance above the line space are in accordance with American-British-Canadian (ABC) agreements.

Table 1-2: ANSI standard tolerances for appropriate selection of holes and shafts tolerances (ANSI B4.I-1967, R1979).

Table 1-3 [16] also displays the type of information that can be gathered and used for tolerance optimisation. In spite of the list found in the handbook, still it is not an exhaustive register. There are many more processes that are available commercially due to the advent of machine tool technology.

Process	Most Suitable Materials	Material Removal Rate	Dimensional Tol. mm(in)	Surface Finish µm (µin)
Turning	Various Machinable	Mild steel up to 21 cu cm (1.3 cu in)/hp.min	±0.025 (0.001)	0.4-6.3 (16-250)
Drilling	Various Machinable	Mild steel up to 300 cu cm (19 cu in)/hp.min	±0.15, ±0.025 (+0.006, -0.001)	1.6-6.3 (63-250)
Milling	Various Machinable	Mild steel up to 6000 cu cm (365 cu in)/min with 300hp	±0.05 (0.002)	0.8-6.3 (32-250)
Planing & Shaping	Various Machinable	Mild steel up to appr. 10 cu cm (0.6 cu in)/hp.min	±0.13 (0.005)	1.6-12.5 (63-500)
Broaching	Various Machinable	Max. of large surface broaches appr. 1300 cu cm (80 cu in)/min	±0.025 (0.001)	0.8-3.2 (32-125)
EDM	Hardened	49 cu cm (3 cu in)/min	±0.05 (0.002)	1.6-3.2 (63-125)
Ultrasonic Machining	Brittle, Hard and non conductive	30-4000 cu cm (1.8-240 cu in)/min	± 0.025 (0.001)	1(40)

Table 1-3: Machining Processes and their tolerance capabilities.

(b) *Cutting Tools* -- several cutter material properties are essential for successful cutting [17]. Unfortunately, an ideal tool material for all conceivable machining conditions has not yet been developed. Some ranges of cutting tools, nevertheless, are available which can be matched to particular applications [18]. An investigation on cutting tool, i.e. tool life, speed and feed etc. for machining processes was conducted at Metcut Research Centre, United States of America. The data has been published and can be used as guidelines to select the speed and feed for machining operations, to predict the period of tool maintenance or to speculate on the occurrence of production variation due to cutting tool faults [19].

(c) *Related Variables* -- process requirements cannot be confined to machine or cutting tools only. Bralla in his editorial says [16]:

> *".... design engineers, manufacturing engineers and industrial engineers in analysing alternatives method for producing a part or a product or for performing an individual operation or an entire process, are faced with the cost variables that relate to materials, direct labour, indirect labour, special tooling, perishable tools and supplies, utilities and invested capital. The interrelationships of these variables can be considerable and therefore a comparison of alternatives must be detailed and complete to assess properly their full impact on total unit costs."*

DelMar and Sheldon elaborate further by describing processes as consisting of any combination of labour, machines and technology [20]. Siddall on the other hand, describes variables for process requirements as [21]: (i) dependent variables - quantities that the designer cannot achieve directly; and (ii) independent variables - quantities that can be specified directly, including the shape, dimensions, tolerances, surface finish, material properties and production volume. A transition is made from dependent variables, such as roundness, to a set of independent variables, such as dimensions and tolerances.

Figure 1-1 is best to describe the aforementioned process requirement influences on dimensions and tolerances.

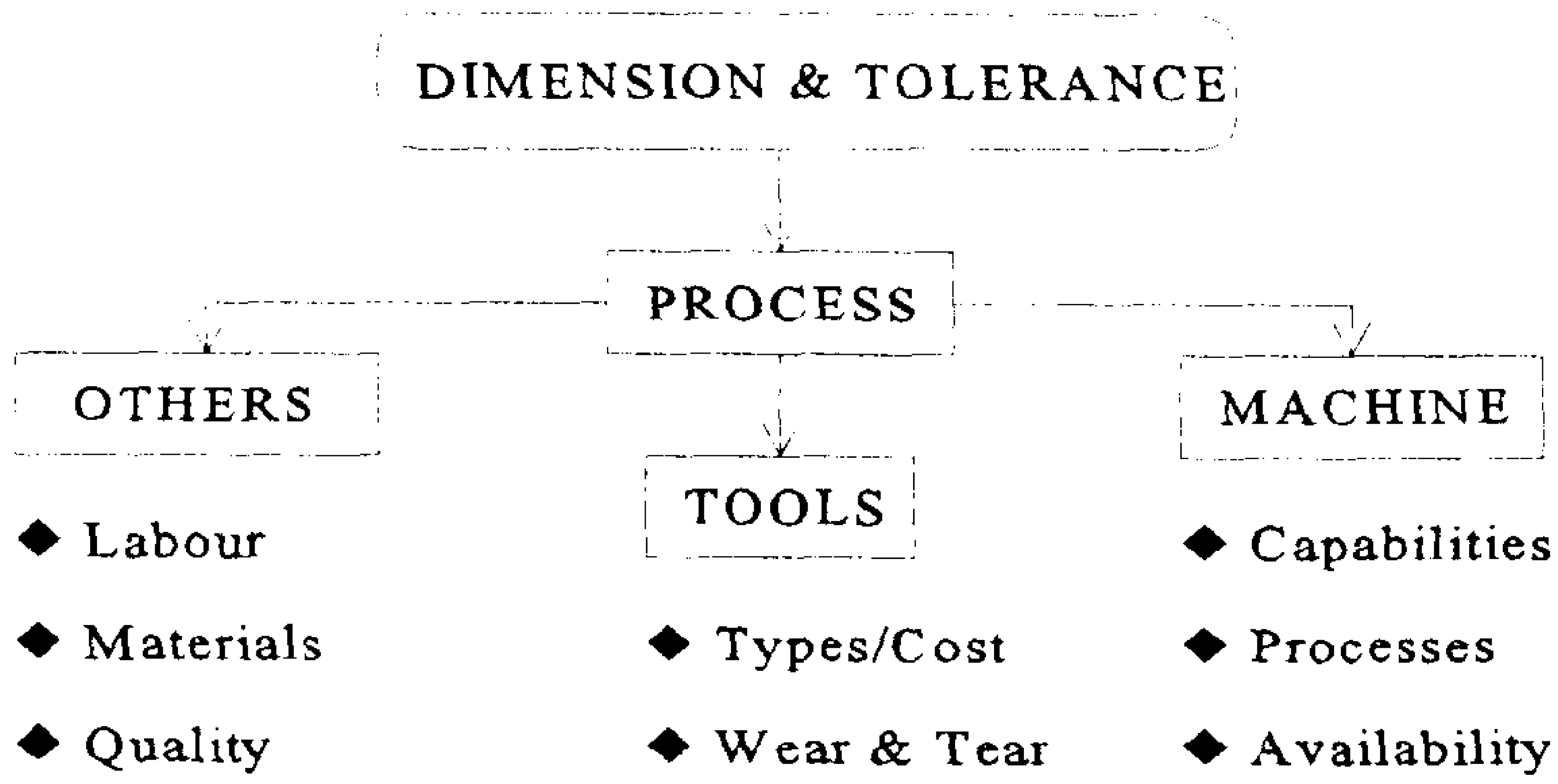

Figure 1-1: Process Requirements

The plight involved in trying to solve bits and details of the requirements is what Simultaneous Engineering methodology is all about. Simultaneous Engineering methodology is where all relevant components of the manufacturing system are made active participants in the design effort from the start. This concept departs from the past practice of serial operation as shown in figure 1-2, which is the cause of fragmentation by specialisation.

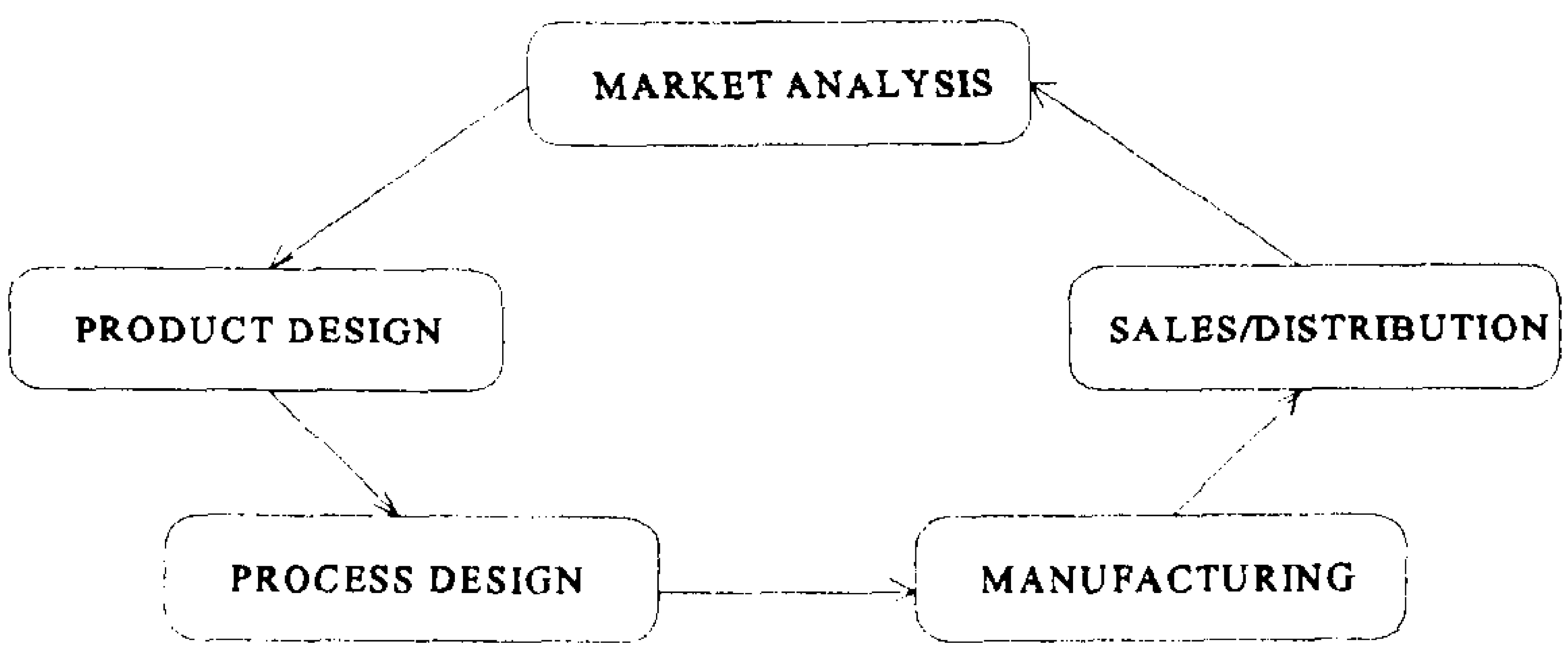

Figure 1.2: Serial Operation

Simultaneous Engineering, also known as Concurrent Engineering, is being recognised as the most efficient and effective approach to aid engineering design. In its new setup shown in figure 1-3, Simultaneous Engineering moves to a more dynamic operational practice.

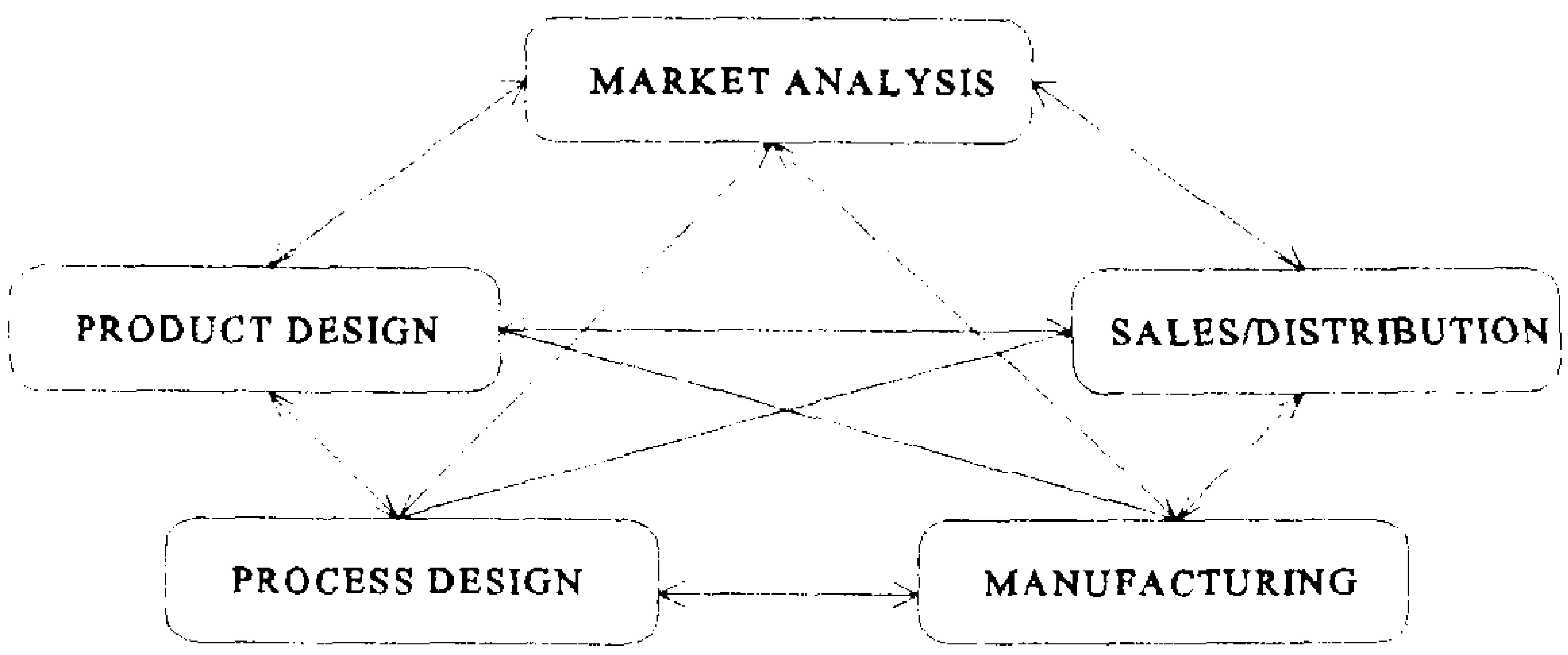

Figure 1.3: Dynamic Operation

Stoll in his remark says [22], *".... a company cannot meet quality and cost objectives with isolated design and manufacturing engineering operations."* To become competitive requires a single engineering effort from concept to production. Thus, the essence of the Simultaneous Engineering approach is the integration of product design and process planning into one common activity. He further commented on the importance of integration is [23], *"to identify product concepts that are inherently easy to manufacture, to focus on component design for ease to manufacture and assembly, and to integrate manufacturing process design and product design to ensure best matching of needs and requirements."* In Simultaneous Engineering, if a manufacturing engineer enters the design circle late in the process and obtains agreement for changes in the design detail then a large part of the design may simply unravel. Many difficult and pivotal choices will have been made for nothing. Simultaneous engineering, nevertheless, is not a product or machine but merely a process. Simultaneous engineering is people, discipline, procedure, methodology and management issues.

Evans described [24].

> *"to achieve Simultaneous Engineering, companies may have to make a special effort to knock down departmental barriers, real or imagined, that have developed over the years. Boundaries between design and manufacturing departments have become institutionalised over the years."*

Gregory [25] commented on the difficulties of carrying out simultaneous engineering is because of two factors: (a) lack of the right tools; and (b) reluctance to change.

The advent of Computer Aided Design (CAD) technology has made it an important acquisition for engineering companies of all sizes and an alternative for implementing simultaneous engineering. The ability to speed up change would certainly overcome the dilemma of constant amendments, corrections and minor improvements due to the intricacy of design and redesign cause by various responses from many scattered users. CAD vendors so far have created excellent geometry engines, tools that can put lines, points, surfaces and mass property calculations on a display. These however are not sufficient for simultaneous engineering. Iwata says [26], *"conventional computer-aided systems developed thus far have machine-like characteristics: representation of internal properties, quantitive representation, sufficient and consistent representation and a routine design process."*

Thus, to enable the distribution of unambiguous product information to all engineering disciplines intelligently, CAD must shift from its present stage to an intelligent system. In the new environment, CAD will be an information supplier, a central database in which all design data resides.

Krause remarked [25]

> *"there were once gurus who knew how to perform certain types of calculations very quickly and adroitly, but that sort of ability is no longer as important with the advent of computerised computational techniques. The same thing is going to happen to these knowledge-based gurus."*

Research on Intelligent Knowledge Based Systems is developing rapidly in respect of concept, techniques and applications. Successes in the construction and application of the techniques in various fields are mentioned in many reports [27,28,29,30]. Application of Intelligent Knowledge Based System concepts working interactively with CAD have been the subject of considerable attention for the past decade. Seely says that knowledge-based and CAD complement each other [31]. This view is supported by Rosenfeld [32], *"unlike traditional CAD programs that capture geometric information only, knowledge-based engineering systems capture the intent behind the product design - the how and why, in addition to the what of the design."* Making CAD to be intelligent is what *Intelligent CAD* technology is all about.

A lot of investigation of design specifications especially on dimension and tolerance optimisation have been undertaken over the years. Unfortunately, little of its understanding found its way into the industry or commercially available software. Tolerance optimisations involve not only extremes of clearances but include the selection of manufacturing processes, cost of making it and quality of the product to be manufactured. McGoldrick [33] in a survey of sixty three different software products reported that, *"there are no CAD systems currently being marketed in the UK which incorporate any sort of tolerance model."*

Intelligent CAD technology can be adopted to optimise the dimensions and tolerances of workpieces intelligently by considering variation factors in manufacturing; for example, process capabilities. The system called Knowledge-based Automatic Tolerance Analysis in short KATA, is used to analyse all cylindrical shapes and simple prismatic parts. Parts are analysed based on deviations after being machined using processes such as turning, milling, grinding etc. The reference for analysis of deviation is the nominal dimension and tolerance specified in the blue print or CAD drawing.

CHAPTER 2

TOLERANCE FUNCTIONS AND CONTROL

2.1 Introduction

Tolerance means,

> *"an allowable variation from standard."* [34]

> *"the amount of variation permitted on dimensions or surfaces of machined parts."*[15]

> *"the total amount of variation permitted for the size of a dimension, a positional relationship of the form of a profile or other design requirement."* [10]

From these definitions, it can be derived that the tolerances fall into two categories namely, one that is related to a dimension and the other that is related to geometric features. These two types of tolerance are commonly known as *dimensional* and *geometrical* tolerances.

2.2 Dimensional Tolerance

Dimensional tolerances also known as conventional tolerances, set limits on the values of linear and angular dimensions of the workpiece. BS 308 Part 2 §8.2, expresses dimensional tolerance in several different ways. In figure 2.1 the tolerance is

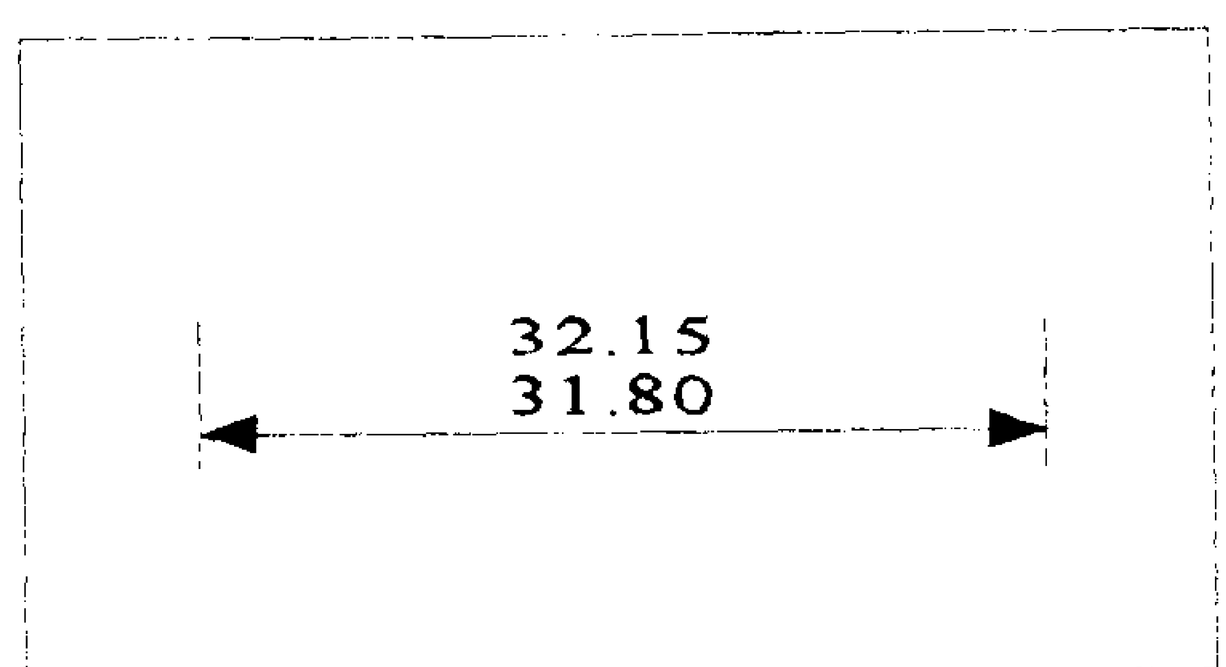

Figure 2-1: Limits presentation

specified through the upper and lower limits of the dimension. A further expression that can be found in BS 308 is the "±" tolerance assigned to a dimension. Three forms of the "±" representation are illustrated in figure 2-2. Where *fits* taken from BS 4500 are used and it is desired to give appropriate designating symbols on drawing, they may be expressed as in figure 2-3:

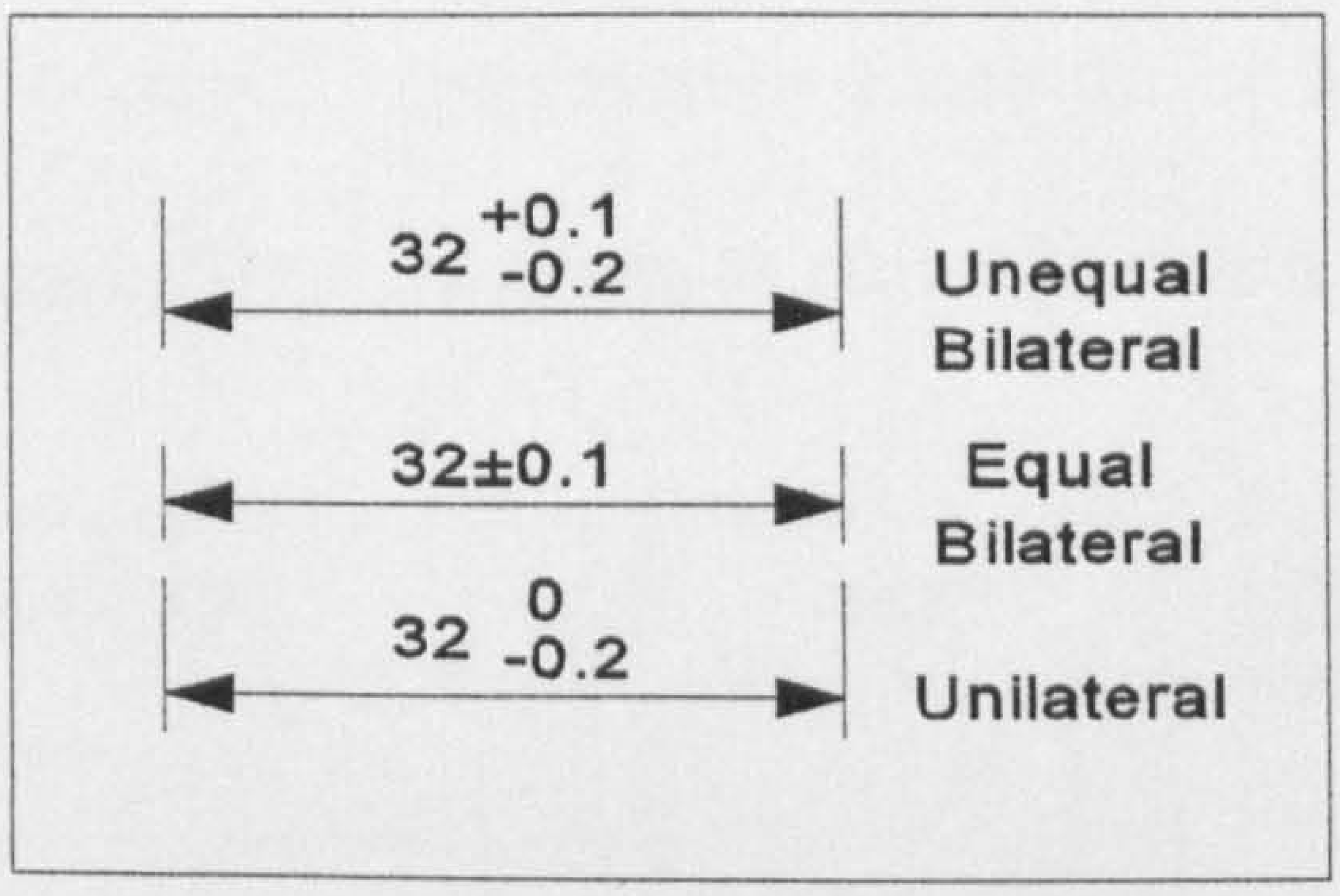

Figure 2-2: Forms of Tolerance

SHAFTS

(i) ⌀30f7

(ii) ⌀30f7 $\begin{pmatrix}-0.020\\-0.041\end{pmatrix}$ or ⌀30f7 $\begin{pmatrix}29.980\\29.959\end{pmatrix}$

HOLES

(i) ⌀30H8

(ii) ⌀30H8 $\begin{pmatrix}+0.003\\0\end{pmatrix}$ or ⌀30H8 $\begin{pmatrix}30.033\\30.000\end{pmatrix}$

Figure 2-3: Limit and Fit Representation

2.3 Geometrical Tolerance

Geometrical tolerance differs significantly from dimensional tolerance. Dimensional tolerances directly constrain the dimensions of the part. However, geometric tolerances have no direct control on it, thus define the size and shape of a tolerance zone within which the feature is to lie. There are four main types of geometrical tolerances. Location, attitude and form, respectively define the relative position, relative orientation and the permissible variation of the surface itself. The fourth type, a composite also known as run-out tolerance, defines all three in a single tolerance.

There are fourteen geometrical characteristics described in BS 308 Part 3 §3. Each characteristic has a combination of attributes as shown in table 2.1.

CHARACTERISTICS	TYPE	FEATURE	SYMBOL
Straightness	FORM	SINGLE	—
Flatness			⏥
Roundness			○
Cylindricity			⌭
Profile of a line		SINGLE OR RELATED	⌒
Profile of a surface			⌓
Parallelism	ORIENTATION	RELATED	//
Squareness			⊥
Angularity			∠
Position	LOCATION		⌖
Concentricity			◎
Symmetry			⌯
Run-out	COMPOSITE		↗
Total run-out			⌰

Table 2-1: Geometrical Tolerance Characteristics.

The feature column represents the type of part feature. Single features are an attribute of a single surface feature. For example, a face may have flatness tolerance and a cylindrical surface may have roundness tolerance. Related features define the relationship between the surface feature and one or more other features referred as datum. A datum is defined by British Standards as a theoretical exact geometric reference such as an axis, plane, straight plane, straight line, etc., to which tolerance features are related. Profile characteristics can belong to either single or related features. A comprehensive description of geometrical tolerancing can be found in [11,35].

2.4 Tolerance Expression

In a production environment tolerance is expressed as either:

(a) a *design tolerance* related to the operational requirements of an assembly or the part specifications; or

(b) a *manufacturing tolerance*, also known as *working tolerance*, recognised mainly as a set of instructions to assist process planning for manufacture.

The relationship between these two set of tolerances can be described as; *manufacturing tolerances are used to ensure the realization of design tolerances.*

2.4.1 Identifying the Problem

In quantity production, a part cannot be machined dimensionally according to design tolerances as shown on the blueprint or CAD drawing, in this case, datum surfaces must be set. These datum surfaces are based on the selection of locating surfaces for fixturing and cutting tool layout. As a result, tolerance accumulate, not withstanding, whether dimensions from individual cuts are added or subtracted. This is often referred to as tolerance stackup. An example of tolerance stackup is shown in figure 2-4. The problem is seldom avoided in machining, regardless of whether the equipment is manual, numerically controlled or is a special purposed transfer line.

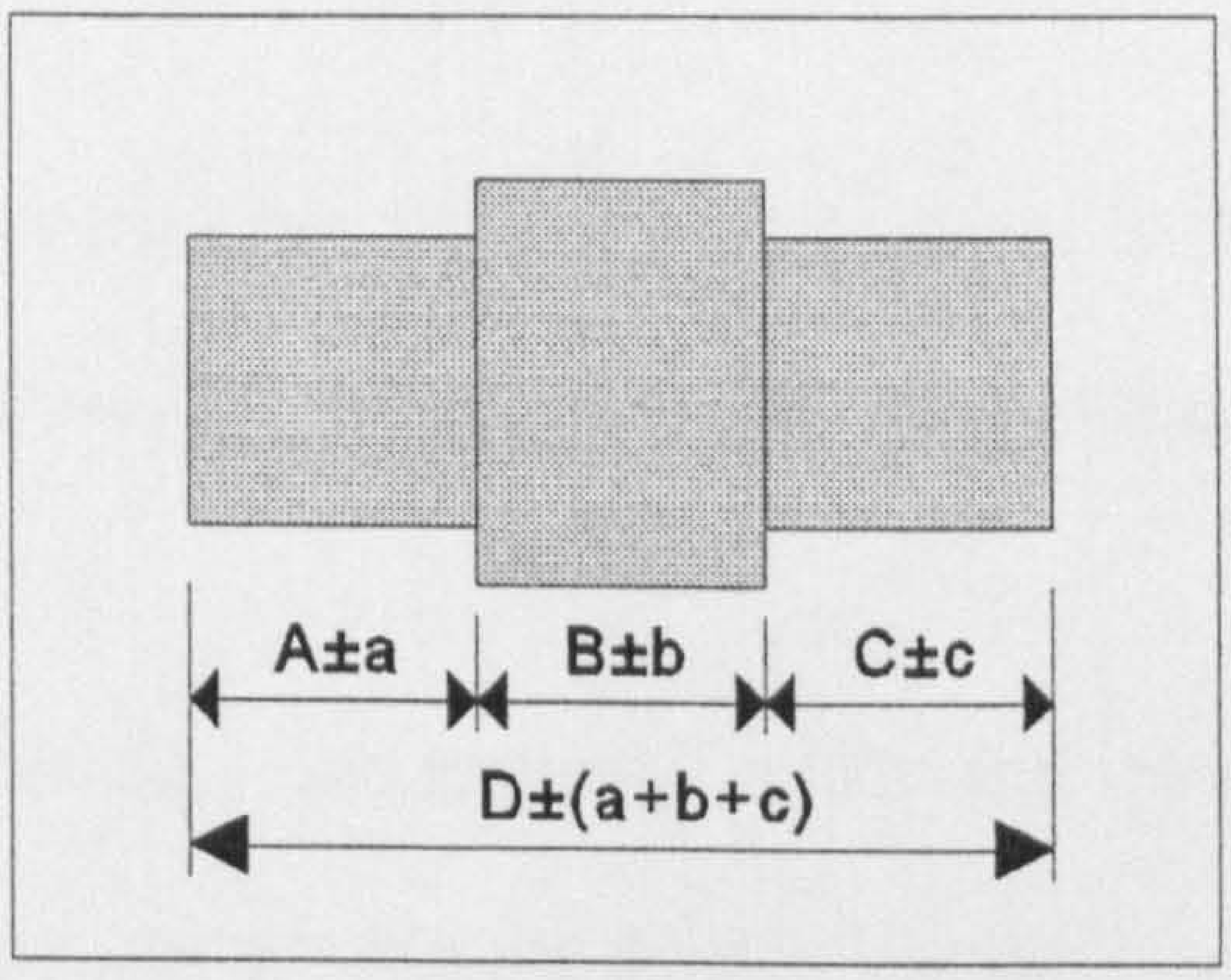

Figure 2-4: Tolerance Stack-up.

2.4.2 Tolerance Control

During the past decades, there has been no structured approach to the control of manufacturing tolerance variation. The general approach is essentially the trial and error technique of postulating workpiece tolerances. Subsequently, the analysis to ascertain whether this postulated set of tolerances fulfils the design specification would be executed. Usually, the chosen set of tolerances would prove to be unsatisfactory. Therefore, the tolerances are changed, the analysis redone and evaluated, and this step iterated until a satisfactory set of workpiece tolerances is obtained. This early method of tolerance control is described by Darwin [36] as;

> *"....it looked as if some of the tolerances were assigned much closer than should be necessary, and I started to find out how they had been fixed ... I*

concluded that in designing a machine the chief engineer drew it free-hand with dimensions to the nearest inch, and sent to the draughtmans to work out the detail to the nearest thousandth, who then gave it to his junior assistant, anxious not to get himself into trouble, would, as a general rule think of the smallest number he knew and halve it."

Since then, a number of experiments and algorithms have been conducted and tested. From the review of related literature [37,38,39], investigations on tolerance control related to one of the following:

(a) *Worst-case tolerancing* -- an approach where the tolerances are assigned to the workpiece in such a manner that the probability that it will not function properly is zero; and

(b) *Statistical tolerancing* -- a technique which simply relaxs the requirements of worst-case tolerancing to allow the probability of not functioning to be non zero.

2.5 Worst-Case Tolerancing

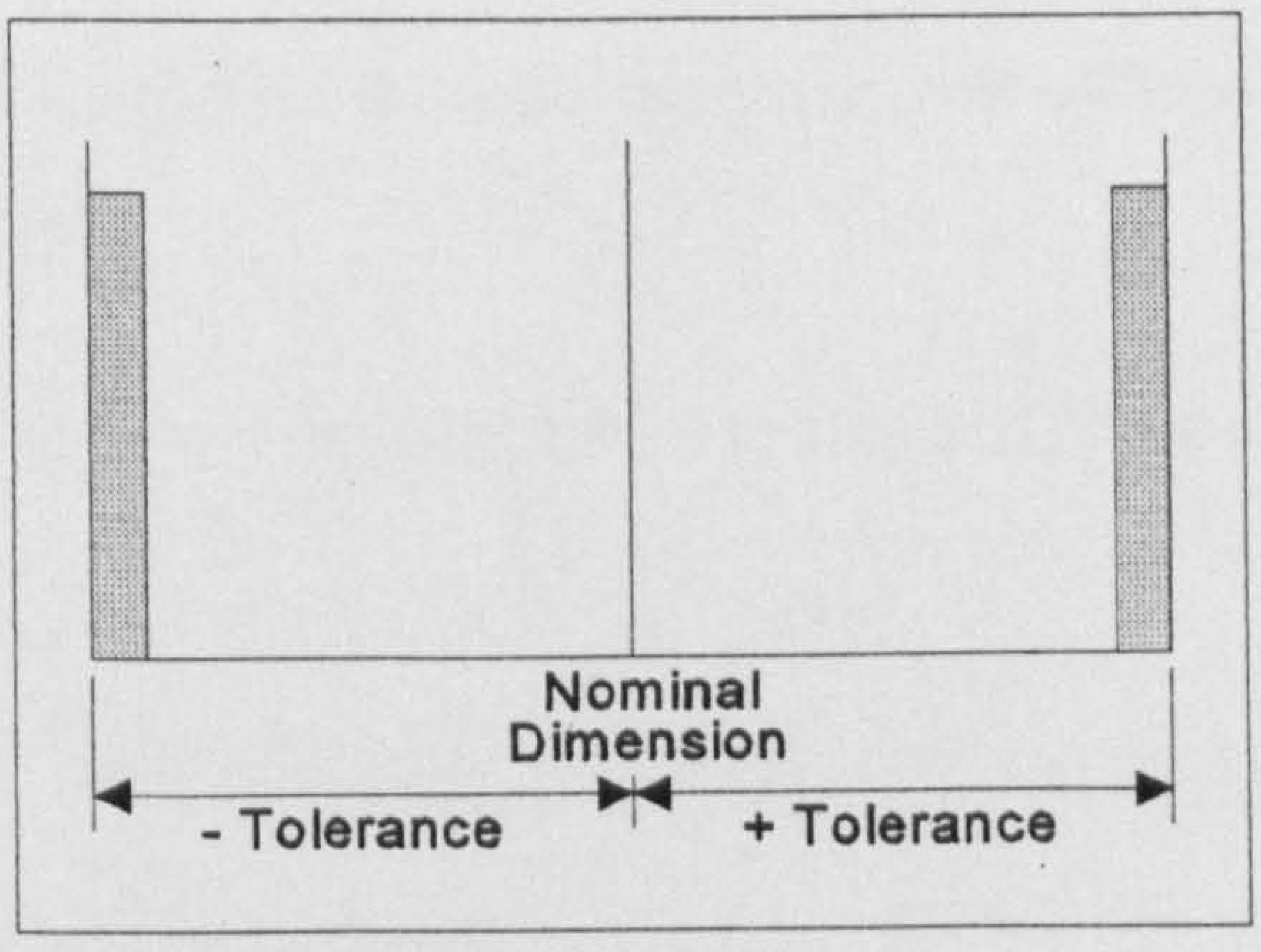

Figure 2-5: Worst-case tolerancing

The worst case approach also, referred to as tolerance stackup analysis, or method of extremes assumes that workpiece dimensions and tolerances are at one of the allowable limits as shown in figure 2-5. If the dimension is defined as $d_o \pm t$, where d_o is the nominal dimension and t the tolerance, then the worst-case occurs when the actual dimension d is taken to be either d_o+t or d_o-t. This gives a standard deviation of $\pm t$ and an equation for the cumulative tolerance across several defined tolerances is the simple sum:

$$t_s = t_1 + t_2 + \ldots + t_n \tag{2-1}$$

where: t_s is the cumulative tolerance and t_l to t_n are the individual tolerances.

2.5.1 Tolerance Charts

One of the prominent worst-case tolerancing techniques experimented with by tolerance optimisation investigators is the tolerance chart. The use of a tolerance chart was reported as early as the fifties [40,41,42,43]. A tolerance chart provides a graphical representation of each operation contained in the process plan. Its purpose is to show how individual cuts combine to produce each blue print dimension. It yields a set of linear algebraic expressions showing the relationship between each desired blue print dimension and the individual cuts that contribute to it.

Some basic principles of preparing a tolerance chart are:

(a) maximum possible tolerances should be assigned to each in-process cut without violating the specified blueprint tolerances;

(b) values of tolerance assigned to a cut should be consistent with the range of process capabilities on the machine tools; and

(c) minimum or maximum stock removal for each cut should be feasible with respect to the process, tool and work material under consideration.

As shown in figure 2-6, inputs required to produce a tolerance chart are:

(a) sequence of operations to be performed;

(b) surfaces of the part to be machined in each operation in thick dark line;

(c) surfaces of location for fixturing and gauging purposes;

(d) surface generated for each operation; and

(e) machine chosen for each operation.

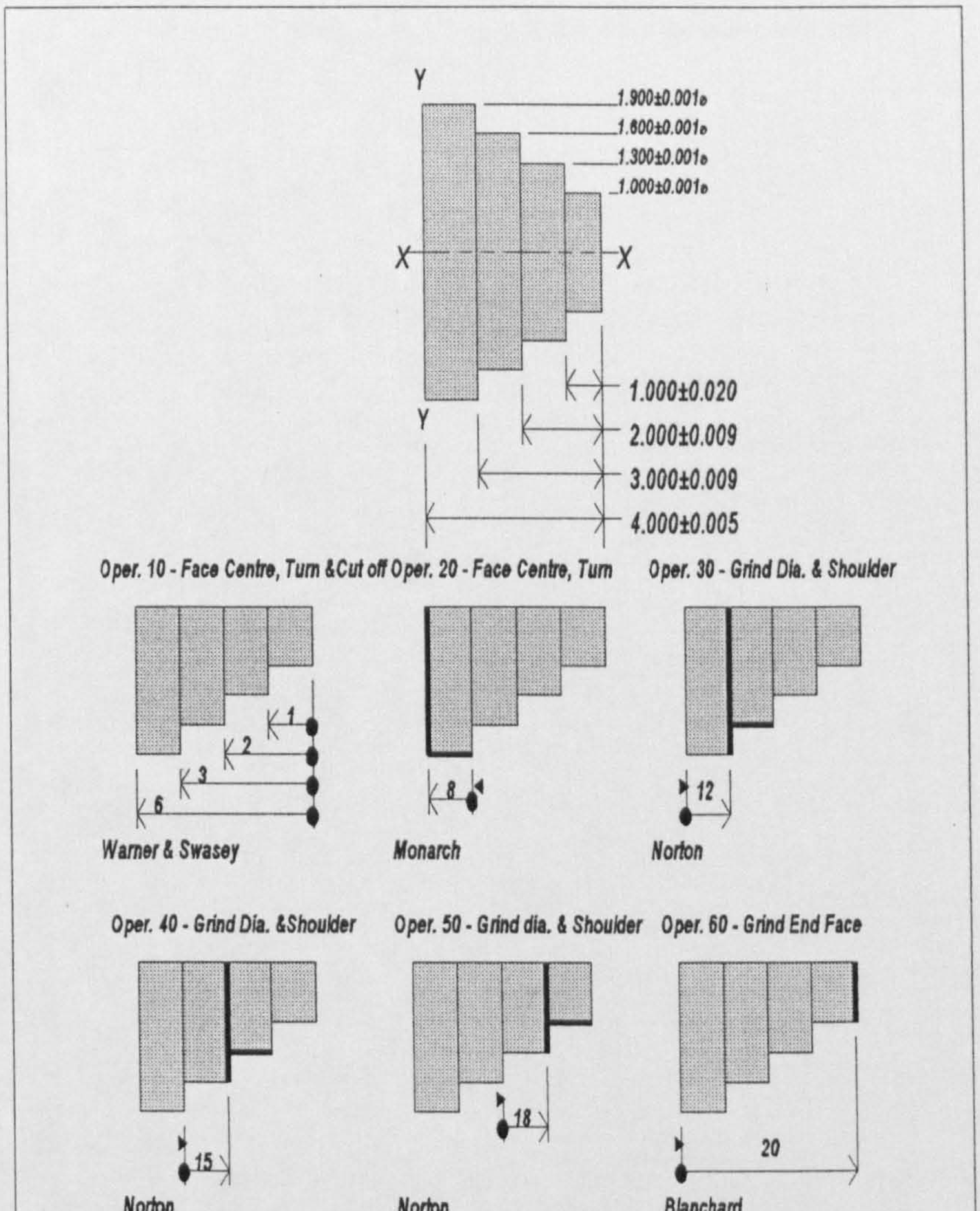

Figure 2.6: Blue print drawing and process plan for steel plug

An example of a manually developed tolerance chart is shown in figure 2.7. A step by step chart's development is explained in detailed in the literature [44,45].

The development of the chart heavily depends on the knowledge and experience of the user. It is difficult for an inexperienced engineer and strenuous and taxing for the expert. To overcome the complexity, investigators have diverted to a more effective and efficient methodology by automating the tolerance chart analysis. These advances are much accelerated with the advent of Computer Integrated Manufacturing (CIM).

(10) 1.015±.002, (50) 1.000±.001
(10) 1.315±.002, (40) 1.300±.001
(10) 1.615±.002, (30) 1.600±.001
(10) 1.916±.002, (20) 1.900±.001

Line No	Oper No	Machine use	(Machine To) Mean	Tol	A B C D E	(Balance Dim) Mean	Tol	Line Involved	Stock Rem Mean	Tol
1	10	W&S	.979	.003					SOLID	
2	10		1.994	.003					SOLID	
3	10		3.003	.003					SOLID	
4						1.009	.006	3 - 2		
5						2.024	.006	3 - 1		
6	10		4.031	.010					SOLID	
7						1.028	.013	6 - 3		
8	20	MONARCH	1.008	.004				7 - 8	.020	.017
9						2.017	.010	4 + 8		
10						3.032	.010	5 + 8		
11						4.011	.007	3 + 8		
12	30	NORTON	1.000	.001				8 - 12	.008	.005
13						1.017	.011	9 - 12		
14						2.032	.011	10 - 12		
15	40	NORTON	1.000	.003				13 - 15	.017	.014
16						2.000	.004	12 + 15		
17						1.032	.014	14 - 15		
18	50	NORTON	1.000	.015				17 - 18	.032	.029
19						3.000	.019	16 + 18		
20	60	BLANCHARD	4.000	.001	Z			11 - 20	.011	.008
21					W	3.000	.002	20 - 12		
22					X	2.000	.005	20 - 16		
23					Y	1.000	.020	20 - 19		
24										
25					A B C D E					
26			**BLUE PRINT**			**RESULTANTS**				
27			4.000	.005	Z	4.000	.001	20		
28			1.000	.020	Y	1.000	.020	23		
29			2.000	.009	X	2.000	.005	22		
30			3.000	.002	W	3.000	.002	21		
31										

Figure 2-7: Tolerance chart for steel plug (figure 2-6)

2.5.2 Automating Worst-Case Tolerancing

The earliest development of automated manufacturing tolerance control was Computer Aided Tolerance Control (CATC) [46,47].

CATC is developed in four segments;

(a) *drawing input*, to allow the user to draw simple geometric shapes, i.e, lines, arcs etc;

(b) *data entry*, where the user interactively enters data such as routing and dimensions and tolerances;

(c) *tolerance analysis*, to compute working D&T and stock removal amounts; and

(d) *output*, which is in the form of a plotted tolerance chart.

The resultant D&T and chart adjustment of CATC are carried out in the procedure using the following equations:

$$W_i = F + C_L \sum_j S_{jL} + C_W \sum_j S_{jW} \qquad j \in (j>i) \qquad (2\text{-}2)$$

where: i, j are the numbers of the working dimensions; L and W are the location and working dimension lines respectively; C_L and C_W are 1 if the locating or working side of W_i is an outside dimension; C_L or C_W will be -1 if the locating or working side of W_i is an inside dimension; F is the final dimension; and S_{jL} and S_{jW} are the stock removal from the locating and working surfaces, after the current working dimension.

A point worth noting of CATC is that it has instigated a significant idea in its approach. An important aspect in allocating tolerance is to consider all related features connected to the dimensions and tolerances being analysed. Information tracing is the foundation for finding the solution to the tolerance stack-up problem. Manually, it is a very meticulous and taxing operation. CATC on the other hand, has eliminated this technical difficulty due to the assistance of a computer.

Fainguelernt, Weill and Bourdet [48], demonstrated the feasibility of computerized tolerancing and dimensioning on an Apple IIe microcomputer. Tolerance allocation is optimized in relation to functional requirements, machining capabilities, influences of tool wear and work settings.

The optimization of tolerances is carried out interactively in the following sequence:

(a) input of workpiece data (drawing of the workpiece and its dimensions);

(b) input of process plan data (sequence of operations, values for setting dimensions and tolerances, etc.);

(c) optimization of setting dimension tolerances, with checking of feasibility;

(d) computation of variations and machine dimensions;

(e) computation of mean values of the setting dimensions; and

(f) output of results as a simplified version of a tolerance chart separated into three different charts: (i) drawing; (ii) setting; and (iii) manufacturing dimensions.

This approach which included other factors of variation such as tool wear and work setting in the analysis is considered to be a significant contribution to tolerance optimisation. It is very much different from the algorithm adopted in earlier approaches.

Computer-Aided Dimensional Planning (CADP) developed by Xiaoqing and Davies [49] who experimented with a matrix-tree-chain technique. The adoption of this technique was to enable the continuous iteration of tolerance allocation until a suitable set of tolerances is found. This enhanced the algorithm used by CATC where the analysis will be terminated if an error is detected, thus, it was unable to adjust the tolerance automatically. In this case the data in CATC has to be adjusted from the start, the analysis redone and the step iterated until a satisfactory set of tolerances is obtained.

In the technique, the matrix which depends on the size of the tolerance chart to be represented is first defined. A row specifies an individual dimension, i.e. the dimension line in the tolerance chart; a column specifies a vertical shoulder surface, i.e. vertical line in the tolerance chart. A series of matrix operations is performed to find out the relationship between a dimension and other dimensions for tolerance calculation purposes. Following the matrix applications, tree operations are conducted by stretching each branch of one sub-tree, and cutting off repeated branches until no more sub-trees can be used to stretch the tree, figure 2.8.

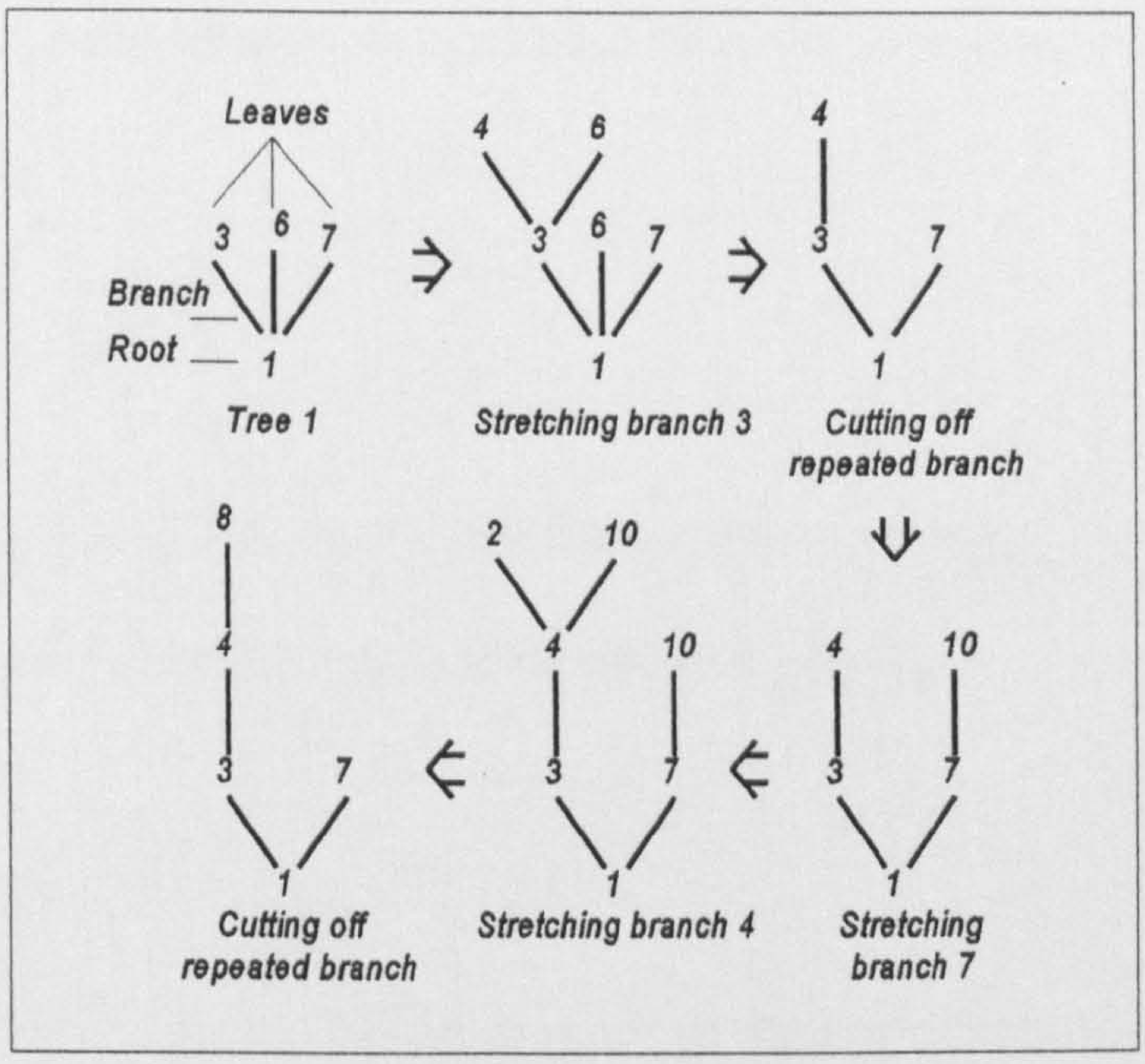

Figure 2-8: Tree operations

The matrix-tree-chain procedure has only solved the problem of checking the accumulation of tolerances and calculations of working dimensions and stock removal. To be able to make an adjustment of tolerances, Xiaoqing and Davies had to use an iterative method. First, the maximum ratio of difference between the resultant tolerance and the drawing tolerance is obtained:

$$X_{\max}=\max\left(1-\frac{R_j}{B_j}\right) \qquad (j=1,2,\ldots,n) \quad (2\text{-}3)$$

where R_j is resultant tolerance (or sum of tolerances) and B_j is blueprint tolerance, and

$$R_j=\sum_i t_i \qquad (2\text{-}4)$$

where t_i is the tolerance of an individual dimension in the tolerance chain obtained by the matrix-tree-chain. R_j is the resultant dimension when X_{max} is a maximum.

Secondly, the proportion of the tolerance of an individual dimension to the sum tolerance R_j in the tolerance chain can be obtained by:

$$S_{ij}=\frac{t_i}{R_j} \qquad (2\text{-}5)$$

Then, an iterative operation is carried out:

$$t_i^{(k+1)}=t_i^{(k)}+t_i^{(k)}*X_{\max}^{(k)}*S_{ij}^{(k)} \qquad (2\text{-}6)$$

where k is the number of iterations.

If the resultant tolerances are less or more than the drawing tolerances or the stock removals are not sufficient for the cutting operation, the procedure must be repeated with a new t_i, X_{max} and S_{ij} until the results are satisfactory. CADP is written in FORTRAN and runs on VAX 11/750 under the VMS operating system. Unfortunately, Xiaoqing and Davies explained that CADP is not able to handle angular tolerancing.

The significant contribution of Xiaoqing and Davies work comes from the use of tree theory technique to comprehend the complexity of the tracing procedure. The procedure proposed diverts from tracing a feature of a single process operation to a more dynamic process and enables the analysis of a multiple operation.

2.5.2.1 Tracing for Relationship

The tree theory technique stems from the linear graph concept. The earliest known paper on linear graph theory was written by a Swiss mathematician, Euler in 1736 [50].

A linear graph is a simple way of displaying the structure of a system by drawing a diagram consisting of points called *vertices* and line segments called *edges*. The vertices are connected by edges so that relationships between the components can be indicated. A *tree* in the theory means *"a connected graph that has no circuits"*; in which, a circuit is an arc that returns to its starting point [51]. This means that there are no multiple edges. Thus, in a tree there is a unique arc connecting any pair of vertices. Apart from Xiaoqing and Davies, tree theory has been used by several investigators to assist in stackup tolerancing.

The *Rooted-tree*, as shown in figure 2.9, has been adopted by Whybrew et al [52]. Each link represents a machining operation with an associated working dimension, and each node represents a machine or a locating surface.

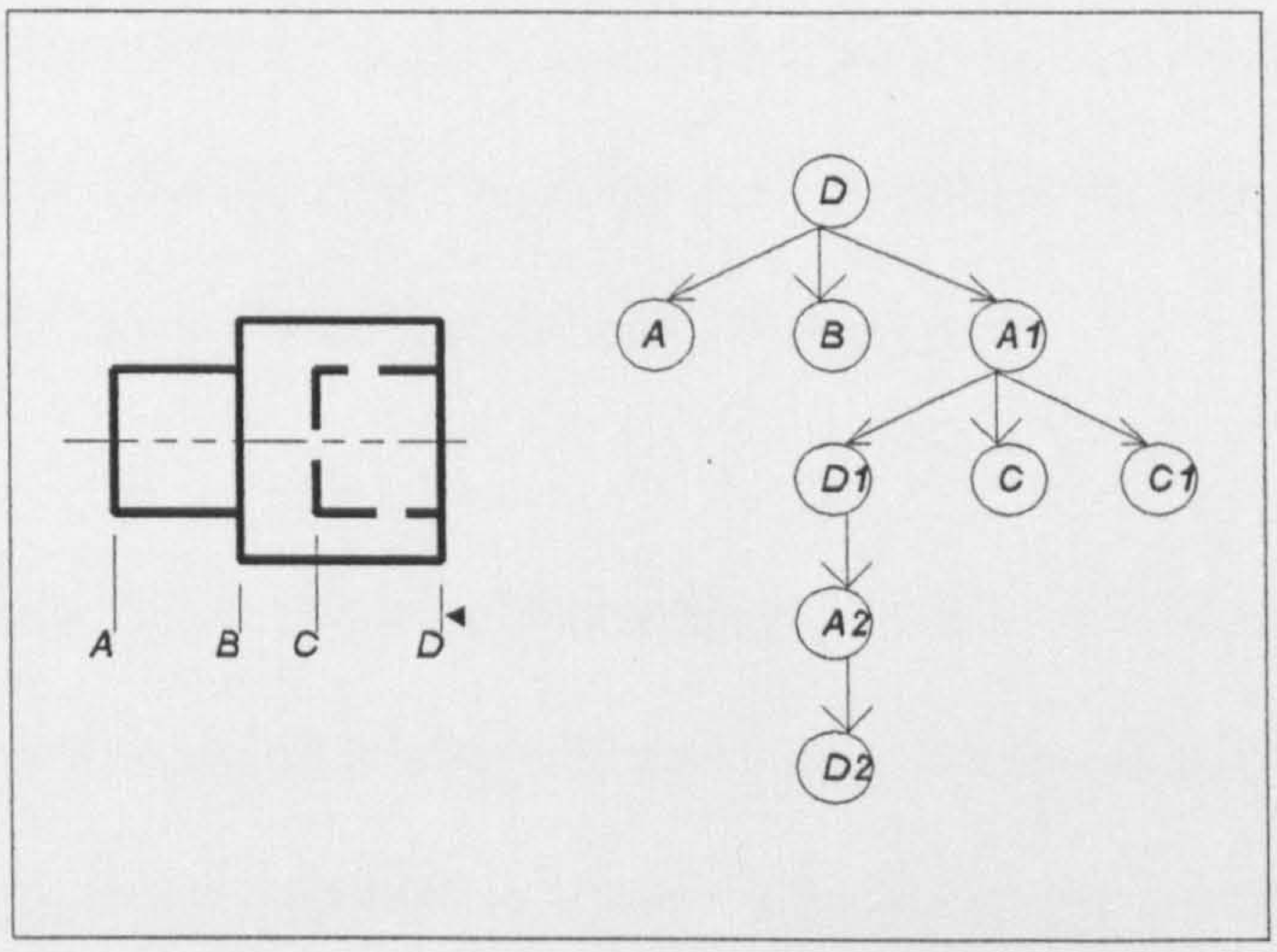

Figure 2-9: Rooted tree

From the validation made by Whybrew et al., a disturbing result has been presented that the manufacturing tolerance results are larger compared to the blueprint requirement is contrary to the basic objective of tolerance control. There is no further explanation of the matter by the investigators. It can only be hypothesised that Whybrew et al had not included the blue print specification in the algorithm resulting in a lack of sensitivity and leading to faulty analysis.

Using a similar *rooted tree*, Ngoi overcomes the insensitivity by including two types of constraints in the mathematical model, namely: (a) the blueprint specification; and (b) process capabilities [53,54]. The mathematical model, which represents a resource allocation problem, is solved using the Linear INteractive Discrete Optimizer (LINDO), a commercial software package for solving linear programming problems.

Using the same mathematical model and tolerance balancing process, Ngoi and Ong [55,56,57] tried another type of tracing approach that they called a *path tracing technique*. The idea behind this alternative is to develop a system of linear equations from the early stage of tracing. However, the number of blue print dimensions is often less than the number of unknown working dimensions. As such, extra information is required to ensure uniqueness in the system of linear equations. Ngoi and Ong made it possible by inputting information on the amount of metal removal for roughing and finishing cuts.

To overcome the predicament of adding information, He et al [58,59] introduced three types of path traces to determine (a) the functional equations, (b) the machining equations and (c) the assembly equations. This special path trace technique gives a procedure for expressing the assembly, functional and machining equations in mathematical forms. Based on the equations derived from the traces procedure, assembly, functional and machining constraints are established to optimise the dimension and tolerance for design or for process planning.

Li and Zhang [60] used a dimensional connection graph to analyse the dimensional connections between part surfaces. Similar in features to a rooted tree, each node of the dimensional graph represents a surface element and each edge represents the dimensional connection between two surface elements.

Zhang, Mei and Dudek [61], however, highlighted several disadvantages in the method proposed by Li and Zhang such as:

(a) NC machines are not included in the discussion;

(b) coordinate of tool movement for working dimension is not even discussed; and

(c) fixture requirements to facilitate the operation are not described.

Thus, Zhang et al have proposed that:

(a) before applying the working dimension and tolerance chain, the selection of datums elements and setup variation are to be decided;

(b) all start and end coordinates of tool movement for all working dimensions are included;

(c) fixture requirements to facilitate the operation are described in the analysis.

Therefore, by these additions, the system is now able to analyse workpieces manufactured using an NC or CNC machine.

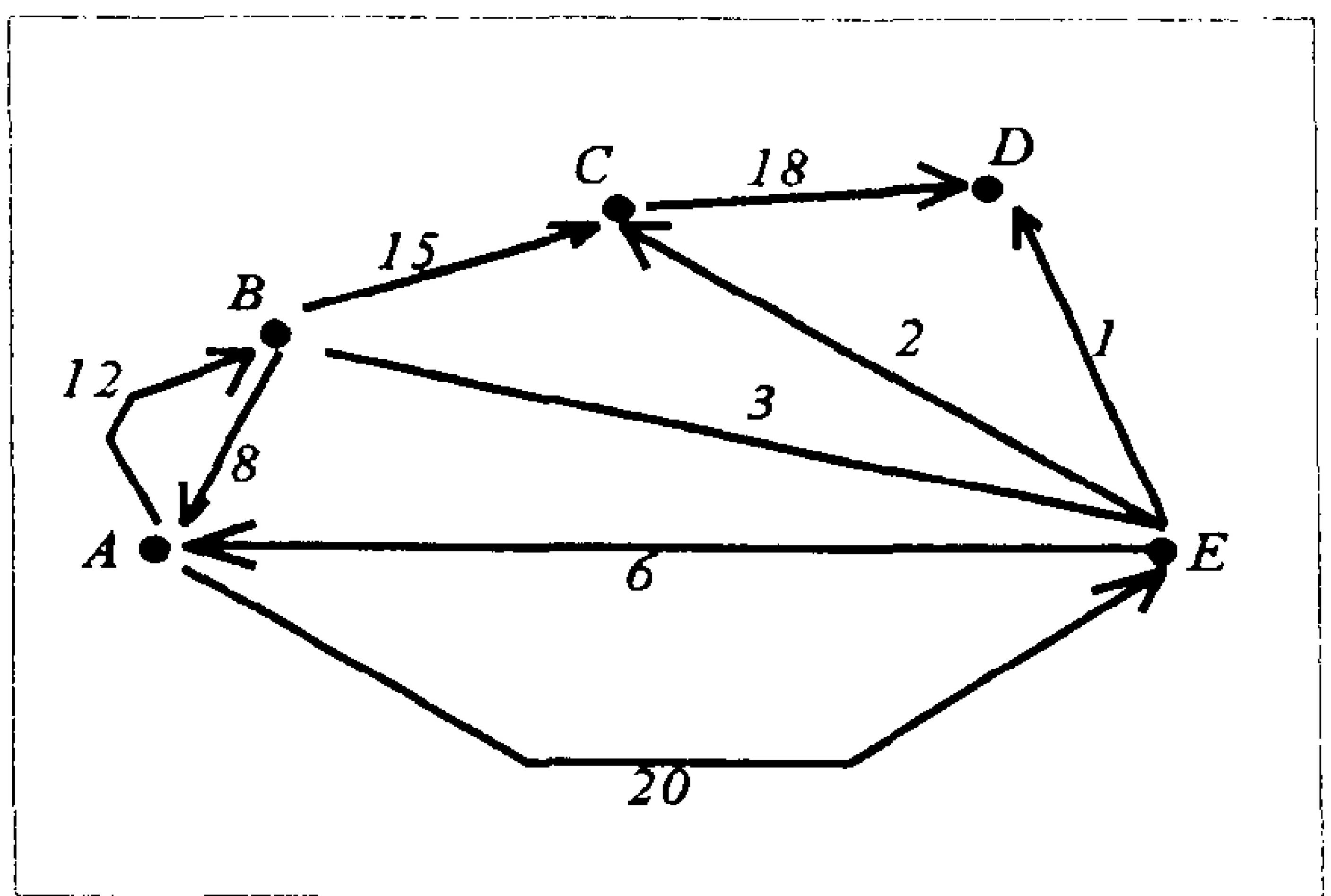

Figure 2-10: Graph Representation of Machining Sequence

A unique *graph* representation for surface machining relationships has been developed by Irani, Mittal and Lehtihet [62]. Figure 2.10 illustrates the machining relationship of the Steel Plug model used by Wade in his validation. Each cut is represented as a numbered arc in the graph, by which, each arc number refers to the corresponding line assigned to the machining cut in the tolerance chart. The tail node of an arc represents the surface used as a datum for the cut. The head node corresponds to the surface on which material removal occurs. Once the relationship is obtained, the cumulative residual tolerance on all blueprint dimensions and stock removal is minimized subject to blueprint tolerances, stock removal and process capabilities. Linear programming software, LINDO running on a microcomputer is used to make the analysis.

Ji [63,64] uses three trees in his algorithm namely:

(a) *blueprint dimension tree,* used to check whether blueprint dimensioning is correct and to create the base for a blueprint and stock removal tree;

(b) *blueprint and stock removal tree* where solid stock removal is identified and used to create the nodes for a working dimension tree; and

(c) *working dimension tree* is to assist in finding the dimensional chains for blueprint dimensions and stock removal.

Tolerances are then assigned automatically to workpieces using a linear programming model on a microcomputer. Ji listed several setbacks of the approach he developed:

(a) consideration is given to only vertical surfaces;

(b) angular cutting would not be covered by the model;

(c) the model does not encompass special types of operation (e.g. heat treatment, electroplating); and

(d) raw material dimensions are also not included.

2.6 Statistical Tolerancing

The use of statistics for analysing engineering parts and processes is not new. Slote's comment on Spotts' article of 1959, on the application of statistics to dimensioning was that, *"there is nothing new in the statistical approach he presents"* [65]. This is because the basic statistical technique is very much well defined. Thus, it is important to understand the technique first before considering any new procedures which have developed from it.

Workpieces produced which do not function properly without the fault being assignable to the tolerance values are probably caused by tool wear, faulty material, workmanship, process capabilities and other unforseen variables.

In view of these factors, Evans has outlined two facets of the problem [66,67,68]:

(a) the problem of ascertaining the distribution of the response of a mechanism for given workpiece tolerance allocations; and

(b) the problem due to the shifting and drifting of workpiece tolerance distributions.

Evans said that the problem can be solved by allowing the part tolerances to be defined as probability distributions, which is the essence of statistical tolerancing.

Four techniques have been studied by Evans:

(a) *linear propagation of error*, where the technique uses the first order of Taylor series for worst-case analysis;

(b) *nonlinear propagation of error* where Taylor's series is extended to the sixth order due to the inaccuracy of the first order, with Evans commenting on it as too analytical;

(c) *quadrature*, is employed when a numerical approach is contemplated and recommended by Evans to be used. Full explanation of the technique is in the literature [69,70,71]; and

(d) *Monte Carlo Simulation*, in which the method allows an unlimited precision compared to the other three techniques, unfortunately, a setback of the technique is that a fairly large sample is needed, therefore, Evans advised not to use the technique.

Sayed and Kheir [72] commented on the Monte Carlo technique that it can only inform the user if the tolerance is too low. However, the technique does not explain which is to be changed and by how much. On the other hand if the tolerance is inadequate, the user is not sure whether the optimum has been achieved.

The objective of the statistical technique is to allow the probability that a workpiece or assembly will not function properly to be non-zero, in contrast with the worst-case approach. Hence, the engineer is forced to choose some theoretical distribution shape that best simulates the frequency distributions obtained in practice to give a close estimate for tolerance allocation.

Mansoor has referred to six statistical distribution shapes used to represent process tolerances as shown in figure 2.11 [73]. The assumptions made or implied by the investigators of the distributions referred by Mansoor are basically identical and are as follows:

(a) the frequency distribution curve spreads symmetrically over the tolerance range;

(b) the frequency distributions of all dimensions have identical shapes;

(c) assembly is obtained from random components selection of sizeable production lots;

(d) the distribution of the dimension would be about the mean dimension; and

(e) no work outside specifications enters the assembly.

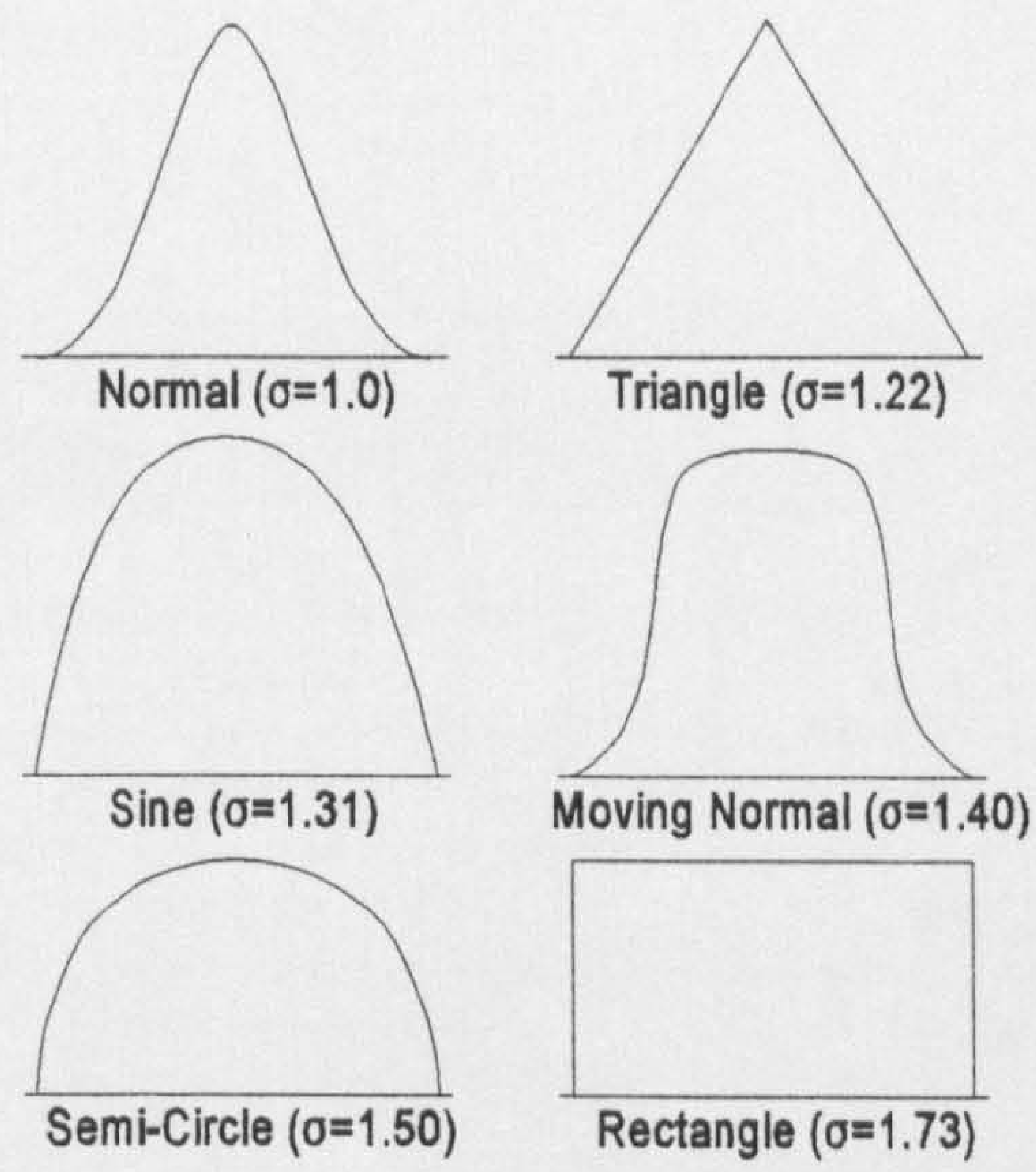

Figure 2-11: Distribution shapes used by different investigators for the application of probability to tolerances.

Thus, the essential difference is in the shape of the frequency distribution adopted, i.e. the magnitude of the standard deviation associated with each curve. Therefore, the formulae could differ from each other in degree and not in the basic form. The literature since then has focused on Normal distributions and its variations.

2.6.1 Normal Distribution and its Variation

The normal distribution has assumed the formula, [74]:

$$f(x) = \frac{e^{-(x-\mu)^2/2\sigma^2}}{\sigma\sqrt{2\pi}} \qquad (2\text{-}7)$$

where: $e = 2.718$; μ is the mean valueof the sample; $\pi = 3.141$; and σ is the standard deviation.

The maximum occurs at $x = \mu$ and there is symmetry about $x = \mu$. The area under the normal distribution curve gives the probability of a particular tolerance being achieved. Examples of probabilities are shown in Table 2.2 [75]:

RANGE (in Standard Deviation)	AREA
-1 to +1	0.6827
-2 to +2	0.9545
-3 to +3	0.9973
-1.96 to +1.96	0.95
-2.576 to +2.576	0.99

Table 2-2: Areas under portions of a unit Normal Distribution.

Two important statistical laws which were used extensively by the investigators were:

(a) *law of addition of variances* which states that the variance of the sum or difference of two or more independent random variables is equal to the sum of their variances; and

(b) *central limit theorem* which states that, a linear combination of independent random variables will approach a normal variate as the number of components becomes large [76]. This results in the use of the following basic equation:

$$\sigma_s = \sqrt{\sigma_1^2 + \sigma_2^2 + \ldots + \sigma_n^2} \tag{2-8}$$

where σ_1, σ_2, ... σ_n are the standard deviations of a number of independent variables and σ_s is the standard deviation of the combined distribution.

When applying the normal distributions to tolerances, it is commonly assumed that the tolerance t is equivalent to three times the standard deviation 3σ. Thus a full range of -3σ to $+3\sigma$ corresponds to an area of 99.73%, which represents 27 measurements in 10,000 being outside the permitted limits. The theoretical spread of the distribution gives an estimate of the process capability and this is defined as the *natural process tolerance*. Other factors have also been suggested; Mansoor for instance, uses 3.09σ corresponds to an area of 99.9% for data analysis.

If a factor, Z, is used so that $\sigma = t/Z$, then the relationship between tolerances can be derived from equation 2-8:

$$t_s = \left(\frac{Z_s}{Z}\right)\sqrt{t_1^2 + t_2^2 + \ldots + t_n^2} \tag{2-9}$$

where t_s is the tolerance obtained by summing tolerances t_1 to t_n Z is the factor used for all the individual distributions and Z_s is the factor required for the sum tolerance. If the natural process tolerance of the components reflect the percentage acceptance required for the assembly, then $Z_s = Z$ and equation 2-9 simplifies to:

$$t_s = \sqrt{t_1^2 + t_2^2 + \ldots + t_n^2} \tag{2-10}$$

This simple statistical method is referred to as the *Root Sum of Squares* (RSS) method. RSS assumes that all components are normally distributed with a mean at the tolerance midpoint.

The occurrence of non-random factors can cause the RSS method to produce more than predicted out-of-tolerance parts. Some of these non-random factors are due to a component part distribution that is not well approximated by normal distributions or the mean is not at the tolerance midpoint. Hence, a modified RSS method is adopted where a conservative RSS analysis is modified through a calculated correction factor [77,78].

Thus, equation 2-10 would become:

$$t_s = C\sqrt{t_1^2 + t_2^2 + \ldots + t_n^2} \tag{2-11}$$

Many correction factors have been suggested. The most common is 1.5, which is recommended by Bender [79]; whereas, Gladman [80] suggested a range of 1.4 to 1.8.

The law of addition of variances has been recommended by Mansoor [73] for the case when the component tolerance affecting an assembly is known and it is required to find the expected range of the assembly dimensions. The law is mainly used when it is possible to gather information on the arithmetic mean and the standard deviation of the component populations from actual production lots.

The equation used for the analysis is:

$$T_{Prob} = T_{Arith} - \sum t_i + \sqrt{\sum t_i^2} \tag{2-12}$$

where T_{Arith} is the arithmetic sum of workpiece tolerance; T_{Prob} is the probable sum; and t_i is the natural process tolerance.

For cases where the assembly requirement is known and it is required to tolerance the component dimensions to satisfy this requirement, Mansoor [73] has introduced a K factor which requires prior knowledge of the parameters obtained during production so that component tolerance can be selected using the following equation:

$$T_i = K t_i \tag{2-13}$$

where T_i is the tolerance specification for dimensions (T_a, T_b,..., T_c), and t_i is the natural process tolerance for A, B,..., N.

In this case, K factor is acquired from equation 2-14:

$$K = \frac{T_{Prob} + \sum t_i - \sqrt{\sum t_i^2}}{\sum t_i} \tag{2-14}$$

Table 2-3 shows the guideline used by Mansoor [73] as an approximate relationship between K and process capabiliy.

K	COMMENTS
K<1	Process will not be able to produce work to the tolerance specification
1≤K≤1.33	Process could produce work to the tolerance specification (to allow for small deviations from the assumptions made, K should be made 1.1). Strict quality control methods must be used particularly at the lower range.
1.33<K≤2	Process will easily produce work to the tolerance specification but a good level of control is still required, especially at the lower range.
K>2	Process will easily produce work to the tolerance specification with controls easing as K increases.

Table 2-3: Relation between K factor and production control.

Mansoor's work on the application of probability design methods is extended by Parkinson [81,82].

Two types of analysis namely, failure function analysis and second-moment analysis, originally developed in connection with problems in reliability theory used to analyse the dimensional tolerances of manufactured components.

When substantial samples of data on the dimensions of the components are obtainable, failure function analysis is used. Second-moment analysis, on the other hand, is less accurate and can only provide limits on the probability of failure to assemble to specification, but is more appropriate to cases of limited information. In this case, instead of sample data, from which the statistics of the failure function may be deduced; the available information takes the form essentially of the means, variances and covariances for the relevant component dimension.

Three types of probability distributions used for the analysis were: (a) normal; (b) truncated normal which provides an alternative distribution where a high degree of dimensional control can be expected to eliminate outliers (Therefore, some or all of the dimensions remain within specified limits); and (c) mixed normal distributions which have random mean values. A set of FORTRAN programs running on VAX/VMS system is used to carry out the risk assessment and cost optimization of dimensional tolerances [83]. One of the setbacks of the technique as mentioned by Parkinson is that in short production runs or small sample sizes, it is possible that the risk figure cannot be interpreted and there will always be an underlying statistical error in which the smaller the sample the larger the error.

Mansoor [73] and later Parkinson [81,82,83] have pointed out the problem of mean shift in which production processes are seldom controlled closely enough to keep the mean dimension exactly centred between the tolerance limits, figure 2-12. Based on this mean uncertainty, Chase and Greenwood introduced a technique called a unified model and also known as a mean shift model [84,85]. The mean shift technique permit the inclusion of mean shifts in the tolerance analysis. The method is based on resolving the component's tolerances into two parts: a mean shift or bias from the tolerance midpoint (first moment of the distribution); and the variability about the mean (second moment). This is accomplished by selecting a shift factor f_i for each component between 0 (RSS) and 1.0 (worst-case). The resulting tolerance sum has the form shown in the following equation:

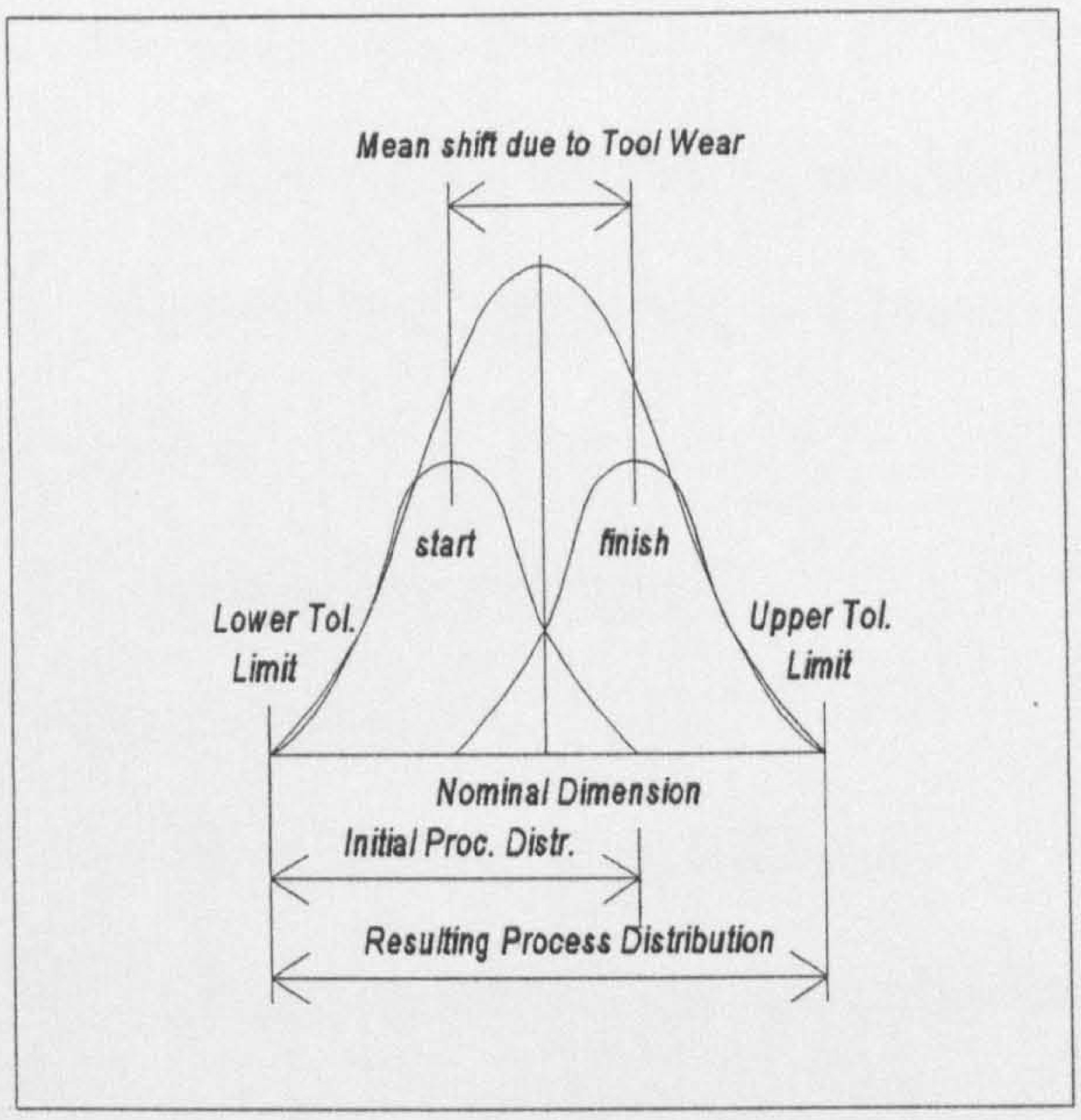

Figure 2-12: Example of cause of mean shift,

$$tol_s = f_1 tol_1 + f_2 tol_2 + \dots + f_n tol_n + [(1-f_1)^2 tol_1^2 + (1-f_2)^2 tol_2^2 + \dots + (1-f_n)^2 tol_n^2]^{1/2} \qquad (2\text{-}15)$$

The first summation is composed of the estimated mean shifts. It is treated as a worst case model as all shifts are assumed to combine to give the greatest assembly shift. The second summation represents the component variability and is treated as the sum of squares. Each component variability in this second summation is reduced by the factor $(1\text{-}f_i)$. This presumes that the process variability is small for parts with a large mean shift, that is, the component variability is still assumed to be three sigma from the mean to the nearest tolerance limit.

Bjørke [86], on the other hand, prefers Beta distributions instead of Normal ones which he considers inflexible. The flexibility of the Beta distribution allows better approximation of the real distribution. Michael and Siddall [87] favoured Beta distribution as a basis of optimisation. The advantages of the Beta distribution are:

(a) it covers a range of distributions from normal to uniform (rectangle);

(b) it has a finite range (the normal distribution is infinite);

(c) it covers asymmetrical cases (the normal distribution is always symmetrical);

(d) it covers confidence levels of up to 100% (normal distribution only does this for infinite tolerances); and

(e) the method applies to tolerance chains with as few as two dimensions.

Bjørke [86], however, reminds us that the penalty of using this model is that it has more computation. The probability density function of a generalised Beta distribution in the interval [a,b] is:

$$f(x,\gamma,\eta,a,b)=\frac{1}{(b-a)B(\gamma,\eta)}\left(\frac{x-a}{b-a}\right)^{\gamma-1}\left(1-\frac{x-a}{b-a}\right)^{\eta-1} \qquad (2\text{-}16)$$

where: $\gamma>0$, $\eta>0$, $a\leq x\leq b$ and $B(\gamma,\eta)$ is the Beta function define by the integral:

$$B(\gamma,\eta)=\int_0^1 z^{\gamma-1}(1-z)^{\eta-1}dz \qquad (2\text{-}17)$$

where $z = (x\text{-}a / b\text{-}a)$.

Figure 2-13 shows examples of symmetrical Beta distributions and figure 2-14 examples of assymmetrical distributions.

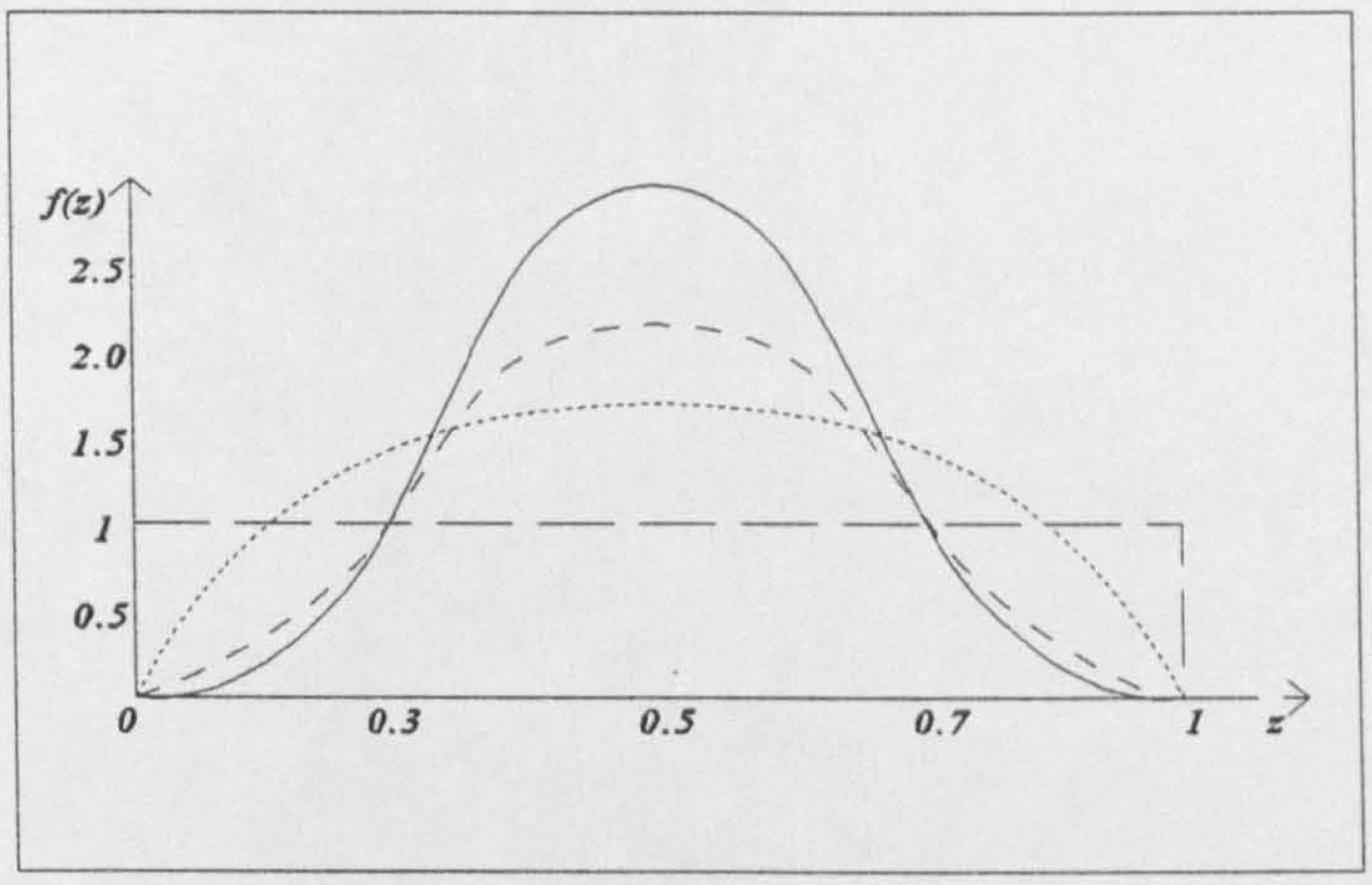

Figure 2-13: Symmetrical Beta Distributions

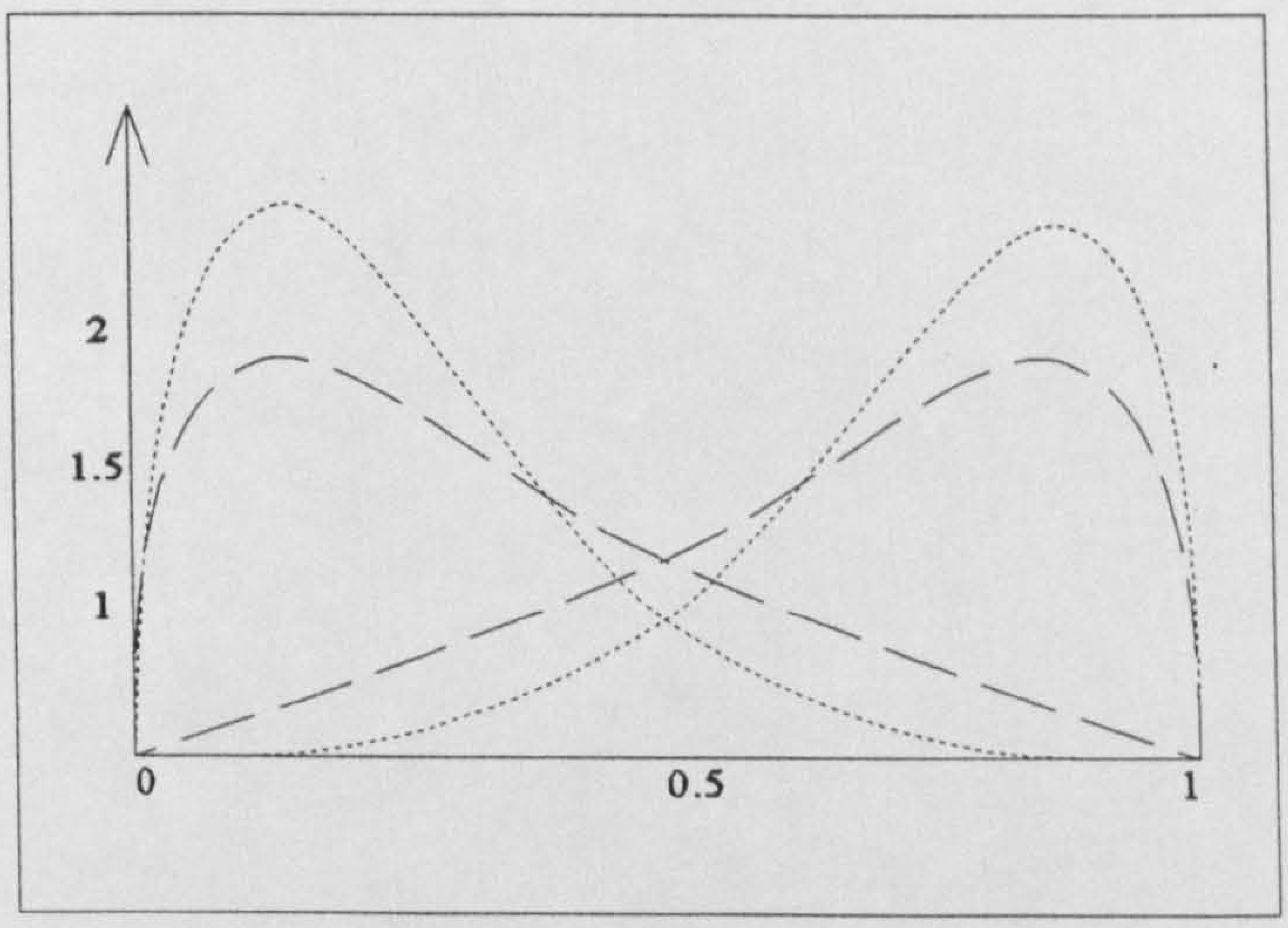

Figure 2-14: Asymmetrical Beta Distributions.

2.7 Tolerance for Minimum Cost

Review of the tolerance control literature thus far has dwelt on using blueprint dimensions, process capabilities and work settings as constraints. Implicitly by doing so, the cost of manufacturing is assumed to be embedded in the analysis objective. Nevertheless, there are several other investigators who explicitly use minimum cost tolerance as their approach constraint.

Cost minimisation is chosen as the objective for the following reason:

(a) The manufacturing engineer must make a process plan that is not only functionally correct but also suitable for manufacture at low cost.

(b) The objective of other tolerance allocation approaches is to obtain the least possible scrap percentage, because the quantity of scrap is assumed to be proportional to cost [88]. However, it does not necessarily follow the bigger the scrap percentage the higher the cost. Cost depends not only on scrap percentage, but also on the machining, tooling, etc., expenses of each operation.

(c) There are many other manufacturing criteria such as maximum profit, maximum quality gain, maximum rate of return, etc., where, most of them are difficult to relate to tolerance, hence, cost is the more common and easier critera.

From a review of the literature, the relationship of component tolerance x to the cost of achieving it in production is a function of the form shown in figure 2-15. This curve shows two well-known basic features, which are obviously essential for a cost-tolerance relationship according to normal manufacturing experience.

The two features are:

(a) when $x = 0$, $C = \infty$; and

(b) C should be a decreasing function of x, tending to become flat as x becomes large.

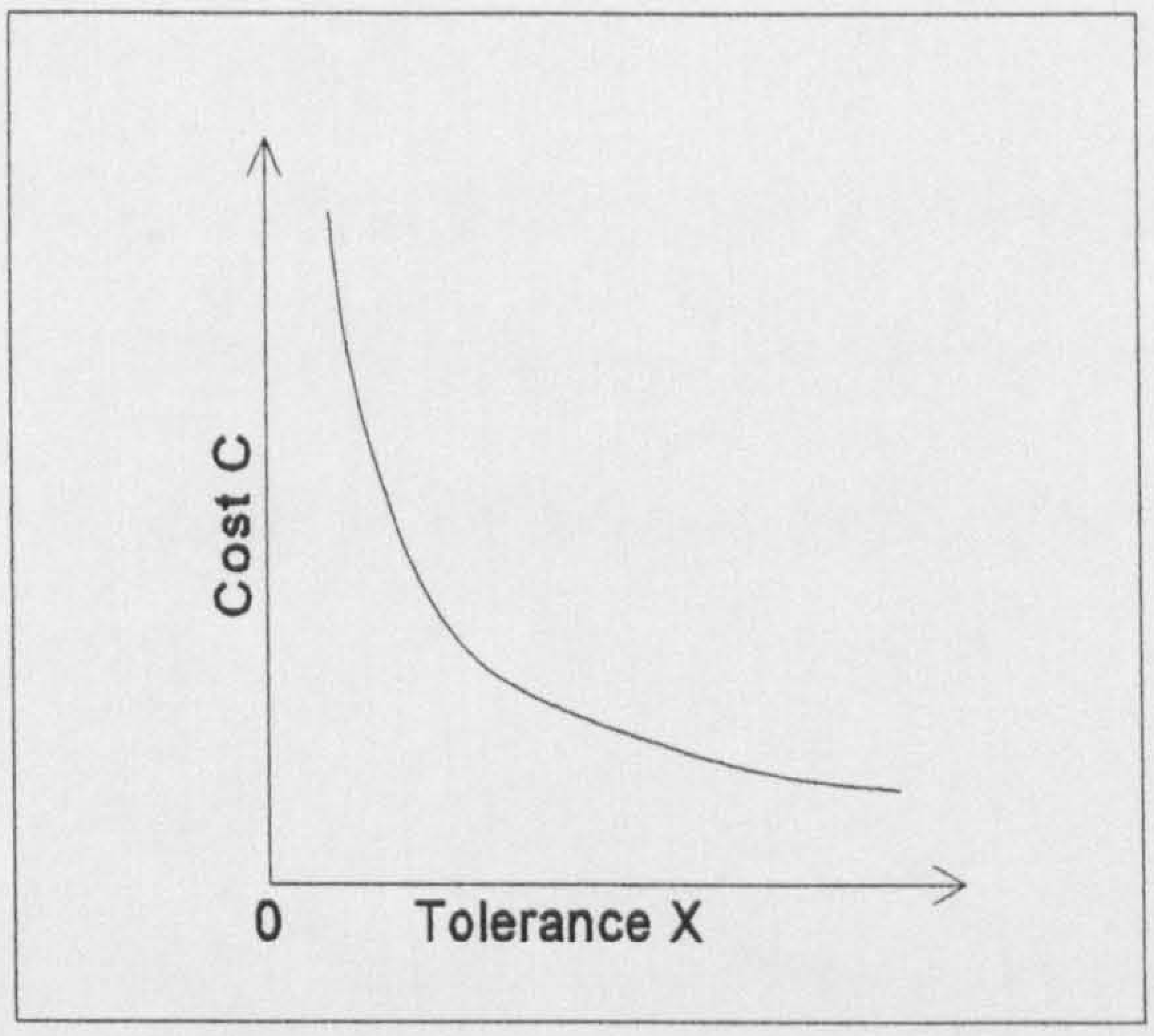

Figure 2-15: Cost-Tolerance Relationships

Three proposed objective functions for cost minimisation were developed by He [88] according to various situations. When cost information is available, average cost per acceptable unit produced is minimized, otherwise the ratio of the number of components to be machined to output is minimised. The idea behind this is that, scrap does not necessarily represent a non-optimum production method. The machining costs for different operations are different (due to different machine tools, tools and machining conditions); the shape and variability of manufactured dimension distributions are also different, so the rates of occurrence of scrap from operations are different.

There are cases in which some sets of inexpensive operations with certain scrap levels will cost less than a set of expensive operations with no scrap. Thus, the total cost should be balanced between these operations. Therefore, the problem is to achieve an economic balance between increased part costs as tolerance increases (scrap occurs), versus increased machining costs as tolerance is made smaller (no scrap).

The last of He's objective functions is that, if residual tolerance exists, sums of manufactured tolerance are maximised by assigning weights according to their process capabilities. Combining these objectives, a computerised optimisation program called CADT was established. CADT determines the optimum set of dimensions and tolerances to satisfy the design specifications and the permissible machining allowances, with the cost of producing to be minimised.

Cost optimisation has been integrated by Irani et al [62] in their second model using a mixed-integer programme assuming that the blue print and stock removal constraints remain unchanged. The ability to select an alternative machine within a desired tolerance range and process capability is included in the existing constraint model to make the system more viable. Due to the significant increase in the number of constraints and decision variables, the programming model was hosted by an IBM mainframe computer.

Tang et al [89] has also utilised the approach of dimensional chains but with a limited set of constraints compared to Irani et al. In the technique, process tolerance and stock removal are linked together through dimension chains. An assignment of the design variables corresponding to the least value of cost function is represented by equation 2-18:

$$T_c=\sum_{n=1}^{M} f_n({}_nX_b, {}_nX_e) \tag{2-18}$$

where f_n, ${}_{n}X_b$, ${}_{n}X_e$ represent the manufacturing cost, working tolerance and stock removal at the nth operation respectively.

The manufacturing cost for each operation is estimated by:

$$f_n({}_{n}X_b, {}_{n}X_e) = {}_{n}A + {}_{n}B \cdot {}_{n}X_b + {}_{n}C \cdot {}_{n}X_e \cdot {}_{n}X_f \tag{2-19}$$

where ${}_{n}A$, ${}_{n}B$, and ${}_{n}C$ are constants depending upon the machine selected. The value of ${}_{n}X_e \cdot {}_{n}X_f$ in the last represents the volume of the material to be removed during the nth operation.

Spotts [90] hypothesised that minimum cost can be achieved by widening the tolerances of the more expensive parts in an assembly to lower the manufacturing cost. On the other hand, the tolerances of some less expensive parts are tightened. This trade off was meant to balance out the cost of producing the parts.

A similar technique has also been described by Speckhart [91]. Relative cost of holding the tolerance and the importance of that tolerance to the operation contribute significantly in allocating the tolerances to the components.

The method of solution is by first expressing the cost-tolerance data by an analytical expression:

$$\$_j(i) = A(i) + B(i)e^{C(i)t_j(i)} \tag{2-20}$$

where $\$_j(i)$ is the cost of holding the tolerance on the ith dimension for the jth restraint condition; $t_j(i)$ is the tolerance ith dimension by the jth restraint condition; and $A(i)$, $B(i)$ *and* $C(i)$ is constraint for the ith dimension that can be obtained by using a nonlinear least-squares curve fit procedure to fit the equation to the discrete cost-tolerance data.

Finally, Speckhart [91] uses Lagrange's multipliers to reduce the cost function subject to the restraint condition. Since the method requires a large number of computations, a Fortran program was developed to handle the general case.

Gladman [92] and Cavé [93] investigations was the basis of Peters [94] study on minimum cost tolerance. Two approaches have been discussed. First, is the description of three known techniques; i.e. worst-case, RSS and mean shift model. The second approach is the consideration of the inverse problem. The object is to find the tolerance to be distributed among the components given the final tolerance of a product or the equivalent tolerance sum.

Peters suggested that the tolerance should be distributed taking into consideration the cost of workpieces and the standard deviation of their production process. This is to ensure a minimum cost including the cost of the fraction rejected by sorting. This comes to the conclusion that the cost of the assembly of two pieces is:

$$K=\frac{C_1}{1-p_1}+\frac{C_2}{1-p_2} \tag{2-21}$$

where: C_i is the cost of a component i and p_i is the rejected fraction.

Assuming the tolerances are at $\pm t$, based on a Gaussian distribution, the accepted fraction is:

$$1-p=\frac{1}{\sqrt{2\pi}}\int_{\mu}^{\mu} e^{-\frac{\mu^2}{2}}\,d\mu \tag{2-22}$$

Introducing equation 2-22 in equation 2-21 and setting the derivative equal to zero, the cost to assemble two pieces of assembly will therefore be:

$$\frac{C_1 e^{-\frac{\mu_1^2}{2}}}{(1-p_1)^2}=\frac{C_2 e^{-\frac{\mu_2^2}{2}}}{(1-p_2)^2}\frac{d\mu_2}{d\mu_1} \tag{2-23}$$

Sfantsikopoulos [95] introduced an approach to optimise tolerances based on the relationship between the size of a particular dimension, its specified tolerance zone and the related manufacturing cost.

The cost-tolerance relationships took the form:

$$C(D,t)=C(D)+C''_0\cdot\frac{i(D)^{r-1}}{t^r} \tag{2-24}$$

where: $C(D,t)$ is the manufacturing cost of dimension D with tolerance zone $\pm t$; $C'(D)$ is the manufacturing cost of dimension D with commercial accuracy; C''_0 is the accuracy cost constant; $i(D)$ is the ISO standard tolerance factor and r is the cost sensitivity to a tolerance exponent.

Lee and Woo [96] optimised tolerance synthesis by treating cost minimisation as the objective function and the stack-up condition as the constraint. Each dimension is assumed to follow the normal distribution. Since errors in the processing are due to small independent sources such as operators, materials, machine, etc., tolerances are then determined by standard deviation and a confidence coefficient.

CHAPTER 3

TOLERANCE REPRESENTATION IN CAD

3.1 Introduction

McGoldrick [33] stated,

> *"There are no CAD systems currently being marketed in the UK which incorporate any sort of tolerance model".*

This can be interpreted into two facets of tolerance modelling which are:

(a) models that can optimise working tolerance on a part; or

(b) tolerance representation models in CAD.

The first model has been reviewed and explained in Chapter 2. Regarding tolerance representation in CAD, this involves a different set of investigations. These investigations relate to the matter of representing, manipulating and analysing dimension and tolerance data specifically in CAD. The use of solid models and variational geometry and its implications for the successful integration of CAD and CAM are discussed in this area of investigation.

3.2 Computer-Aided Design (CAD)

CAD is defined as, *any design activity that involves the effective use of the computer to create or modify an engineering design* [99]. It has the ability to speed up change. Furthermore, it can overcome the dilemma of constant amendments due to the intricacy of design and redesign cause by feedback variation from many scattered users.

The only reference to CAD in the British Standards [100] catalogue is concerned with graphic presentation rather than with the structure of CAD models. Increasingly, engineering drawings are being generated using computer-based tools rather than the drawing board. The most common CAD systems are two dimensional draughting packages. Other packages available include three-dimensional wireframe systems, surface modellers and solid modellers. These enable the three aspects of engineering drawings, namely shape, dimensions and tolerances and other information to be generated.

3.2.1 Shape

In two-dimensional (2D) draughting systems, methods of construction based on manual techniques are provided as system functions. For example, a line may be constructed to join two points or a circle constructed with a given radius and tangent to two lines. Simple methods of creating more complex curves are often included. Ellipses are common, parabolae and hyperbolae are rare, but more general curves fitting through a sequence of points, using bi-arcs for example are useful enhancements. Woodwark [101] describes how the basic shapes are represented as points, lines and circular arcs within typical draughting packages. Other common facilities enable sections to be drawn with crosshatching, symbols to be created and patterns and other symmetry to be used.

Shapes can also be created which are beyond the limits of those usually associated with traditional drawing board techniques. These include offset paths for numerically controlled (NC) machine tools and three dimensional (3D) representations such as wireframes, surface models and solid models. Woodwark in his book also discusses mathematically defined curves and surfaces that include interpolation, Bézier and B-splines. Other characteristics of computer-based geometry systems are the ability to define the geometry parametrically. Instead of a point being defined numerically as (5.5, 10.5), it may be defined parametrically as ($x1$, $y1$) where $x1 = 5.5$ and $y1 = 10.5$. This results in greater flexibility, a complex modification being effected by simply editing the numerical value of the parametric concerned.

3.2.2 Dimensions and Tolerances

Within draughting packages, dimensions and tolerances are often constructed using similar methods to the manual techniques. The points to which the dimensions relate are selected and the position and style of the dimension are chosen. Tolerances are then added to the dimensions. Geometrical tolerances are attached in the same manner. DIAD 2D [102] in its datasheet says,

> *"Comprehensive dimensioning and tolerance facilities to BS 308, DIN, AFNOR or ANSI standards. ... Geometric Tolerancing symbols supplied".*

Since their first launch in 1982, the AutoCAD [103] suite of tools for 2D and 3D draughting and designs have held a preeminent place in the market; they claim over 400,000 installations worldwide and availability in 12 languages. Under the headline *AutoCAD Technical Specification* with the heading of *Dimensioning*, is printed:

> *"Multiple standards support (including global editing and update), linear, angular, radial, ordinate, oblique, baseline and continuous dimensions (all with full associativity). Text notes and labels, dynamic editing of dimensions and text".*

These are only two of the many CAD systems available in the market explaining their ability for dimensioning and tolerancing.

In many systems the values of the dimensions and tolerances must be defined when they are created. More sophisticated systems have *associative dimensioning* that enables the dimensions to be related to the geometric shape entities [103,104]. The value of the dimension is usually evaluated from the host geometry, although it can often be overwritten if required. The advantage of this approach is that when the geometry is changed, the dimensions will remain associated with the relevant pieces of geometry and carry the correct values.

A *dimension driven geometry*, which is the reverse approach to associative dimensioning, allows geometry to be edited using the dimensions. Parametric Technologies, Pro/ENGINEER [105] offers this facility.

Nevertheless, whichever approach is used, the dimensions and tolerances within these systems are used for graphical output and do not support applications such as process planning or tolerance analysis.

3.2.3 Other Information

Current computer-based systems mimic manual methods in their representation of the other technical and nontechnical information found on drawings. The names, numbers and dates usually exist only as strings of text positioned somewhere on the drawing. The only relationship between this information and the shape and the dimensions and tolerances are that it resides in the same file.

Some more advanced systems are beginning to hold it in a structured database. This enables the computer-based management systems, and also the draughting system, to make use of it. Several add-ons are available for AutoCAD, and examples are in the literature [106,107]. The limitation of these systems is that they can normally manage the data definitions only within the single suite of software.

3.3 Model Representation

An object modelled in CAD can be of one of the following three forms:

(a) wireframe which can only define edges;

(b) surface that can define edges and surfaces; and

(c) solid modelling that can define surfaces and edges and distinguishs between space and solid material.

3.3.1 Wireframe Model

Wireframe system uses the geometric entities, i.e. lines, circles, arcs and curves and defined the locations in space either X, Y and Z coordinates for 3D model or X and Y coordinate for 2D.

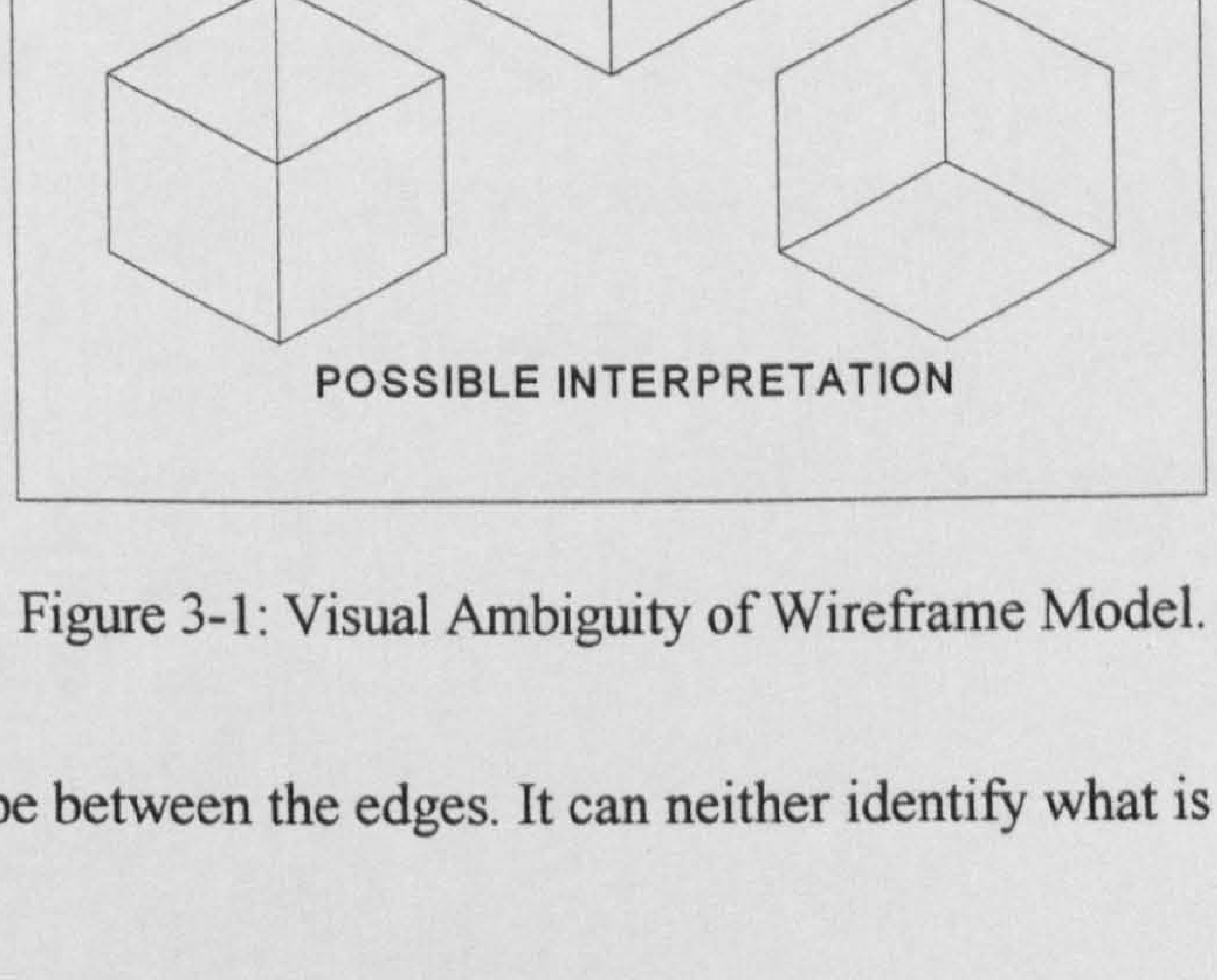

Figure 3-1: Visual Ambiguity of Wireframe Model.

In other words a wireframe model can only define the edges of a component and has no knowledge of the surface shape between the edges. It can neither identify what is solid and what is not. Thus, this leads to some weaknesses of the wireframe system. Firstly, the model will be visually ambiguous as shown in figure 3-1.

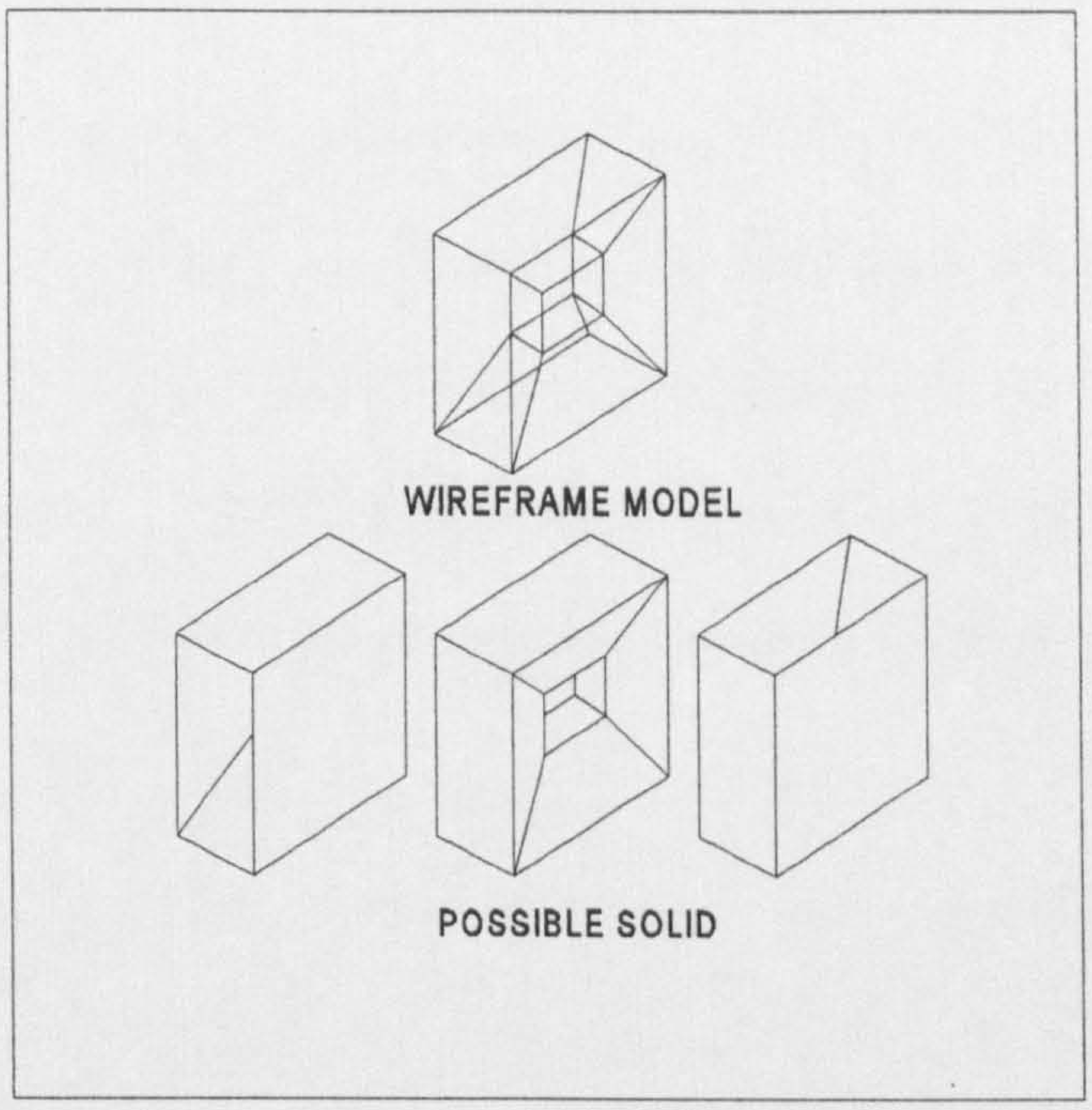

Figure 3-2: Incomplete Definition for Solid Interpretation

Secondly and more seriously, the same wireframe model in some instances, may represent more than one solid, figure 3-2.

3.3.2 Surface Model

The objective of surface modelling is to completely define the surface form of an object so that the computer can calculate accurately the X, Y and Z coordinates of any point on the surface. There are two approaches to do the calculation in which:

(a) to start with a 3D wireframe model and then add the definition of the surfaces between the edges of the model; and

(b) to take measured coordinates information from a physical model and then try to define surfaces that fit the data points.

Most of the surface models work on a patching system of four-sided surfaces positioned edge to edge. Once the basic model is complete, adjacent surface patches may be blended or concatenated to remove sharp edges. They may also be filleted with circular or elliptical fillets or offset to define the through thickness form of a component. An advantage of the surface model is that the surface form of the component is completely and unambiguously defined. Thus, it can be used directly to program NC machine tools to manufacture the form accurately.

3.3.3 Solid Model

Like surface modelling, solid modelling defines the edges and the surface form of the component. The difference is that, a solid modeller is also able to tell where there is solid material. Hence, it can decide whether a given point is in space, on the surface of the component or in solid material.

There are six basic representation schemes that give unambiguous representations of solid bodies, although only two of these are mainly used today [99]:

(a) *Pure Primitive Instancing Scheme* - It is based on the group technology concepts where there are families of basic objects. Each family is termed a generic primitive, for example, blocks, spheres, cones, wedges, cylinders and tori. Individual objects within a family are termed primitive instances. Thus, a particular object is represented by its family name and its particular dimensions.

(b) *Spatial Occupancy Enumeration* - It is a list of cubes of fixed size called spatial cells occupied by the solid. Each cell is usually represented by a single coordinate and the list of cell coordinates which make up the solid body is called a spatial array.

(c) *Cell Decomposition* - It represents a solid object by decomposing it into several small cells. Spatial occupancy enumeration is a special case of this, where all of the cells lie in a fixed grid and are cubical.

(d) *Sweep Representation Schemes* - In this representation scheme a solid is described not as a generic primitive but by a two dimensional cross section. This cross section is then swept along a trajectory that is usually a translation sweep or a rotational sweep to form a solid body.

(e) *Constructive Solid Geometry* (CSG) - CSG represents a solid object by a *tree* of set or Boolean operations. The CSG tree consists of primitive leaves that represent the solid primitive instances such as blocks, spheres, cones, wedges, cylinders and tori. The user then combines these primitives' shapes to create a subtree by specifying a Boolean operation to be performed. The first is a union operation that adds two primitives to create a new solid body. A difference operation will subtract one primitive from another and an intersection operation calculates the common volume of two primitives.

(f) *Boundary Representation Scheme* (B-rep) - A solid body is represented by dividing its boundary into several subsets called *faces*. Each face is then represented by its bounded edges and vertices. Thus, the solid object created by B-rep is represented by two types of linked data. First, is geometric data which defines the faces, edges and vertices of the object. Second is topological data which describes how the faces, edges and vertices are linked together.

CSG and B-rep are used mainly today compared to the rest of the representation methodsl. *Primitive instances* are of little use by themselves as they only allow the primitives to be defined. It is most useful when the schemes are combined with other representations such as CSG. *Spatial enumeration* was reported to be costly in terms of processing time and the representation of any curved surfaces was poor [99]. *Sweep representation* is now generally used only to input geometric information, and is often combined with other schemes similar to *primitive instances*.

3.4 Dimension Representation

Apart from systems specifically marketed as draughting packages, few systems attempt to include dimensions or tolerances; the geometry represents a nominal object. Nominal geometry, whether in 2D or 3D, wire frame or solid are defined using *control parameters*.

The control parameters may be either the numerical values used to define the geometry directly or the parameters as used by parametric systems. The most significant problem to be addressed in associating dimensions with geometry is establishing relationships between the dimensions and the control parameters. A model containing dimensions, D, control parameters, P, and geometric elements, G, can be represented as:

$$D = \{d_1, d_2, \ldots, d_n\}$$

$$P = \{p_1, p_2, \ldots, p_n\}$$

$$G = \{g_1, g_2, \ldots, g_n\}$$

Requicha [108] identifies two approaches in relation to these entities namely,

(a) direct parametrization; and

(b) indirect or inverse parametrization.

3.4.1 Direct Parametrization

The geometry is defined by the control parameters as a function of the dimensions:

$$P = f_1 (D)$$

$$\text{and } G = f_2 (P) \text{ or } G = f_3 (D)$$

Advantages:

(a) The geometry is properly constrained so that any alteration to a dimension *d* will have the desired effect on the geometry. This is *dimension driven geometry.*

(b) The control parameter's P can be the natural parameter of the model. Examples include the size of a block or vertex coordinates.

(c) The dimensions become the true control parameters and therefore form a part of the geometric description. If the geometry is completely defined by dimensions, then the set of dimensions must be complete. If all the dimensions are used to define the geometry then they must also be non-redundant.

(d) Tangencies and other constructions can be defined using algebraic expressions to ensure correct behaviour even when the dimension values are altered.

Disadvantage :

(a) The form of the equations does not define which geometric elements are related by each dimension. The inverse function, $D = f_3^{-1}(G)$ is required and in the general case, is not easy to determine. This is because the geometric elements related by the dimensions are not known, the approach is not able to support applications.

3.4.2 Inverse Parametrization

The geometry is defined using control parameters. The dimensions are defined as functions of the geometry:

$$G = h_1(P)$$

$$\text{and } D = h_2(G) \text{ or } D = h_3(P)$$

Advantage:

(a) The geometric elements related by each dimension are defined explicitly making it suitable for supporting applications such as drawing annotations with *associative dimensioning*.

Disadvantages:

(a) The dimensions can only be changed indirectly by altering the geometry control parameters. To rectify this, the inverse $P = h_3^{-1}(D)$ is required which is also not easy to determine for the general case.

(b) It is not possible to check whether the set of dimensions is complete.

3.5 Tolerance Representation

The ability of CAD to associate dimensions with geometry has already been discussed. In some of the CAD systems, tolerances can be attributed to the dimensions but how this tolerance information is used or what it means is often far from clear. A simplistic approach, using direct parametrization, is to allow the parameters of the model, which represent the engineering dimensions to vary within the tolerance range. This results in a set of models comprising every combination of every dimensional value within its permitted variance range. This is unsatisfactory because not only does it produce an infinite number of possible configurations but also all instances are of perfect form. The approach can be used for testing components and assemblies at Maximum Materials Condition (MMC) which exhibit perfect form. MMC defines a finite but often large set of instances only one of which can be represented by a geometric modeller at any one time. Geometric modelling capabilities would enable interference and clearance checks to be made within these limitations.

Requicha [108,109,110] identifies some of the issues in tolerancing. His premise is that components need to have some concept of being *in spec*, by which he means being functionally equivalent and interchangeable in assembly. The latter, although desirable, is not always attainable in practice. Requicha believes that conventional tolerancing is inherently ambiguous and although modern tolerancing is easy to formulate there are still some gaps. He also floats some ideas on tolerance representations based on variational classes.

The theory uses a three plane master datum system and a concept of an offset solid that he uses to illustrate control of size, surface form and orientation, curve forms and MMC. Although some principles are known to ensure validity, for example, the nominal representation must be valid and the union of all the nominal features must equal the nominal object's boundary, the conditions to ensure that there are sufficient constraints are not known.

3.6 Review on Dimension and Tolerance Representation

The solid model based investigation of dimension and tolerance was initiated by Requicha [111]. He started the investigation by relating several representational issues of dimension and tolerance in a CSG based solid modeller, Parts and Assembly Description Languages I (PADL-1) [112]. The object was built procedurally incorporating the dimensional values into the definition of the object itself, thus, producing the dimensioned drawing from this information. Following this work several investigators have explored this field of dimension and tolerance and studied several aspects of its implications for the successful integration of CAD and Computer Aided Manufacturing (CAM). Minagawa, Okino and Kakazu [113] reported the success of a fully automatic dimensioning system based on a CSG based solid modeller, *TIPS-1* and *AUTDIM*. TIPS-1 and AUTDIM operate based on the sequence of:

(a) recognition of the geometric pattern from the object (CSG model) data structure;

(b) extraction of the dimensions of each feature (depending on the types of primitives);

(c) the connection of the extracted dimensions to each other so that a dimension chain may be formed;

(d) verification of the consistency of the obtained dimensions, and

(e) graphical visualisation or representation of the output.

Yuen, Tan and Yu [114] have also presented a general scheme for the automatic dimensioning of objects from their boundary representation. Its application on the CSG based modeller

PADL2 has been reported. Dimensions are defined by distances and angular relationships between a pair of entities (e.g. points, lines and surfaces). To represent dimensioning adequately for the whole object, a dimension tree is constructed so that all the boundaries' surfaces of the object are present in the tree. Under or over-dimensioning is thus easily detectable from the dimensioning tree. It is reported that the current application does not fully agree with engineering drawing standards and that the automatically generated dimensioning link may require manual modification.

Requicha [108,109] developed a theory based on the *variational class* concept. Variational classes are families of objects that are similar to a nominal object, interchangeable in assembly and are functionally equivalent. By his definition, an object is considered to be in tolerance if its features' boundaries lie within the specified range of the tolerance zone as shown in figure 3-3.

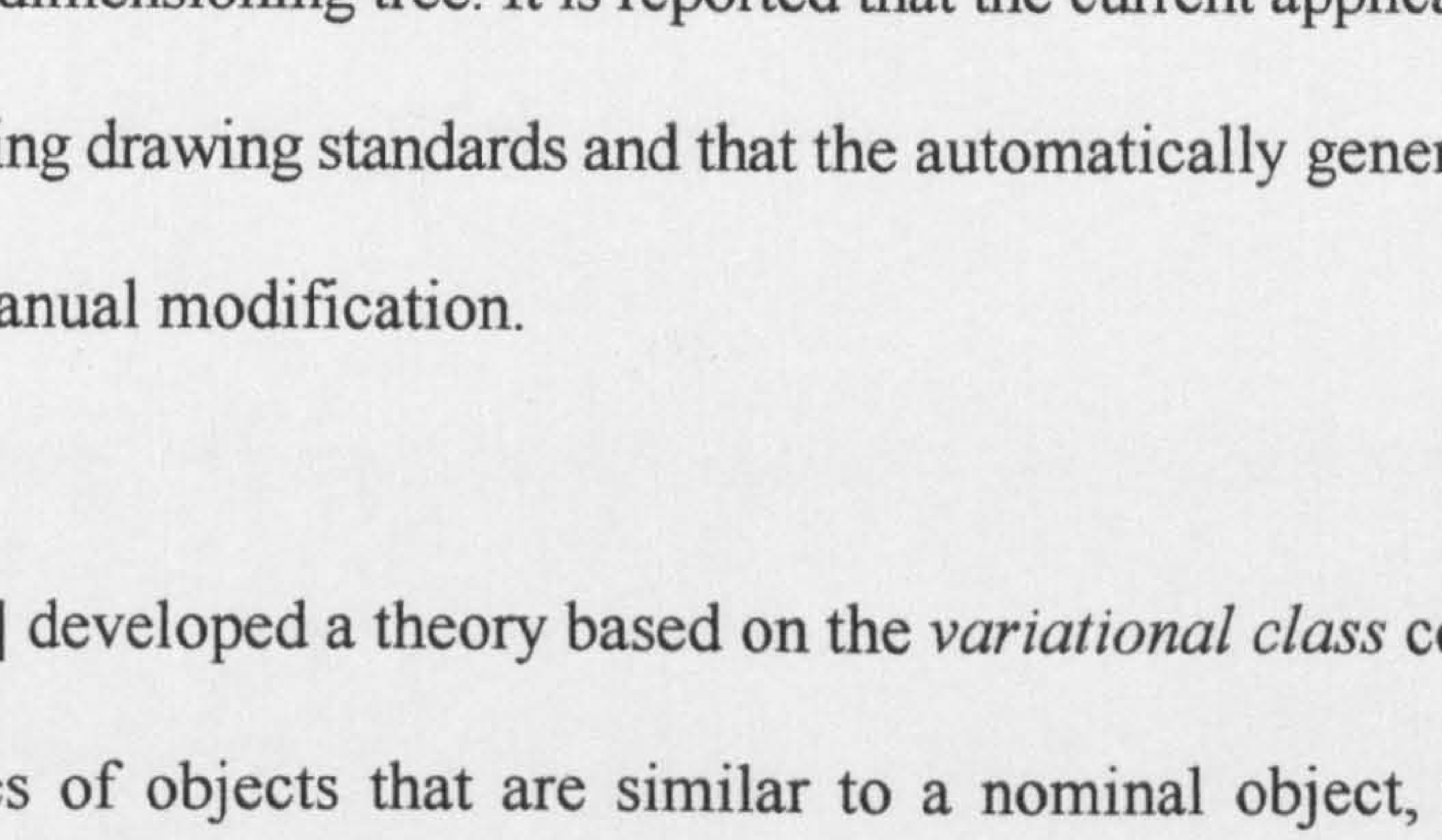

Figure 3-3: Tolerance Zone

The tolerance zone is defined over a domain of feasible region constructed by *offsetting*, i.e. expanding or contracting for plus/minus tolerances, the part's nominal boundaries. The tolerance information is specified of an object boundary surface feature set of geometric attributes and it dictates the offsetting criteria for the boundary surfaces.

Rossignac [115] discussed at length the offsetting operations in an attempt to combine them with other Boolean and rigid motion operations in an extended CSG scheme. Farmer and Gladman [116], nevertheless, commented that this differed in some respects from the ISO system. Moreover as the handling of dimensions and tolerances in the general case requires the ability to access the bounded entities of objects, the CSG based tolerance theory of Requicha raises some manipulation problems during implementation.

Requicha and Chan [117] look in more detail at a theory of tolerancing, which treats tolerances as properties or attributes of an object's features and other variational information in a CSG based modeller, particularly at some implementational issues. The variational information is associated with the solid model by means of a graph, called a Variational graph or VGraph. They describe VGraph which is observed by Pratt [118] to be a move to B-rep.

Requicha says that it could be attached to a B-rep, but describes the VGraph as a facility for CSG based modellers. It consists of *VFaces* (Variational Faces), *NFaces* (Nominal Faces), *SFeats* (Surface Features), *VEdges* (Variational Edges), *CFeats* (Curve Features), *Attributes* and *DatSys* (a datum system consisting of an ordered set of datums -usually three). The system uses the idea of zones described by offset surfaces and constraints divided into intrinsic (size and form) and extrinsic (position, orientation and run-out).

Jayaraman and Srinivasan [119,120] have examined the issues of representing the geometric tolerances in solid models from the perspective of functional requirements related to the geometry of mechanical parts. Their investigation is mainly concerned with the positioning of parts with respect to each other in an assembly and with maintaining material bulk in critical portions of parts. They develop specific virtual boundary requirements' to reflect the required functional conditions of the assembly. Discussion was made on the theoretical basis of the interpretation of those virtual boundary requirements. This discussion was assisted by the theory of solid model based offsetting as proposed by Rossignac and Requicha.

Elgabry [121] addressed a framework for representing and analysing solid based tolerances. His framework contains a practical set of geometric entities and minimizes the changes needed to the underlying CSG modeller. Elgabry uses CSG primitives in the approach to create critical dimensions. He uses combinations of parameter values, surface offsets and volume sweeping to generate models of MMC or LMC objects for analysis. Apart from querying the meaning of the tolerances, Elgabry raises some implementational issues such as the problem of modifying the tolerance definitions if the geometry is modified and how tolerance data can be associated with a boundary file generated from a CSG modeller.

In GEOTOL, Turner [122] has attempted to associate the tolerancing information with the evaluated boundary representation of the part. All variations are applied to the part faces of the nominal model which is limited to the planar and cylindrical faces only. Because of the parametric approach taken, the derived representation cannot be edited to change the original geometry. A prototype representational module has been built to provide IBM's Geometric Design Processor solid modelling system with a generalised CSG architecture.

Extending GEOTOL ability, Turner [123] introduced a feasibility space approach. Once a variational model has been identified by associating model variables with each surface of the nominal model, a Cartesian space is defined in which n-tuple provides a value of each n

model variables. Each point of this Cartesian space corresponds to a particular instance of the variational model. If a collection of tolerances has been associated with the model, then each of these tolerances can be interpreted as some constraints on this Cartesian space. The tolerance specifies which of the n-tuples of this space correspond to part instances that are in-tolerance. Collectively the tolerances define a feasible region of the space. Each point within this feasible region corresponds to a part instance that satisfies all the tolerances.

Roy and Liu [124] showed the necessity of having a hybrid CSG/B-rep data structure for the tolerance representation so that the advantages of both CSG and B-rep models can be exploited. The tolerance module is attached at the top of this hybrid structure. The user interacts with the solid model at each hierarchical level of object construction for associating tolerance and other technological information (such as material data, surface roughness, etc.) rather than waiting until the entire part geometry has been defined. This investigation is carried out on the Sun Workstation based on the polyhedral B-rep TWIN solid model.

Mullins and Anderson [125] in an approach to tolerance representation used volumetric feature geometry created and represented using an object-oriented design framework. A hybrid CSG/B-rep modelling system was also used in which a B-rep of the entire part is formed from the feature. The final B-rep is an analysis tool for the extraction of information from the model. In the model, dimensions and tolerances of position, orientation and size are represented using a tolerance vector approach. This allows the features to encapsulate the necessary information in a compact and computational manner.

Lu and Wilhelm [126,127] favoured the virtual boundary requirements (VBR) approach of Jayaraman and Srinivasan. Their CASCADE-T extends the VBR approach to provide tolerance primitives that allow conditional tolerances to be used by designers while specifying tolerance relationships. Thus, the conditional tolerance relationships may be derived to link functional requirements with geometric tolerances. Initially in VBR, conditional tolerances are descriptions of geometric surfaces and relationships between surfaces that will provide the measurement of conformance. CASCADE-T also extends the work of Requicha and Chan. A facility for specifying and storing tolerance information was provided. Using tolerance primitives, conditional tolerance equations are generated according to each primitive and its associated attributes for the solid.

It should be noted that dimension and tolerance representation in CAD is feature-based. This is to enable geometry information be identified and stored in a CAD database. Proper identification of geometric features is required to be able to use the information stored effectively.

Two kinds of features are involved:

(a) lower level features, such as points, lines, arcs, splines and surfaces; and

(b) higher level features, which are combinations of the lower level features such as holes, slots, pockets, countersink or complex features that maintain certain relationships among themselves.

Lower level features are basic topological entities, whereas higher level features are design specific and the choice for their selection depends on the function and the context of application. For the dimension and tolerance application, this feature information must be extracted or recreated from the solid model.

3.7 Feature-based Modelling

A feature is defined by Gandhi as *"a geometric entity that defines the attributes of a part's nominal size and shape"* [128]. Requicha and Vandenbrande define a feature as a primitive in a high-level language or representation scheme for defining mechanical parts [129]. Henderson and Anderson describe that sometimes features are thought of as volumes to be removed by machining operations [130]. A recent definition of a feature is given by Dixon et al. [131] is; *"a feature is any named entity with attributes of both form and functions"*. In general, many investigators in this field accept that features are abstract entities that relate to form and function.

There are two approaches to features:

(a) *Pre-defined features* - These are selected in the construction of a product and form part of its definition. This is a parametric approach to features.

(b) *Recognized features* - The definition does not include the features but are recognised in the geometric definition or an evaluation of it. This is an inverse parametric approach.

The interaction of design features, primitive features, geometric primitives and dimensions and tolerances are complex. Ranyak and Fridshal [132] have a hierarchy starting with a design feature, moving down to a primitive feature and at the bottom of the hierarchy is the geometric primitive. This is restrictive in that the relationship between primitive features and the entities related by dimensions and tolerances, is forced to conform to the relationships defined in the design feature. For example, an open pocket feature defined by the width and depth cannot be constrained by relating the sides to other entities external to the slot, rather than to each other, unless a separate slot definition is created for each possible method of constraint.

Roy and Liu [124] propose a feature-based representational scheme for dimensions and tolerances. Although the influence of earlier research can be seen, there are some distinctive ideas. Features are central and are identified as two types: Low level features that include points, lines, arcs, splines and surfaces and are equivalent to Ranyak's primitive features and higher level features that are combinations of lower level ones which equate to Ranyak's design features. The scheme requires several graph-based representations to be maintained. These include a CSG type definition for geometry, a Structured Face Adjacency Graph (SFAG) which includes features for the topology. Within the SFAG is the Face Adjacency (FAG), a Feature Tree (FT) a Spatial Relationship Graph (SRG) and a datum dependency graph.

The modelling system is described as a CSG/B-rep hybrid with the FAG being similar to a B-rep except it is face-based rather than edge-based. The system includes data structures for dimensions, dimensional tolerances and geometrical tolerances. It also allows for other technological information such as threads, knurls and surface texture. Although the model is fairly comprehensive and addresses some important issues such as the need for relationships in a face-based system, it is limited in its modelling capabilities and requires a great deal of user interaction to drive it.

Each feature is built in a Datum Reference Frame (DRF) which requires the user to identify three coordinate planes, axes or features. The system must be used in a particular way; first the structural representation of the model is formed, then the attributes are applied. It is not clear whether the model can be edited at a later stage without the possibility of both parametrized and inverse parametrized representations. The tolerance information is attributed only and no geometrical tolerance representation is defined.

CHAPTER 4

INTELLIGENT CAD

4.1 Introduction

Existing CAD systems are mostly geometric in nature. Their databases consist of connected geometric entities which constitute the object model. CAD is unable to represent the functionality of the object model nor infer the knowledge used by the designer.

Intelligent CAD (ICAD) systems, on the other hand, have knowledge processing capabilities such as inference, knowledge-based management or search mechanisms (or heuristics) [133]. The intelligence in CAD is credited to the growing investigations in Artificial Intelligence (AI) in which it has now become one of AI's subfields.

Earlier attempts to apply expert system technology into CAD systems were mainly directed towards solving specific and often very limited domains of engineering design problems as described by Smithers [134]. The vast majority of the development is directed to or based on the area of electronic circuit design [135]. To have a better grasp of the subject, before reviewing the status of ICAD investigations, the chapter commences with a general overview of the field of AI especially the subfield of Expert Systems. Subsequently, ICAD status will be reviewed and the feasibility of this intelligent tool analysed.

4.2 Artificial Intelligence

The inspirational origins of AI are often traced back to the work of the distinguished British mathematician, Alan Turing, and in particular to his paper *"Computing machinery and* *intelligence"* [136]. AI as a term, did not emerge until 1956 when John McCarthy proposed a conference on making an intelligent machine in that year. Since then many systems have been developed and it seems unfair to pick out a few as representative examples. Appendix A1 will provide an overview of the range of activity, however, it hardly represents an exhaustive list. The list quoted can be found in the literature [27,28,29,30]. AI can be defined as;

> *"the part of computer science concerned with designing intelligent computer systems, that is, systems that exhibit the characteristics we associate with intelligence in human behaviour - understanding language, learning, reasoning, solving problems, and so on."*. [27]

AI programs traditionally demanded large computers. In addition special list processing languages (primarily LISP, PROLOG, POPLOG and POP-II) and programming environments designed to make routine parts of programming easier and less time consuming are also required. However, a new trend of developing AI programs (or packages) that use conventional computational languages, such as C and run on smaller computers and Personal Computer's is emerging.

Expert Systems is a term that is often used synonymously with AI. In actual fact the technology is a branch of study and application in AI. An expert system is:

> *"an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solutions"*[137]

"a class of computer programs that can advise, analyse, categorise, communicate, consult, design, diagnose, explain, explore, forecast, form concepts, identify, interpret, justify, learn, manage, monitor, plan, present, retrieve, schedule, test and tutor" [138]

"problem solving programs that solve substantial problems generally conceded as being difficult and requiring expertise. They are called Knowledge-based because their performance depends critically on the use of facts and heuristics used by experts" [139].

This definition illustrated the used of knowledge-based to enhance the analysis performance which is related to the premise of this investigation.

4.3 Expert Systems

Another name used synonymously with expert systems is Intelligent Knowledge-based Systems (IKBS). This synonym explains the fundamental features of an expert system, which contains a large amount of knowledge or expertise specific to a particular topic - the so called 'knowledge domain'. Black [140] nevertheless, suggested that IKBS and expert systems are two different entities. According to Black, the distinction between IKBS and expert systems is based on the level of competence of the system, the area of application and the mode of operation. In general, however, this notion is not supported by other investigators [141,142,143,144].

The expert system approach in its intelligent manner departs from the deterministic approach of conventional programming. The differentiating features are represented in table 4-1.

EXPERT SYSTEMS	CONVENTIONAL PROGRAMMING
Within their chosen field they can demonstrate expert capabilities.	It is deterministic in which it follows a predetermined sequence for every problem it must solve.
Incorporate *rules of thumb* (heuristics).	It is constructed primarily of linear relationships.
Can handle uncertainty (most conventional programs will not accept answers such as `don't know' or `80% sure' to question)	The typical objectives are to compute values, store constants and retrieve data.
Can provide the user with explanations for the advice that they offer.	It deals with universally accepted processes, for example, value of log or the product of 1x1.
Programmed in a declarative style, usually by means of rules.	It follows established mathematical rules.

Table 4-1: Expert Systems and Conventional Programming Features

4.3.1 The Structure of an Expert System

A human expert uses knowledge and reasoning to arrive at conclusions. Similarly an expert system relies on knowledge and performs reasoning. The reasoning carried out in an expert system attempts to mimic human experts in combining pieces of knowledge. Thus, the structure or architecture of an expert system partially resembles how a human expert performs. An analogy between human experts and an expert system can be illustrated in figure 4-1.

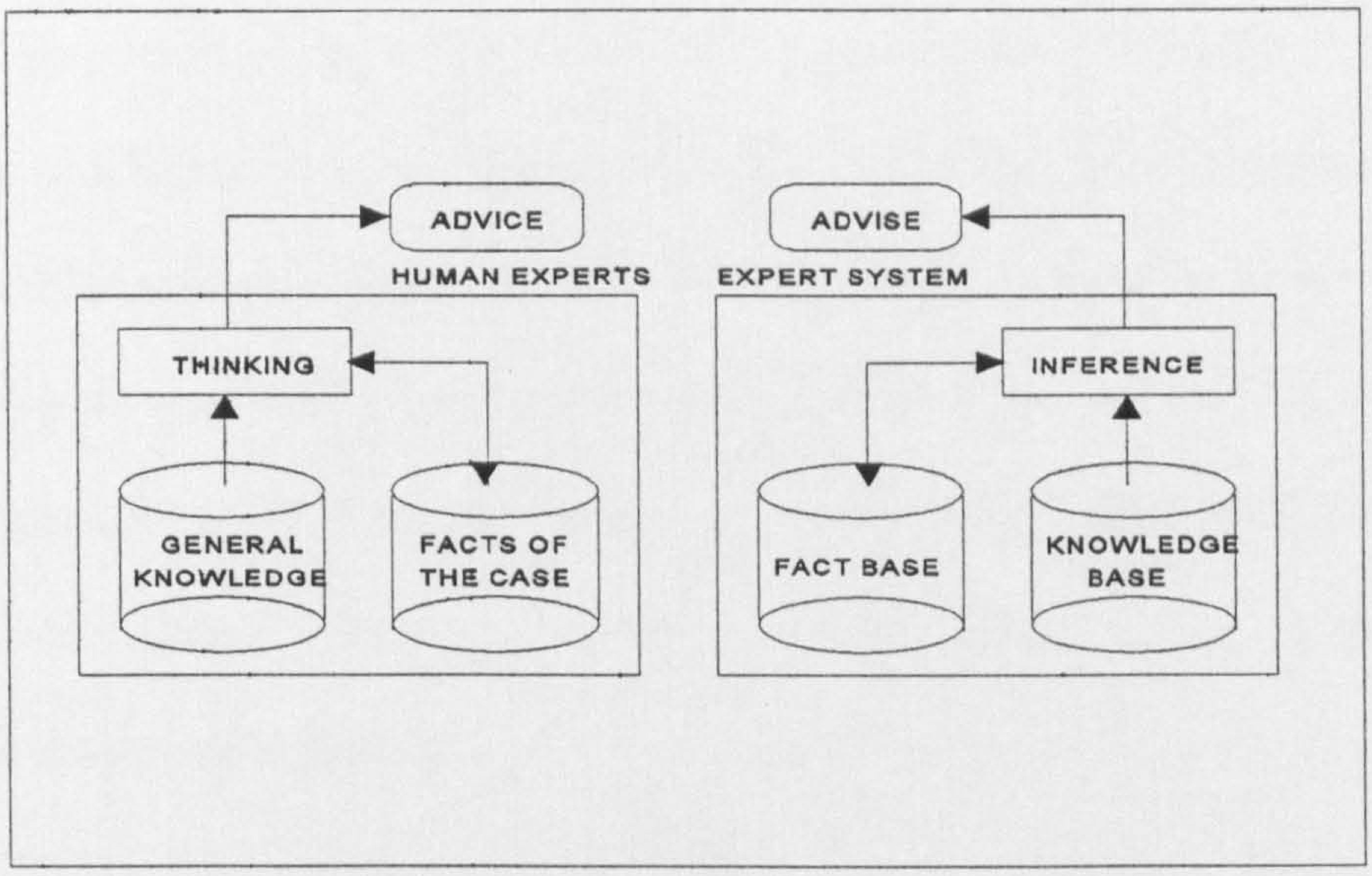

Figure 4-1: Analogy of human experts and expert system.

The first part of human expertise is a long term memory of facts, structures, and rules that represents expert knowledge about the domain's expertise. The analogous structure in an expert system is called the *knowledge base*.

The second part of human expertise is a method of reasoning that can use expert knowledge to solve problems. The part of an expert system that carries out the reasoning function is called the *inference engine*.

Inference engine and knowledge base are the key components of an Expert System. The separation of control (the inference engine) from knowledge (knowledge base) is a hallmark of an expert system. An Expert System derives most of its power from its knowledge rather than its inferencing ability [145,146].

Expert Systems are applied to the class of problems in which no simple algorithmic solution is known. To qualify as an Expert System it must attain levels of performance roughly equivalent to a human expert. Hence, most Expert Systems can reason about their own inferences. Other important features are that, it does not forget, considers all details, does not overlook remote possibilities and does not jump to sudden conclusions. Still it has its own shortcomings such as, lack of common sense, can be slow compared to humans, and not good at approximate pattern matching [147,148]. A further breakdown of an expert system structure is illustrated in figure 4-2.

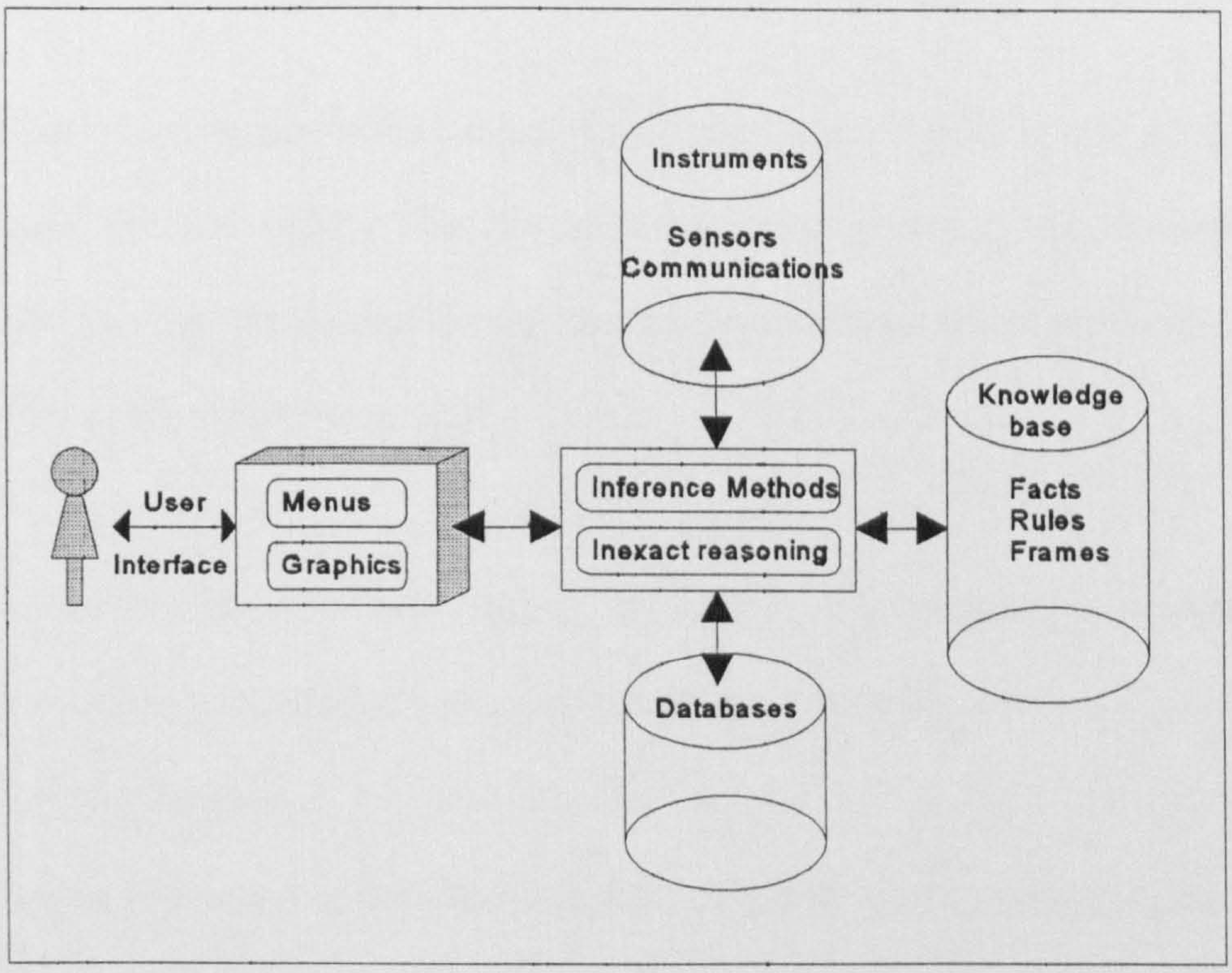

Figure 4-2: Expert System Structure.

4.3.2 Knowledge base

The knowledge base is one of the two critical parts of an expert system, the other is the inference engine. It is the portion of an expert system that consists of the *rules* and the *facts* of a particular domain of knowledge [146].

Rules in the knowledge base give a picture of how a human expert would tackle the problem in his expert domain. It is also used to represent certain scientific formulae or specific pattern that can be deduced from any standard data or graphical representation of certain empirical relationships. For example, a rule can be written as:

IF *conditions* THEN *action*

This is important since the inference mechanism actually compares or tries to match the condition of the rules against the facts in the knowledge base. The rules set in the knowledge base can be regarded as an open-ended set in which the extension of the set is achieved by merely adding new rules.

As new rules are added especially when the knowledge base becomes considerably large, the integrity of the rule set must be considered. The following possible conditions that can happen are to be avoided:

(a) *Infinite chaining* - occurs when sub goals are set up to prove an initial goal; and

(b) *Contradictions between two or more rules* - for systems with a very large rule base.

A means of overcoming such problems is to introduce *Metarules*. A metarule is a rule that controls or manages other rules, by which it operates as an added means of controlling the search process in an expert system. A metarule may decide such things as ruling out problem solutions that probably will not work, or remembering and reusing successful strategies.

An example of metarule might be:

IF	**machine cylindrical parts**;
AND	**there are rules on types of materials**;
AND	**there are rules of D&T required**;
AND	**there are rules on type of tool used**;
THEN	**use the capable machine selection rules**.

4.3.3 Inference Engine

The inference engine is sometimes called the control structure, the rule interpreter or the interpreter. It is that part of the expert system that is responsible for inference and control [137,145,146]. Using rules in the knowledge base, it makes inferences by deriving new facts from old facts.

The types of inference can include:

(a) *modus ponens* - a rule of logic in which if X implies Y and X is true then Y is true [27,137,142,146,148];

(b) *universal instantiation* - the principle is that, given a universally quantified variable in a true sentence, then any substitution by any appropriate term in the domain will produce a true sentence [148,149];

(c) *uncertainty reasoning* - also known as inexact reasoning, under which it is a version of logic that uses multivalued statements. Statements in logic generally are either true or false whereas in this type of reasoning it is either true, false or unknown. Bayesian statistics, fuzzy logic and uncertainty factors are means of applying inexact reasoning [27,150,151]; and

(d) *resolution* - used to determine the truth of an assertion in logic systems. The most common version of resolution is resolution by negation in which the axioms of a theorem and the negation of the theorem must produce a contradiction if the theorem is true. The resolution principle is summarized as follows: *(A or B)* and *(Not A or C)* implies *(B or C)*. Thus, *(B or C)* is the resolvent [27,142,144,146,148,149,150].

The inference engine controls the order in which rules are examined. It selects the rule to start with and the next rule to check. Then, decides when a conclusion has been found, when to ask the user and when to give up.

Different aspects of control can include the choice of backward and forward chaining.

(a) *Forward chaining* - it is a chain of inferences that start from an initial state and moves to a goal state. The initials set of facts are compared to the condition elements of the rules in the rule base. When matches are found, the eligible rules are fired, adding new facts. The new facts can in turn fire more rules. This process is continued until no more rules can be executed. Forward chaining is synonymous with forward reasoning, data driven search, bottom-up processing and antecedent reasoning [137,145,146,148,149].

An example of Forward Chaining is:

Assuming **B** and **E** are true:

IF E AND C THEN F	rule ignored since not all IF parts are known;
IF A THEN C	rule ignored since IF part not known;
IF B THEN D	rule is fired, since B is true, therefore D is added to the knowledge base;
IF D THEN A	rule is fired and A is now added to the knowledge base;
IF A THEN C	rule is fired and C is added;
IF E AND C THEN F	rule is fired, no more rules to prove so the inference engine stops.

(b) *Backward chaining* - it is a chain of inferences from a goal state to an initial state. One begins with a hypothesis and attempts to prove the hypothesis by proving the assumptions true that support the hypothesis. In other word, Backward Chaining assumes a goal and verifies the sub goals required. If a sub goals fails, assume the next possible goal and try to verify this. For example; to prove a person has a cold, one must find the presence of the necessary symptoms, i.e. sneezing, fever etc. [137,142,145,146,149].

(c) *Mixed chaining* - one main advantage of Forward Chaining is that it does not require the information be added in any particular order. On the other hand, if the user is entering information then the system can offer no assistance in directing the user to enter the required information. In contrast, Backward Chaining is entirely directed. It asks the user for specific items of information and it can usually justify why it is requesting the information. The problems with it occur when the systems involve heuristics. For example; it is irrelevant to ask the material composition to determine the carbon equivalent once the material grade which has carbon equivalent associated with it is known. In those circumstances where the inference strategy being used cannot allow the disadvantages stated above, an alternative compromise strategy is possible. This involves using backward chaining as the main inference strategy whilst using forward chaining to provide the advantage of data driven inferencing. A drawback of this mixed chaining is that it is possible for the two modes of reasoning to miss meeting each other [145].

4.3.4 Knowledge Acquisition

Since expert systems deal with knowledge and expertise, the problem of how to acquire the required knowledge is among the early issues to be solved in building expert systems. There are two types of knowledge acquisition:

(a) *manual knowledge acquisition* - refers to such procedures as a knowledge engineer interviewing a domain expert, and verbal protocol analysis (i.e., human experts, empirical data, manuals, and alike). Manual knowledge acquisition can also be formally broken down into the stages of identification, conceptualization, formalization, implementation and testing;

(b) *automatic knowledge acquisition* - refers to the process of getting computers to learn from external sources, especially from experts. Two computer-automated techniques used to achieve this goal are computer interviewing of experts and secondly, entering examples into the computer. Rules can then be constructed from the knowledge gained.

4.3.5 User Interface

The communication link between the program and the user is the user interface. At its most basic level, it is only what the user sees on the monitor, however, its importance should not be underestimated. A user interface may include menus, questions, text explanations and natural language. The operation of an expert system is extremely important and a poor design of the User Interface module has been a recurring criticism of interactive computer systems.

A user approaching an expert system is a similar step as approaching a human expert. The expert must choose the right question so that the problem description can be described more precisely. The question must be simple yet explicit so that it can be easily understood. Finally, as with any other computer to human interface, the environment of user friendliness must be installed to the system. Due to the subjective nature of an expert system, this factor should be taken far more seriously.

4.3.6 Knowledge Representation

Expert systems derive their power from knowledge. The heart of any expert system is the knowledge it contains, and it is the effective use of this knowledge that makes its reasoning successful. Two different techniques of knowledge coding are procedural and declarative knowledge.

(a) *Procedural knowledge* - is an outgrowth of traditional programming. It is knowledge that specifies how to solve a given problem rather than describing or specifying the problem and is represented by a step-by-step procedure. As a result, procedural knowledge cannot easily be scrutinised unless one traces through the different program steps. Its outstanding advantage is that it is efficient in terms of execution time.

(b) *Declarative knowledge* - it is knowledge that emphasizes concepts and their relations with other ideas, rather than procedures that manipulate the concepts. In other words, declarative knowledge is a specification of a problem rather than a procedure for solving the problem. The knowledge is explicitly specified, rather than being intertwined in a procedure.

Parsaye and Chignell [145] uses the definition of a grandfather to illustrate a declarative and procedural description of knowledge. Three *declarative* definitions of what a grandfather is:

i. A grandfather of a person is the father of that person's father or the father of that person's mother.

ii. A grandfather of a person is the father of that person's parent.

iii. A grandfather of a person is a male grandparent of that person.

The corresponding *procedural* interpretations are:

i. To find the grandfather of a person find the father of that person's father or the father of that person's mother.

ii. To find the grandfather of a person, find the father of that person's parent.

iii. To find the grandfather of a person, find a male grandparent of that person.

The knowledge used by an expert system needs to be represented and employed in a form that can be used for reasoning. Some inspiration for developing knowledge representation methods has come from observing how humans cope with the problem of representing and organizing knowledge. The earliest explicit attempts at knowledge representation in AI reflected psychological models of human memory. They drew on the analogy between knowledge and natural language to build structures that represented the meaning of words. This approach resulted in the definition of *semantic networks* [152,153].

Approaches to *semantic networks* rely on two fundamental units, namely, *Nodes* which represent objects, concepts or events and *Links* that represent a relation between nodes. Graphically, nodes are drawn as boxes, ovals or circles. Links also called arcs, are drawn as arrows connecting the nodes, as in figure 4-3. Nodes within a semantic network can be instances of other nodes using a special link called the *is-a* link. The *is-a* links provides the concept of inheritance. Inheritance is the process of making the information stored in a high-level concept available to lower-level instances of that concept.

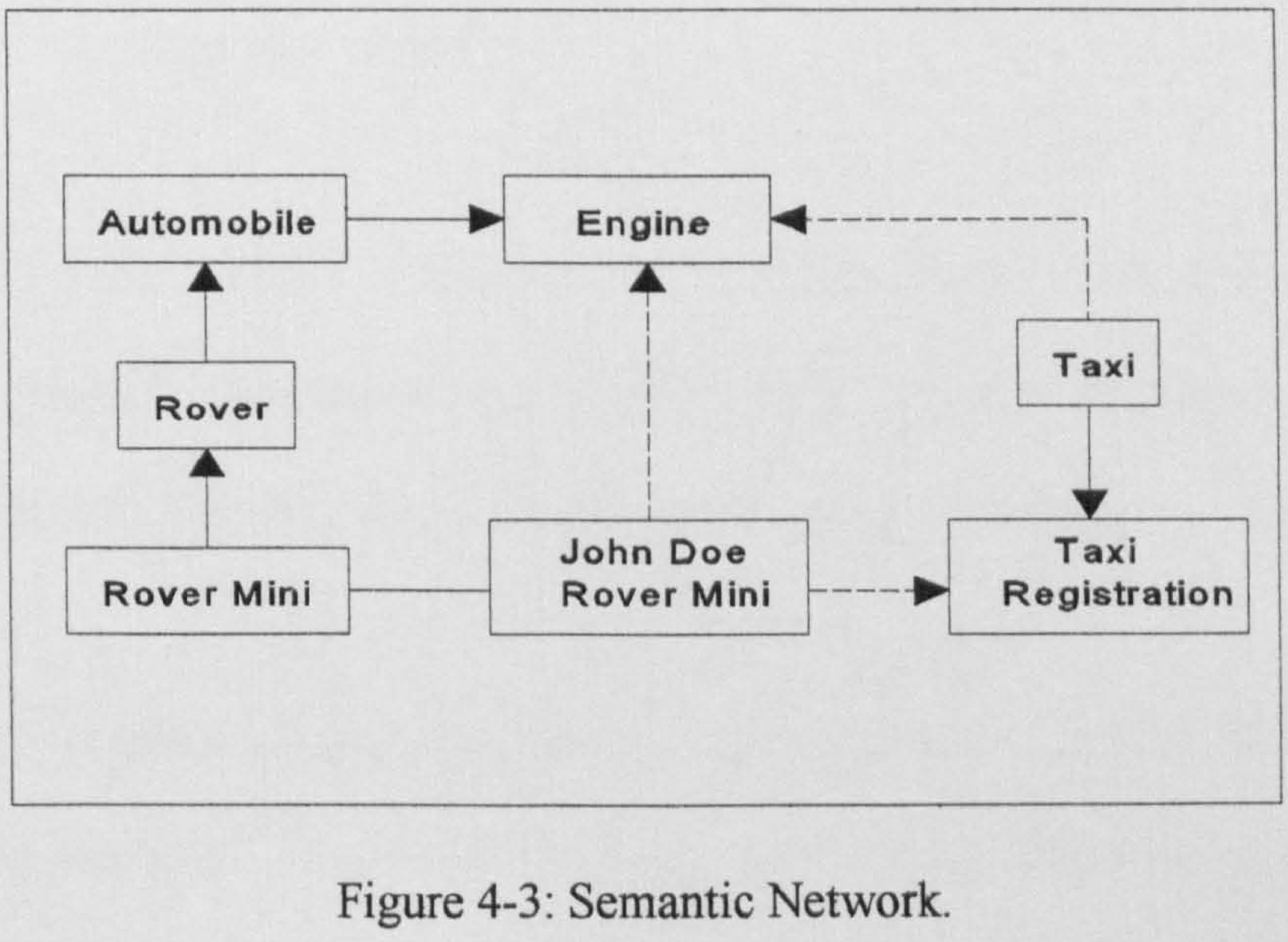

Figure 4-3: Semantic Network.

Another early approach to knowledge representation was based on *Production Rules*. Newell and Simon showed that a surprising amount of human problem solving could be explained using such production rules [154]. In this approach, knowledge is represented as a series of *If-Then* rules based on propositional logic. Propositional logic is a language and a tool for detecting the truth or falsity of propositions.

Production systems as a whole consist of three parts:

(a) a rule base containing a set of production rules;

(b) a database (also called context) containing facts that describe the status of the system; and

(c) an inference engine (also called an interpreter) which makes use of the production rules and the database of facts to make inferences, make diagnoses, suggests action, etc.

A production rule is a statement cast in the form:

IF **this condition(s) holds**

THEN **do this action(s)**

The *IF* part of the production is called the condition and describes the required conditions for the particular rule to be applicable or 'fired'. The *THEN* part is called the action, and describe what is the action to be taken if a particular production rule is fired.

Formal Logic is another early approach of knowledge representation [27]. It has two important and interlocking branches. The first are the *axioms*, which are used to represent *what can be said*. The other is the *rule of inference*, which are used to decide *what is the possible inferences* that can be made if certain axioms are true. In logic, a proposition can have a truth value of true or false. An example of a proposition, *"The system is working"*. A proposition is taken as a whole sentence and not to be broken down into its constituent parts. More complex propositions can be formed from many simple propositions being connected by *sentential connectives*. The five commonly used connectives are shown in table 4.2.

CONNECTIVE	SYMBOL
AND	$\wedge$
OR	$\vee$
NOT	$\neg$
IMPLIES	$\rightarrow$
EQUIVALENT	$\equiv$

Table 4-2: Sentential Connectives

The use of these connectives brings us to propositional calculus, where statements like *"The book has been borrowed or it has been sold"* can be expressed.

Facts and rules are important knowledge structures, nevertheless, a way of packaging knowledge that makes it easily accessible is needed. Packages provide modularity, hierarchical organization and compactness of expression. Knowledge structure that fulfils these purposes which are frequently used in expert systems is the *frame* and *script* [155,156].

The basic idea of a frame was outlined by Minsky [145], who in introducing the notion of frames, wrote:

> *".... the ingredients of most theories both in AI and in Psychology have been on the whole to minute, local and unstructured to account ....for the effectiveness of common-sense thought. The 'chunks' of reasoning, language, memory and perception ought to be larger and more structured; their factual contents must be more intimately connected in order to explain the apparent power and speed of mental activities."*

Frames and scripts represented one line of research on how to combine declarations and procedures within a knowledge representation environment. The fundamental organizing principle underlying frame system is the packaging of knowledge.

Scripts allow reasoning based on expectations about what should happen next in stereotyped situations. A script is an outline of an episode of a certain type. This outline serves two purposes. First, it organizes a set of actions. Second, it predicts the presence of activities that are not specifically referred to.

A typical machining script is shown in figure 4.4.

Script : Machine Shop Track : Lathe Machine Prop : Cutting Tools Machine Capabilities Machinist	Codes X = Part Name MC = Machine Capabilities

ENTRY CONDITION	**RESULTS**
X is to be machined X has specification	X has tight specification MC has the capability X is machined X is within tolerance

Event 1	Set-up part to be machined
Event 2	Machine to specification
Event 3	Check for out of specification
Event 4	Machining done

Figure 4-4: Sample of a machining script.

This script can also be represented as rules:

Part machined `*machined*'
If
Part machined `*machined*'
and
Part set `*setup*'
and
Part specification `*D&T*'

Part machined `*machined*'
If
Machine capable `*capability*'
and
Tool type `*cutter*'

Object oriented methods provide another alternative for representing knowledge. The methodology for representing knowledge shares a number of features with frames and semantic networks. In this approach knowledge is viewed as a set of objects, each of which can exhibit certain behaviour. Each object is in a network or hierarchy and can access properties and information from higher-level objects [145,157]. In the object oriented paradigm, objects communicate with each other by sending and receiving *messages*. An object that has received a message checks the database and decides what action to take. Action can be taken by invoking a *method*. Any action that it does decide to take is again passed on as a message.

Finally, an interesting approach to knowledge representation suitable for application on a multiprocessing or distributed computer system is the *blackboard* model [142,145]. The blackboard models aim to address three distinct problems that appear as the size of a knowledge-base grows:

(a) the system becomes harder to understand, since there are many rules, facts, etc.;

(b) different types of knowledge and different knowledge representation and inference methods need to be integrated; and

(c) response time begins to deteriorate as the amount of required computation increases.

The blackboard model deals with these issues by separating knowledge into modular knowledge sources that use different knowledge representation and inference methods and that may reside on separate computers. Thus, the blackboard architecture is made up of three basic components:

(a) the *blackboard* (a global database),

(b) the *knowledge source* (independent sources that have access to the blackboard), and

(c) the *scheduler* (to control knowledge activity).

Three distinct advantages provided by the blackboard model:

(a) It can be used to organize knowledge in a modular way.

(b) It can easily integrate different knowledge representation methods.

(c) It may be executed in a distributed computing environment for greater efficiency.

4.4 Expert System Techniques

There are three broad types of software tools used to develop expert systems, namely:

(a) shells,

(b) toolkits and

(c) languages.

To choose a software tool, first, one needs to look at the demands of the knowledge representation. What is the data structure needed to hold the expert's knowledge? The system developer must also look at the other things a tool can do. Examples can include the need to communicate with programs written in other languages or with some databases or may have to provide an easy user interface, with pop-up menus and windows.

The task of system development will be adroit and swift for the experienced knowledge engineer. However, the novice will face many difficulties and challenges through not having the understanding of the tools' capabilities during development.

4.4.1 Shells

The expert system shell can be considered as a reasoning system out of which all the knowledge has been emptied. When knowledge about a new domain is entered into the shell appropriately, an expert system is created. The original shell can then be used to create a new expert system in similar fashion.

An expert system shell performs three major functions:

(a) assists in building the knowledge-base by allowing the developer to insert knowledge into knowledge representation structures;

(b) provides methods of inference or deduction that reason on the basis of information in the knowledge-base and new facts enter by the user; and

(c) provides an interface that allows the user to set up reasoning tasks and query the system about its reasoning strategy.

There are many expert system shells available commercially as listed in Appendix A2 [158,159,160,161]. These shells are constantly updated and the price varies according to the capabilities. The advantage of an expert system shell is that it cuts down on the development time in building an expert system. The disadvantage is that it may not be flexible enough to handle all the problems the knowledge engineer will encounter.

4.4.2 Toolkits

A toolkit is a cross between a language and a shell. It is a set of integrated tools that can be used to solve AI problems, i.e. a tool that possesses a frame-based component, a logic component, a forward chaining component, etc. However, a tool does not have a knowledge base which must be supplied by the programmer [161,162]. Most of the toolkits available are based on LISP and run on workstations. Due to the nature of toolkits, they are best suited to experienced knowledge engineers.

4.4.3 Languages

The idea of developing expert systems by means of a *language* are most familiar especially in earlier expert systems such as MYCIN, DENDRAL and PROSPECTOR (all are written using languages). There are several languages used to develop expert systems. The most common of them are LISP or PROLOG [27,137,142].

(a) **LISP** - the major language of AI in the United States and was developed by John McCarthy. LISP stands for LISt Processing and is based on recursive function theory. It is characterized by symbol manipulation as opposed to numerical manipulation. LISP is an interpreted, type-free, flexible language, in contrast to other languages, one can change the syntax of the language, add new data structures or define new functions with little difficulty. LISP is especially useful in prototyping and its environment is characterized by excellent editing and debugging tools. Nevertheless, LISP is criticized because it takes up a lot of memory and it is not as portable and is slow compared with other language such as C. LISP also has difficulty interacting with other computer languages.

(b) **PROLOG** - was invented around 1970 by Alain Colmerauer and his associates at the University of Marseilles. PROLOG which stands for PROgramming in LOGic is a declarative language based on logic programming. It is based on predicate calculus. Predicate calculus consists of statements about individuals or objects, their properties and their relationships with other objects that return a value of true or false. The clause is the primary representational scheme in PROLOG and it is used to represent relationships among objects in facts and rules. The control features of PROLOG can be divided into two classes, explicit and implicit. Explicit control features include the ordering of facts and rules, the ordering of subgoals in a query, the cut, fail, true, not, call, repeat and recursion. Implicit control features are depth-first search and backtracking. A PROLOG programmer defines relationships among objects in facts and rules and carries out the explicit control features to control the program flow partially. Then, all that is required is to query the program. The implicit control features of depth-first search and backtracking take command of the control flow to solve the query. An advantage of PROLOG is that if an attempt to solve a query fails, the underlying backtracking mechanism of PROLOG automatically uninstantiates variables. New values for the variables will then be evaluated in an attempt to find an alternative solution. A second advantage is that it is relatively easy to build bridges to relational data bases. In comparison with LISP, PROLOG is faster and does not take up as much memory. PROLOG may be considered a production-system language. It has a built-in inference engine, its body of facts may be considered a type of working memory and its rules may be considered as a rule base.

4.5 Intelligent CAD

The ability of CAD to speed up change would certainly overcome the dilemma of constant amendments, corrections and minor improvements due to the intricacy of design and redesign. CAD vendors so far have created excellent geometry engines, tools that can put lines, points, surface and mass property calculations on a display; however, CAD is incapable of capturing design intent and does not allow engineers to perform uncertainty analyses. The inherent limitations of conventional CAD systems and the requirement of CAD systems to be intelligent and flexible in manipulating design knowledge are presented in many papers [133,134,163,164].

Due to these incapabilities, application of expert systems working interactively with CAD is an option for further investigation as mentioned by Seely and Rosenfeld [31,32]. Many developments in Intelligent CAD systems are on engineering design processes and can be found in the literature [134,163,165,166,167,168,169]. The major differences between these systems are the representation and control of knowledge and the AI system(s) used, for example, representation schemes, languages and knowledge-bases. The contributing factor to such differences resulted from the developers' perception of design and the approach towards the formalization of a process model.

Although these developments do not specifically connect to the analysis of dimensions and tolerances, the information gathered from the review will be valuable for setting up the proper approach to develop knowledge-based automatic tolerance analysis.

4.5.1 Intelligent CAD Review

Ohsuga [133] designed a knowledge processing system called *Knowledge Acquisition and Utilisation System (KAUS)*. The system, which has been partly implemented, is based on a new knowledge representation language called, *'Predicate Logic+Data Structure'*, which can express knowledge in the *object oriented* way.

He also introduced:

(a) *object level* knowledge and the concept of *model consistency* to represent the object model and its transformation rules;

(b) *metalevel* knowledge (or metaknowledge) to represent the design process and its control strategy.

Using a less complicated approach than that of Ohsuga, Tomiyama et al. [163] introduced the concept of a *metamodel*, which serves as a key modelling basis of CAD and allows flexible (evolutionary) transformations between various models used in design and supports integration of these models. This modelling scheme is based on the *General Design Theory* [170,171]. A *physical phenomenon* modeller, based on qualitative physics is used as a mechanism to model physical phenomena acting on design objects. Thus it will find out what effect that phenomena will have in the observed system. Meanwhile, design knowledge is described using an *Integrated Data Description Language*. Both systems are implemented out in *Smalltalk-80*, which is an object oriented language.

Jakiela and Papalambros [165] developed an Intelligent CAD system prototype which serves as an aid in improving the quality of initial designs during the conceptual design phase. The approach taken is *proper codification of knowledge* within the CAD system and a *production rule* mechanisms used for coding intelligence. The system's programming language is called DAL, a Pascal/FORTRAN hybrid language.

Akagi and Fujita [166] developed an expert system for engineering design based on an object oriented knowledge representation concept. The system is encoded in LISP, combined with

FORTRAN programs for graphic and large scale numerical computations. The developed system was applied at the preliminary ship design stage and in the design of a marine power plant.

Lukas and Pollock [167] came up with an interesting idea of automating design *handbooks* or standards, i.e. encoding the engineering rules of thumb, equations and data into a computerised format. This allows the designer to manipulate the design standards at the conceptual design stage while allowing special features to be added on. A Concept Modeller software package developed by Wisdom Systems was used to achieve this idea.

Phelps et al. [168] introduce the concept of a knowledge-based Taguchi design assistant and have developed the architecture for an expert system for Taguchi methods for engineering design. Common LISP is the basis for intelligence coding and design knowledge is represented as *frames*.

Smithers [134] discusses the need for formal and expressive knowledge representation schemes, the type of automated reasoning (or inferencing) techniques required to support the different types of problem-solving activities, the suitable technique for dealing with the dependencies between design elements and maintaining their consistency, and finally detailing the sort of intelligent control required for knowledge-based systems. He [172] later tested the idea: first, by the construction of a design support system that integrates various knowledge based systems techniques to support the design activities of pharmaceutical drug designers, and secondly, by a design case study from another design domain.

An expert system shell, GoldworksII, is utilised to control the invocation of knowledge sources. The architecture of the exploration-based model of Smithers et al. are shown in figure 4-5.

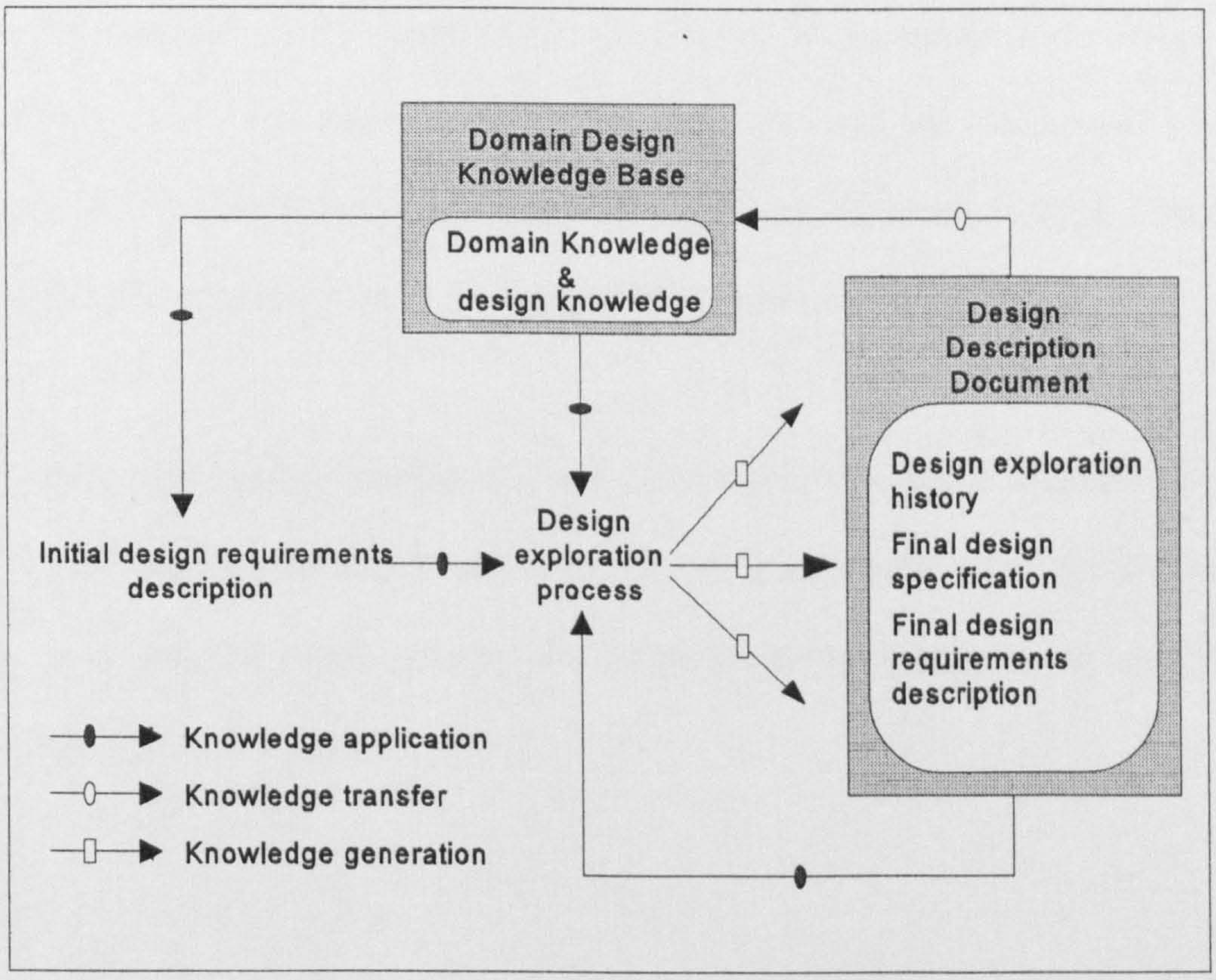

Figure 4-5: Knowledge process underlying exploration based design

Based on the knowledge gathered from the investigations on Intelligent CAD, the current working model of the system can advise a designer in a real-time design activity. It also exhibits a considerable amount of knowledge about the nongeometric attributes of the objects drawn and the real environment for the design operation.

Nevertheless, Myers and Pohl [173] explained that the current Intelligent CAD model has two major problems:

(a) difficulty with knowledge acquisition; and

(b) design attributes are evaluated in different ways by different designers.

Most of the Intelligent CAD systems listed and reviewed are experimental prototypes. Only two, namely, *ICAD* (developed by ICAD Systems Inc.) and the *Conceptual Modeller* (developed by Wisdom Systems) are commercially available. However, in 1993, *Conceptual Modeller* has been included under the wing of ICAD Systems Incorporation.

The growing importance of Intelligent CAD technology has motivated other developers to develop such a unique system. Hence, in early 1994, 2 more commercial Intelligent CAD systems were marketed, namely, *Design*$^{++}$ by Design Power Incorporation [174] and *Defigner* of LoftTech Incorporated [175].

CHAPTER 5

KNOWLEDGE-BASED AUTOMATIC TOLERANCE ANALYSIS

5.1 Introduction

Based on the literature findings, an intelligent knowledge-based system working interactively with a CAD system has improved the design ability significantly. Thus, this approach can also ameliorate the present technique of tolerance optimisation.

A knowledge of processing will produce more accurate tolerance results. This is because of the ability to optimise tolerance allocation based on real production data and not imaginary or random number generation.

5.2 Architecture

The system has been designated as Knowledge-based Automatic Tolerance Analysis's (KATA) and is composed of four modules namely:

(a) feature recognition;

(b) detailing;

(c) manufacturing, and

(d) tolerance analysis.

The system architecture is shown in figure 5-1.

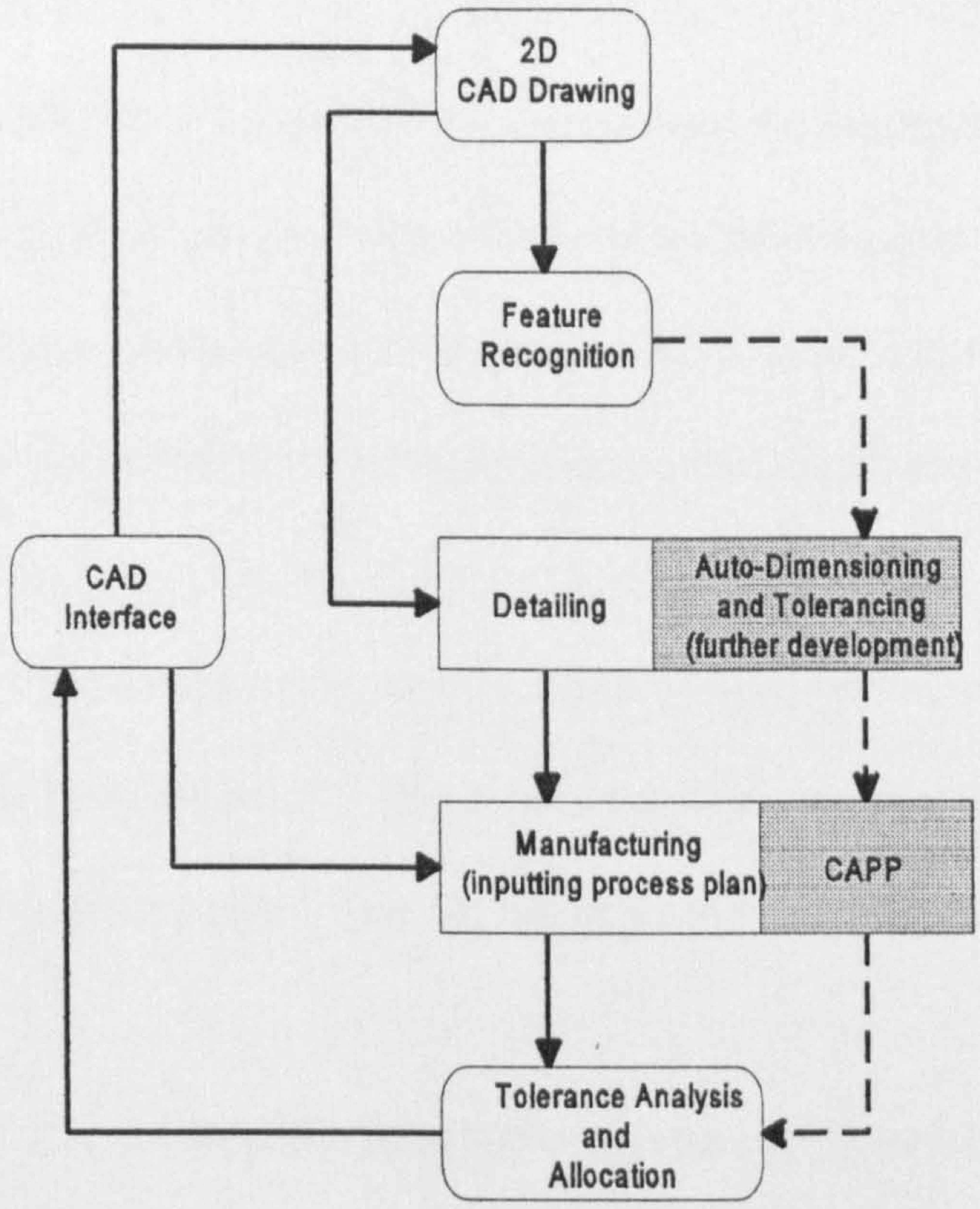

Figure 5-1: KATA's system architecture.

The process flow begins with the user creating the geometry of the workpiece using a CAD system. Following the creation, two approaches are available for getting geometry data for tolerance optimisation.

The first approach is using the *feature recognition module*. In this module the workpiece feature is identified and the data collected from the identification is stored in a CAD database. This ability is a significant advantage of KATA over other tolerance optimisation systems developed thus far. Instead of dwelling on only tolerance allocation optimisation as previous approaches, it has the capability of recognising a feature which can improve the system in the area of process planning profoundly. Hence, it will be very useful for KATA's future development.

In the second approach, the workpiece is semiautomatically dimensioned and toleranced in the *detailing module*. All the information obtained from the workpiece is stored in a CAD database for later used in the analysis. Dimension and tolerance representation is based on BS 308 specification. The representation output on the CAD screen of this module is equivalent to the detailed drawing distributed by the designer to the manufacturing engineer.

Subsequently, using the *manufacturing module*, a process plan for the workpiece is examined. The method of cutting, type of process, the depth of cut etc., are analysed in the module. All manufacturing information obtained from the manufacturing module, is stored in the database.

In the *tolerance analysis module*, the information entered previously in the aforementioned modules will be expertly analysed. This analysis is conducted using a linear programming model. The analysis result is verified with the plant capabilities data already installed in the system database to gain the optimal manufacturing tolerances. When the analysis arrives at a feasible solution, the manufacturing tolerances will be updated automatically in the CAD graphics screen. Output of this analysis can be used as a process sheet to be distributed to relevant process cells as a reference for manufacture.

5.3 Software

KATA's developments are based on CAD and an Intelligent Knowledge-based package. CAD, is KATA's platform of part representation and communication. On the other hand, the intelligent knowledge-base is KATA's brain. It is the knowledge processing capabilities of KATA where inferences and search mechanisms reside.

5.3.1 CAD

AutoCAD will be the CAD host for KATA. Release 11 has been licensed to be used at the Advanced Technology Centre, University of Warwick.

This CAD package is quite popular, used by Universities and Industries alike. In August 1992 alone, it is reported that 650,000 copies of AutoCAD had been sold worldwide [176]. At a price tag of approximately 3,000 pound sterling for a complete package of AutoCAD Release 12, it is affordable for any small or medium scale manufacturer.

Apart from the basic CAD utilities, AutoCAD's main asset is AutoLisp, the software programming language embedded in the CAD system. AutoLisp closely follows the forms, conventions and syntax of the *Common Lisp* (list processing) language. It is based on *xlisp*, a program developed by an American, David Betz [177].

Another important feature of AutoCAD is the AutoCAD Development System (ADS). ADS is an advanced programming language environment created to support the development and integration of C-language programs either directly with AutoCAD or to assist the interfacing of AutoCAD with other software applications such as databases.

5.3.2 Intelligent Knowledge-Based System.

An intelligent knowledge-based shell that can work interactively with AutoCAD was investigated and built for KATA. PROLOG is used to built KATA's feature recognition module. AutoLISP is used to programme KATA's knowledge-base. Finally, tolerance optimisation is conducted using the Linear INteractive Discrete Optimizer (LINDO) software.

LINDO is an interactive linear, quadratic and integer programming system. The command programme is simple and can be illustrated by the following example:

To maximize $2X + 3Y$ subject to $4X + 3Y < 10$ and $3X - 5Y < 12$ [178];

MAX 2X + 3Y

SUBJECT TO

4X + 3Y < 10

3X + 5Y < 12

END

Typing *GO* will cause this problem to be solved. A significant difference with other numerical programming methods is that the strict inequality < is interpreted to mean the loose inequality $\leq$.

5.4 Hardware

Since AutoCAD is the host CAD for KATA, the hardware selection is much more obvious. Based on the system requirement suggested by Autodesk [179], developer of AutoCAD, KATA is built on the following host's specification:

(a) 80486-DX33 system running MS-DOS 5.01;

(b) Logitech serial mouse, used to facilitate the coordinate movement in AutoCAD drawing environment;

(c) 1.2 Mb, 5¼ inch floppy drive and a 1.44 Mb, 3½ inch floppy drive;

(d) 8Mb random-access memory, and

(e) 250 Mb Hard disk.

5.5 Detail Structure

Instead of using an existing expert system shell to work interactively with AutoCAD, it was decided to develop and built KATA's own shell. This will allow greater flexibility in terms of the architecture development and the system's optimisation processing. An interface problem is also eliminated because the languages used are compatible and embedded in AutoCAD itself.

5.5.1 Feature Recognition

The structure for the feature recognition module is illustrated in figure 5-2. The upper part of the feature recognition structure is used to preprocess the *Data eXchange Format* (DXF) file to a PROLOG format file [180].

The drawing file (DWG file) stores the drawing compactly and is rapidly accessed by AutoCAD. However, it is in an undocumented binary format that requires an expert programmer to read. AutoCAD provides another file format, in ASCII text, which is just as complete. This DXF file is more readable than the DWG file and accessible to process to another format. Further explanation of DXF files and codes can be found to in the literature [181,182,183].

In this experiment, the DXF file is preprocessed to a PROLOG fact format file. Then this file is used to recognise a part feature. The external knowledge-base is later consulted to verify the recognised feature.

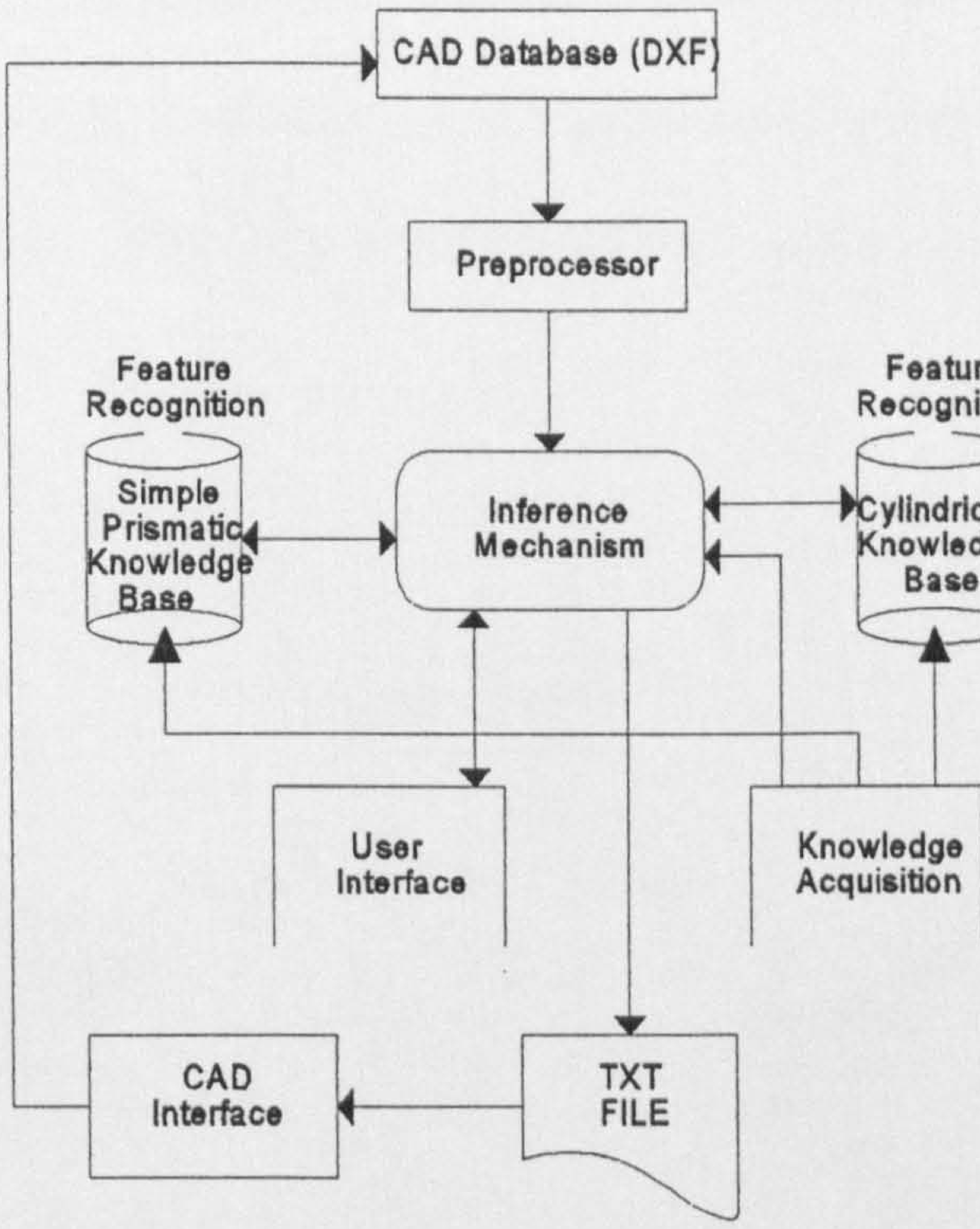

Figure 5-2: Feature Recognition Structure.

The workpiece to be analysed is represented in 2D. Meeran and Pratt explained that automated recognition of machining features from 2D drawing is a practical possibility [184]. There are also other investigators who worked on the feature recognition problem to reconstruct a 3D solid model from 2D drawings [185,186,187,188,189]. 2D drawing according to them is the form in which most of the design data currently exists. Many designs also exist as 3D wireframe models or less frequently solid models. All drawings have been encoded as DXF files. DXF is a *de facto* standard through its wide use as a transfer format.

5.5.1.1 Knowledge Identification

A *feature* is a component of the real object or a boundary of its geometric model such as a surface or a hole. This type of feature is called a simple feature. A compound or composite feature such as a flange, slot or pocket is one that is a combination of simple features. To identify and simplify the situation, workpiece features can be classified as two main categories; namely rotational (cylindrical) and non-rotational (prismatic) features. Shown in a tree diagram, figure 5-3, are categories of workpiece features.

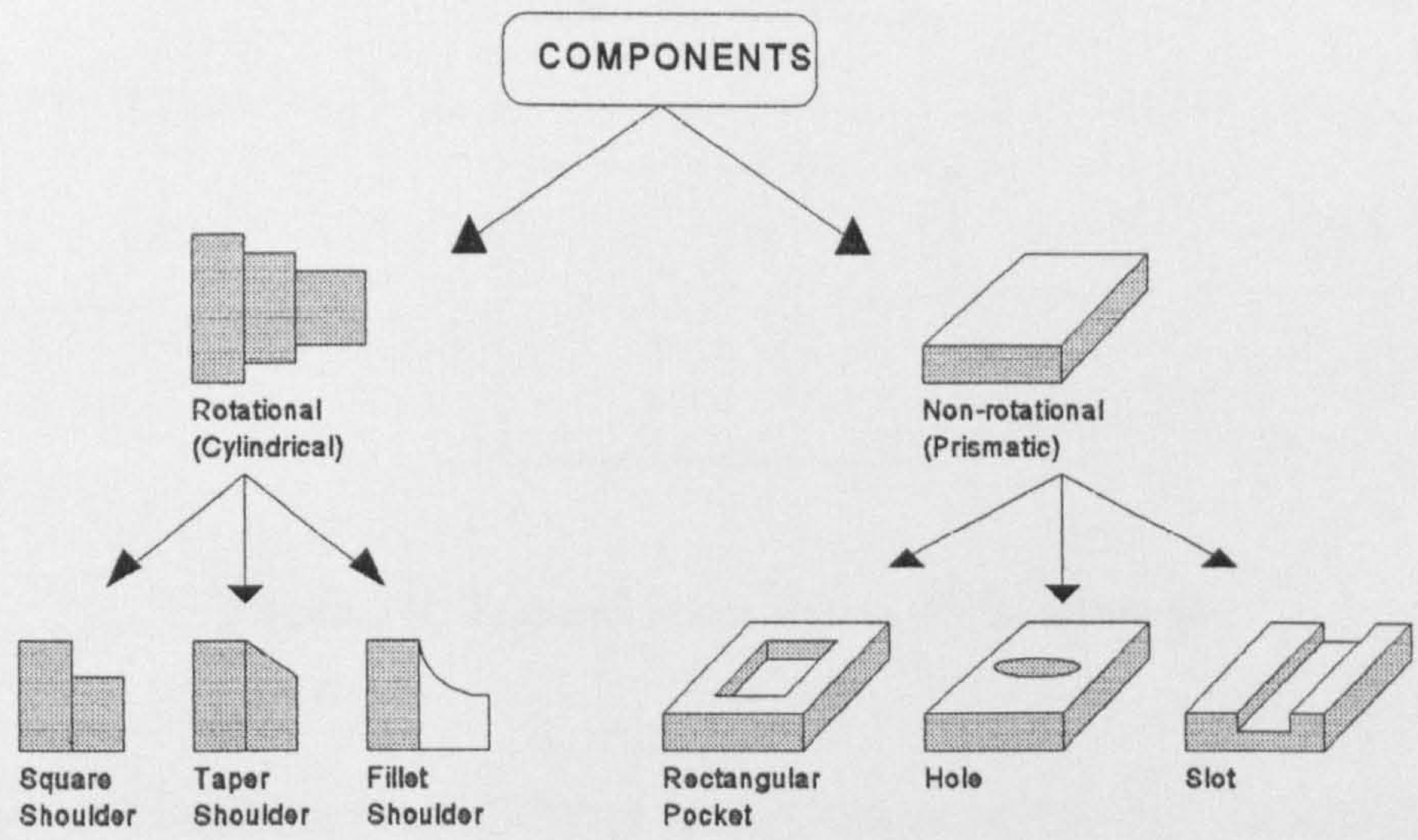

Figure 5-3: Manufacturing Feature Tree

5.5.1.2 Knowledge Representation

Each manufacturing feature consists of some group entities such as line, circle and arc. Referring to figure 5-3, the feature *square shoulder* can be represented by one horizontal solid line linking with another vertical solid line. Similarly, the feature *fillet shoulder* consists of one vertical solid line, one solid arc and one horizontal solid line. While a representation of one inclined solid line can simply mean that the feature is a *taper*.

For non-rotational workpieces, to recognize features involves checking entities from several different planes. The *hole* feature in figure 5-4, for example, is represented in two views.

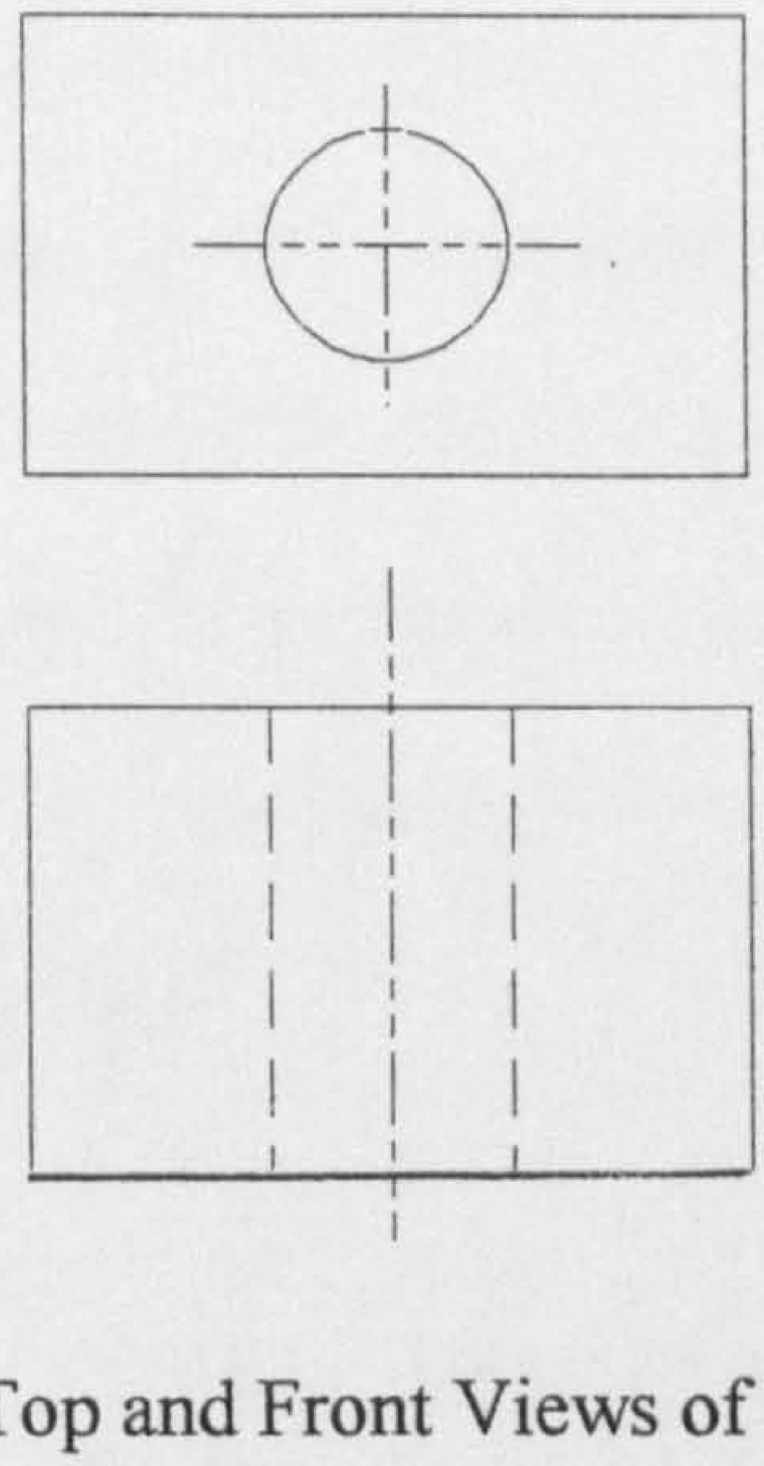

Figure 5-4: Top and Front Views of Hole feature.

Thus, using the following PROLOG fact format, the geometric entities in a 2D wireframe CAD database can be extracted:

vector(RNO,View_no,Line_type,Entity_name,[Start_x,

Start_y,End_x,End_y,Centre_x,Centre_y, Radius, Angle]).

In the PROLOG functor *vector*, *Rno* is the record number of each geometry entity. *View_no* is used to recognise the corresponding view of the entity, such as front view, end view, etc. *Line_type* contains the information about the type of line of each entity in the CAD database, for example, hidden line, solid line and centre line. *Entity_name* records the name of basic entities such as, lines, circles and arcs. Geometric relationships of the features, on the other hand, can be represented in the following PROLOG fact formats:

link(Rno1,Rno2).

touch(Rno1,Rno2).

The functor *link* defines that the end point of one entity is equal to the start point of another entity. The functor *touch* defined that the end or start point of the entity is touching a segment of another entity. These two functors can be used as a graphic grammar rule. The low level features of rotational workpieces can be simply recognised by matching all entities of the profiles of the workpiece with the feature template. For example, the feature *external cylinder* can be recognised by the following rule:

```
feature_rule(FRNO, Name,      /* feature name: external cylinder
        Type,                 /* line type: solid
        View _no,             /* view_no: 1- front view
        Ename,                /* entity name: line
        Direction,            /* direction (e.g. LHS or RHS): both
        [LD1,UD1,             /* lower and upper limit of D1: nil
        LD2,UD2,              /* lower and upper limit of D2: nil
        LL,UL,                /* lower and upper limit of L: nil
        LA,UA,                /* lower and upper limit of angle: 0
        LRAD,URAD])           /* lower and upper limit of Rad: 0
feature_db(Fno,Name,Type,View_no,Direction,Datalist_Rno).
```

Each *vector* is matched with the above template. The recognised feature can be recorded into the *feature_db* database for later feature recognition process. Take the feature *LHS fillet shoulder*, as in figure 5-3, the rule to recognise this features can be written as:

```
lhs_fillet_shoulder(S):-                                        /* fillet shoulder
        feature_db(Fno1,"external cylinder",_,V,B,_Rno1),       /* external shoulder
        feature_db(Fno2,"external_fillet",_,V,L,_Rno2),         /* external LHS fillet
        link(Rno1,Rno2),
        S=[Fno1,Fno2],
lhs_fillet_shoulder(S).
```

The above PROLOG clause defined that, if the low level feature *Fno1* is *external_cylinder* and *Fno2* is *external fillet*, and *Rno1* and *Rno2* has a *link* relationship, then the feature is *LHS_fillet_shoulder*. The list *S* is used to register the record number of all related entities of this feature. For non-rotational workpieces, the same concept can be applied. The rule for *hole* is shown as follow:

```
hole(S):
        vector(Rno1,Plan_view,"hidden","line",[X1,Y1,_,_,_,_90]),
        vector(Rno2,Plan_view,"hidden","line",[X2,Y1,_,_,_,_90]),
        middle(X1,X2,CX),
        distance(X1,X2,Dia),
        vector(Rno3,Plan_view,"centre","line",[CX,_,_,_,_,_90])
        vector(Rno4,Front_view,"solid","circle",[_,_,_,CX,_Rad,_])
        Rad=Dia/2,
        S=[Rno1,Rno2,Rno4],
hole(S).
```

The mechanism of inferencing of this template-matching process is the natural backward-chaining depth first search strategy of PROLOG. The control predicate is therefore straightforward in its operation. The predicate is described as:

```
template-matching:-
            hole(_),fail;
            neck(_),fail;
            .....
            .....,fail
template-matching.
```

The predicate first tries to prove the first subgoal, *hole( )*. It seeks to identify hole from the corresponding views of the workpiece. If it succeeds, control moves on to the next subgoal, *fail*. Fail is the standard predicate in PROLOG that forces backtracking. In this way, all hole patterns will be recognised within the geometry and the related entities will be deleted from the PROLOG database.

After an exhaustive search of the database, the control predicates move on to the next subgoal, *neck( )*. The semicolon represents the Boolean OR operator. Therefore, the control predicate will attempt to prove all listed subgoals, thus invoking all feature templates. This backward-chaining depth-first inference mechanism can make sure that all features can be recognised by matching all feature templates without missing any. For a sample workpiece as in figure 5-5, the feature identified by KATA is shown in figure 5-6.

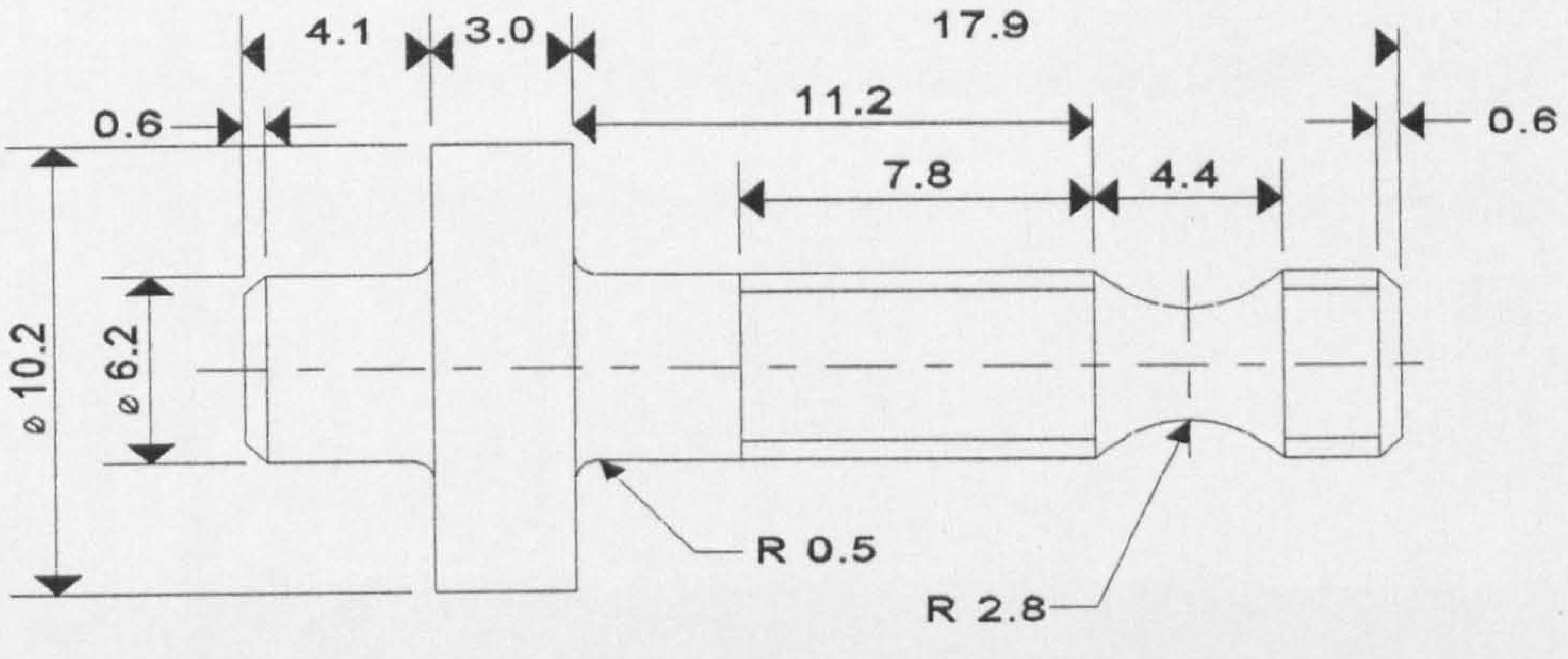

Figure 5-5: Feature Recognition Sample Workpiece.

CAD file c:\acad\part3.kb1			
Fno	Name	type	direction
found 1	Face	External	-
found 2	Face	External	-
found 3	Face	External	-
found 4	Face	External	-
found 5	Chamfer	External	Right
found 6	Chamfer	External	Left
found 7	Diameter	External	-
found 8	Diameter	External	-
found 9	Diameter	External	-
found 10	Diameter	External	-
found 11	Thread	Thread	-
found 12	Thread	Thread	-
found 13	Fillet	External	Right
found 14	Fillet	External	Left
found 15	R_groove	External	-

Figure 5-6: Workpiece Characteristic Identified in Feature Recognition Module.

5.5.2 Detailing

The tolerance in KATA's detailed drawing is represented in an *unequal bilateral form*. A set of detailing menus were built specifically for this module. It is to assist the user to dimension and tolerance the workpieces. Examples of this menu are shown in figure 5-7 and 5-8.

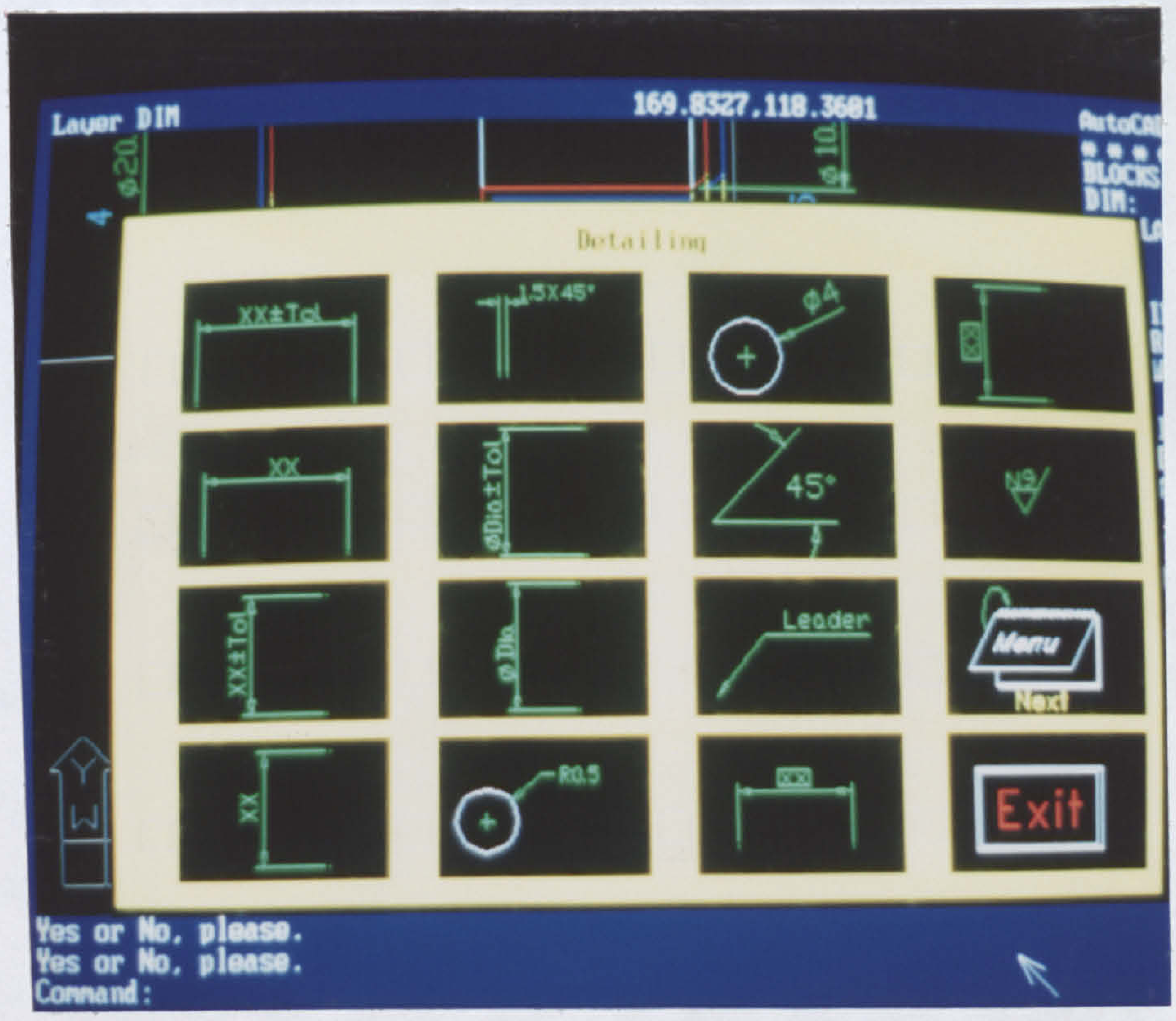

Figure 5-7: Dimensional Detailing Menu.

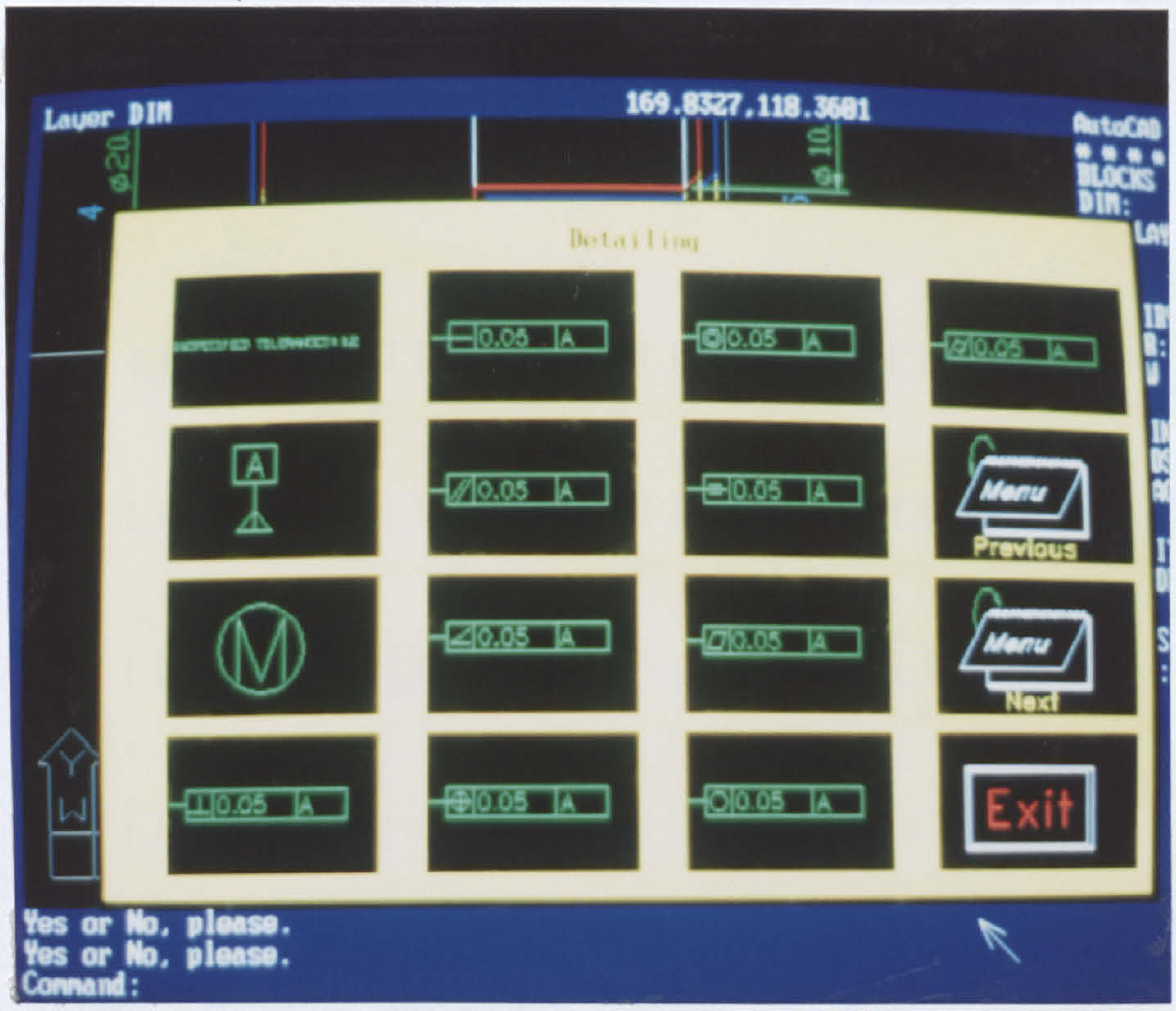

Figure 5-8: Geometrical Detailing Menu.

Tolerance representation can be divided into two categories, namely, dimensional tolerance and geometric tolerance representations. Numerical and geometric data gathered from these representations is required to proceed to a tolerance analysis module. In dimensional tolerance, all linear dimensions and tolerances information is stored in the CAD attributes' database. This information can be retrieved from a CAD drawing editor as shown in figure 5-9. Information stored includes the coordinates of start points and end points of horizontal and vertical dimensions in the drawing, information on the type of features, ie. shoulder, chamfer, taper, dimension identification numbers and blueprint tolerances. A significant advantage of KATA over other systems developed thus far is that these values can be updated and edited without terminating the job. After saving all data in a CAD database, it is later used by an external program for the analysis and allocation of tolerance.

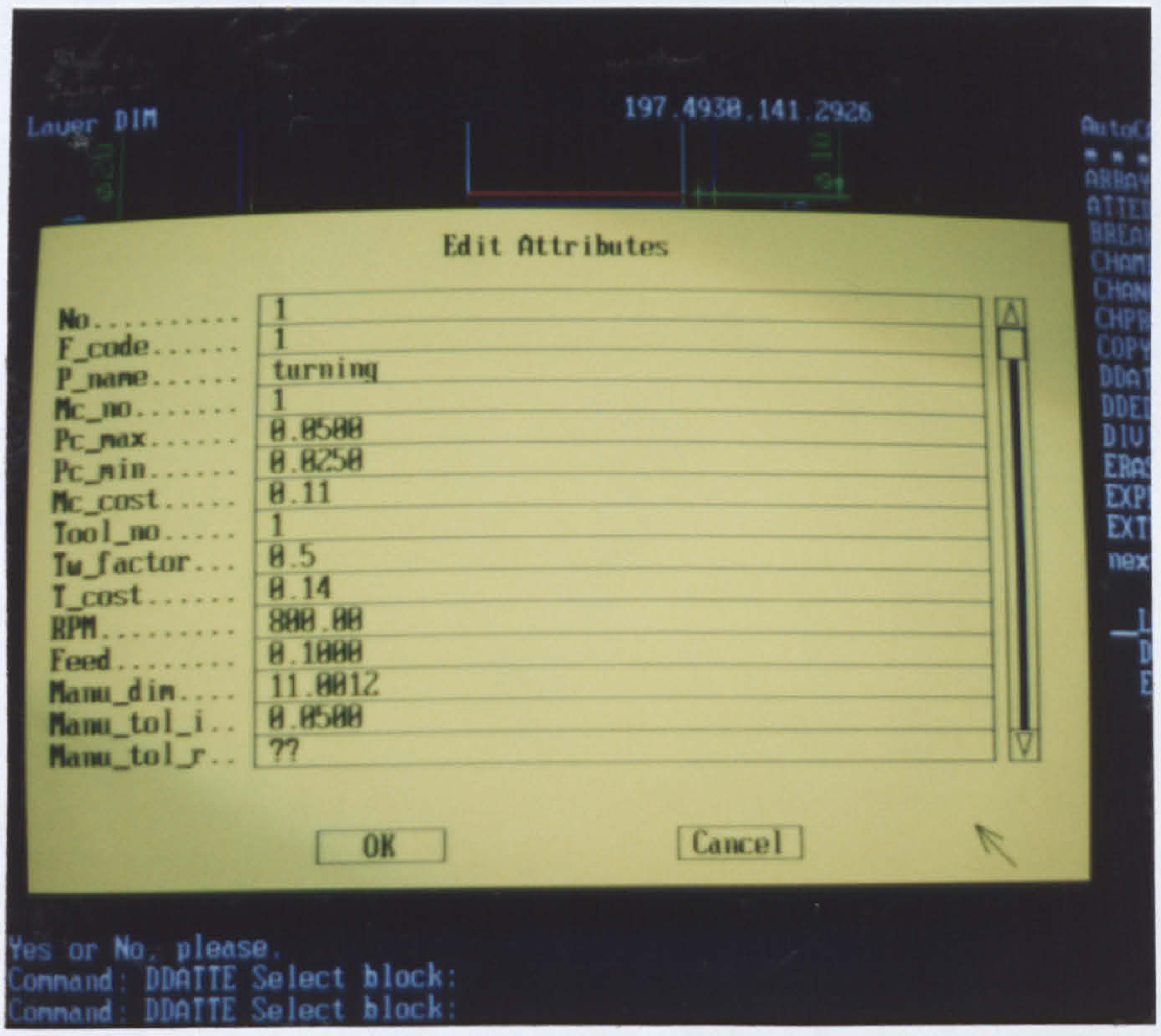

Figure 5-9: Model Attribute from CAD database.

One setback of working with a 2D wireframe model is that not all geometric tolerance characteristics can be analysed. The ability to analyse these geometric tolerances is by transforming its attribute into linear form. For example, the tolerance zone of the location tolerance is represented by a circle and its centre is measured from the X-datum and Y-datum. The diameter of this circle is equal to the value in the block as shown in figure 5-10(a). Therefore, the analysis can still be conducted if these features are transformed into two linear dimensional tolerances as in figure 5-10(b).

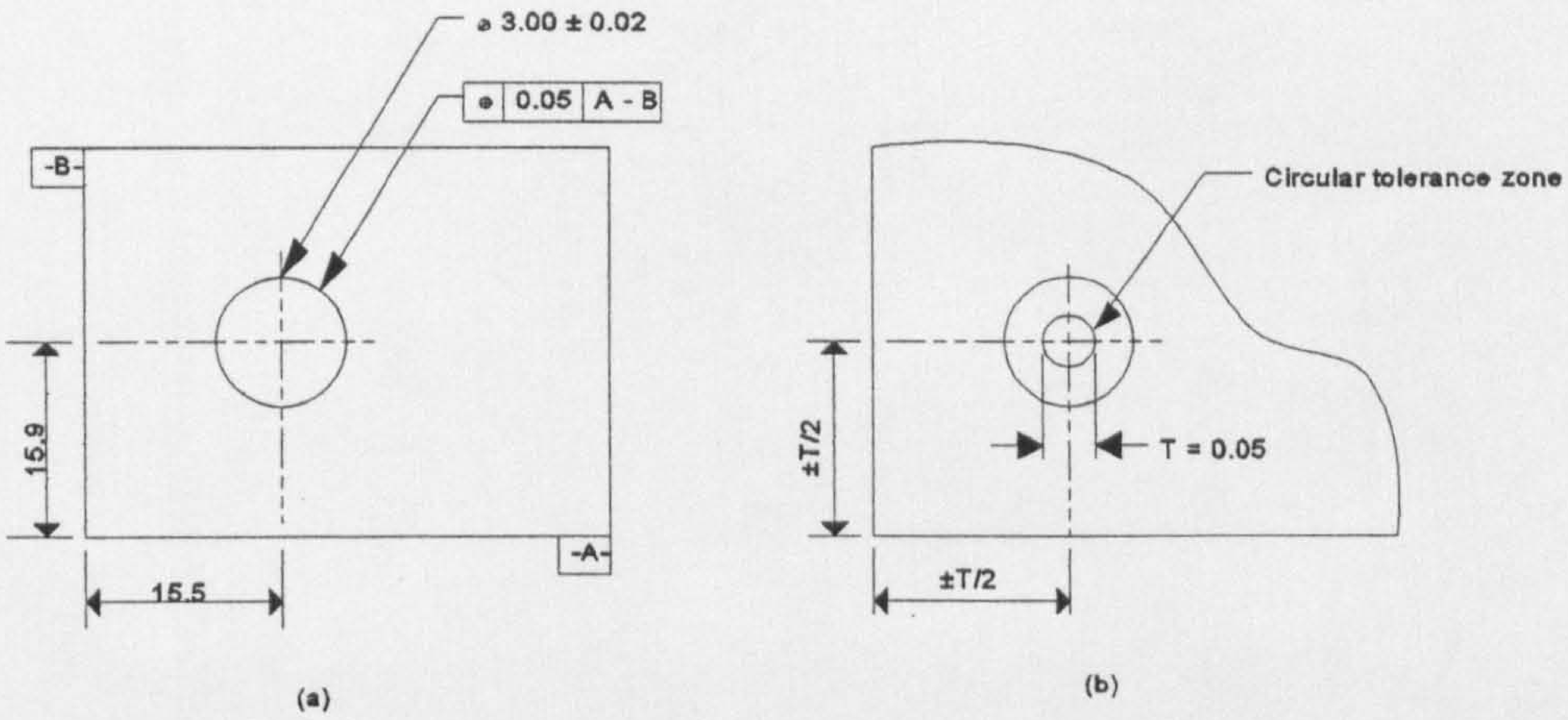

Figure 5-10: Location Tolerance Translation in KATA

The concept of symmetrical tolerance is to regulate the position of the control centerline within the tolerance zone. For symmetry to exist between two features, the centerline of one feature must lie in the plane of the centerline of the other feature; or in other words, the two centerlines must be coincidental. Thus, the control centerline is called a datum centerline.

Like a location tolerance, a symmetrical tolerance zone can be transformed into a set of linear tolerances and these values as shown in figure 5-11.

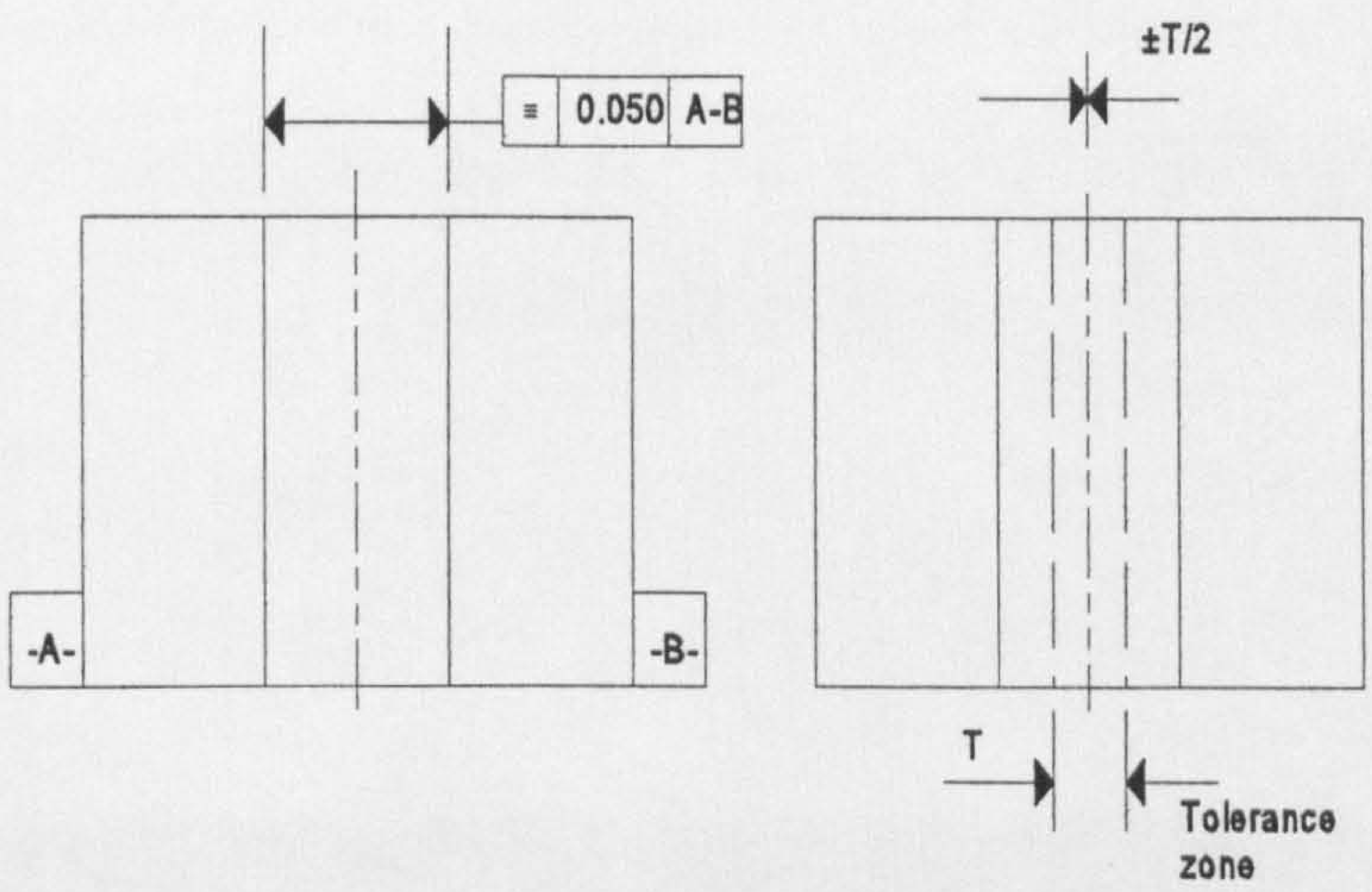

Figure 5-11: Symmetrical Tolerance Translation.

5.5.3 Manufacturing Module

Before the dimensions and tolerances are optimized, prior knowledge of process capabilities, cutting tools, available machines and cutting parameters is important. This is to ensure that the analysis process is being supported by actual production capabilities. Thus, the resultant value acquired from the analysis will be more accurate at the time of production. Having this knowledge will also inform the engineer of the plant capability. This imparts a significant advantage to the user rather than wasting valuable time and money with trial and error production runs. A notable advantage due to the used of modular development is that the module can further be expanded according to the needs of manufacturing.

Three types of knowledge bases developed for KATA, thus far and currently available in the module are:

(a) *manufacturing knowledge* -- having the abilities of updating information required for inferencing during manufacturing and analysis of part components to be manufactured, such as types of machines, tooling, materials to be processed etc., figure 5-12;

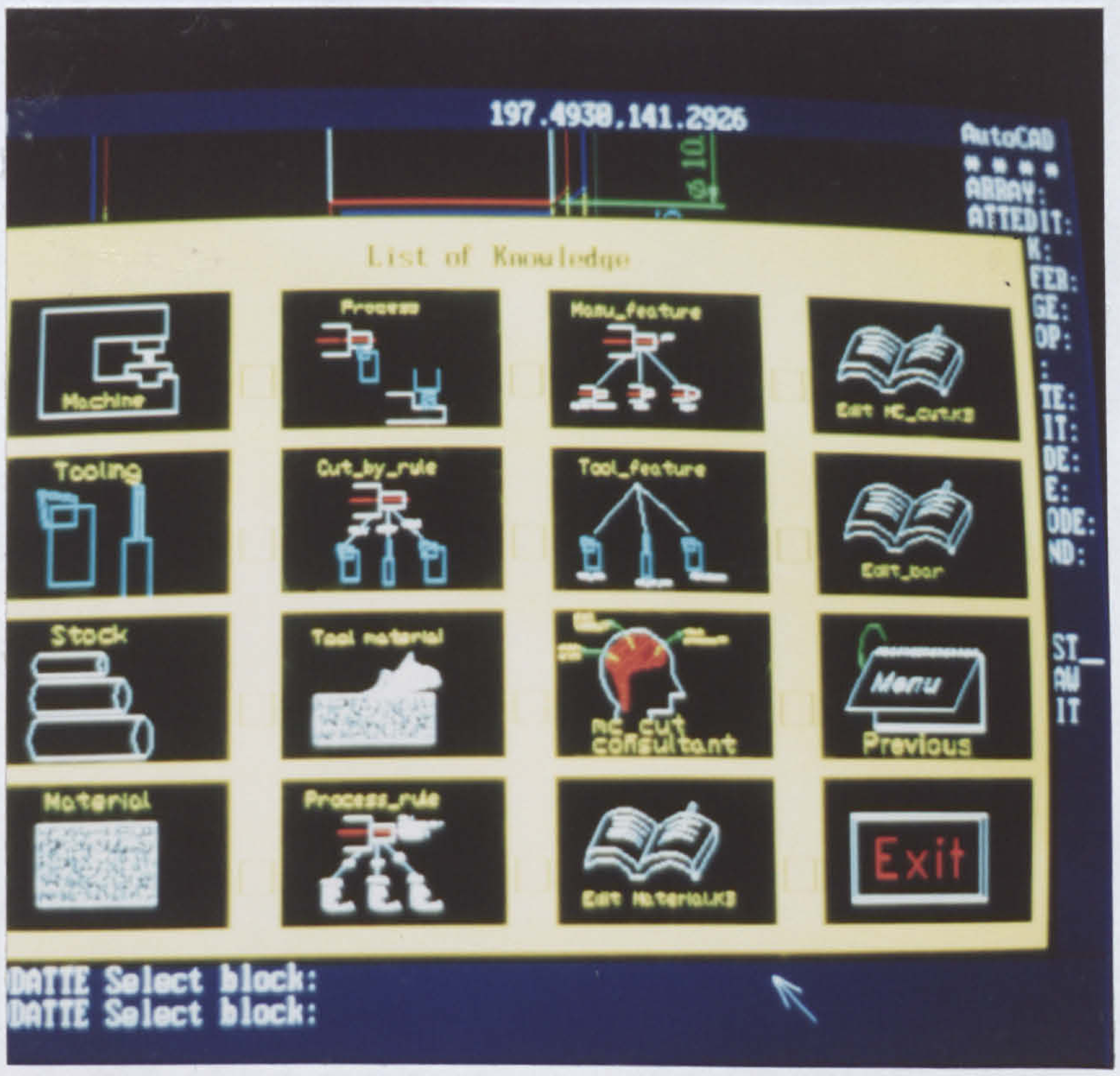

Figure 5-12: Menu to Access Manufacturing Knowledge-base.

(b) *rotational workpiece machining knowledge* -- the method and approach of cutting is made easy using this user friendly menu, figure 5-13; and finally

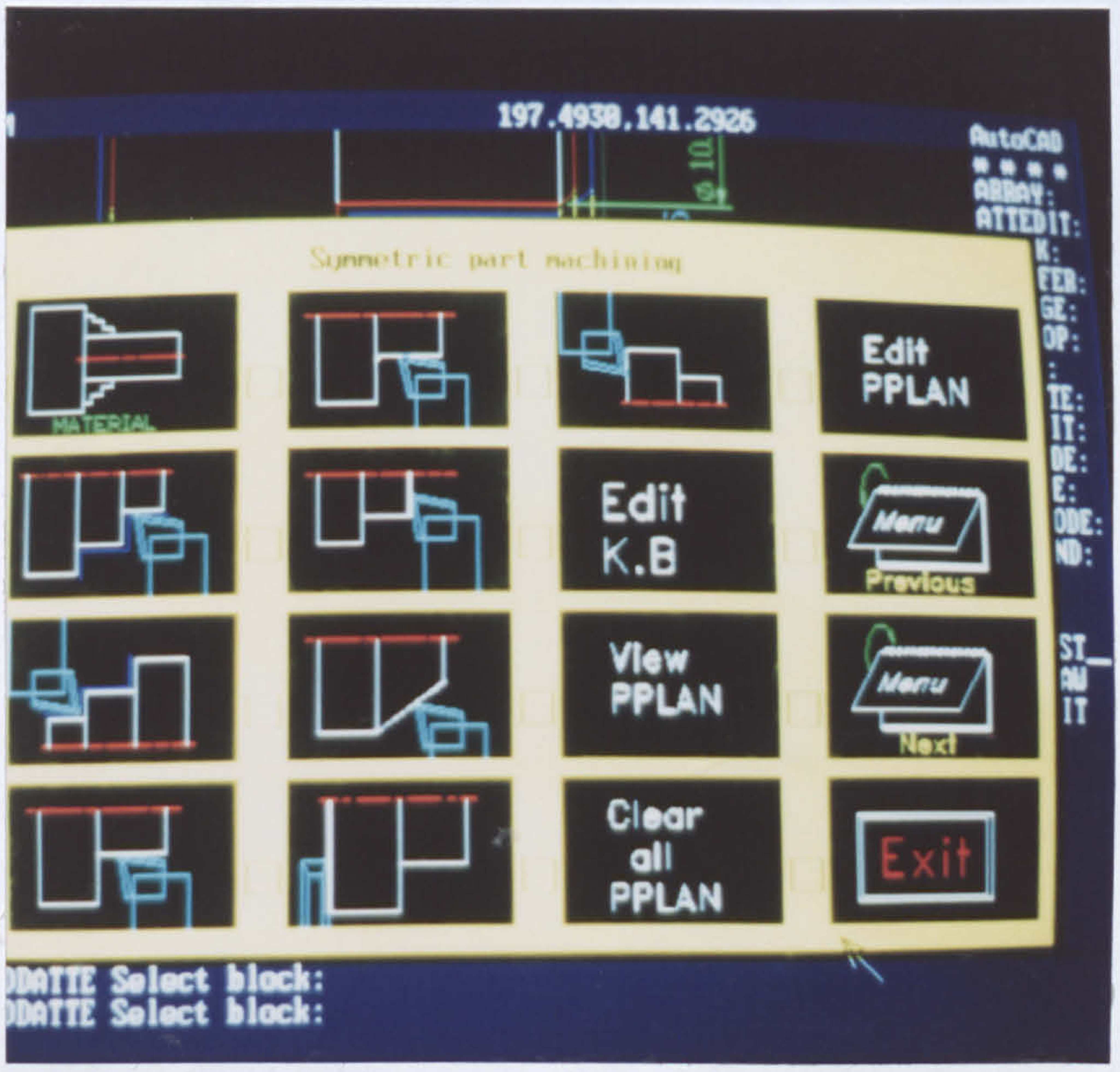

Figure 5-13: Rotational Part Machining Menu Knowledge-base.

(c) *milling process knowledge* -- using the same approach as rotational workpiece machining, the knowledge is expanded to include milling and similar processes.

Similar to the technique used in the feature recognition module, this module can retrieve the information from the CAD attribute database. Hence, manufacturing operations such as machining can be conducted automatically. The manufacturing module system structure is shown in figure 5-14.

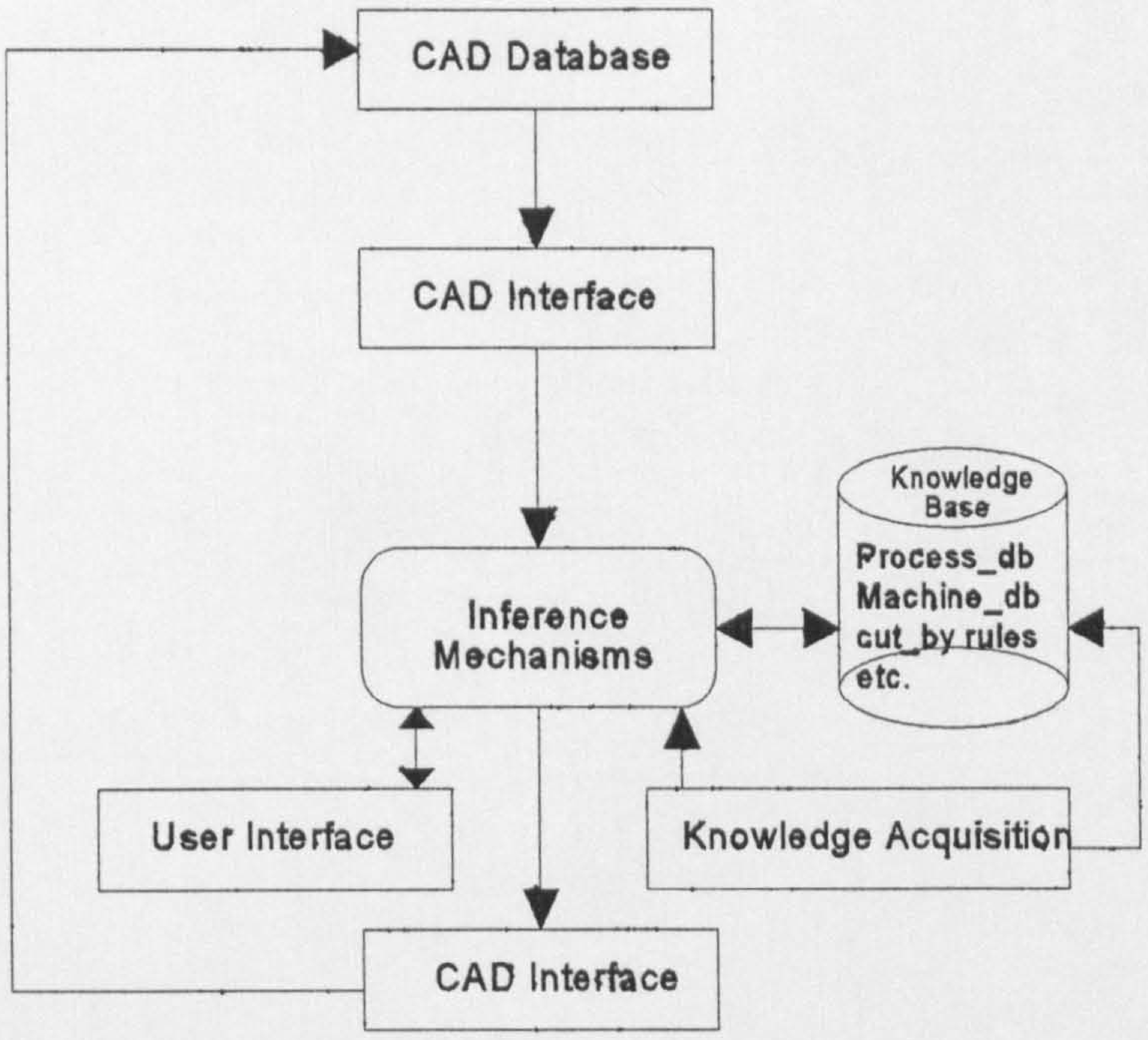

Figure 5-14: Knowledge-base Structure.

5.5.3.1 Knowledge Identification

Features can be processed by one or more manufacturing processes. For example, a flat surface of a metal block can be machined by a shaping machine or a milling machine. The type of machine selected to process the workpiece depends upon the tolerance and surface roughness requirements already specified in the blueprint.

Contemplating the idea of relationships, several factors have to be considered. The size, shape and type of tooling are selected based on the manufacturing feature and the type of process. Suitable tool material and cutting parameters, i.e. speed, feed and depth of cut have to be selected based on the type of workpiece material. This is to achieve the prespecified tolerance and surface roughness found in the blueprint drawing.

The interrelationship of these factors can be represented in a semantic network diagram as shown in figure 5-15.

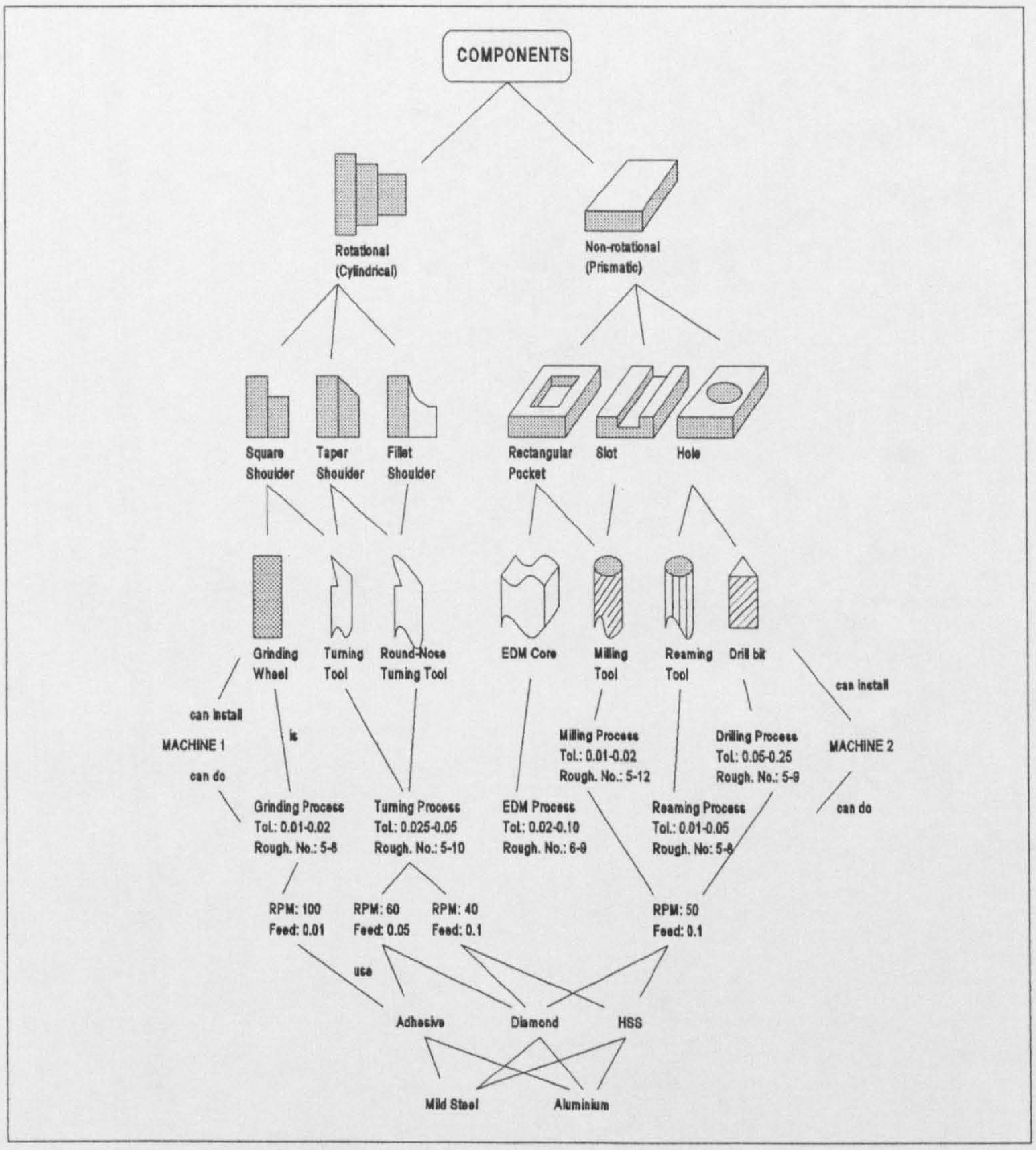

Figure 5-15: Semantic Network of the Related Information

Based on this semantic network diagram, the database for KATA can be stored in a frame-like format. The following section explains the knowledge representation used to accommodate the identification of semantic relationships.

5.5.3.2 Knowledge Representation

The machine and tool database structure are listed as:

```
machine_db(MC_no,        /* Machine no
    Name,                /* Name
    [Tol1,               /* practical stock removal tolerance (max. tol)
    Tol2,                /* Process capability (min. tol)
    Max_x,               /* Max-X-axis-length
    Min_y],              /* Min-Y-axis-length
    Mc_cost,             /* Machine operating cost
    Toolist,             /* Tool list
    Process_no).         /* Process_no
tool_db(Tno,             /* Tool no
    Name,                /* Name
    tool_dwg,            /* Tool drawing name
    Process_no,          /* Process no
    Tool_shape_no,       /* Tool feature no
    Tmatl_no,            /* Tool material
    [Max_rpm,            /* Max  rpm
    Max_feed,            /* Max feed
    Max_dc],             /* Max depth of cut
    Tw_factor,           /* Tool wear factor
    Tool_cost).          /* Tool operation cost
```

Examples of these knowledge contents are shown in figure 5-16 and 5-17.

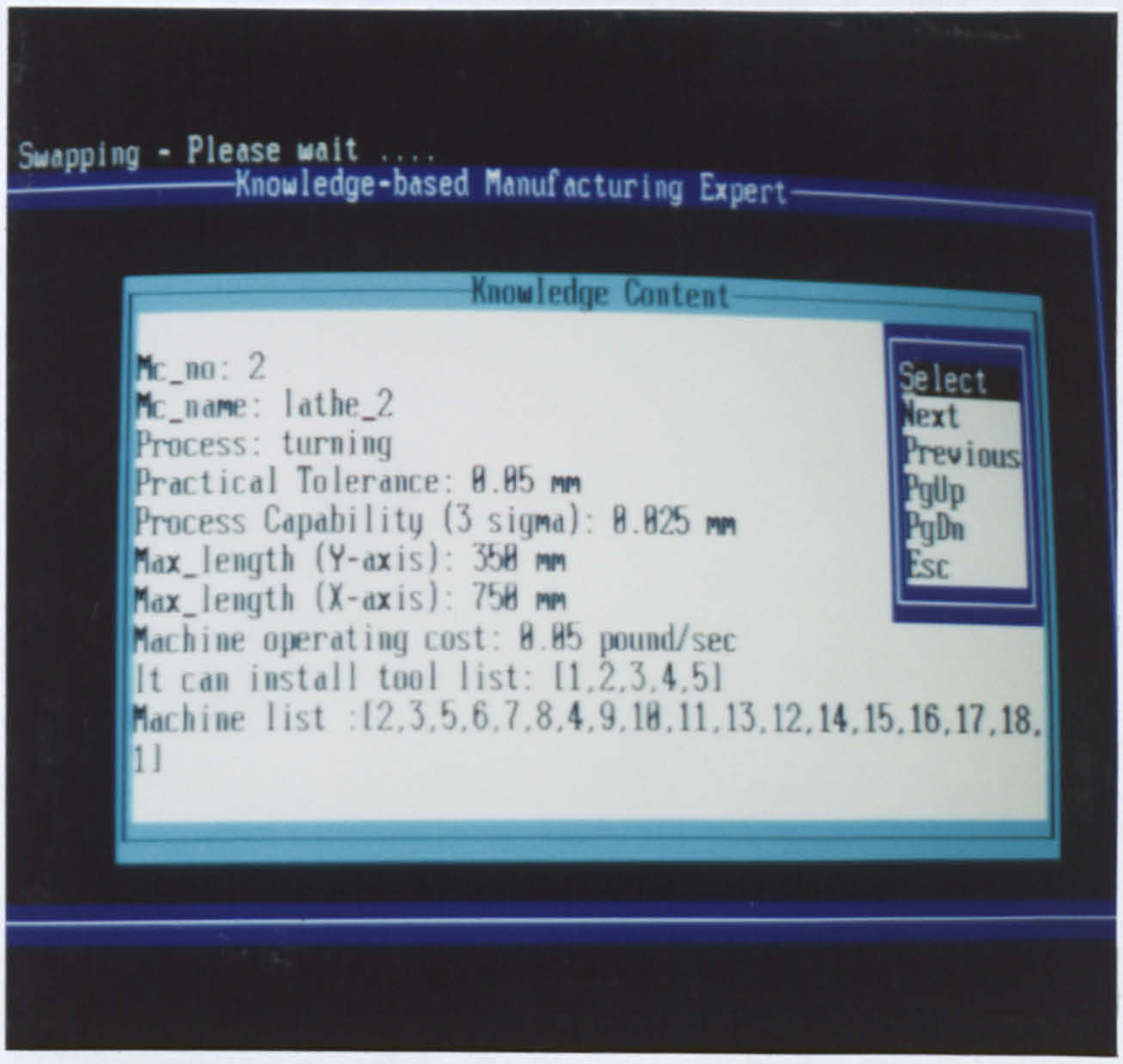

Figure 5-16: Machine Knowledge Representation.

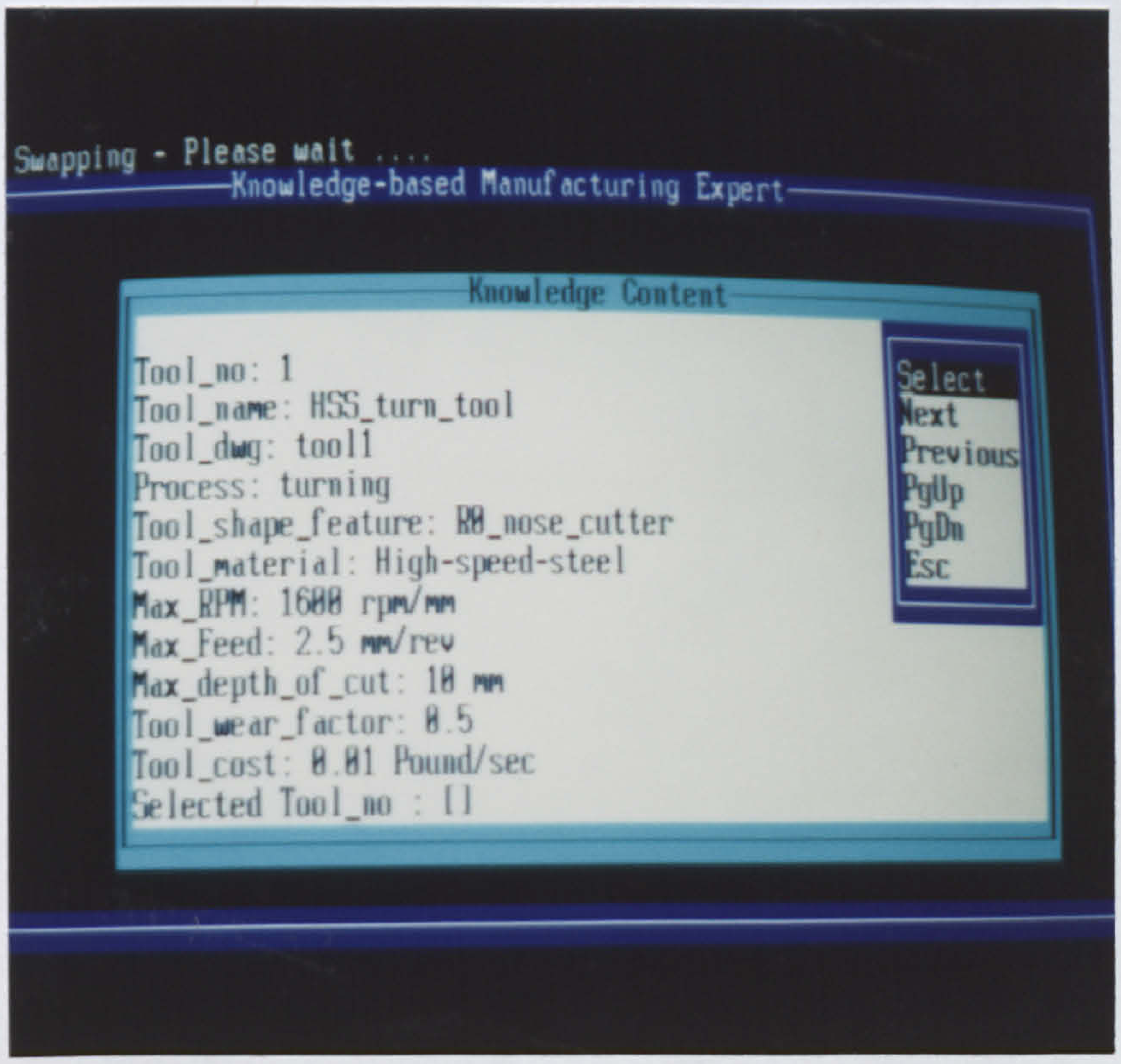

Figure 5-17: Cutting Tool Knowledge Representation.

The rule for selecting a tool can be represented by the following PROLOG predicate:

```
cut_by(Rno,                       /* Rule no
    Feature_no,                   /* manufacturing feature code
    [Y_len,                       /* Length (Y-axis) of the feature
    X_len,                        /* Length (X-axis) of the feature
    Fillet,                       /* Fillet of the feature
    Angle],                       /* Angle of the feature
    [Rel1,Rel2,Rel3,Rel4],        /* Comparison operators (eg. ≤,≥,=)
    Tool_shape_no).               /* recommended tool feature no
```

The predicate defines that, if the *feature_no, Y_len, X_len, Fillet* and *Angle* are matched, the recommended cutting tool is *Tool_shape_no*. The *process_db* predicate contains the process knowledge such that, if the *Matl_no, Lsr, Usr, Utol, Ltol* and *Rof* are matched, then the recommended process is Process_no and the cutting parameters are: *Rpm, Feed and D1*.

```
process_db(Rno,              /* Rule no
    Process_no,              /* Process_no
    Matl_no,                 /* Type material being cut
    Tmatl_no,                /* Tool material no
    [Lsr,Usr,                /* Lower and Upper limits of roughness no.
    Ltol,Utol],              /* Lower and Upper limits of process capability
    [Rpm,                    /* Recommended rpm (revolution/minute)
    Feed,                    /* Recommended feed
    D1],                     /* Depth of material for next cut
    Rof].                    /* Rough cut = 1, Final cut = 0
```

Examples of the knowledge content for tool and process selection is shown in figure 5-18 and figure 5-19.

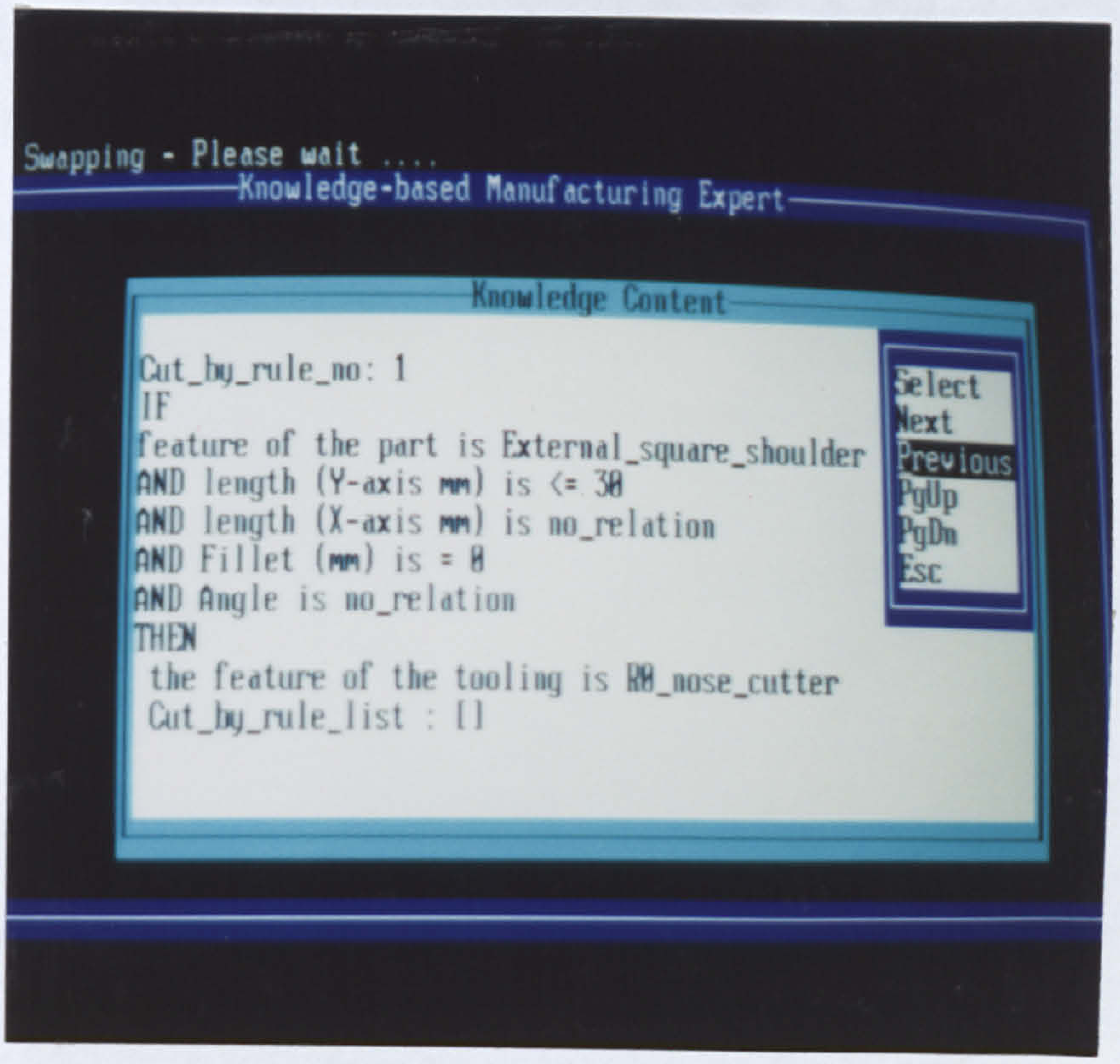

Figure 5-18: Cutting Tool rule.

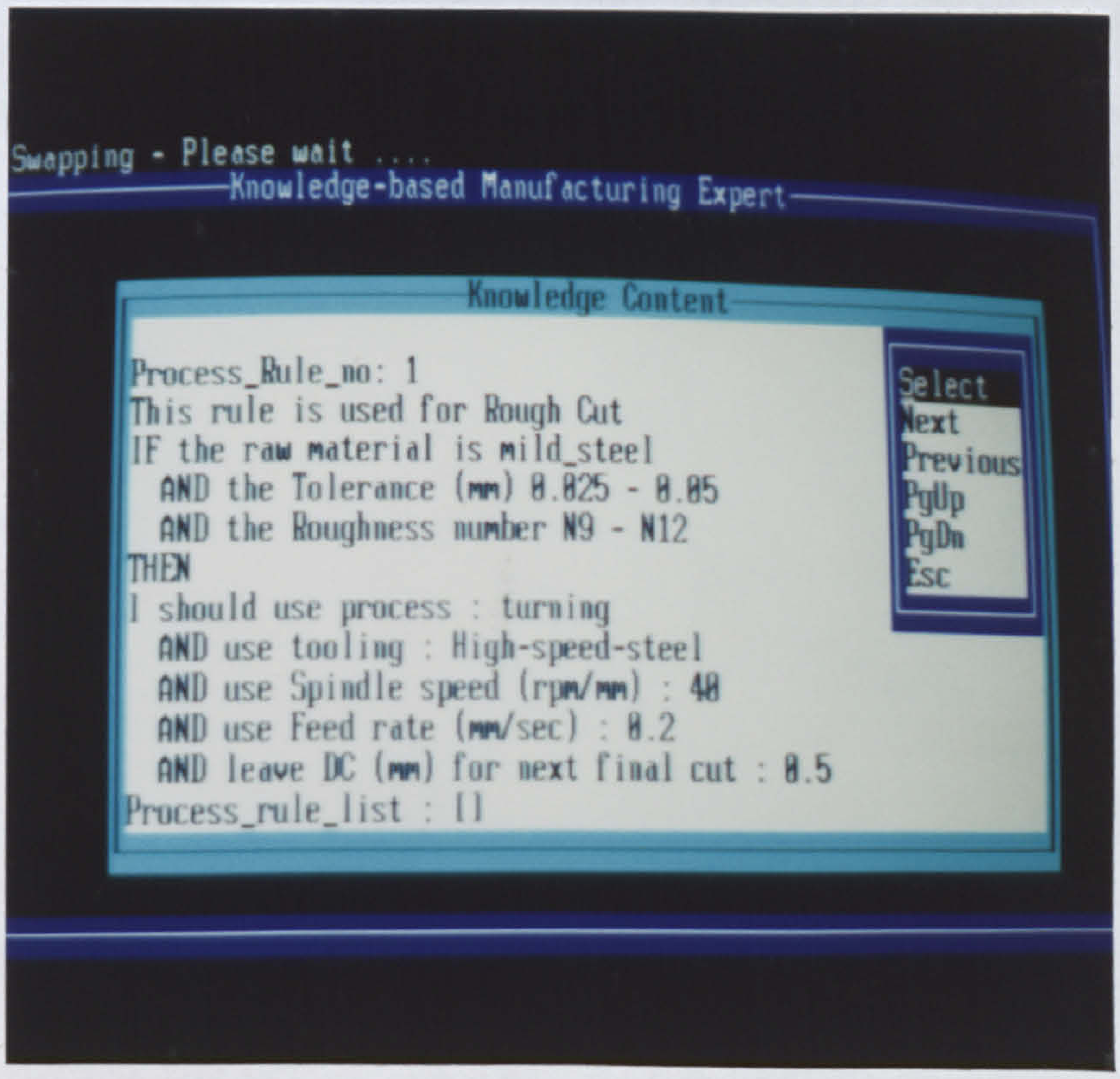

Figure 5-19: Process Selection Rule.

After all the information such as *Feature_no, X_len, Y_len, Angle, Fillet, Matl_no, Matl_X, Matl_Y, Rof, DC,* the blueprint and roughness number are known, the module can select the type of machine, type of process, type of cutters and cutting parameters by referring to the knowledge base. An advantage of PROLOG built-in backtracking mechanism is that the search approach can be either a top-down or bottom-up strategy as in figure 5-15.

The control predicate is as follows:

```
select(R):-
        t_shape_select(R1,Feature_no,Feature_dim),                 /* search cut_by rule
        process_select(R2,R1,Tol,Roughness_no,Matl_no,Rof),        /* search process_db
        tool_select(R1,R2,DC,R3),                                   /*search tool_db
        mc_select(R2,R3,Max_X,Max_Y),                               /* search machine_db
        R=[R1,R2,R3,R4],                                            /* else
        error([R1,R2,R3,R4]).                                       /* display
```

The inferences start with matching the conditions of *cut_rule*. If any of these rules is matched, the rule number *R1* will be fired. Then the control moves to the subgoal, *process_select*. This predicate matches the conditions of process rules, *process_db* with the value of *Tol, Surface_no,* and *Matl_no*. If it succeeds, the corresponding rule number will be stored to *R2*.

The backtracking occurs when the previously selected tool shape is not compatible to the selected process. For example, the *Flat_surface* can be cut by either a shaping or milling tool. The *t_shape_select* predicate selects the shaping tool first, it may be due to the position of the rule in the knowledge base. However, the second subgoal cannot find a suitable process in the knowledge base because shaping cannot machine the workpiece to the required tolerance and surface roughness specified in the design. Hence, the built-in backtracking mechanism is invoked and searches for an alternative tool from the *cut_by rules*.

The rule number *R1* and *R2* are subsequently sent to the third subgoal, *tool_select*, therefore, tools which matched the recommended *tool material, rpm, feed, depth of cut, process_no, tool shape*, etc., can be selected. Finally, the next subgoal, *mc_select* can select suitable machine by matching all conditions that come from rules *R1, R2, R3* and *Max_X* and *Max_Y*. If any one of these four subgoals is unsuccessful during firing, the system will display the error message and show which rules are unsuccessful.

5.5.4 Tolerance Analysis

KATA uses the worst-case tolerancing technique. The ability of the system to analyse tolerance using real time production capability data enables the system to improve the accuracy of the expected results after manufacturing.

As reported, the worst-case approach assumes that dimensions relating to processes or workpieces are at an allowable limit. If the dimension is defined as $d_o \pm t$, where d_o is the nominal dimension and t the tolerance, then the worst-case occurs when the actual dimension d is taken to be either d_o+t or d_o-t. This simple straight forward approach is very much appropriate for cases where substantial data on workpieces and production capabilities are obtainable. It is a pivotal advantage compared to *moment analysis* adopted widely in Statistical Tolerancing, which is less accurate and provides only limits on the probability of failure to assemble to specification.

In this module, the first half of the structure is required to sort out all linear dimensions and convert any special dimensions, eg. angular dimension, into linear dimensions. Then, the tolerance graph of dimensional chain for blueprint dimensions and stock removal can be generated as shown in figure 5-20.

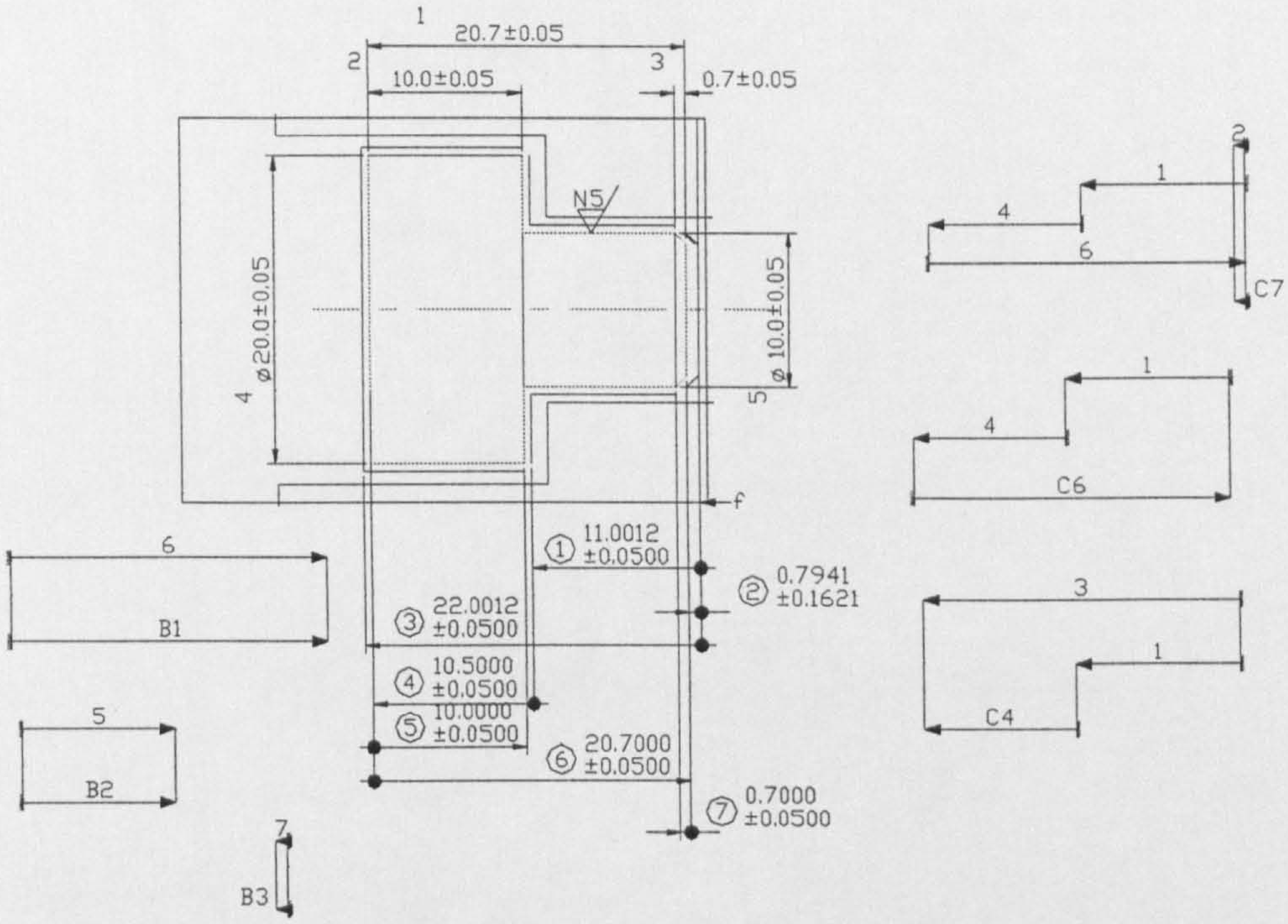

Figure 5-20: Processed Part Dimensional Chain.

This graphic representations of tolerance chains are transformed into linear programming models which can then be stored in the ASCII text file. Thus, this transformed data can be entered directly into any commercial linear programming software in this case LINDO, to do analysis of tolerance optimisations. Eventually, the result is sent to the CAD system and the attribute value is updated. The system structure for the module is shown in figure 5-21.

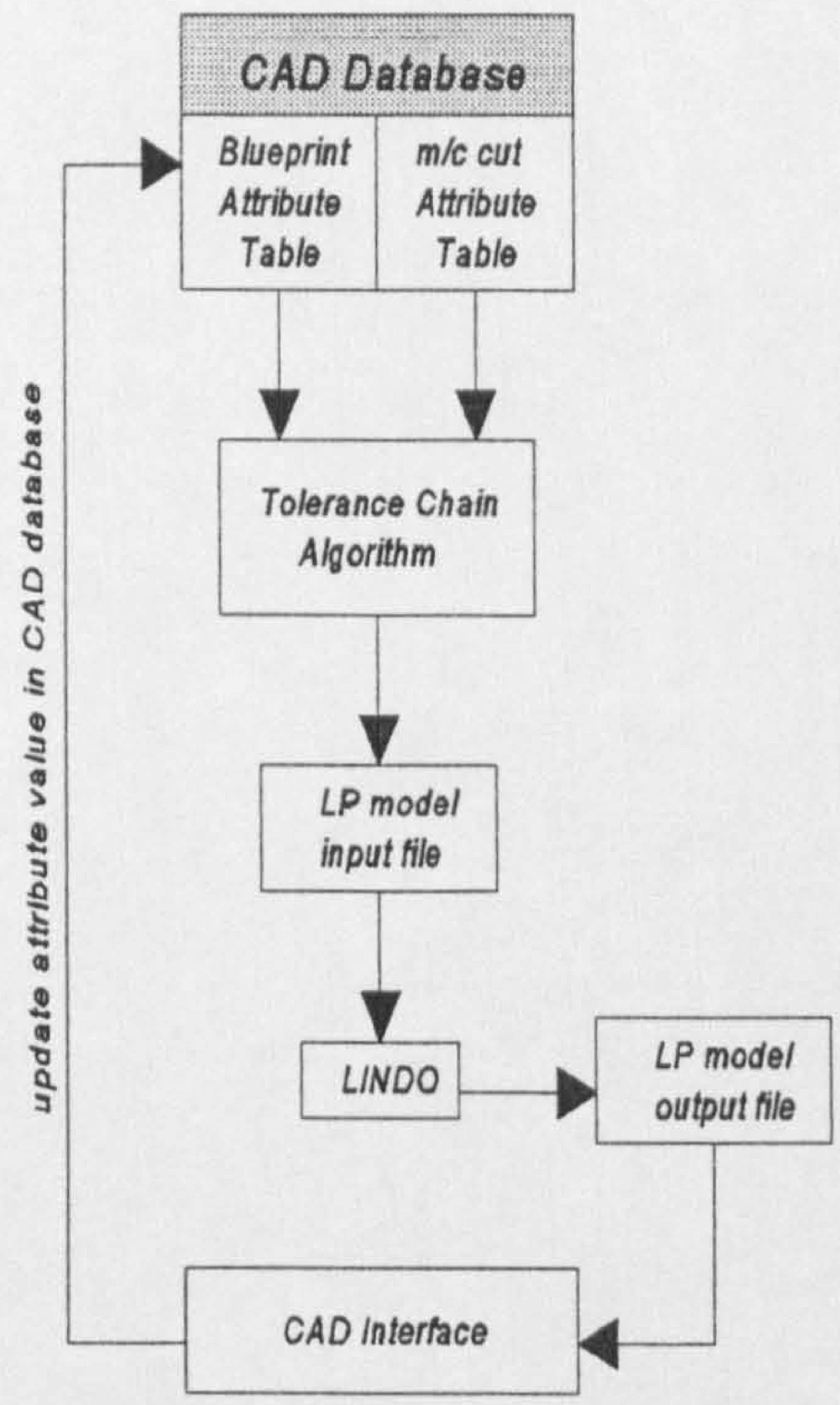

Figure 5-21: Tolerance Optimisation Structure

5.5.4.1 Feature Relationships

Graph theory is an important technique to trace out feature relationships. These relationship findings will aid the analysis of optimizing tolerance allocation to the workpiece. KATA has been developed to use the graph theory to form a set of linear equations. A graph representation for surface machining cut associativity based on the theory is generated. For the Steel Plug model used by Wade's example for formulating tolerance chart, figure 2.6, the graph representation of machining sequence, is shown in figure 5-22.

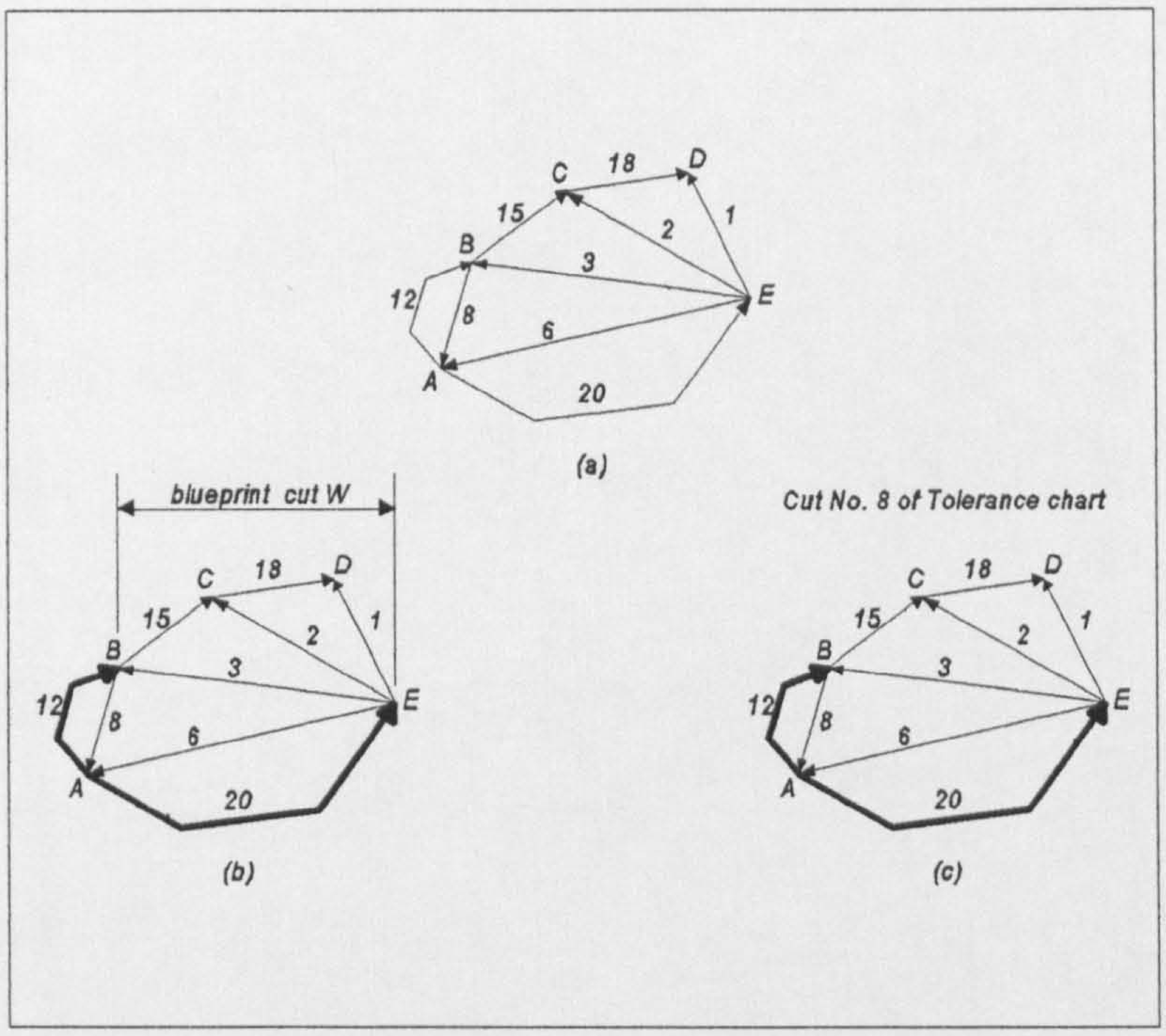

Figure 5-22: Graph Representation of Machining Sequence

This approach was first introduced by Irani, Mittal and Lehtihet to formulate linear dimensions and tolerances from a tolerance chart into a linear programming model. Each cut is represented as a numbered arc in the graph. Each arc number refers to the corresponding line assigned to the machining cut in the tolerance chart. The tail node of an arc represents the surface used as a datum for the cut. The head node corresponds to the surface on which material removal occurs. All machining cuts that contribute to the blueprint dimension and stock removal tolerances have to be combined and identified.

In the linear programming (LP) model, these set of cuts will constitute the decision variables appearing in the constraints of the tolerance optimisation model.

The search algorithms for the graph are:

(a) let (i, j) be the node pair corresponding to a blueprint dimension or stock removal arc and let N_{arc} be its line number, ie. the arc label;

(b) place nodes i and j in list L_i and place (i, j) in list L_{arc}, Set k = i;

(c) examine all machining cut arcs and identify each arc (n, k). Of these arcs, choose the arc with the maximum line number (N) less than N_{arc}. Place node n in L_i and (n, k) in L_{arc}. Let N_{arc} = N and k = n;

(d) repeat step (c) until no arc (n,k) is found;

(e) place node j in list L_j. Identify arc (m, j) as per step (c). Place node m in L_j, then stop; else repeat step (e).

The subsequent objective of tolerance analysis is to assign the maximum feasible tolerance to each cut without violating constraints on blueprint dimensions, stock removal and process capabilities. Thus, from the graph created, relationships between each feature have been identified and now a set of constraint equations can be formulated. This will automatically assign maximum permissible values to machining tolerances.

5.5.4.2 Linear Programming Model

The next objective of tolerance optimisation is to assign the maximum feasible tolerance to each cut without violating constraints on blueprint dimension and process capabilities. Three sets of constraints are used to achieve this objective.

Constraint set one is to express accumulated tolerances on cuts generating each blueprint dimension in equality form. Constraint set two represents the accumulation of tolerance on combinations of cuts occurring in schematics for stock removals. Limits on stock removal tolerance are defined by the process planner or suggested by the knowledge base. Constraint set three reflects the influence of process capability of each machine tool on the tolerance produced in a cut. The maximum accuracy available on the machine tool is the restriction on achievable tolerance.

This LP model for allocation of the tolerances to cuts, eliminates the need for assumptions and surmise values. Such an approach as this has made the use of current production capabilities when installed in the knowledge base a significant advantage for the analysis. Constraints can be appended to the formulation, or weights attached to certain slack variables to interactively modify the solution. A linear programming analysis and its constraint's formulation can be derived as:

Minimize:

$$\sum_{i=1}^{n} z_i + \sum_{j=1}^{m} w_j$$

Subject to:

$$\sum_{k \varepsilon BPi} t_k + z_i = b_i \qquad i=1,2,....,n, \qquad \text{(constraint 1)}$$

$$\sum_{k \varepsilon SRj} t_k + w_j = s_j \qquad j=1,2,....,m, \qquad \text{(constraint 2)}$$

$$t_k \geq LPC_k \qquad k=1,2,....,p, \qquad \text{(constraint 3)}$$

where:

z_i = residual tolerance in the ith blueprint constraint;

t_k = tolerance on the kth machining cut;

BP_i = set of machining cuts in the schematic for the ith blueprint dimension;

b_i = maximum design tolerance allocated for the ith blueprint dimension;

n = number of blueprint dimensions specified on the component drawing;

SR_j = set of machining cuts in the schematic for the jth removal;

w_j = residual tolerance in the jth stock removal constraint;

s_j = maximum tolerance on the jth stock removal;

m = number of stock removal lines on the tolerance chart;

LPC_k = lower limit on the tolerance for cut k defined by the process capability of the machine tool assigned; and

p = number of machining cuts in the process plan.

5.5.4.3 Angular Cut

Workpieces sometime have one or more angular features. Such angular features can range from a simple chamfer to a tapered surface. In suitable combinations, BS 308 recommends the following detailing in the drawing:

(a) the diameter (or height) at each end of the tapered/angle feature;

(b) the length of the tapered/angle feature;

(c) the diameter (or height) at a selected cross-sectional plane that may be within the tapered/angle feature or outside; and

(d) the dimension locating a cross-sectional plane, at which the diameter (or height) is specified; and the rate of taper or the included angle.

Figure 5-23 shows some typical combinations recommended by BS 308 for specifying the size and form of angle features.

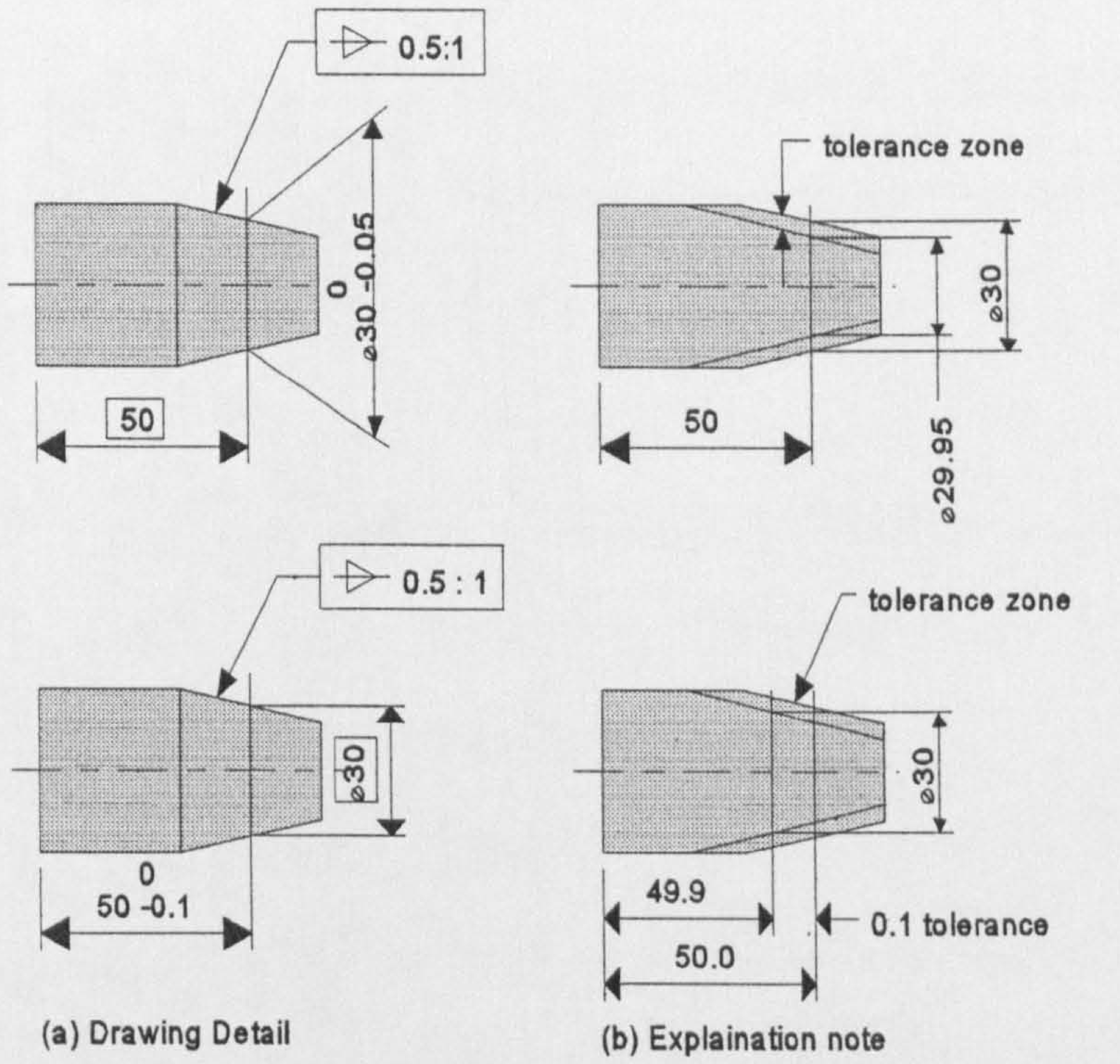

Figure 5-23: Dimension and Tolerance an Angular Object.

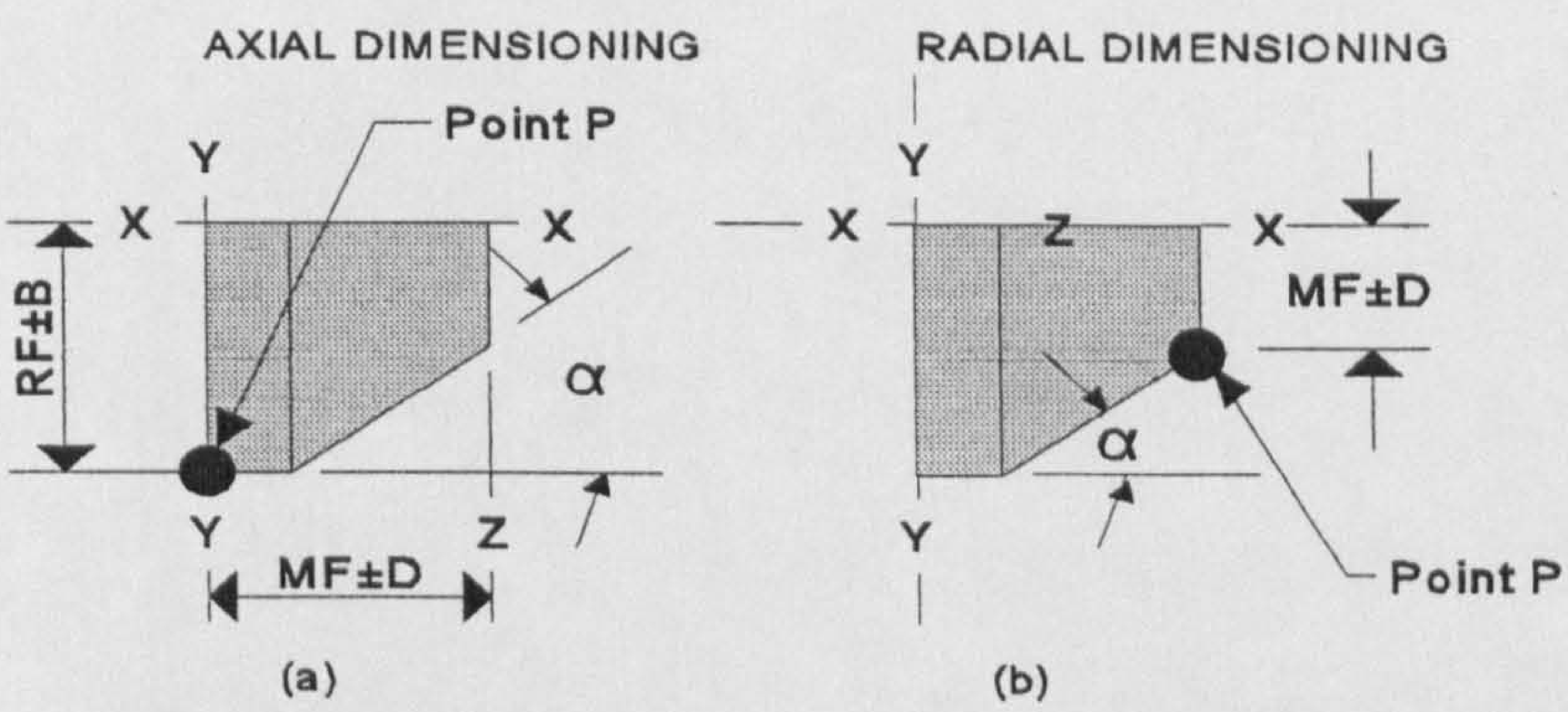

Figure 5-24: Detail of Angular Dimensioning Scheme.

Figure 5-24 shows an axial and radial dimensioning detail. In an axial dimension, the blueprint angle dimension, *MF±D*, to the point *P* is measured along the X-X axis of the part (like square shoulder features). In a radial dimension the $MF \pm D$ dimension to the point P is measured along the Y-Y axis of the part (like radial or cylindrical features). Trigonometric formulas are used to convert diameters of cylindrical workpieces, or from the angular surface itself to sets of linear values along the X-X axis.

For proper use of the formula, the angle α, must satisfy the following conditions:

(a) must be the mean value. If the angle given on the blueprint is not in equal bilateral form, it must be converted to obtain the mean value; and

(b) the angle α, is between the angular surface and the X-X axis and not the Y-Y axis. If the angle of the angular surface is given on a blueprint taken from the Y-Y axis, this angle β, should be converted to its complementary value α.

To produce the corresponding blueprint angle as defined in figure 5-24, a standard angle machining cut method describe extensively in [44] is employed. In each standard angle machining method, a rough angle machining cut, $MR1 \pm C1$, initially establishes a point P. Following this, one or two subsequent machining cuts can be taken, moving the point P to the finished location defined by the $MF \pm D$ blueprint dimension. The angle cut machining methods are presented as both two and three stage machining processes and are classified as Group I, II, III-A and III-B. Group I machining methods cover the remachining of the boundary surfaces that delimit the length of the angular surface. Group II methods cover the remachining of the angular surface; Groups III-A and III-B cover methods that are combinations of Groups I and II.

Using these standard angle machining cut methods as a basis, trigonometric formulae are derived for calculating the rough angle machining cut *MR1±C1* for Group I and II and the rough *MR1±C1* and finish machining cut *MR2±C2* for Groups III-A and III-B. Detailed trigonometric calculations of these stages can be referred to in [44]. Thus, these formulae for angular cut machining can be used as a constraint in KATA when performing the rough and finish angle cut. This constraint will then seek to reduce the cumulative residual tolerance.

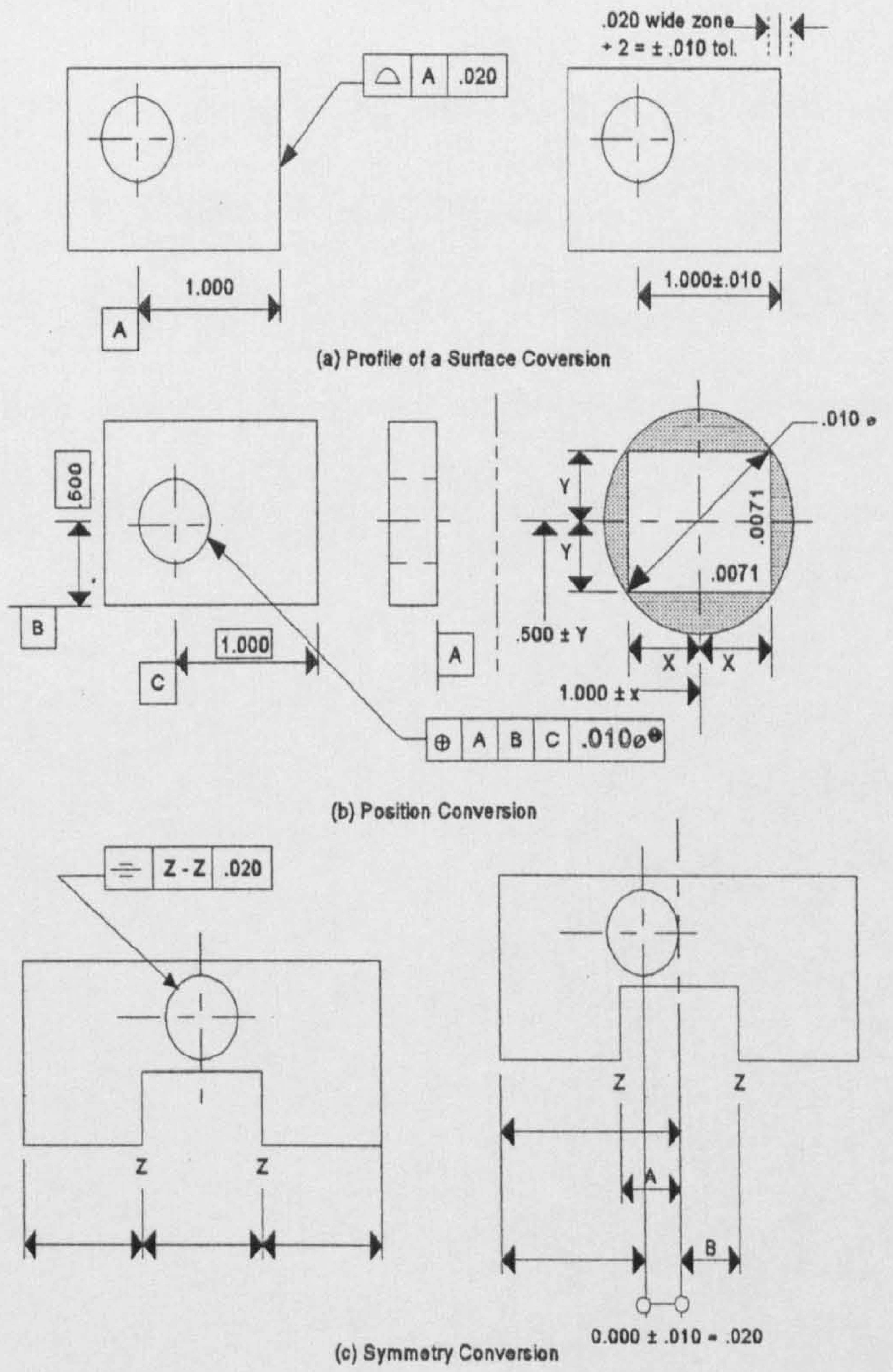

Figure 5-25: Conversion of Geometric Characteristics to mean size ± tolerance.

5.5.4.4 Geometrical Tolerance Analysis

Four out of fourteen geometric tolerances namely, profile of a plane, position, symmetry and concentricity are controlled in KATA. In order for these geometric tolerances to be factored into KATA, the blueprint defined values must be converted into the equivalent equal bilateral form. The required conversions for profile of a plane, position, symmetry and concentricity are shown in figure 5-25. Concentricity conversion is also handled in the same manner.

5.6 KATA's Knowledge

A significant advantage of KATA over other systems is it's generic nature. The ability to add, edit, and delete types of machines, cutting tools and materials in the CAD database at any time allows KATA to be always up to date with its analysing capabilities. Thus, the manufacturing engineer is consistently informed about plant capabilities allowing the manufactured part to be competitive with other vendors.

CHAPTER 6

EXPERIMENTAL METHOD

6.1 Workpiece

Three sample workpieces will be used to validate the system. The three workpieces are: (a) Plug 1 (P1); (b) Plug 2 (P2) and (c) Block 1 (B1). Specification for P1, P2 and B1 is shown in figure 6-1, 6-2 and 6-3 respectively.

P1 and B1 are sample workpieces made by this investigator to test KATA's ability and knowledge-base. P2 is used by several investigators [44,46,53,62] to validate their optimisation technique. Thus, a comparative study can be made on the analysis result produced by KATA and previous approaches.

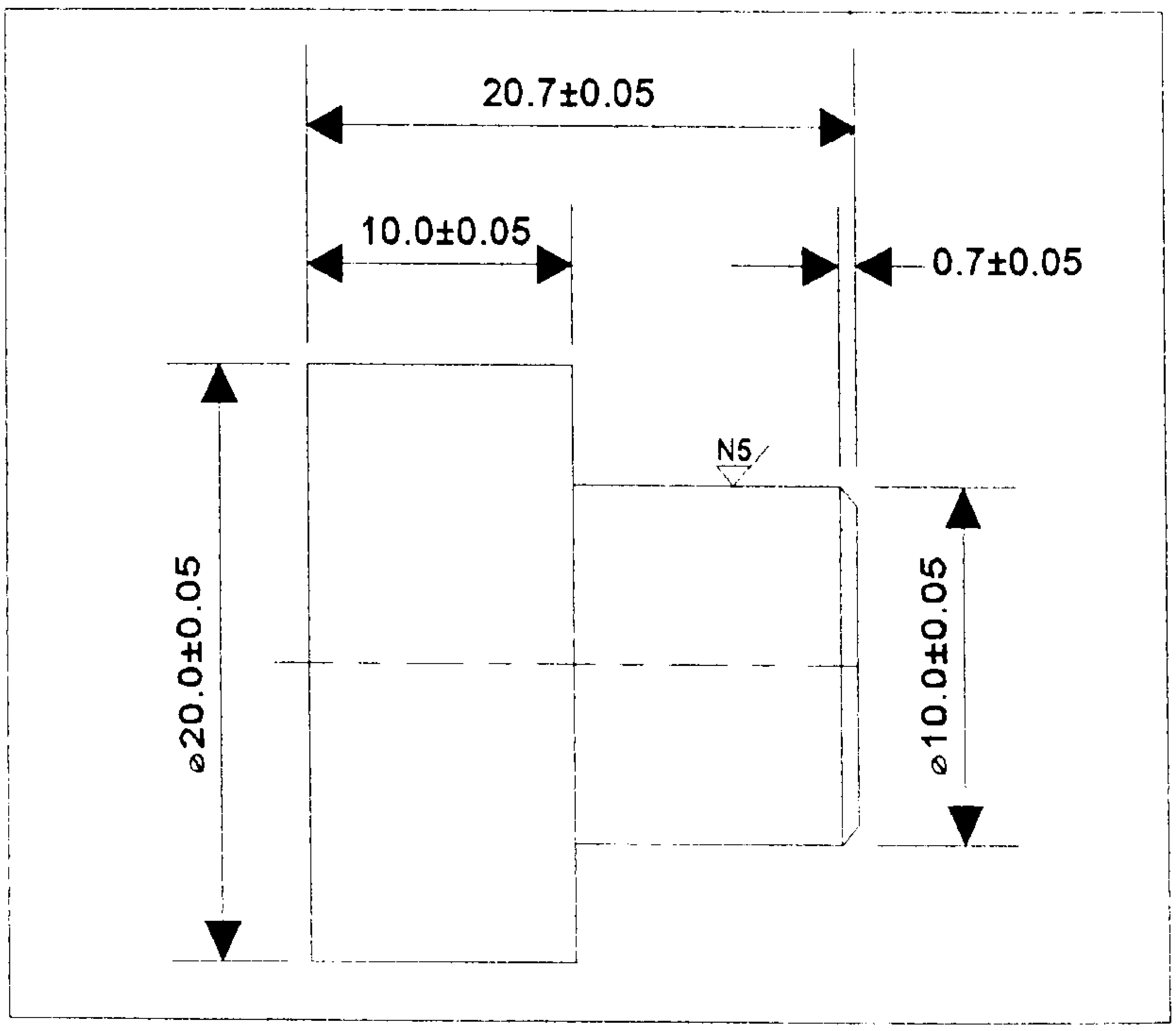

Figure 6-1: Plug P1

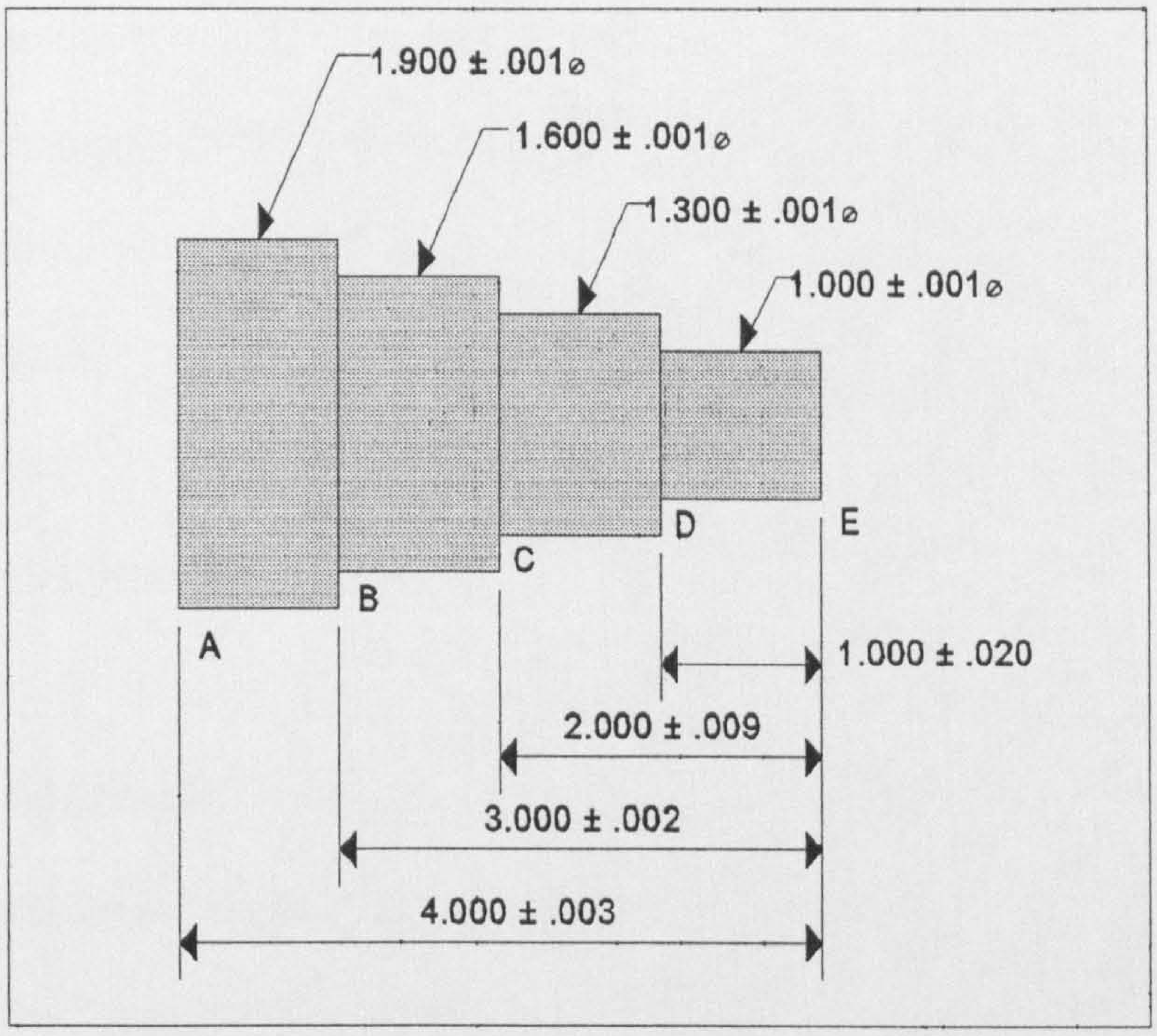

Figure 6-2: Plug P2.

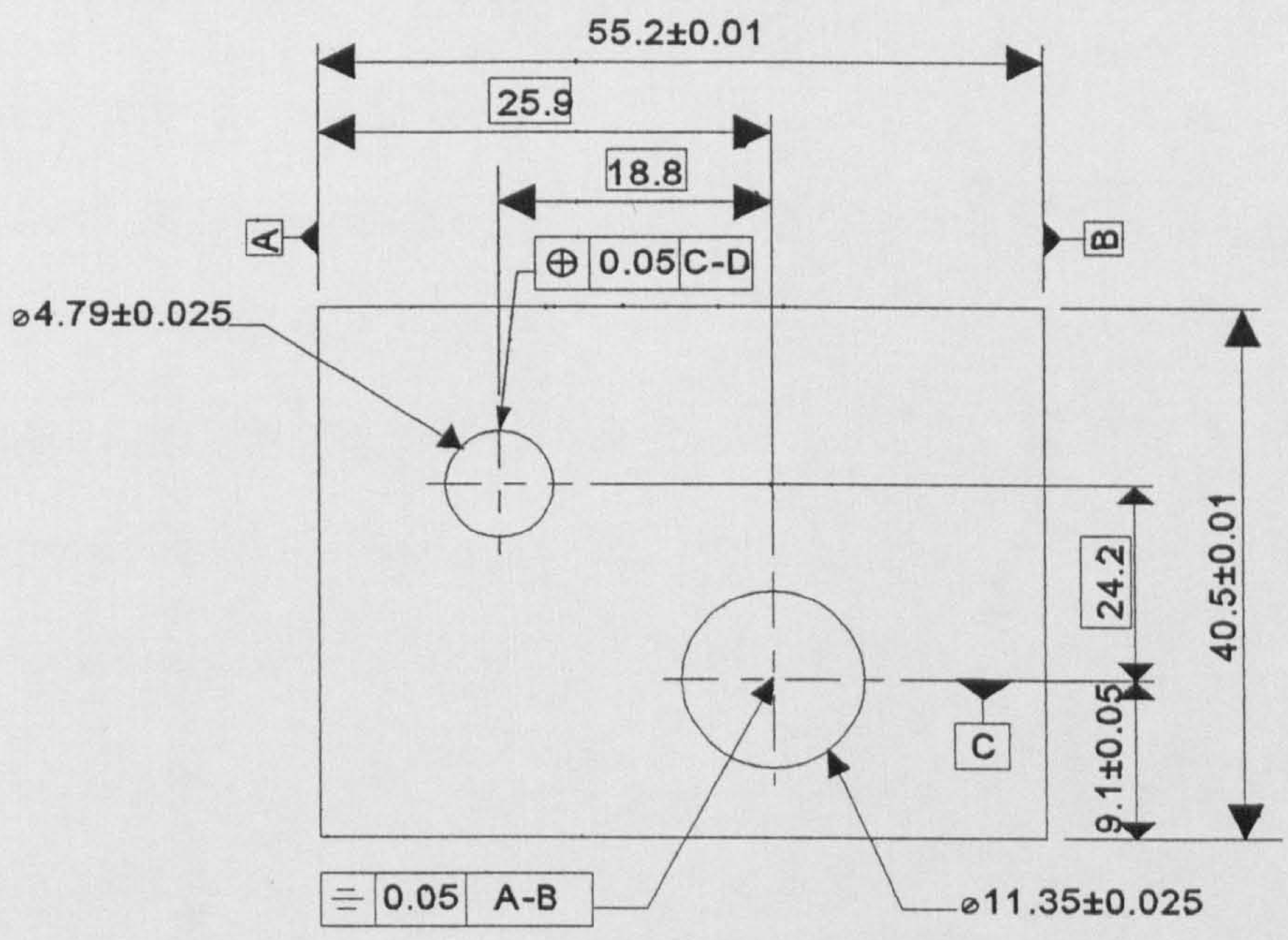

Figure 6-3: Block B1.

6.2 Workpiece Material

Three types of material specification have been preinstalled in the knowledge-based system. The available expert system rules are specifically fitted to process these three types of materials, namely:

(a) Aluminium;

(b) Brass; and

(c) Mild steel (Low Carbon Steel 1010, 1020).

6.3 Knowledge-based System

Detailed explanation of the development of the knowledge-based tolerance analysis system can be found in Chapter 5. The system known as KATA and AutoCAD, is hosted by the following computer hardware specifications:

(a) 80486-DX33 system running MS-DOS 5.01;

(b) Logitech serial mouse;

(c) 1.2 Mb, 5¼ inches floppy drive and 1.44 Mb, 3½ inches floppy drive;

(d) 8 Mb random-access memory;

(e) Super VGA colour monitor and card;

(f) 2 parallel and 1 serial port, and

(e) 250 Mb hard disk (KATA and AutoCAD occupy only 25 Mb).

6.4 Machining

A range of machines are preinstalled in KATA. Practical tolerances produced by the machine along with other characteristics are included in the database as processing constraints. The list can be found in table 6-1.

6.5 Tooling

Recommended cutting speeds and feeds can be referred to in the literature [16,17,18]. High speed steel and cemented carbides are the most commonly used cutting tool materials. Other materials used to make cutting tools are cemented oxides or ceramics, cermets, cast nonferrous alloys (stellite), single crystal diamonds, polycrystalline diamonds and cubic boron nitride.

For this experiment, only a high speed steel type of cutting tool is installed in KATA. The list is in table 6-2. In the database, the recommended processing characteristics such as speed, feed and depth of cut are included as constraints for processing.

Machine Number	Machine Name	Process	Practical Tolerance (mm)	Process Capability (3σ)	Maximum Length Y-axis (mm)	Maximum Length X-axis (mm)	Operating Cost ($/sec)
1	Lathe 1	Turning	0.05	0.025	350	750	0.01
2	Lathe 2	Turning	0.05	0.025	350	750	0.05
3	Grinder 1	Cylindrical grinding	0.02	0.01	250	600	0.1
4	Drilling	Drilling	0.25	0.05	100	100	0.5
5	Reaming	Reaming	0.05	0.01	100	100	0.5
6	Boring	Boring	0.05	0.025	300	300	0.7
7	Milling	Milling	0.02	0.01	500	1000	0.5
8	Broaching	Broaching	0.05	0.01	300	300	0.6
9	Shaping	Shaping	0.125	0.025	1000	1000	0.2
10	EDM	EDM	0.1	0.02	100	100	0.5
11	Press	Punch & Die	0.04	0.01	100	100	0.4

Table 6-1: Range of Machines in KATA's database.

Tool No.	Tool Name	Process	Tool Shape	Tool Matl.	Max. RPM	Maximum Feed (mm/rev)	Maximum Depth of Cut (mm)	Tool Wear Factor	Tool Cost ($/sec)
1	Turn	Truning	Round Nose	HSS	1600	2.5	10	0.5	0.01
2	Cut-off	Turning	Cut-off	HSS	1600	2.5	10	0.5	0.01
3	Round Nose 0.5	Turning	Round Nose	HSS	1600	2.5	10	0.5	0.01
4	Round Nose 1.0	Turning	Round Nose	HSS	1600	2.5	10	0.5	0.01
5	Round Nose 1.5	Turning	Round Nose	HSS	1600	2.5	10	0.5	0.01
6	Grinding Wheel	Cylindrical Grinding	Grind Wheel	Abrasive	2000	0.5	5	0.05	0.03
7	Grinding Wheel	Cylindrical Grinding	Grind Wheel	Abrasive	2000	0.5	5	0.05	0.03
8	Twist Drill	Drilling	Hole Cutter	HSS	200	0.5	10	0.5	0.1
9	Reamer	Reaming	Hole Cutter	HSS	100	0.5	10	0.5	0.5
10	Boring	Boring	Hole Cutter	HSS	100	0.5	10	0.5	0.5
11	Centre Drill	Centre Drilling	Hole Cutter	HSS	100	0.5	10	0.5	0.5
12	Mill Cutter	Milling	Mill Cutter	HSS	200	0.5	10	0.5	0.5

Table 6-2: Cutting Tool Characteristics in KATA's database.

CHAPTER 7

RESULTS

7.1 Introduction

The objectives of this experiment are two fold. First is to optimise tolerance allocation to the workpieces. The optimisation procedure is subjected to constraints such as blue print specification and manufacturing process capabilities.

Second is to prove that knowledge-based engineering can assist the tolerance optimisation procedure. This assistance can therefore *intelligently* inform the process engineer the plant ability to process the workpiece concerned. Hence, this approach will significantly improve analysis of tolerance concerning real time production and residual tolerance accuracy.

7.2 Rotational Machining

Two tests have been conducted using part P1 and P2. Based on the preliminary test, KATA can simulate actual production procedure in detail. Every inch of machining is categorically registered starting with the rough cut until the part is fully machined.

7.2.1 Part P1

Figure 7-1 illustrates part P1 before the working tolerance is optimised. The detail of figure 7-1 include the blueprint dimension and tolerance, the rough machining cut and the finish cut. Cut numbers ①, ②, ③ and ④ are the rough machining cuts. Cuts ⑤, ⑥ and ⑦ are the finish cuts.

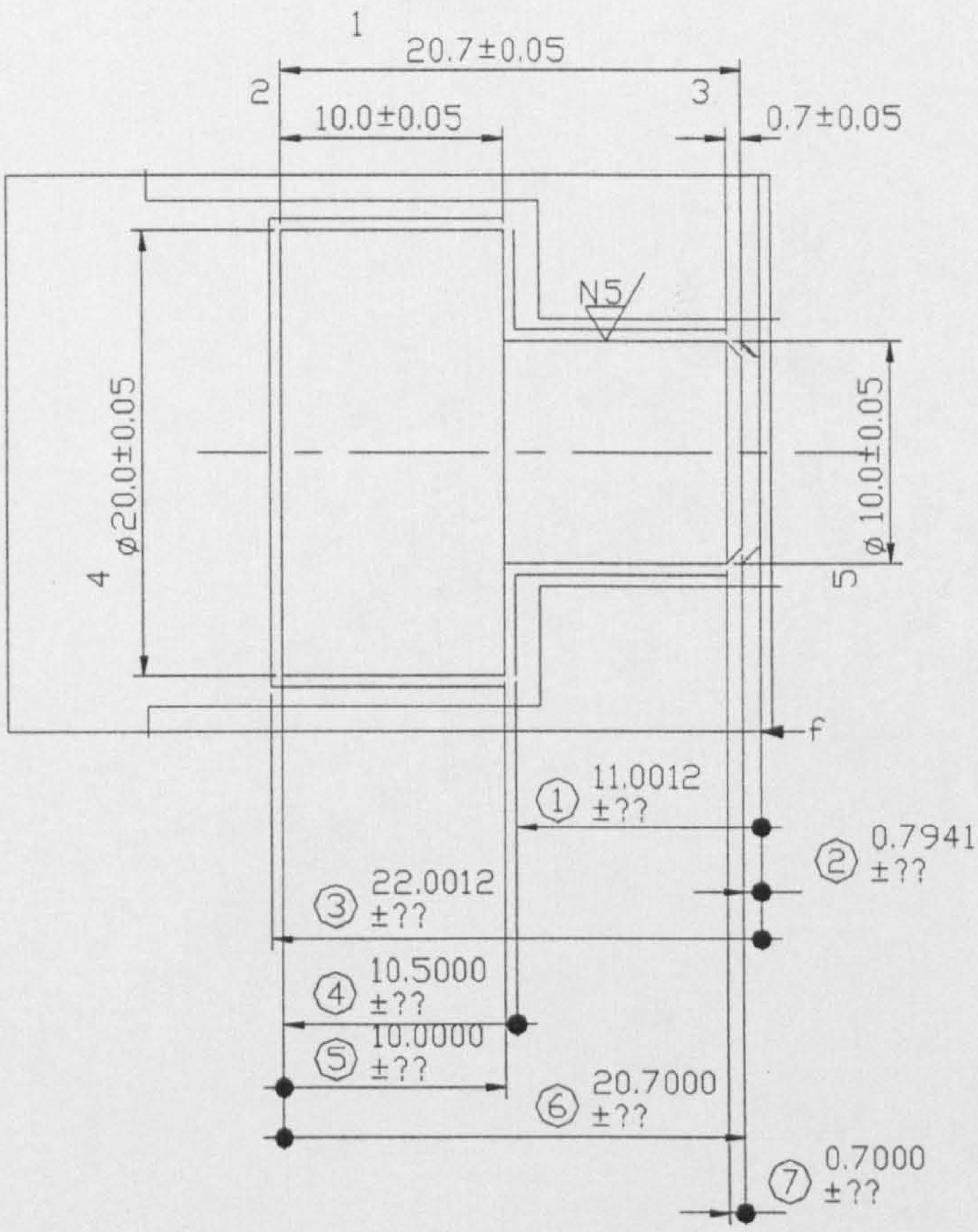

Figure 7-1: P1 before Tolerance Optimisation.

Figure 7-2 illustrates part P1 after the working tolerance is optimised. In figure 7-2, it also includes the blueprint resultant graphs, B1, B2 and B3 and the stock removal graphs, C4, C6 and C7. Based on the blueprint resultant graphs, B1, B2 and B3 are directly decided by the final cut number ⑥, ⑤ and ⑦ respectively. The stock removal graphs on the other hand, explain that the accumulation of cut C6 for example is equal to the combination of cut 4 and cut 1.

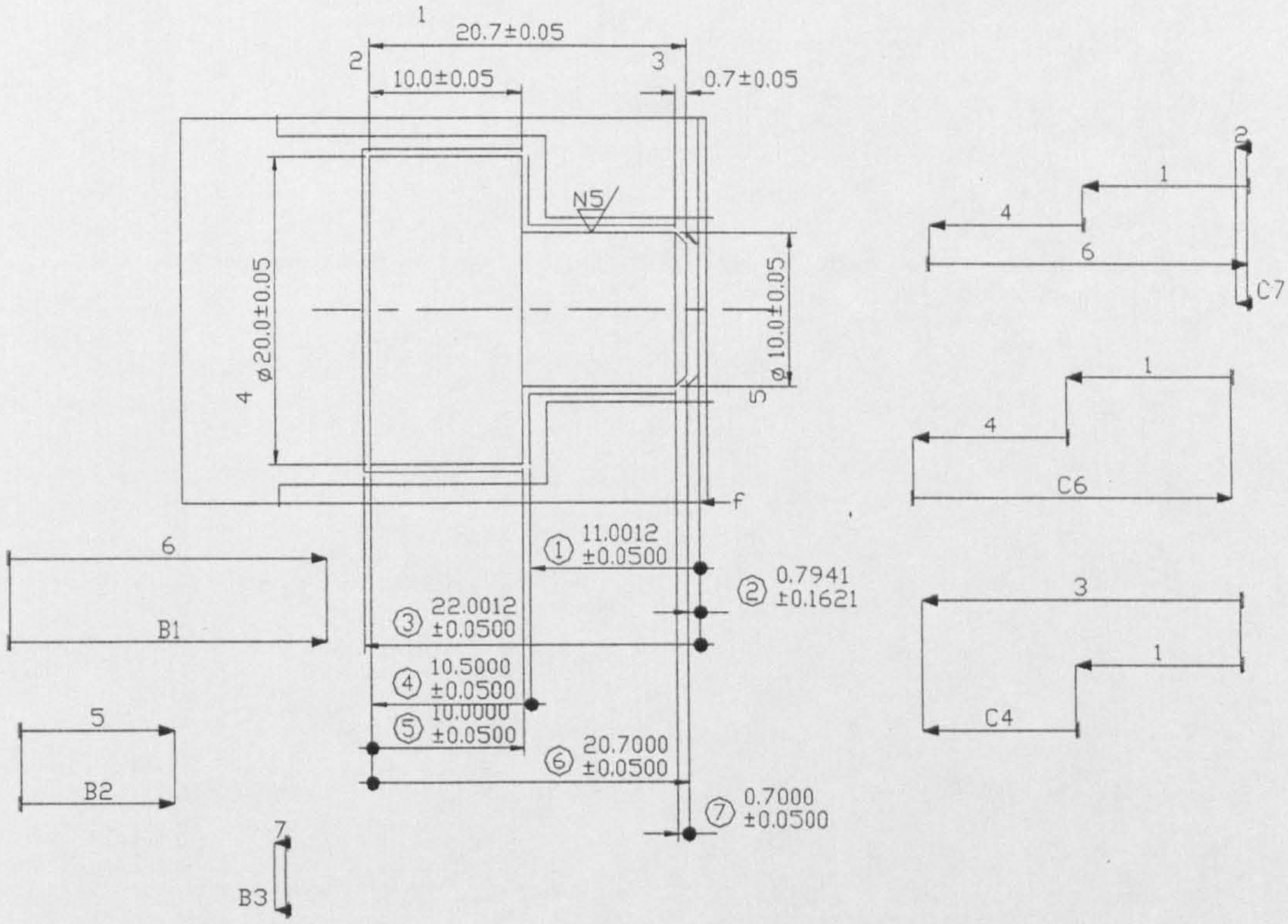

Figure 7-2: P1 after Tolerance Optimisation.

After processing part P1, KATA stores all information pertaining to the machining sequence and its related data. This information is important especially when preparing the process and tool sheets to be handed out to the shop floor for reference to machine part P1. The information can also contain the cost of machining and tooling which can assist to evaluate the cost of the finish part per production. Partial detail of the information stored is given in table 7-1 for rough cut and table 7-2 for finish cut.

CUT NUMBER	1	2	3	4
PROCESS NAME	Turning	Turning (chamfer)	Turning	Turning
MACHINE NUMBER	1	1	1	1
MAX. PROCESS TOL. (mm)	0.05	0.05	0.05	0.05
MIN. PROCESS TOL. (mm)	0.025	0.025	0.025	0.025
MACHINING COST ($/sec)	0.11	0.01	0.12	0.39
TOOL NUMBER	1	1	2	1
TOOL WEAR FACTOR	0.5	0.5	0.5	0.5
TOOL COST ($/sec)	0.14	0.01	0.21	0.2
RPM	800	400	1060.05	310
FEED (mm/rev)	0.1	0.1	0.1	0.1
DIM (mm)	11.0012	0.7941	22.0012	10.5
B/P CONS. TOL. (mm)	0.05	0.2121	0.05	0.05
RESULTANT TOL. (mm)	0.05	0.1621	0.05	0.05

Table 7-1: KATA's Rough Cut Data.

CUT NUMBER	1	2	3
PROCESS NAME	Turning	Turning	Turning (chamfer)
MACHINE NUMBER	1	1	1
MAX. PROCESS TOL. (mm)	0.05	0.05	0.05
MIN. PROCESS TOL. (mm)	0.025	0.025	0.025
MACHINING COST ($/sec)	0.23	0.15	0.03
TOOL NUMBER	1	1	1
TOOL WEAR FACTOR	0.5	0.5	0.5
TOOL COST ($/sec)	0.15	0.04	0.01
RPM	400	171.72	200
FEED (mm/rev)	0.1	0.1	0.1
DIMENSION (mm)	10	20.7	0.7
B/P CONS TOL. (mm)	0.05	0.5	0.05
RESULTANT TOL. (mm)	0.05	0.5	0.05

Table 7-2: KATA's Finish Cut Data.

Table 7-3 compares the results produce by KATA with the dimension and tolerance required by the blueprint.

BLUEPRINT	10.0 ± 0.05	20.70 ± 0.05	0.70 ± 0.05
KATA	10.0000 ± 0.05	20.7000 ± 0.05	0.7000 ± 0.05

Table 7-3: Blueprint and KATA's Results.

7.2.2 Part P2

P2 is analysed by first converting the specification value that is in an imperial units into metric. This is largely due to KATA's knowledge since its database is written for metric analysis. The conversion is shown in Table 7-4.

ORIGINAL VALUES (jn)	CONVERTED VALUE (cm)
4.000 ± 0.005	10.16 ± 0.013
3.000 ± 0.002	7.62 ± 0.005
2.000 ± 0.009	5.08 ± 0.023
1.000 ± 0.020	2.54 ± 0.051

Table 7-4: P2 Value Conversion.

In addition to P2 that is different from the original validation made by previous investigators is the inclusion of a surface roughness number. Surface roughness can further refine the finish requirements of the workpiece, for example, a surface should be finer because of tight fitting during the assembly of the product. The inclusion of a surface roughness number in the analysis will make better refinement in the process selection. Here a roughness average number of *5* is used. This surface roughness number is typical for turning and grinding processes.

The result of P2 analysis using KATA is shown in figure 7-3.

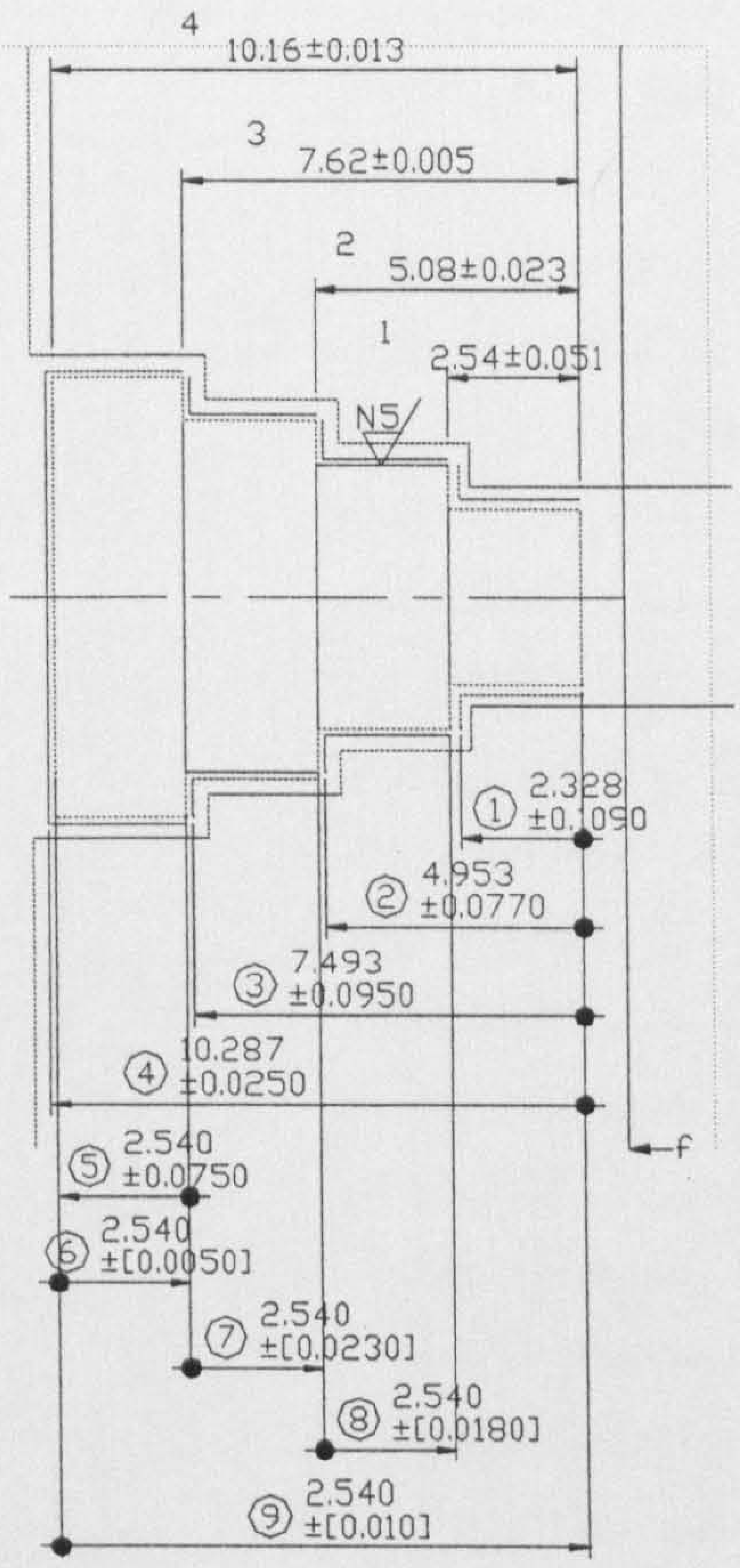

Figure 7-3: P2 Analysis.

Cuts ① to ④ are roughing cuts, thus, the residual tolerance of each cut is negligible. However as a benchmark for a further operation, the rough cut is always set to be greater than or equal to the maximum process tolerance.

Cuts ⑤ to ⑨ are the finish cuts for part P2. From the analysis, KATA produces information that is based on the available process capability, there is no feasible solution for P2. This is represented by the bracket tolerances in cuts ⑥ to ⑨ and shown in detail in KATA's LP model:

```
Min   w1+w2+w3+w4+w5+z1+z2+z3+z4

st
t5+t4+w1=0.100                  /* stock removal cut 5
t6+t3+w2=0.100                  /* stock removal cut 6
t7+t2+w3=0.100                  /* stock removal cut 7
t8+t1+w4=0.100                  /* stock removal cut 8
t9+t5+w5=0.100                  /* stock removal cut 9
z1+t9+t8=0.051                  /* blueprint resultant 1
z2+t9+t7=0.023                  /* blueprint resultant 2
z3+t9+t6=0.005                  /* blueprint resultant 3
z4+t9=0.010                     /* blueprint resultant 4
t1>=0.025                       /* minimum process tolerance (selected by KBS)
t2>=0.025                       /* minimum process tolerance (selected by KBS)
t3>=0.025                       /* minimum process tolerance (selected by KBS)
t4>=0.025                       /* minimum process tolerance (selected by KBS)
t5>=0.025                       /* minimum process tolerance (selected by KBS)
t6>=0.025                       /* minimum process tolerance (selected by KBS)
t7>=0.025                       /* minimum process tolerance (selected by KBS)
t8>=0.025                       /* minimum process tolerance (selected by KBS)
t9>=0.025                       /* minimum process tolerance (selected by KBS)
end
```

Mathematically, the following calculations illustrated the negative slack that occurs during optimisation analysis. KATA's preselection of machine has chosen a 0.05 maximum process tolerance and a minimum of 0.025 process tolerance. *t1* to *t9* in the LP model shows that the process tolerance for each process is greater than or equal to 0.025. Therefore, the residual tolerance after the blueprint constraints are:

$$z1 = 0.051-(t9+t8) = 0.051-0.050 = 0.010$$

$$\left.\begin{array}{l} \mathbf{z2 = 0.023-(t9+t7) = 0.023-0.050 = -0.027} \\ \mathbf{z3 = 0.005-(t9+t6) = 0.005-0.050 = -0.045} \\ \mathbf{z4 = 0.010-t9 = 0.010-0.025 = -0.015} \end{array}\right\} \quad \text{Negative Slacks}$$

These negative slack figures have proven that the present process capability installed in KATA is not able to process the type of tolerance required by P2. Thus, the next action that should be taken is either:

(a) change the blueprint dimension accordingly, or

(b) upgrade the process ability.

Table 7-5 illustrates the remaining P2 machining parameters.

CUT NO.	1	2	3	4	5	6	7	8	9
PROCESS NAME	Turning	Turning	Turning	Turning	Turning	Turining	Turning	Turning	Turning
MACH. NO.	1	1	1	1	1	1	1	1	1
MAX. PROCESS TOL. (mm)	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
MIN. PROCESS TOL. (mm)	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025
MACHINING COST ($/sec)	0.09	0.07	0.06	0.09	0.22	0.06	0.08	0.2	0.08
TOOL NO.	1	1	1	2	2	1	1	1	1
TOOL WEAR FACTOR	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
TOOL COST ($/sec)	0.07	0.07	0.07	0.17	0.16	0.08	0.08	0.08	0.04
RPM	480	640	800	1160	440	800	640	240	320
FEED (mm/rev)	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
DIMENSION (mm)	2.328	4.953	7.493	10.287	2.54	2.54	2.54	2.54	10.16
B/P CONS. TOL. (mm)	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
RESULTANT TOL. (mm)	0.109	0.077	0.095	0.025	0.075	[0.005]	[0.023]	[0.018]	[0.010]

Table 7-5: P2 Machining Data.

7.3 Simple Prismatic Machining

Part B1 is used to test KATA's ability to machine simple prismatic parts. Thus, processes such as milling, drilling and the like can also be tested apart from the regular turning and facing processes. Another aspect of verification tested using part B1 is in the area of geometric tolerance. Two types of geometric tolerances are tested namely symmetrical and positional.

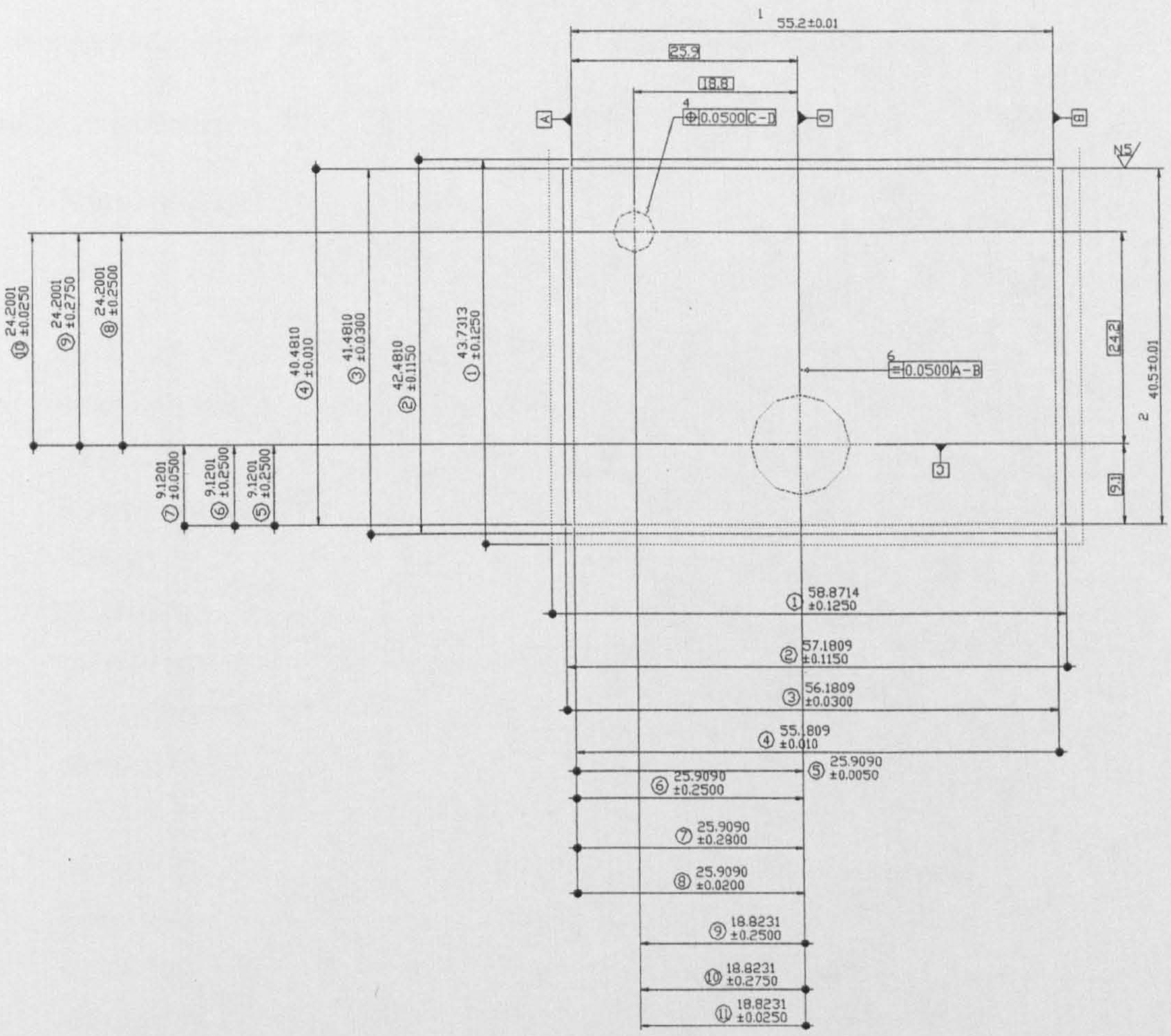

Figure 7-4: B1 Tolerance Optimisation.

7.3.1 Part B1

Figure 7-4 describes B1 after it is processed. Geometric tolerance number 4 controls the positional tolerance of the smaller hole (hole 1). The datum point for this small hole is the centre of the large hole (hole 2). Symmetry for hole 2 is controlled by the geometric tolerance number 6. Therefore, a dummy cut number 5 to compensate an optimised calculation for the cut is added to the model just after cut number 4. Tolerance accumulation for cut number 6 is the combination of tolerances number 5 and 8. Further verification can be looked into from the dimensional chain process. The position tolerance of hole 1 is shown to be controlled directly by cut number 11. LP model for the lower half of model B1 is as follows:

```
Min w1+w2+w3+w4+z1+z2+z3

st
t3+t2+w1=0.1450          /* stock removal cut number 3
t4+t3+w2=0.0400          /* stock removal cut number 4
t8+t7+w3=0.3000          /* stock removal cut number 8
t11+t10+w4=0.3000        /* stock removal cut number 11
2t5-t4=0                 /* dummy cut constraint
z1+t4=0.010              /* blueprint resultant 1
z2+t11=0.0250            /* blueprint resultant 2
z3+t8+t5=0.0250          /* blueprint resultant 3
t1>=0.125                /* minimum process tolerance for shaping
t2>=0.025                /* minimum process tolerance for shaping
t3>=0.010                /* minimum process tolerance for milling
t4>=0.010                /* minimum process tolerance for milling
t6>=0.250                /* minimum process tolerance for drilling
t7>=0.050                /* minimum process tolerance for drilling
t8>=0.010                /* minimum process tolerance for reaming
t9>=0.250                /* minimum process tolerance for drilling
t10>=0.050               /* minimum process tolerance for drilling
t11>=0.010               /* minimum process tolerance for reaming
t5>=0.005                /* dummy cut after cut number 4 - milling process
end
```

Mathematically verifying the finding:

$$z1 = 0.010 - t4 = 0.010 - 0.010 = 0.000$$

$$z2 = 0.025 - t11 = 0.025 - 0.010 = 0.015$$

$$z3 = 0.025 - (t8+t5) = 0.025 - 0.015 = 0.010$$

CUT NO.	1	2	3	4	5
PROCESS NAME	Shaping	Shaping	Milling	Milling	Dummy
MACHINE NUMBER	9	9	7	7	-
MAX. PROCESS TOL. (mm)	0.125	0.125	0.02	0.02	0.02
MIN. PROCESS TOL (mm)	0.025	0.025	0.01	0.01	0.005
MACHINING COST ($/sec)	14.13	13.72	33.71	33.11	-
TOOL NO.	14	14	12	12	-
TOOL WEAR FACTOR	0.5	0.5	0.5	0.5	-
TOOL COST ($/sec)	5.89	5.72	28.09	27.59	-
RPM	100	100	100	100	-
FEED (mm/rev)	0.5	0.5	0.5	0.5	-
DIMENSION (mm)	58.874	57.1809	56.1809	55.1809	25.909
B/P CONS. TOL. (mm)	0.125	0.125	0.02	0.02	0.01
RESULTANT TOL. (mm)	0.125	0.115	0.03	0.01	0.005

Table 7-6 (a): Machining Attribute for B1.

Based on the blueprint tolerance constraints, the resultant is equal to or less than the blueprint specifications. This shows that the resultant working tolerance suggested by KATA is at its optimum, based on the available capabilities and do not violate the rule of blueprint requirements. Attribute information in table 7-6 (a) and (b) explained further how this model achieved its optimal solution.

CUT NO.	6	7	8	9	10	11
PROCESS NAME	Centre Drill	Rough Drill	Reaming	Centre Drill	Rough Drill	Reaming
MACHINE NUMBER	4	4	5	4	4	5
MAX. PROCESS TOL. (mm)	0.25	0.25	0.05	0.25	0.25	0.05
MIN. PROCESS TOL (mm)	0.25	0.05	0.01	0.05	0.05	0.01
MACHINING COST ($/sec)	0.63	0.63	0.63	1.5	1.5	1.5
TOOL NO.	11	8	9	11	8	9
TOOL WEAR FACTOR	0.5	0.5	0.5	0.5	0.5	0.5
TOOL COST ($/sec)	6	1.2	6	6	1.2	6
RPM	1134.84	1134.84	1134.84	478.5	478.5	478.5
FEED (mm/rev)	0.5	0.5	0.5	0.5	0.5	0.5
DIMENSION (mm)	25.909	25.909	25.909	18.8231	18.8231	18.8231
B/P CONS. TOL. (mm)	0.25	0.25	0.05	0.25	0.25	0.05
RESULTANT TOL. (mm)	0.25	0.28	0.02	0.25	0.275	0.025

Table 7-6 (b): Machining Attribute for B1 (continued).

7.4 Single Reference Analysis

A significant finding resulting from P2's validation is the way the design tolerances were assigned to the features.

The basis of tolerance stackup theory is that tolerance accumulates, notwithstanding, whether dimensions from individual cuts are added or subtracted. Nevertheless, if the sum of the tolerances of smaller features exceed the tolerance of the overall feature, then stackup theory is no longer valid. An example is shown in figure 7.5.

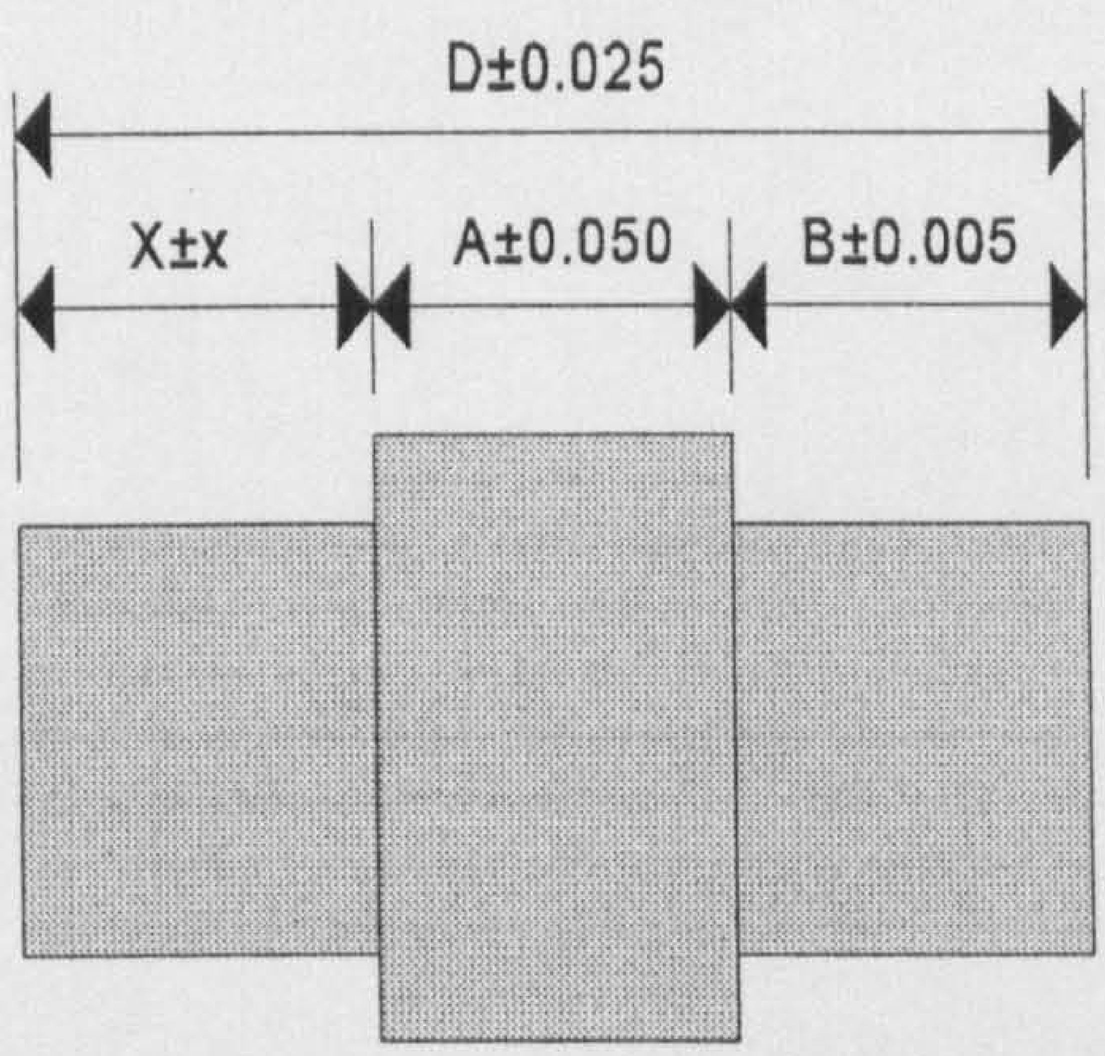

Figure 7.5: Reference tolerance.

Based on tolerance stackup theory, dimension D would be **D=A+B+X** and tolerance for D will then be **0.025=0.005+0.050+x**. Following the tolerance stackup theory, dimension X would have a negative slack figure for its tolerance:

$$\mathbf{x = 0.025-(0.005+0.050) = -0.030}$$

In theory, the tolerance is acceptable and can be assigned to a part. However, it is not a common practice to allocate a negative value as a working tolerance to a workpiece.

The aforementioned analysis uses the maximum value of individual feature tolerance assigned to a part to calculate the tolerance that should be assigned to x. This is common practice in allocating tolerance to a part [45-49, 52, 53, 58, 60-63]. In other words, these investigators are utilising multiple tolerances for analysis reference. Due to the variety of values assigned to a part, the multiple tolerance method of analysis should be validated.

Thus, using the basic premise that tolerance will accumulate (tolerance stackup theory); this investigation requires reassessment of the validity of present approach to tolerance analysis in comparison to single tolerance reference technique.

7.4.1 Experimental Method

It was proposed to validate the single tolerance reference analysis using the same sample workpieces, namely: (a) Plug 1 (P1); (b) Plug 2 (P2) and (c) Block 1 (B1). Supporting the analysis, types of material, tooling and machining parameter will be the same for both multiple tolerances reference and single tolerance reference validation. Ealier results using a multiple tolerance reference could then be compared with the results of single tolerance reference analysis.

7.4.2 Validation Result

After processing part P1 using a single tolerance reference, KATA has given the same solution as the multiple tolerance reference method. Results from the validation are shown in table 7-7.

CUT No.	MULTIPLE REFERENCE (±) TOL.	SINGLE REFERENCE (±) TOL
1	0.0500	0.0500
2	0.1621	0.1621
3	0.0500	0.0500
4	0.0500	0.0500
5	0.0500	0.0500
6	0.0500	0.0500
7	0.0500	0.0500

Table 7-7: P1's Single Tolerance Reference Validation

This table shows identical results from the two techniques because the tolerance assigned for the overall feature of Part P1 is the same as the tolerances assigned to individual smaller features.

CUT No.	MULTIPLE REFERENCE TOL. (±)	SINGLE REFERENCE TOL (±)
1	0.109	0.025
2	0.077	0.052
3	0.095	0.07
4	0.025	0.05
5	0.075	0.025
6	[0.005]	0.075
7	[0.023]	0.05
8	[0.018]	[0.0230]
9	[0.01]	[0.0050]

Table 7-8: P2 Process Results Comparison.

In the case of P2, the result from the single reference analysis differs from the multiple reference. It is expected in this case that the optimisation analysis for the test will fail based on the earlier P2 analysis shown in figure 7-3. Nevertheless, the objective of this test is to validate the single tolerance technique. Table 7-8 describes the differences in test results. Using the single reference technique, the number of negative slack values caused by inbalance allocation is eradicated. Significant shift in the results occur when the differences between one tolerance to the other is small as with single tolerance reference when compared to the larger differences obtained by the multiple tolerance reference technique

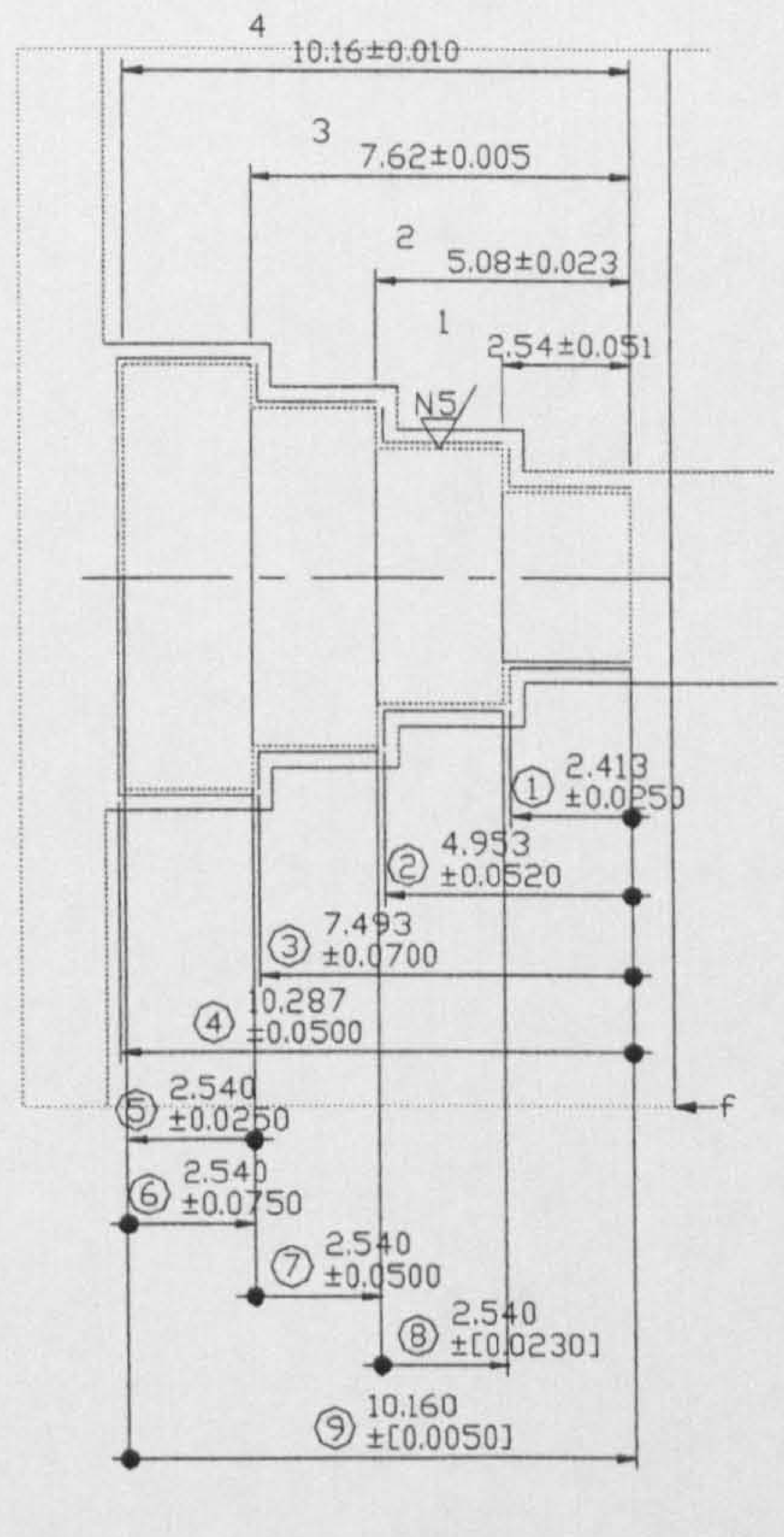

Figure 7-6: Single Reference Analysis Validation.

Figure 7-6 illustrates the result using the single tolerance reference technique. In the illustration, blueprint tolerance number *4* is 0.010, while blueprint tolerance number *1* is 0.051. Hypothetically, if blueprint number *4* is divided equally into four parts, the tolerance for each part would be 0.0025. Thus, the tolerance allocated is not violating any other tolerance requirement. However, if tolerance number *1* is divided equally into four parts, the tolerance for each part would be 0.0128. By doing so, tolerance number *3* has been violated. Thus, tolerance number *1* cannot be used as a constraint for optimisation.

Single reference analysis test LP model is as follows:

```
Min w1+w2+w3+w4+w5+z1+z2+z3+z4

st
t5+t4+w1=0.1000
t6+t3+w2=0.1000
t7+t2+w3=0.1000
t8+t7+t1+w4=0.1500
t9+t5+w5=0.1000
z1+t9+t8+t7=0.510
z2+t7=0.023
z3+t6=0.005
z4+t9=0.010
t1>=0.0250
t2>=0.0250
t3>=0.0250
t4>=0.0250
t5>=0.0250
t6>=0.0250
t7>=0.0250
t8>=0.0250
t9>=0.0250
end
```

B1 is used to validate two types of geometrical tolerances namely; symmetrical and positional. Using the single tolerance reference technique, the result of B1's validation has shown no differences in the result compared to the multiple tolerance reference. This is because, tolerances assigned for both geometric features are specific to the features themselves, thus, they do not substantially affect any other linear tolerances.

CHAPTER 8

DISCUSSION

8.1 Introduction

The dimension and tolerance of a product to be manufactured should be optimized according to some criteria, such as minimum production cost. Such a criterion is achieved using systematical analysis. Several methods of analysis have been developed by many investigators, namely; worst-case analysis, simple RSS, modified RSS and shifted normal. Differences in the result can be seen in an example using a journal bearing with a diameter of 40.0 ± 0.015 and a shaft of diameter 39.96 ± 0.01 as shown in table 8-1.

ANALYSIS METHOD	CALCULATED CLEARANCE
Worst-case	0.04 ± 0.025
Simple RSS	0.04 ± 0.018
Modified RSS (C_f = 1.4)	0.04 ± 0.025
Modified RSS (C_f = 1.5)	0.04 ± 0.027
Modified RSS (C_f = 1.8)	0.04 ± 0.032
Shifted normal (f = 0)	0.04 ± 0.025
Shifted normal (f = 0.5)	0.04 ± 0.022
Shifted normal (f = 1)	0.04 ± 0.018

Table 8-1: Calculated Clearance for Various Analysis Method.

A wide variation of results occur due to the effect of the various analysis methods. To have a full grasp of this occurrence, historical factors and characteristics of the research trend on the development of the tolerance analysis must first be understood.

8.2 Tolerance Optimisation Investigations

Worst-case analysis began during the fifties. Development of the analysis has centred on the most reliable technique to trace the process feature relationships. When this process feature relationship is identified, the tolerance can be distributed accordingly to each feature subjected to the blueprint specification.

Though blueprint specification, stock removal, machine capabilities, etc., have been used as constraint factors during allocation of tolerance, the margin of error still exists. This question of error is either low or high depending on the volume of production.

Much of the predicament in tolerance allocation is due to variables such as tool wear and tear, misalignment of a machine, a setup problem etc. The objective of worst-case analysis is to assign tolerances to the workpiece such that the probability it will not function properly is zero. This will become more difficult if the number of parts produced is large. The problem arises because worst-case analysis makes no assumptions on how parts are distributed within the tolerance zone. It is imperative, that no parts should fall outside the zone. Hence, investigators shifted to the statistical tolerance approach.

Statistical tolerancing is a method to assign tolerance to a part such that the probability it will not function should be non-zero. Simple RSS analysis is preferred since the distribution of parts will show some central tendency toward the midpoint. Unfortunately, the occurrence of non random factors can cause the RSS method to produce more than predicted out-of-tolerance. Some of these non random factors are component part distributions that are not well approximated by normal distributions or the mean is not at the tolerance midpoint.

Limitations to the application of normal distribution have been listed in [73,79,87,97,98]. These limitations and their causes are shown in table 8-2.

LIMITATION	CAUSE
Tolerance distribution are not always symmetrical	Tool wear, temperature changes and other factors cause dimension to drift.
Process tolerances do not necessarily cover the whole of the permitted drawing tolerance range	Knowledge about process capabilities is limited and a process may be working well within the design tolerances. It is commonly assumed that the required drawing tolerance matches the natural process tolerance.
A multi model distribution may exist	Pooling work from several machines or settings.
The distribution may be uncontrolled	Setting up each piece individually as a one-off.
Distributions may be truncated.	Rejecting out-of-tolerance parts or grading parts according to accuracy.
The nominal dimension and the process average do not always agree.	Some operators set tools on the high tolerance side to compensate for tool wear, some set to the low side so that any scrap can be reworked.
Normal distribution can only be summed if they are independent	Using the same machine tools, setups and cutting tools for several dimensions on several components causes correlation in many cases.
The central limit theorem is not always appropriate	Small production runs
The normal distribution model is inaccurate	The normal distribution has an infinite range, hence the closer the area gets to 100% the less accurate the model is likely be.
The law of addition of variances cannot always be applied.	Not all tolerances can be summed linearly

Table 8-2 Limitations of Using Normal Distribution for Process Tolerances

Due to these limitations of normal distribution, some investigators [86,97] have adopted a Beta distribution. It covers a range of distributions from normal to uniform to overcome the limitations of normal distribution. The Beta curve is the most flexible of the distributions used for tolerance analysis and can represent the actual distributions more accurately than the normal distribution and its variation. It is, however, more complex, mathematically intensive and requires more detail data to drive it.

Although the application of statistical methods to tolerance problems is well understood, there is little data available to enable the various approaches to be fully assessed. Nevertheless, studying research trends and the character of the distributions, it can be summarised that two main factors of the investigation in tolerance analysis are:

(a) to allocate the maximum possible tolerance to the workpiece as long as it is within the required specification; and

(b) able to make the best estimate of the actual production environment or criteria to facilitate the analysis.

8.3 Computer Assisted Tolerance Control

While studying the growth of tolerance control techniques, one other significant fact of its development is the increasing used of a computer to assist the analysis. Table 8-3 shows a summary of computer-assisted tolerance control.

From the table it can be derived that many of the methods used in Computer-Assisted Tolerance Control are worst-case instead of statistical analysis. Whenever the analysis becomes more computational or graphical intensive, a minicomputer will be used instead of a micro. There is no explanation on the matter. Nevertheless, it can be hypothesised that the power of a microcomputer in such cases is not able to support the demands of the system to be developed. An astonishing fact, however, is that not many systems developed thus far have geometrical tolerance or an angular cut, i.e. taper, chamfer, as part of the analysis package.

INVESTIGATORS	METHOD OF ANALYSIS	HOST HARDWARE	ALGORITHM CONSTRAINT	Dim Tol	Geo Tol	Ang Cut
Tolerance Chart (1950)	Worst-case	Chart	b/p, machine capabilities, stock removal	✓	Partial	✓
Bjørke (1977)	Statistical (Beta Distribution)	N/A	b/p, process capabilities.	✓	✗	✗
Ahluwalia/Karolin (1984)	Worst-case	Mini-computer	b/p, process capabilities, stock removal	✓	✗	✗
Parkinson (1986)	Worst-case, RSS.	Mini-computer (VAX)	b/p, process capabilities.	✓	✗	✗
Fainguelernt et al (1986)	Worst-case	Micro-computer (Apple IIe)	b/p, process capabilities, cutting tools, work setting	✓	✗	✓
Xiaoqing/Davies (1988)	Worst-case	Mini-computer (VAX 11/750)	b/p, process capabilities, materials, process.	✓	✗	✗
Irani et al (1989)	Worst-case	Micro-computer, Mini-computer (cost model).	b/p, process capabilities, stock removal; include machine precision for cost model.	✓	✗	✗
Li/Zhang (1989)	Worst-case	N/A	b/p, process capabilities	✓	✗	✗
Whybrew et al (1990)	Worst-case	Micro-computer	*Resultant flaw during validation raised some question on the type of constraint used*	✓	✗	✗
Lee/Woo (1990)	Statistical	Mini-computer (IBM 3090)	b/p, cost criterion.	✓	✗	✗
Zhang et al (1992)	Worst-case	N/A	b/p, process capabilities.	✓	✗	✗
He et al (1992)	Worst-case	N/A	b/p, stock removal (cutting tools)	✓	Partial	✗
Ngoi (1993)	Worst-case	Micro-computer (IBM-PC)	b/p, process capabilities	✓	✗	✗
Ngoi/Ong (1993)	Worst-case	Micro-computer (IBM-PC)	b/p, process capabilities, stock removal	✓	✗	✗
Ping Ji (1993)	Worst-case	Micro-computer (IBM-PC)	b/p, process capabilities.	✓	✗	✗
TI/TOL (1993)	Worst-case, RSS, modified RSS	Mini-computer (Pro-Engineer)	b/p, process capabilities.	✓	Partial	✓

Table 8-3: Computer-Assisted Tolerance Control

In the findings, CAD was not mentioned as being used as a stand alone system or working interactively with another system in any analysis. An important fact is that throughout all these years it is only in 1993 that a tolerance analysis package was marketed for commercial used. Despite the advent of computer technology, especially CAD, its capability is unfortunately still not fully utilised. Much of the development related to computer-assisted techniques dwell on ways and means of automating feature relationships and developing equations that can best emulate the actual production environment. Based on this analysis, further development on tolerance control should be on:

(a) a technique that can support the analysis using actual production capabilities; and

(b) the ability of CAD must be fully used, especially, since much of the design work and specifications are stored in it.

8.4 CAD Issues

There are two issues of concern on dimension and tolerance in CAD:

(a) current CAD status on the subject; and

(b) future development to enrich CAD's ability on the matter especially concerning tolerance.

Current CAD packages include 2D and 3D wireframe systems, surfaces and solids modelling. They all enable shape to be represented and many enable dimensions and tolerances to be defined. Sophisticated systems allow some associativity between the nominal geometry and the dimensions. However, none enable the dimension and tolerance to be used for applications such as process planning or tolerance analysis. There are two ways in which dimensions can be related to geometry.

Direct parametrization defines the geometry using the dimensions. This dimension driven geometry has the advantage that the geometry is well behaved, adopting the correct shape according to changes in dimensions. The second approach, *inverse parametrization*, adds dimensions to existing geometry. This associative dimensioning supports drawing annotations but does not allow dimensions to be amended directly and cannot guarantee that a particular dimensioning is complete.

On the matter of tolerances, the status is still very obscure. CAD can present tolerances to a part model according to the specification of BS 308. However, it does not have the ability to analyse the tolerance allocated to the model if required. Thus, effort has been made by several investigators (refer to section 3.5 and 3.6) to enhance this setback.

Two prominent approaches used by these investigators were:

(a) variational class concept, and

(b) virtual boundary approach.

In variational class concept, a tolerance zone is defined over a domain of a feasible region constructed by expanding or constructing the nominal geometry. The virtual boundary approach, on the other hand, is the theoretical limit boundary of a feature of size when the combined effects of all associated tolerances are considered. Significant contribution from this area of investigation is on the matter of how to comprehend the enigma of Geometric Tolerances. Ability to identify a feature region has made geometric tolerance analysis possible. This type of technique certainly departs from the linear algorithmic analysis.

8.4.1 Knowledge-based Systems

Based on the development made on dimension and tolerance representation in CAD, the work on tolerance analysis can further be extended. Knowledge-based systems merging with CAD will be a significant solution to tolerance optimisation. To acquire such mergers, CAD and knowledge-based system have to work interactively. The common term used for such a system is Intelligent CAD.

8.5 Intelligent CAD

Present CAD systems are only able to support a subset of the engineering design activities. Such activities are those corresponding to geometric modelling, engineering analysis, design reviews and evaluation and automated draughting. Optimizing dimension and tolerance for manufacture, nonetheless, requires intelligent analysis supported by heuristics and knowledge of the plant capabilities.

The limited capability of today's CAD has led to research in developing a new generation of Intelligent CAD systems. Success of Intelligent CAD investigation in design provided an ample solution for it to be used in the area of dimension and tolerance study.

The approach towards intelligent CAD involves extensive use of AI technology. The inclusion of expert systems in CAD refines and improves the design analysis quality. One main issue in this interactive technology is the difficulties in interfacing separate technologies.

This difficulty arises since most software vendors will not disclose their internal data structure to other parties. Thus, it can lead to inefficient interfaces within the core system leading to a slower solution time. There are two basic approaches to this problem.

One is to write a converter to go from each system to every other. As conversion is required in both directions between a pair of systems, two converters are needed for each pair. The other approach is to devise an independent format, called a neutral format and write a pair of converters for each system: A pre-processor to convert outgoing CAD data into the neutral format and a post-processor to convert incoming data from the neutral format.

The disadvantage of neutral formats compared with special converters is that one cannot assume that once the neutral format has been devised every CAD system in the future will have data that can be converted into it. To counter such difficulties, a standard of some kind is essential to any transfer of data from one program to another. The stream of binary digits constituting the data cannot be interpreted by the receiving program unless its format or a coding scheme is known. Several initiatives towards finding the solution are as follows:

(a) *CAM-I* - Computer-Aided Manufacturing International Inc. was formed in 1972. Over a hundred industrial companies, educational institutions and government agencies in North America, Europe and Japan pooled resources to form this nonprofit organisation. It provides a convenient conduit for transferring information between companies using computers in design, analysis and manufacturing.

(b) *IPAD* - The Integrated Program for Aerospace Vehicle Design (IPAD) is a NASA sponsored project to Boeing Co., underway since 1976 aimed at developing a software program for integrating existing CAD functions and for developing efficient ways to handle the huge amounts of data involved in such systems. Essentially, IPAD will provide a basic software framework in which all the separate functions of CAD will operate in an integrated manner.

(c) *ICAM* - Integrated Computer-Aided Manufacturing is a United States of America Air Force program attempting to develop one master program that will coordinate all the sophisticated design and manufacturing techniques employed by industry. In cooperation with IPAD, the ICAM program attempts to integrate such diverse engineering functions as design, analysis, fabrication, materials handling and inspection.

(d) *IGES* - A spin off of the IPAD and ICAM programs is the Initial Graphics Exchange Specification (IGES), which attempt to standardize the communication of geometric data between computer systems. It is an international standard of which version 5.0 is now current. In an extensive test carried out in 1988 by the Motor Industry Research Association and the Organisation for Data Exchange and Tele-Transmission in Europe with the assistance of the Cadcam Data Exchange Technical Centre, six pairs of engineering companies from the automotive, aerospace and rail industries carried out drawing exchanges between four pairs of different CAD/CAM systems. Of 70 faults discovered, only one or two were attributed to inadequacies in IGES. About a third were connected with misinterpretations of the specification and the remaining two-thirds or so were software bugs.

(e) *STEP and PDES* - Standard for the Exchange of Product model data (STEP) produce by ISO is to provide a basis for exchanging product information. It draws on the experience of previous data exchange standards and promises to become a really effective interchange standard for a variety of products. It aims to provide a neutral exchange mechanism capable of completely representing product definition data throughout the life cycle of a product. Product Data Exchange Specification (PDES), on the other hand, has been separately funded by the United States defence industry.

The work is being done in cooperation with that on STEP and will be an American contribution to STEP.

This set of standards will give the system developer a guideline to follow and use as utility in the software. Thus, the problem of data conversion can be solved.

8.6 KATA

KATA was developed using the technique in which an expert system is embedded in the CAD system. Embedding expert system in CAD and making it to work interactively will provide a competitive advantage over areas, such as:

(a) overcoming the dilemma of constant amendments of the design;

(b) corrections and improvements according to the plant capabilities;

(c) simultaneous engineering; and

(d) time save for process analysis.

Many issues had to be considered when building such a system. The following issues needed to be defined before commencing with any development.

(a) *Theoretical basis* - What is the syntax and semantics of tolerances to a product?

(b) *Applications* - What is the requirement for analysing part tolerances? What requirements are there for tolerances to support computer-based applications in design and manufacture?

(c) *Association with geometry* - How should the tolerances be represented so that they are consistent with the drawing and usable for engineering application? How should tolerance relate to part shape? Should one be defined or derived, are they independent or can they coexist? How can the shape be generated and recognised?

From the refinement of these issues, the plan and construction of the system was deployed in a logical manner. Integration of CAD and Knowledge-based systems in some respect gives a solution for the above inquiries.

CAD has long been known to have the ability of constructing geometry and able to store numerical data. Therefore, as long as adhering to the drawing principle, the workpiece will be represented in clarity and completeness. However if CAD could be developed to work interactively with a knowledge-based system, the new system could also capture the intent of making the part, the how and why the part is to be produced.

KATA was built in a *modular structure*. This approach is quite significant. It can allow any amendment or addition being made to the module concerned. Instead of troubleshooting the whole system for mistake or addition, the approach has already narrowed the working area. It will also be very beneficial for future development of KATA. For instance at present, 2D representation is used to analyse the workpiece to be manufactured, it is intended that later the ability will also include a 3D solid modelling representation to enhance its analysis capabilities. The main issue of concern for the development was KATA's host.

8.7 Host System

It was imperative to select a proper host for the system to assure software compatibility and ease of use. The selection of the host was based on two factors:

(a) the objectives of the research, and

(b) the area that the system would serve to meet its purposes.

These factors can be summarised as:

(a) ability to analyse the workpiece tolerances intelligently by considering several factors such as machine capabilities, cutting tools, etc.;

(b) ability to allocate tolerances to workpiece as optimum as it can be specified according to the required blueprint specification;

(c) ability to perform dimensional (conventional) and geometrical tolerance; and

(d) ability to be used by small and medium scale production plant.

8.7.1 Hardware

The host hardware of many computer-based tolerance analyses reviewed previously is either on a mainframe-based or minicomputer system. Based on table 8-3, only Fainguelernt et al., Irani et al., Ngoi and Ji have attempted to use a microcomputer to host the algorithm developed. On the other hand, to the best of this investigator knowledge, all the work on tolerance representation in CAD is hosted by either a mainframe-based or minicomputer system.

Issues with respect to host hardware selection were dependent on the system available at the investigation site and whether it was accommodative to the specification of software or CAD system selected. While developing a prototype or experimental-based system, the constraint of host hardware or software cost was the least priority. Many systems discussed earlier can be said to be experimental-based systems, thus, it is of no surprise that manufacturer cannot procure commercially the techniques or systems developed by these investigators. Pro/ENGINEER broke this stalemate when TI/TOL was marketed, nonetheless, TI/TOL is hosted by a minicomputer that is too expensive for the small and medium scale manufacturer to acquire. Apart from the initial cost of purchase, the cost entailed from this type of system includes licensing, training, etc., and can only be afforded by the giant manufacturer such as Texas Instrument and alike. In retrospect, if the system to be developed is specifically for industrial use and targeted for small and medium scale group of manufacturer. The selection of host hardware and CAD package had therefore to include the ability of these manufacturers to obtain such a system. This ability included the matter of cost (i.e. initial purchasing, licensing, operating) and training.

Weighing these factors and supported with the advent of microcomputer technology, to select a microcomputer as the host system for KATA was an obvious decision. The idea of building KATA was not only experimentation, but also to provide a final system that would enabl small and medium scale manufacturers the possibility of its application in competitive market.

8.7.2 Software

Selecting CAD software to build KATA was rather intricate. There were many CAD software systems available in the market. Each had its own advantage and disadvantage, nevertheless, AutoCAD was selected. This selection was due to its wide circulation and most important of all was its the ability to interact with a third party software, a programming language such as C or PROLOG. Another pivotal advantage of AutoCAD over others is that a higher level programming language capable of making an expert system shell is naturally embedded in it. Hence, having this natural advantage the problem of data conversion is eliminated.

8.8 KATA's Performance

KATA's performance can be evaluated and discussed in four categories:

(a) detailing (model representation),

(b) manufacturing (process plan),

(c) tolerance optimisation, and

(d) features recognition.

8.8.1 Detailing

The geometry to make KATA's workpiece model was generated through AutoCAD's own menu and draw utility. AutoCAD is so easy to use that within a couple of hours a novice user can create complex geometry.

Dimensioning and drawing symbols, on the other hand, were specifically built for KATA. This could be assessed easily through a pull down a detailing menu. The dimension, tolerance and drawing symbols in KATA conformed to BS 308. However, KATA's prototype was built to analyse metric specification. Thus, all specifications given in imperial units had to be converted into metric first.

The detailing menu built for KATA was not as comprehensive as the one listed in BS 308. Nonetheless, it gives a complete description on the procedure with KATA's manufacturing and tolerance optimisation modules.

An important aspect of KATA's detailing shell was that the value assigned to the workpiece is stored in CAD's database. This value can be retrieved by other shells, i.e. manufacturing and tolerance optimisation for further operation. It also can be edited using AutoCAD's edit attribute utility. Thus, the user does not have to redo the detailing or forthcoming operation if an error is made.

8.8.2 Manufacturing

In order to optimize the manufacturing tolerance as if the workpiece was actually doing a trial run needed KATA to be developed having the knowledge and facts of real time production capabilities. The task of evaluating what capability should be included and thus become a constraint for production was rather subjective.

However, based on this this author's knowledge and on the literature findings, the following were included:

(a) machine capability to hold the tolerance,

(b) recommended process tolerance,

(c) tool wear factors,

(d) surface roughness,

(e) blueprint requirement, and

(f) types of material.

The value for all these variables can be added, deleted and edited through a pull down expert system using production rules within AutoCAD. The user does not have to open up the expert system shell. This is very much different from previously built systems in which the user had to open the programme and edit it line by line. Therefore, a novice user can easily approach and adapt to the system as if working with AutoCAD itself.

The next step is rudimentary. In this stage the user makes the process plan for the workpiece with all the workpiece data and the current production capabilities data flashing on the screen to be agreed by the user. In case the user does not agree with the suggestion, KATA allows the user to select a newer process by script conversing. This is a significant difference compared to the system such as CATC [46,47]. KATA tries to find the best solution and allows the job to continue without terminating it first, as happens with CATC.

If any error occurs, such as the machining parameters required by the workpiece cannot be processed by KATA, an error message will be given stating the problem.

8.8.3 Tolerance Optimisation

The next stage is to optimise the tolerance. KATA tolerance optimisation procedure follows the well established tolerance stackup rule. The equation used to find an optimum solution is the same as used by previous investigators (refer to section 2.5.1 and 2.5.2) and can be traced back to Wade's original tolerance chart. An extra equation is needed to include the constraint listed previously in the original equation. These equations agree with the one derived by Irani et al. and used in their system.

KATA was developed to also include in its search algorithm angular cuts, i.e. chamfer, taper etc. and geometric tolerance. However, KATA geometric tolerance capability is limited to position and symmetry. This limited ability is due to KATA's two dimensional representation. The feature recognition ability is also partly responsible.

8.8.4 Feature Recognition

KATA's feature recognition ability is at its infancy stage. At this point it can only identify low-level feature such as thread, angular etc. Therefore, it cannot fully support KATA's processing ability. If KATA is to be fully automated, this module need to be upgraded.

8.9 Validation

Three tests have been undertaken to validate KATA using test workpieces P1, P2 and B1 as explained in detail in Chapter 7. P1 and B1 were contrived workpieces meant to test KATA's ability to optimise dimensional tolerance, geometrical tolerance and angular cuts such as chamfer and taper. P2, on the other hand, was used by several investigators [44,45,53,62]. KATA's result was compared with these investigators' results to evaluate the differences.

The results from the validation show improvement in tolerance analysis. Based on the available production capability information already installed in KATA, test workpiece P1 was successfully optimised (refer to figure 7-2 and table 7-3). The working dimension and tolerance suggested by KATA do not violate any constraints, thus, KATA can give an optimum solution. The recommended working tolerance for dimensional cut and chamfer cut shows that it does not exceed the required specifications.

B1 produced the same result as P1. Position and Symmetry tolerances were successfully optimised (refer to figure 7-4). Nevertheless, the requirement to convert the geometrical tolerance into an equal bilateral form imposed an addition to the analysis procedure. A dummy cut was required to obtain the final geometric tolerance result. This was to compensate the result value under which the dummy cut results is always equal to half of the previous cut value. The constraint is set in the LP model using the equation **2t2-t1=0**. t2 is the dummy cut and t1 is the previous cut. This unquestionably handicapped the ability to process any other geometric tolerance characteristics.

P2 was not able to be processed by KATA. This is because of P2's tight specification that could not be processed using the available process capability already installed in the database. Two alternatives for this situation were,

(a) change the specification, or

(b) upgrade the process capabilities.

To upgrade the process capabilities, KATA offered an assistance during processing. Any machining parameter failure was given an error message and a solution for the error. Thus, the user is always informed of the workpiece and production capability error. Due to the failure, a comparative study on the result cannot be made.

Nevertheless, the consequence of P2's analysis failure have produced two significant results. These results enriched KATA's competence with the ability of seeking an alternative process plans and eradicating the tolerance stackup error.

8.9.1 Alternative Process Plans

Due to the failure of P2's analysis, an alternative process plans can be tested on the same workpiece. The ability to continue processing and seek for an alternative process plan without terminating the deliberation describes KATA's flexibility and determination to search for a new solution.

This ability mimic the human approach of seeking other alternative using the available process capabilities. This shows that KATA does not rely totally on a mathematical solution.

8.9.2 Single Tolerance Reference Method

Secondly, the selection of a reference tolerance for the analysis is important and affect the result of tolerance optimisation. Present tolerance optimisation analyses use the multiple tolerance reference technique. From the validation, the multiple tolerance reference technique has shown to produce a lot of negative slack values when the overall feature tolerance is tighter than the individual smaller feature tolerances. As an alternative approach, the author had developed and introduced a single tolerance reference method within KATA to eliminate this defect and improve on the analysis of tolerance optimisation. Based on the validation, the new technique has shown a significant improvement when the overall feature tolerance is tighter. However, the method does not have any affect when the overall feature is greater than or equal to the individual smaller feature tolerance.

KATA was developed not only to use the multiple tolerance reference technique but also to give the flexibility to select the reference specification, i.e. dimension and tolerance, to analyse and optimise the working tolerance. Single tolerance reference technique could be used as the basis of analysis when the tolerance of the overall larger feature has a tighter tolerance than the individual tolerances of the smaller features. Every cut during the manufacturing processes in KATA will refer to this single tolerance instead of the related feature tolerances. The tolerance suggested by KATA using this single reference method eliminates the problem of negative slack value. Negative slack is now no longer due to the problem of tolerance stackup but solely a result of machining parameters.

8.10 Future Work

KATA has proven to be a breakthrough in finding the optimum solution for manufacturing tolerance. The solution given will not be only mathematically sound and free from human error but also strengthen the results based on present process capability.

Since it is generic in nature, any plant can use KATA and install it in its own production capabilities. This will give an advantage by which a component from a first party manufacturer can easily be processed by the third party vendor. Part number DBM 0715, figure 8-1, from a former partner of Rover is a classic example of the enigma faced by the second party assembler and the third party vendor. The part which lack in information and an unclear written specification can be processed by KATA without and further information or explanation required. Nevertheless, there are two other factors that can really enhance KATA's ability: (a) feature recognition, and (b) process control linkup. Feature recognition can fully automates and upgrade KATA's ability to analyse geometric tolerance. To do this, KATA's representation has to be three dimensional. With the advent of microcomputer technology and research on feature recognition this vision can be materialised. Besides geometric tolerance, this would allow the analysis to be conducted from a single position rather than working with present X and Y axis.

The paramount beauty of KATA is the process capabilities decision support. Linking KATA with a process control system such as Statistical Process Control or even to a coordinate measuring machine will enable KATA to update its information automatically. This will enhance the analysis accuracy and thus complete the manufacturing simultaneous engineering cycle

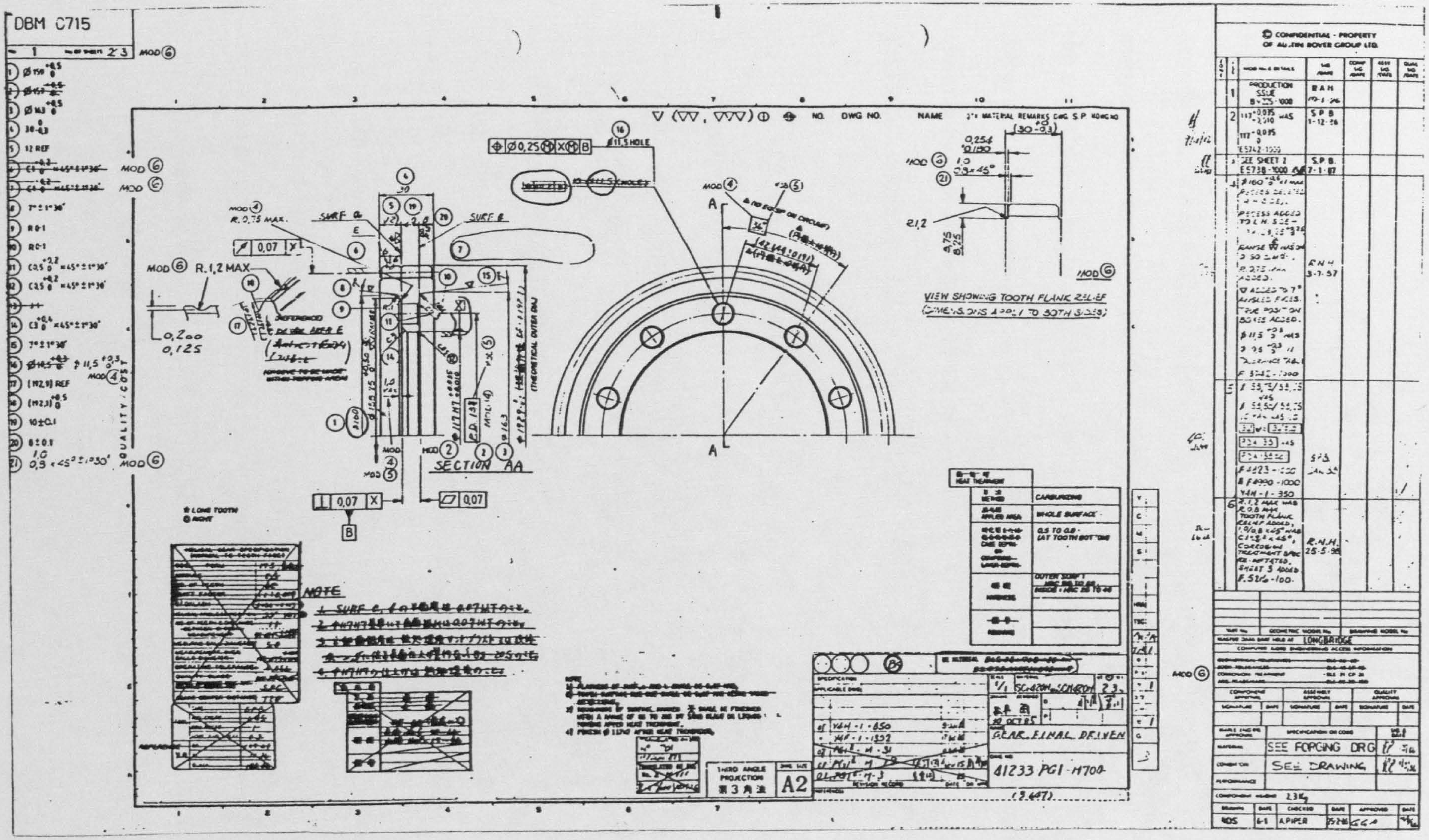

Figure 8-1: Final Drive Gear (Part number DBM 0715).

CHAPTER 9

CONCLUSION

Dimensions and tolerances are pervasive in all stages of engineering. This thesis has described the development of a Knowledge-based Automatic Tolerance Analysis that can assist the manufacturer to manufacture a part with good quality and at a low cost. During the investigation a new technique known as Single Tolerance Reference Technique is discovered. This technique has shown a major departure in the analysis of tolerance optimisation. In conclusion the investigation can be summarised into three facets; (a) KATA's Development, (b) KATA's Ability and (c) Single Tolerance Reference Method.

A. KATA'S DEVELOPMENT

1. KATA's was developed in this investigation with the objective of optimising the allocation of manufacturing tolerance automatically and systematically using the production criteria as constraints of analysis. Machine capability, stock removal, workpiece material, cutting tool and machining cost have been used as constraints in the development.

2. The initial premise in the construction of this knowledge-based system was to incorporate tolerance stackup theory. This has been used as the basis of calculating the accumulation of tolerance during machining to acquire the resultant tolerance of the working feature. This resultant tolerance which will be assigned to the working feature is subjected to the constraint imposed during production as mentioned in A1.

3. Departing from the common practices of manual technique or tolerance analysis automation through computing, KATA was developed using AutoCAD as the host CAD system. This diversion is to enable the designer and manufacturing engineer to share a common platform of design and analysis. Knowledge sharing from the two disciplines can now be done with the stroke of a keyboard.

4. The knowledge of tolerance optimisation analysis incorporated in the intelligent knowledge-based system which developed by the author and embedded in AutoCAD. The system designated as KATA was developed using AutoCAD's own higher-level language AutoLISP. AutoCAD's ability to interact with other languages such as PROLOG and C and third party software LINDO enabled the author to link all the programmes without any interface problem.

5. To allow any amendment or addition to KATA's programme, KATA was built in a modular structure. Hence, instead of troubleshooting the whole system for a mistake or to make a program addition, the module has already narrowed down the working area. The approach is also very beneficial for future development of KATA.

6. KATA's prototype is able to optimise tolerance allocation using both the single tolerance reference technique and the multiple tolerance reference method.

C. KATA'S ABILITY

1. KATA's ability to optimize tolerance with the support of actual production capability has improved the result accuracy. Analysis is now based on fact and not imaginary or random number generation as with Monte Carlo Simulation.

2. Trial production runs can now be made at the front end of the design stage. This ability will lower the scrap percentage. Several possible processes can be evaluated in a matter of minutes and the best result could be selected. Thus cost of production and the time taken to produce the part will be at a competitive advantage over other manufacturers.

3. Using CAD as the media of communication between human and machine will enable the process engineer to visualise and simulate the actual machining process. The use of the CAD system makes KATA a useful tool for not only developing product detail but enables analysis of the manufacturing processes.

4. The nature of KATA's structure is the assimilation of production capability information which makes the system itself work as a central database whereby all information about the product and the plant resides. Thus, the *knowledge guru* can now be accessible to everybody.

5. The knowledge-based system within KATA mimics the human ability to analyse and optimise the tolerance heuristically. This enables an inexperience process engineer to conduct the analysis without the assistance of the experienced engineer.

6. Failure to process the part will be analysed and reported with a possible solution. Thus, corrective measures can be taken. This saves the time and running cost of trouble shooting and testing. KATA also gives an implicit recommendation when failure to process occurs. Machine capability and types of tools can either be upgraded or serviced when a machining parameter error is given.

7. KATA's ability can be greatly enhanced if the model is represented in three-dimensional solid modelling. All of the geometric tolerance characteristics can be analysed and optimised. Better visualisations of the finished product can be made and the machining operation will be as good as making a trial cut using CNC graphic capability.

8. KATA's ability can also be greatly enhanced if it can be a linked to a process control system or a coordinate measuring machine. This will complete the manufacturing simultaneous engineering cycle. Having such a link up, the process capability will be updated automatically at a presepecified interval. Thus, KATA and the engineer will always be aware of the production status. Any minor amendment can be made without interrupting the production run.

C. SINGLE TOLERANCE REFERENCE METHOD

1. The investigation has led to the significant development of the single tolerance reference technique which departs from the common practice of multiple referencing. Utilisation of the single reference technique has shown that the tolerance stackup error is eradicated.

2. Validation of the technique has shown that if the tolerance of the overall feature is greater than or equal to the individual smaller features of the part, both the multiple tolerance reference technique and the single tolerance reference technique can be utilised.

3. If the tolerance of the overall feature is less than the tolerance of individual smaller features, the multiple tolerance reference will give a negative slack result. However, the single tolerance reference will balance the tolerance and allocate it to the smaller features. This allocation of tolerance is determine according to the value of the overall feature tolerance. Thus, the single reference technique gives a better result compared to the multiple tolerance reference techique.

4. There is no difference in result between multiple and single reference methods when geometrical tolerances are analysed. This is because the tolerances assigned for the geometric features are dedicated to that feature.

REFERENCE

1. Singer C. et al, eds., *A History of Technology*, vol. II, London: Oxford Univ. Press, 1956.

2. Holt JC., *Magna Carta*, London: Cambridge Univ. Press, 1927.

3. Singer C. et al, eds., *A History of Technology*, vol IV, London: Oxford Univ. Press, 1958.

4. Rohan TM., Customers Say Quality Program Helps Makes Sales, *Industry Week*, Sept 1984.

5. Nieble BW., *Product Design and Process Engineering*, New York: McGraw-Hill, 1974.

6. Brooker., ed., *Manual of British Standards in Engineering Metrology*, BSI in association with Hutchinson, 1984.

7. Sanders TRB., ed., *The Aims and Principles of Standardisation*, Switzerland: The International Organisation for Standardisation.

8. Woodward CD., *Standards for Industry*, London: Heinemann, 1965.

9. BS 308 Part I, *Engineering Drawing Practice: Recommendations for General Principles*, BSI, 1984.

10. BS 308 Part II, *Engineering Drawing Practice: Recommendations for Dimensioning and Tolerancing for Size*, BSI, 1985.

11. BS 308 Part III, *Engineering Drawing Practice: Recommendations for Geometrical Tolerancing*, BSI, 1990.

12. Walleigh R., Product Design for Low-cost Manufacturing, *The Journal of Business Strategy*, 10(4), 1989, pp 37-41.

13. Parker M., ed., *Manual of British Standards in Engineering Drawing and Design*, BSI in association with Hutchinson, 1984.

14. BS 4500 Part 1, *ISO Limits and Fits*, §1.1 and §1.2, BSI, 1986.

15. Oberg et al, eds., *Machinery's Handbook*, 22nd Edition, New York: Industrial Press, 1984.

16. Bralla JG., ed., *Handbook of Product Design for Manufacturing*, New York: McGraw-Hill, 1986.

17. Trent EM, *Metal Cutting*, London: Butterworths, 1984.

18. Pashby IR, *The Effect of Heat Treatment on the Machinability of Austempered Ductile Iron*, PhD Dissertation, Warwick University, 1992.

19. Metcut Research Associates, *Machining Data Handbook*, Part I & II, 3rd Edition, Ohio: Metcut Research Inc., 1980.

20. Delmar D. and Sheldon G., *Introduction to Quality Control*, St. Paul: West Publishing Co., 1988.

21. Siddal JN., *Analytical Decision Making in Engineering Design*, Englewood-Cliffs: Prentice Hall, 1972.

22. Martin JM., The Final Piece of the Puzzle, *Manufacturing Engineering*, 101(3), September 1988, pp 46-51.

23. Whitney DE., Manufacturing by Design, *Harvard Business Review*, 66(4), Jul/Aug 1988, pp 43-91.

24. Evans B., Simultaneous Engineering, *Mechanical Engineering*, 110(2), February 1988, pp 38-39.

25. Gregory A., The Tool to Automate Common Sense, *Engineering Computers*, September 1991, pp 49,51 & 53.

26. Kuo W. and Hsu JP., Simultaneous Engineering Design in Japan, *Industrial Engineering*, 22(10), October 1990, pp 23-26.

27. Barr A. and Feigenbaum E., *Handbook of Artificial Intelligence*, vol I & II, California: William Kaufmann, 1981.

28. Cohen PR. and Feigenbaum E., *Handbook of Artificial Intelligence*, vol III, California: William Kaufmann, 1982.

29. Wolfgram DD., Dear TJ. and Galbraith CS., *Expert Systems for Technical Profession*, New York: John Wiley & Sons, 1987.

30. Buchanan BG., Expert Systems: Working Systems and the Research Literature, *Expert System*, 3(1), January 1986, pp 32-51.

31. Seely MJ., *Knowledge-Based Engineering: A Weapon in the MCAD Arsenal*, Dataquest Research Newsletter, No 00078-9, California Dataquest Inc., 1990.

32. Rosenfeld LW., Using Knowledge-based Engineering, *Production*, 101(1), November 1989, pp 74-76.

33. McGoldrick PF., Tolerance Representation in Computer Aided Design, *CIRP International Working Seminar on Computer Aided Tolerancing*, The Penn State University, May 16-17, 1991.

34. *The Penguin Pocket English Dictionary*, 4th Edition, London: The Penguin Group, 1990.

35. Spotts MF., *Dimensioning and Tolerancing for Quality Production*, Englewood Cliffs, New Jersey: prentice Hall, 1983.

36. Darwin CG., Statistical Control of Production, *Nature*, 149, 1942, pp 573-575.

37. Zhang HC and Huq ME., Tolerancing Techniques: The State of the Art, *International Journal of Production Research*, 30(9), 1992, pp 2111-2135.

38. Juster NP., Modelling and Representation of Dimensions and Tolerances: A Survey, *Computer Aided Design*, 24(1), 1992, pp 3-17.

39. Roy U., Liu CR and Woo TC., Review of Dimensioning and Tolerancing: Representation and Processing, *Computer Aided Design*, 23(7), 1991, pp 466-483.

40. Mooney CT., How to Adjust Tolerance Charts, *Tool Engineer*, 35(10), 1955, pp 75-81.

41. Johnson A., Index Tolerance Chart Simplifies Production, *Tool Engineer*, 32(2), 1954, pp 53-62.

42. Marks CJ., Tolerance Charts Control Production Machining, *American Machinist*, 97(3), 1953, pp 114-116.

43. Anderson Jr. JF., Assembly Tolerance Charts Save Times and Money, *Tool Engineer*, 37(6), 1956, pp 95-97.

44. Wade O., Tolerance Control, in Drozda TJ and Wick, eds., *Tools and Manufacturing Engineering Handbook*, vol 1, Detroit:SME, 1983, pp 2.1-2.9.

45. Wade O., *Tolerance Control in Design and Manufacturing*, New York: Indusrial Press, 1967.

46. Ahluwalia RS and Karolin AV., CATC - A Computer Aided Tolerance Control System, *Journal of Manufacturing System*, 3(2), 1984, pp 153-160.

47. Karolin AV., Computer Aided Tolerance Analysis, *Proceedings of Autofact 6*, California: SME, 1984, pp 15.71-15.82.

48. Fainguelemt D, Weill R, Bourdet P, Computer Aided Tolerancing and Dimensioning in Process Planning, *Annals of the CIRP*, 35(1), 1986, pp 381-386.

49. Xiaoqing T and Davies BJ., Computer Aided Dimensional Planning, *International Journal of Production Research*, 26(2), 1988, pp 283-297.

50. Euler L, Solutio Problematis as Geometriam Situs Pertinantis, *Academimae Petropolitanse*, 8, 1736, pp 128-140.

51. Ore O, *Graph and their Uses*, New York: Random House, Inc., 1963, p 34.

52. Whybrew K, Britton GA, Robinson DF and Sermsuti-anuwat Y., A Graph Theoretic Approach to Tolerance Charting, *International Journal Advanced Manufacturing Technology*, 5(2), 1990, pp 175-183.

53. Ngoi BKA, Applying Linear Programming to Tolerance Chart Balancing, *International Journal Advanced Manufacturing Technology*, 7, 1992, pp 187-192.

54. Ngoi BKA and Ong CT, A Complete Tolerance Charting System, *International Journal of Production Research*, 31(2), 1993, pp 453-469.

55. Ngoi BKA and Ong CT, Process Sequence Determination for Tolerance Charting, *International Journal of Production Research*, 31(10), 1993, pp 2387-2401.

56. Ngoi BKA and Ong CT, Computer Aided Process Dimensioning, *International Journal Advanced Manufacturing Technology*, 8, 1993, pp 371-376.

57. Ngoi BKA and Ong CT, A Matrix Approach to Tolerance Charting, *International Journal Advanced Manufacturing Technology*, 8, 1993, pp 175-181.

58. He JR and Lin GCI, Computerised Trace Method for Establishing Equations for Dimensions and Tolerances in Design and Manufacture, *International Journal Advanced Manufacturing Technology*, 7, 1992, pp 210-217.

59. He JR and Gibson PR, Computer-Aided Geometrical Dimensioning and Tolerancing for Process-Operation Planning and Quality Control, *International Journal Advanced Manufacturing Technology*, 7, 1992, pp 11-20.

60. Li JK and Zhang C, Operation Dimensions and Tolerances Calculation in CAPP Systems for Precision Manufacturing, *Annals of the CIRP*, 38(1), 1989, pp 403-406.

61. Zhang HC, Mei J and Dudek RA, Operational Dimensioning and Tolerancing in CAPP, *Annals of the CIRP*, 40(1), 1991, pp 419-422.

62. Irani SA, Mittal RO and Lehtihet EA, Tolerance Chart Optimisation, *International Journal of Production Research*, 27(9), 1989, pp 1531-1552.

63. Ji P, A Tree Approach for Tolerance Charting, *International Journal of Production Research*, 31(5), 1993, pp 1023-1033.

64. Ji P, A Linear Programming Model for Tolerance Assignment in a Tolerance Chart, *International Journal of Production Research*, 31(3), 1993, pp 739-751.

65. Spotts MF, An Application of Statistics to Dimensioning of Machine Parts, *Journal of Engineering for Industry, Trans. of ASME*, 81(4), 1959, pp 317-322.

66. Evans DH., Statistical Tolerancing: The State of the Art Part 1- Background, *Journal of Quality Technology*, 6(4), 1974, pp 188-195.

67. Evans DH, Statistical Tolerancing: The State of the Art, Part II: Methods for Estimating Moments, *Journal of Quality Technology*, 7(1), 1975, pp 1-12.

68. Evans DH, Statistical Tolerancing: The State of the Art, Part III: Shifts and Drifts, *Journal of Quality Technology*, 7(2), 1975, pp 72-77.

69. Evans DH, An Application of Numerical Integration Techniques to Statistical Tolerancing, *Technometrics*, 9(3), 1967, pp 441-455.

70. Evans DH, An Application of Numerical Integration Techniques to Statistical Tolerancing, II-A Note on the Error, *Technometrics*, 13(2), 1971, pp 315-328.

71. Evans DH, An Application of Numerical Integration Techniques to Statistical Tolerancing, III- General Distributions, *Technometrics*, 14(1), 1972, pp 23-37.

72. Sayed EY and Kheir NA, An Efficient Technique for Minimum Cost Tolerance Assignment, *Simulation*, 44(4), 1985, pp 189-195.

73. Mansoor EM, The Application of Probability to Tolerances used in Engineering Designs, *Proceeding of Institute of Mechanical Engineers*, 178(1), 1963, pp 29-51.

74. Clarke GM and Cooke D, *A Basic Course in Statistics, 2nd Edition*, Maryland: Edward Arnold, 1983, p 184.

75. Burr IW and James RE, Specifying the Desired Distribution in Lieu of Tolerance Limits - A System Approach, *Paper 74-DE-9*, ASME, 1974.

76. Hines WW and Montgomery DC, *Probability and Statistics in Engineering and Management Science*, 2nd Edition, New York: John Wiley and Sons, 1980, pp 181-183.

77. Greenwood WH and Chase KW, A New Tolerance Analysis Method for Designers and Manufacturers, *Journal of Engineering for Industry, Trans. of ASME*, 109(2), 1987, pp 112-116.

78. *TI/TOL Tolerance Analysis Tool*, Pro Engineer Product Leaflet, 1991.

79. Bender A, Statistical Tolerancing as it Relates to Quality Control and the Designer, *Society of Automotive Engineers*, SAE Paper No. 680490, 1968.

80. Gladman CA, Applying Probability in Tolerancing Technology, *Transaction of Institute of Australian Mechanical Engineers*, ME5(2), 1980, p 82.

81. Parkinson DB, Tolerancing of Component Dimensions in CAD, *Computer Aided Design*, 16(1), 1984, pp 25-32.

82. Parkinson DB, Application of Reliability Methods to Tolerancing, *Journal of Mechanical Design, Trans. of ASME*, 104, 1982, pp 612-618.

83. Parkinson DB, Assesment and Optimisation of Dimensional Tolerances, *Computer Aided Design*, 17(4), 1985, pp 191-199.

84. Chase KW and Greenwood WH, Design Issues in Mechanical Tolerance Analysis, *Manufacturing Review*, 1, 1988, pp 50-59.

85. Greenwood WH and Chase KW, Worst-case Tolerance Analysis with Nonlinear Problems, 110(3), 1988, pp 232-235.

86. Bjørke Ø, *Computer Aided Tolerancing*,Trondheim:Tapir Publishers, 1978.

87. Michael W and Siddall JN, The Optimal Tolerance Assignment with Less than Full Acceptance, *Journal of Mechanical Design, Trans. of ASME*, 104, 1982, pp855-860.

88. He JR, Tolerancing for Manufacturing via Cost Minimization, *Machine Tools Manufacturing*, 31(2), 1991, pp 455-470.

89. Tang GR, Kung R and Chen JY, Optimal Allocation of Process Tolerances and Stock Removals, *International Journal of Production Research*, 32(1), 1994, pp 23-35.

90. Spotts MF, Allocation of Tolerance to Minimze Cost of Assembly, *Journal of Engineering for Industry, Trans. of ASME*, 95(3), 1973, pp 762-764.

91. Speckhart FH, Calculation of Tolerance Based on a Minimum Cost Approach,*Journal of Engineering for Industry, Trans. of ASME,* 94(2), 1972, pp 447-453.

92. Gladman CA, Techniques for Applying Probability to the Tolerancing of Machined Dimensions, NSI, Technical Paper No. 11, CSIRO, Melbourne, 1959.

93. Cavé R, Le Control Statistique des Fabrications, *Eyrolles*, Paris, 2e, ed., 1964.

94. Peters J, Tolerancing the Components of an Assembly for Minimum Cost, *Journal of Engineering for Industry, Trans. of ASME*, 92(3), 1970, pp 677-682.

95. Sfantsikopoulos MM, A Cost-Tolerance Analytical Approach for Design and Manufacturing, *International Journal Advanced Manufacturing Technology*, 5(2), 1990, pp 126-134.

96. Lee WJ and Woo TC, Tolerances Their Analysis and Synthesis, *Journal of Engineering for Industry, Trans. of ASME*, 112(2), 1990, pp 113-121.

97. Brandt DA, How not to use Statistical Tolerancing, *Machine Design*, Feb. 1973, pp 111-113.

98. Brooks KA, How to Setup and Co-ordinate a Statistical Dimensioning Program, *Machine Design*, 33(19), Sept. 1961, pp 140-145.

99. Campbell I, *CAD/CAM*, Module Handout, Univ. of Warwick: Manfacturing Engineering Systems Group, 1992.

100. BS1192: Part 5, Guide for Structuring of Computer Graphic Information, London: British Standards Institution (Related to ISO/TRIO127), 1990.

101. Woodwark J, *Computing Shape*, London: Butterworths, 1986.

102. DIAD 2D, Datasheet Ref. No. EM/AP/002-Oct89, London, ICL-AN STC Company, International Computer Limited.

103. AutoCAD, Release 11, Specification Sheet, Guildford, Autodesk Limited, Nov. 1990.

104. DRAGON 2D DRAUGHTING AND DESIGN, Specfication sheet, Bedfords, MASTA Corporation Limited.

105. Pro/ENGINEER, Marketing brochure DCN PE2-1090-1, Waltham, MA, Parametric Technology Corporation.

106. Mechanical Designer, Publicity poster, Letchworth, Learning Curve.

107. Mechslide, Datasheet, Stockholm, EMT AB.

108. Requicha AAG, Representation of Tolerances in Solid Modelling: Issues and Alternative Approaches, In Pickett MS and Boyse JW, eds., *Solid Modelling by Computers: From theory to Applications*, New York: Plenum Press, 1984, pp 3-22.

109. Requicha AAG, *Toward a Theory of Geometric Tolerancing*, Technical Memorandum TM-40, Production Automation Project, New York: University of Rochester, March 1983.

110. Requicha AAG and Chan SC, Representation of Geometric Features, Tolerances and Attributes in Solid Modelers based on Constructive Geometry., *IEEE Journal of Robotics and Automation*, RA-2(3), 1986, pp 156-166.

111. Requicha AAG, *Mathematical Models of Rigid Solid Objects*, Technical Memorandum, TM-28, Production Automation Project, New York: University of Rochester, 1977.

112. Requicha AAG, *Part and Assembly Description Languages: I. Dimensions and Tolerancing*, Technical Memorandum, TM-19, Production Automation Project, New York: University of Rochester, 1977.

113. Minagawa M, Okino N and Kakazu Y, Automatic Dimensioning on 3D Solid Geometry, *Proceedings of 5th ICPE*, Tokyo: Japan, 1984.

114. Yuen MM, Tan ST and Yu KM, Scheme for Automatic Dimensioning of CSG Defined Parts, *Computer Aided Design*, 20(3), 1988, pp 151-159.

115. Rossignac JR and Requicha AAG, Offsetting Operations in Solid Modelling, *Computer Aided Geometric Design*, 3, 1986, pp 129-148.

116. Farmer LE and Gladman CA, Tolerance Technology - Computer Based Analysis, *Annals of the CIRP*, 35/1/1986, 1986, pp 7-10.

117. Requicha AAG and Chan SC, Representation of Geometric Features, Tolerances and Attributes in Solid Modellers Based on Constructive Geometry, *IEEE Journal Robot & Automation*, RA-2 (3), 1986, pp 156-166.

118. Pratt MJ, Synthesis of an Optimal Approach to Form Feature Modelling, *ASME International Computers in Engineering Conference*, vol 1, August 1988, pp 263-274.

119. Jayaraman R and Srinivasan V, Geometric Tolerancing, Part I: Virtual Boundary Requirements, *IBM Journal of Research and Development*, 33(2), 1989, pp 90-104.

120. Srinivasan V and Jayaraman R, Geometric Tolerancing, Part II: Conditional Tolerances, *IBM Journal of Research and Development*, 33(2), 1989, pp 105-124.

121. Elgabry AK, A Framework for a Solid based Tolerance Analysis, *ASME International Computers in Engineering Conference*, July 1986, pp 79-84.

122. Turner JU, *Tolerances in Computer Aided Geometric Design*, PhD Dissertation, Rensselaer Polutechnic Institute, USA, 1987.

123. Turner JU, A Feasibility Space Approach for Automated Tolerancing, *Trans. of ASME, Journal of Engineering for Industry*. 115, 1993, pp 341-346.

124. Roy U and Liu CR, Feature based Representational Scheme of a Solid Modeller for providing Dimensioning and Tolerancing Information, *International Journal of Robot & Computer Integrated Manufacturing*, 4(3/4), 1988, pp 335-345.

125. Mullins SH and Anderson DC, Feature-based Tolerance Representation for Design and Analysis, *Journal of Design and Manufacturing*, 1, 1991, pp 107-118.

126. Lu S C-Y and Wilhelm RG, Automating Tolerance Synthesis: A Framework and Tools, *Journal of Manufacturing Systems*, 10(4), 1991, pp 279-296.

127. Wilhelm RG and Lu S C-Y, Tolerance Synthesis to Support Concurrent Engineering, *Annals of the CIRP*, 41/1/1992, 1992, pp 197-200.

128. Gandhi A and Myklehurst A, A Natural Language Approach to Feature-based Modelling, *Proceedings of the ASME Conference on Advances in Design Automation*, Montreal, 1989, pp 69-77.

129. Requicha AAG and Vandenbrande, Form Features for Mechanical Design and Manufacturing, *Proceedings of ASME International Computers in Engineering Conference and Exposition*, Anaheim, 1989, pp 47-52.

130. Henderson MR and Anderson DC, Computer Recognition and Extraction of Form Features: A CAD/CAM Link, *Computers in Industry*, 6(4), 1984, pp 315-325.

131. Dixon JR, Libardi Jr EC and Nielsen EH, Unresolved Research Issues in Development of Design With Features Systems, In Wozny MJ, Turner JU and Priess, eds., *Geometric Modelling for Product Engineering,* IFIP, North Holland: Elsevier Science, 1990, pp 183-196.

132. Ranyak PS and Fridshal R, Features for Tolerancing a Solid Model, *Proceedings of ASME Computers in Engineering Conference*, vol 1 1988, pp 275-280.

133. Ohsuga A, Toward Intelligent CAD System, *Computer Aided Design*, 21(5), 1989, pp 315-336.

134. Smithers T, AI-based Design versus Geometry-based Design or Why Design Cannot be Supported by Geometry Alone, *Computer Aided Design*, 21(3), 1989, pp 141-150.

135. Dixon JR, Artificial Intelligence and Design: A Mechanical Engineering View, *Proceeding of AAAI 5th National Conference on AI*, vol 2, 1986, pp 872-877.

136. Turing AM, Computing Machinery and Intelligence, *Mind*, 59(236), 1950, pp 433-460.

137. Harmon P and King D, *Expert Systems*, New York: John Wiley & Sons, 1985.

138. Michaelsen RH, Michie D and Boulanger A, The Technology of Expert Systems, *Byte*, April 1985, pp 303-312.

139. d'Agapayeff A, Expert Systems Go to Work, *Data Processing*, 28(3), April 1986, pp 120-135.

140. Black WJ, *Intelligent Knowledge-based System: An Introduction*, Wokingham: Van Nostrand Reinhold, 1986.

141. Buchanan BG and Shortliffe EH, *Ruled-based Expert System*, Reading, MA: Addison Wesley, 1984.

142. Hayes-Roth F, Waterman D and Lenat D, *Building Expert Systems*, Reading, MA: Addison Wesley, 1983.

143. Johnson T, *The Commercial Application of Expert System Technology*, London: Ovum Ltd., 1984.

144. Waterman DA, *A guide to Expert Systems*, Reading, MA: Addison-Wesley, 1986.

145. Parsaye K and Chignell M, *Expert Systems for Expert*, New York: John Wiley and Sons, 1988.

146. Townsend C and Feuct D, *Designing and Programming Personal Expert Systems*, Blue Ridge Summit, PA: TAB Books, 1986.

147. Smith R, *The Facts on File Disctionary of Artificial Intelligence*, New York: Facts on File, 1989.

148. Charniak E and McDermott D, *Introduction to Artificial Intelligence*, Reading, MA: Addison Wesley, 1985.

149. Luger G and Stubblefield W, *Artificial Intelligence and the Design of Expert System*, Menlo Park, CA: Benjamin/Cummings Publishing Co., Inc., 1989.

150. Jackson P, *Introduction to Expert Systems*, Reading, MA: Addison Wesley, 1986.

151. Lecot K and Parker DS, Control Over Inexact Reasoning, *AI Expert*, 1989, pp 28-34.

152. Quillian MR, Semantic Memory, in Minsky M, ed., *Semantic Information Processing*, Cambridge, MA: MIT Press, 1968, pp 227-270.

153. Anderson JR and Bower GH, *Human Associative Memory*, Washington DC: V.H Winston & Sons, 1973.

154. Newell A and Simon H, *Human Problem Solving*, Englewood Cliffs, NJ: Prentice-Hall, 1972.

155. Minsky M, A Framework for Representing Knowledge, in Winston P, ed., *The Psychology of Computer Vision*, New York: McGraw-Hill, 1975.

156. Schank RC and Abelson RP, *Scripts, Plans, Goals and Understanding*, Hillsdale, NJ: Erlbaum, 1977.

157. Stefik M and Bobrow DG, Object Oriented Programming: Themes and Variations, *AI Magazine*, 6(4), 1986, pp 40-64.

158. Langley N, Expert System Shells, *Expert Systems User*, May 1985, pp 16-19.

159. Langley N, Expert System Shells: 2, *Expert Systems User*, June 1985, pp 20-21

160. Lehner PE and Barth SW, Expert Systems of Microcomputers, *Expert System*, 2(4), 1985, pp 198-207.

161. Mercadal D, *Dictionary of Artificial Intelligence*, New York: Van Nostrand Reinhold, 1990.

162. Citrenbaum R, Geisman JR and Schultz R, Selecting a Shell, *AI Expert*, 1987, pp 30-39.

163. Tomiyama T, Kiriyama T, Takeda H, Xue D and Yoshikawa H, Metamodel: A Key to Intelligent CAD Systems, *Research in Engineering Design*, 1, 1989, pp 19-34.

164. Yoshikawa H and Warman EA, eds., *Design Theory for CAD*, Amsterdam: North Holland, 1987.

165. Jakiela MJ and Papalambros PV, Design and Implementation of a Prototype Intelligent CAD system, *Journal of Mechanisms, Transmissions and Automation in Design*, 111, 1989, pp 252-258.

166. Akagi S and Fujita K, Building an Expert System for Engineering Design Based on the Object-oriented Knowledge Representation Concept, *Proceeding of the ASME Conference on Advances in Design Automation*, Florida, DE-14, 1988, pp 309-313.

167. Lukas MP and Pollock RB, Automated Design Thorugh AI Techniques, *Technical Paper*, WTP88-1, Wisdom Systems, Ohio, 1988.

168. Phelps TA, Karra C and Ragsdale KM, Development of Knowledge Based System Implementing Taguchi Class Methods for Engineering Design, *Proceeding of the ASME Conference on Advances in Design Automation*, Florida, DE-14, 1988, pp 303-308.

169. Premkumar P and Kramer SN, A Generalised Expert System for Implementing Mechanical Design Applications: Review, Introduction and Fundamental Concepts, *Proceeding of the ASME Conference on Advances in Design Automation*, Florida, DE-14, 1988, pp 314-319.

170. Yoshikawa H, General Design Theory and a CAD System, in Sata T and Warman EA, eds., *Proceeding of the IFIP W.G. 5.2-5.3 Working Conference on Man/Machine Communication in CAD/CAM 1980*, North-Holland: Amsterdam, 1981, pp 35-53.

171. Tomiyama T and Yoshikawa H, Extended General Design Theory, in Yoshikawa H and Warman EA, eds., *Design Theory for CAD, Proceedings of the IFIP Working Conference on Man/Machine Communication in CAD/CAM 1985*, North-Holland: Amsterdam, 1987, pp 95-124.

172. Smithers T, Tang MX, Tomes N, Buck P, Clarke B, Lloyd G, Poulter K, Floyd C and Hodgkin E, Development of a Knowledge based Design Support System, *Knowledge-based Systems*, 5(1), 1992, pp 31-40.

173. Myers L and Pohl J, ICADS Expert Design Advisor: An Aid to Reflective Thinking *Knowledge-based Systems*, 5(1), 1992, pp 41-54.

174. Design++ 3.0, Product Information Brochure, Design Power Incorporation, Cupertino, California, USA, 1994.

175. Defigner, Software Information Leaflet, LoftTech Incorporated, Marlborough, Massachusetts, USA, 1994.

176. Wood D, AutoCAD 12, *PC User*, 191, Aug 1992, pp 28-30.

177. Booth P, Speaking in Tongue, *CADDESK*, Sept 1990, pp 16-18.

178. Schrage L, *LINDO - User's Manual Release 5.0*, San Francisco: The Scientific Press, 1991.

179. AutoCAD Release 11, *Installation and Performance Guide*, Autodesk Ltd., Oct 1990.

180. Baharin S, Bobrowski R and Smith GF, Intelligent Computer Aided Tolerance Studies, Ahmad MM and Sullivan G, eds., in *Flexible Automation and Integrated Manufacturing*, Florida: CRC Press Inc., 1993, pp 777-788.

181. AutoCAD Release 11, *ADS Reference Manual*, Autodesk Ltd., Oct. 1990.

182. AutoCAD Release 11, *AutoCAD Reference Manual*, Autodesk Ltd., Oct. 1990.

183. Smith J and Gesner R, *Maximizing AutoCAD: Inside AutoLisp*, volume II, Orgon: New Riders Publishing, 1989.

184. Meeran S and Pratt MJ, Automated Feature Recognition from 2D Drawings, *Computer Aided Design*, 25(1), Jan. 1993, pp 7-17.

185. Aldfeld B, On Automatic Recognition of 3D Structures from 2D Representations, *Computer Aided Design,* 15(2), 1993, pp 59-64.

186. Idesawa M, A System to Generate a Solid Figure From a Three View, *Bulletin Japanese Society of Mechanical Engineers*, 16(92), 1973, pp 216-225.

187. Wesley MA and Markowsky G, Fleshing Out Projections, *IBM Journal of Research and Development*, 25(6), 1981, pp 934-954.

188. Preiss K, Constructing the Solid Representation from Engineering Projections, *Computer and Graphic*, 8(4), 1984, pp 381-389.

189. Sakurai H and Gossard DC, Solid Model Input Through Orthographic Views, *ACM Computer Graphic*, 17(3), 1983, pp 243-252.

APPENDIX A-1

EXPERT SYSTEM

APPLICATIONS ACROSS VARIOUS INDUSTRIES

(Sources Wolfgram et. al. [29] and Buchanan BG [30])

AEROSPACE

Communication network diagnosis

Deep space station designer

Diagnosis of airplane engines

Fault isolation in Avionic

Helicopter repair (Boeing)

Navigator for reentry control (NAVEX, NASA)

Spacecraft malfunction diagnosis

AGRICULTURE

Control disease in winter wheat crops (WHEAT COUNSELLOR, ICI)

Controlling plant life in ponds (North Texas State University)

Crop rotation

Management of apple orchards (POMME, Virginia Polytechnic Institute)

Rice disease diagnosis

CHEMICAL

Chemical synthesis planning (SYNCHEM, SUNY at Stony Brook)

Disease of Metals (Westinghouse Electric Company)

Herbicide advisor (British Gas)

Screener for new chemicals (Shell Institute)

Structure elucidation (DENDRAL, Molecular Design Ltd.)

Welding material selector (WELDSELECTOR, Colorado School of Mines)

COMPUTERS AND COMMUNICATIONS

Analyse VMS dump files after system crash (CDX, Digital Equipment Corporation)

Analyse telephone switching system (COMPASS, GTE)

Assist circuit designers with logic design (DAS-LOGIC, Digital Equipment Corporation)

Check order entry (CONAD, Nixdorf)

Computer configuration (XCON, XSEI and XSITE, Digital Equipment Corporation)

Configuring system layouts (Hitachi)

Database management system selection

Diagnose circuit fabrication lines (PIES, Fairchild)

Diagnosing failures in data processing equipment (DIAG8100, Travellors)

Diagnosing failures in disk drive (FAULTFINDER, Nixdorf)

Diagnosing failures in tape drives (A1-SPEAR, Digital Equipment Corporation)

Diagnosis of Cyber NOS-VE system (CDC Dump Analysis, Control Data Corporation)

Hardware diagnosing interpretation (DC, Prime Computer)

Managing resources for chip designers (CALLISTO, Digital Equipment Corporation)

Monitoring MVS operating systems (YES/MVS/ IBM)

Robot sensor interpretation

Sequence steps in PC board assembly (HI CLASS, Hughes Electronic & Data Systems)

Software job coating (COCOMO 1, Level Five Research)

Software selection consultant

Software system troubleshooter

Troubleshooting photolithography steps in circuit fabrication (PHOTOLITHOGRAPHY ADVISOR, Hewlett Packard)

Troubleshooting Ethernet networks (NTC, Digital Equipment Corporation)

Troubleshooting telephone lines (ACE, Southwest Bell)

Troubleshooting digital voltage sources (DIG VOLTAGE TESTER, Lockheed)

Troubleshooting communication hardware (BDS, Lockheed)

DRILLING

Analysis of oil well logging datas (DIPMETER ADVISOR, Schlumberger)

Diagnosing drilling problems (MUDMAN, N.L. Industries)

Problem analysis on drill bits (SECOFOR, Elf-Acquitaine)

EDUCATION

Debugging PASCAL programs

Expert general library reference (Drexel University)

Learning disability classification advisor

Speech pathology advisor

Student behaviour consultant

Technical engineering education

Test result interpreter

Textbook selection advisor

Tutor designers in design checking (DECGUIDE,Lockheed,Sunnyvale)

Tutoring users of VMS operating system (Digital Equipment Corporation)

Worksheet generator

ENGINEERING

Carburetor fault diagnosis

Construction project planning and evaluation

Design of foundations for bridges and buildings (Carnegie-Mellon University)

Design of motor components

Engineering change order manager

Fastener selection

Front end for engineering design package

Linear programming system (American University)

Machine room safety

Material handling equipment selector (North Caroline State University)

Material selection

Road barrier requirements

Site planning for chemical plant

Statistical analysis tool selector

Statistical consultant (Carleton College)

Symbolic integral calculus (MACSYMA,M.I.T.)

troubleshooting steam propulsion plants (STEAM, Navy Research Centre)

Weight estimator for evolving designs

ENVIRONMENT

Environmental regulations

Mineral deposit relationships (PROSPECTOR, SRI)

Water discharge permit review (Environmental Protection Agency)

Weather forecasting

FINANCIAL SERVICES

Advise in disclosure of confidential information (EDDAS, Environmental Protection Agency)

Advice on insurance underwriting

Analysis of risk insurance (UNDERWRITING ADVISOR, Syntelligence)

Assess commercial insurance risks

Bank services advisor

Brokerage legislation

Claim estimation

Commodity buying

Conflict-of-interest consultant

Credit approval (LENDING ADVISOR, Syntelligence)

Electronic banking services

Financial analysis (Palladin)

Financial planning advisor (PLANPOWER, Applied Expert Systems)

Financial statement analysis

Foreign exchange rates

International tax crediting

Legal analysis of contract claims

Loan application assistant

Performance evaluation of dealerships

Predicting business insolvency (University of Texas at Dallas)

Staff loan scheme

Stock broker marketing advice

Stock exchange regulations

Tax advisor

MANAGEMENT

Analyze battlefield intelligence (TRW)

Business productivity tool (GURU, Micro Data Base Systems)

Contingency planning

Corporate distribution anaylsis (INET)

Corporate structure

Corporate takeover

Creating documents (DOCUMENT MODELLER, Model Office Company)

Database management system purchase advisor (Boeing)

Internal auditing

Inventory management advisor

Management portfolio expert

Management training

Naval acquisition management (ACQUISITION MANAGER ASSISTANT, U.S. Army)

Personnel planning and processing U.S. Army)

Project management

Process management and information systems (MOD 300, Taylor Institute)

Qualitative reasoning for long-range planning (ROME)

Quality assurance standards

Tactical mission planning

MANUFACTURING

Analysis and prevention of mechanical failures (Duke University)

Chemical material selection

Continuous-process manufacturing advisor

Control railroad train braking system

Detecting cracks in billets (Kawasaki Steel)

Diagnosis of electronic controls

Diagnosis of computer board faults (ITT)

Diagnosis of railroad locomotives (DELTA, General Electric)

Diagnosis of hydrostatic sterilizers (Campbell Soup)

Drilling advisor for machining

Electrical system fault diagnosis

Fault diagnosis for auto subsystem

Gas turbine engine fault diagnosis

Maintenance advisor for hydraulic system

Maintenance of epireactor (IMP, Texas instruments)

Newspaper layout design (Composition Systems)

Optimum performance maintenance (Ingalls Shipbuilding)

Power supply fault diagnosis

Process control applications (PICON, Lisp Machine, Inc.)

Refinery process control

Sensor verification for power plant

Sequencing computer board assembly (HI CLASS, Hughes)

Tooling selection for machining

Troubleshooting circuits (Hewlett-Packard)

Troubleshooting paper plants (ACID)

MEDICAL

Adverse drug reaction

Arthritis and rheumatism expert (AI/RHEUM, Knowledge research)

Cancer management (ONOCOCIN, Stanford University Oncology)

Diagnosis of poisoning via telephone hotline (Johns Hopkins university)

Diagnosis of learning disabled (CLASS. L.D. Utah State University)

Diagnosis of pulmonary diseases (PUFF, Pacific Medical Centre)

Diagnosis of infectious diseases (MYCIN)

Health care billing advisor (Ohio State University)

Learning to use MYCIN (GUIDON, Stanford University)

Medical expert (CADAUCEUS)

Serum protein analysis (Helena Laboratories)

ORDERING SYSTEMS AND MARKETING

Configures VAX orders (XCON, XSEL, and XSITE, Digital Equipment Corporation)

Order checking (OCEAN, NCR Corporation)

Order entry checking (Nixdorf)

Promotions of goods (PROMOTER, MDS)

APPENDIX A-2

EXPERT SYSTEM SHELL

(Sources Mercadal [16] and Langley [58,59]).

ACQUAINT	Supports forward and backward chaining, certainty factors, contextual rule sets, frames, demons and fuzzy comparison. There is a database facility and a forms facility.	Lithp System BV, PO Box 65, 1120 AB Landsmeer, The Netherlands.
Actor	An object-oriented programming language.	The Whitewater Group, Technology Innovation Centre, 906 University Pl., Evanston, IL 60201, USA.
AI-NET	A neural net that learns by example and automatically generalizes to solve similar problems. The expert system can be embedded into existing programs.	AI-WARE, Inc., 11000 Cedar Ave., Suite 212, Cleveland, OH 44106, USA.
ALEX	Based on Smalltalk/V for IBM-PC. Features include debugging facilities, windowing and the ability to add new features. It can interface with Smalltalk, PROLOG\V, graphics, databases, spreadsheet and assembly language.	Harris & Hall Associates, PO Box 1900, Port Angles, WA 98362, USA.
Arborist	A decision -support program that utilised operation research techniques to build decision trees.	Texas Instrument Corp., PO Box 809063, Dallas, TX 75380, USA.
Arity/Expert	A backward chaining expert system written in PROLOG for IBM-PC. The program can connect to databases.	Arity Corporation, 30 Domino Dr., Concord, MA 01742, USA.
ART	Features include: forward and backward chaining, logic-based programming, opportunistic rule application, incremental rule compilation, rule priorities, subgoaling, comparative evaluation, time-based reasoning, planning, simulation, meta-knowledge and colour graphics. The primary control mechanism is blackboard architecture and runs on Symbolic LISP machines.	Inference Corporation, 5300 West Century Blvd., Los Angeles, CA90045, USA.

CxPERT	This program can be embedded in C programs. It has backward and forward chaining, attribute value pairs and rules. The rule can include multiple antecedents and multiple consequent. The program's frame-based component includes arrays of frames, dynamic creation of frames, multiple inheritance and attached procedure. The programs runs on IBM-PCs.	Software Plus, 1652 Albermarie Drive, Crofton, MD 21114, USA.
ES/P ADVISOR	A text-oriented expert system, that is the system will function well with knowledge in the form of complex instructions or regulations. It has open-ended architecture so the user can resort to PROLOG when needed. Variable types include fact, number, category and phrase. The program runs on IBM-PCs.	Expert Systems International, 1700 Walnut St., Philadelphia, PA 19103, USA.
Expert Ease	A decision-support expert system. The program develops rules from examples but can only achieve one goal. The program runs on IBM-PCs. DEC Rainbow and Victor 9000 computers.	Human Edge Software, 2445 Faber Pl., Palo Alto, CA 94303, USA.
EXPERT EDGE	A rule-base Bayesian probability expert system. The program will interface with popular spreadsheet and databases. It runs on personal computers.	Human Edge Software, 2445 Faber Pl., Palo Alto, CA 94303, USA.
EXSYS	A forward and backward chaining, if/then/else rule-based system. The system can handle more than 5,000 rules. Rules can have multiple antecedents and consequent. The OR connective is allowed. Knowledge is represented as object value pairs. Depth first search is used. EXSYS can interface with databases and external programs. It can runs on IBM-PCs.	EXSYS Inc., PO Box 75158, Control Stn. 14, Albuquerque, NM 87194, USA.
Envisage	A PROLOG-based tool which is an OPS5 type of production expert system. Features include demons, simulation, fuzzy logic and Bayesian probability. The program runs on VAX mini and mainframes and MicroVAX.	Systems Designers Software Inc., 444 Washington Street, Suite 407, Woburn, MA 01801, USA.
GoldWorks	An expert system shell closely linked with Golden common LISP. The program offers frames, forward chaining rules and object programming. The frame-based system uses demons and is capable of multiple inheritance. Knowledge-based partitioning and metalevel inference are present. It can interfaces with databases, spreadsheet and C. It can address up to 15 Mb of memory. Use IBM-PC.	Gold Hill Computers Inc., 26 Landsdowne St. Cambridge, MA 02139, USA.

GURU	An expert system with its own database, spreadsheet, graphics, natural language facility and report generator. Features include metarules, mixed forward and backward chaining and multiple rule firing capability. Each rule can have a priority, cost and test method. IBM PC, DEC VAX 11 and DEC VAX Mate computers can use this program.	MDBS, PO Box 248, Lafayette, IN 47902, USA.
Intelligence / Compiler	A hybrid expert system with forward and backward chaining, inexact reasoning, semiexact reasoning, frames, tables and a relational database. The rule base can be partitioned. The program includes multiple inheritance with acceptance and attached procedures as part of the package. The program is written in C and will interface with C and DOS. Access to dBaseIII, Lotus and other databases and spreasheets is possible.	IntelligenceWare Inc., 9800 South Sulveda Blvd., Suite 730, Los Angeles, CA 90045, USA.
Knowledge Craft	A high-end frame-based tool with procedural attachment. The inheritance mechanism is highly flexible and does not limit the user to the standard is-a inheritance. Logic programming, rule-based programming, object oriented programming and alternate worlds are all available. Able to use LISP, the underlying language. Use with Sun, Apollo, MicroVAX and Lisp computers.	The Carnegie Group Inc., 650 Commerce Court Station Sq., Pittsburgh, PA 15219, USA.
KnowledgePro	An expert system with hypertext and interfaces to dataBaseIII, Lotus 123 and computer languages. It can handle list processing, string manipulation, tracing facilities, menu making facilities, procedural control, inheritance and topics. The windows facility allows direct links to Excel and other program that use Windows.	Knowledge Garden Inc., 473A Malden Bridge Rd., Nassau, NY 12123, USA.
M1	A backward chaining rule-based system. It does not have rule-base partitioning. Features include certainty factors, multiple instantion, an input forms facility, multiple windows, metafacts, list processing, metarules, metapropositions, an interactive database, limited forward chaining and able to check responses for validity. Program runs on IBM-PC.	Teknowledge, 1850 Embarcadero Rd., Palo Alto, CA 94303, USA.
MacSMARTS	A rule-based forward and backward chaining system with an induction component. It has the capacity for object oriented secondary links with HyperText and Hyper Graphics. It has graphics capabilities and takes advantage of the Macintosh interface. The program runs on the Macintosh.	Cognition Technology, 55 Wheeler St., Cambridge, MA 02138, USA.

NEXPERT OBJECT	A midrange hybrid expert system tool. The program's features include object and rule representation, integrated forward and backward chaining, inexact reasoning, incremental compilation, automatic goal generation, demons, methods, rule priorities, multiple and user defined inheritance and metaslots. Computers that can run this program include: IMB-AT, Macintosh, MicroVAX, Apollo and Sun Microsystems.	Neuron Data Inc., 444 High Street, Palo Alto, CA 94301, USA.
Personal Consultant Plus	The program is designed on a frame-based system and employs rules in either a backward or forward chaining mode. Demons are available. The knowledge base consists of contexts, parameters and production rules. Contexts are produce by breaking up a large problem into subproblems. A context can inherit values and rules. Parameters can be multiple valued. Rule bases can also be partitioned. It can access popular databases and the underlying LISP language. Use with IBM-PC.	Texas Instruments Corp., PO Box 809063, Dallas, TX 75380, USA.
Personal Consultant Easy	A rule-based backward chaining expert system patterned after MYCIN. Some limited forward chaining is available. Knowledge is represented as attribute value pairs. The program's knowledge base consists of production rules and parameters. It is upwardly compatible with Personal Consultant Plus. The program runs on IBM-PC.	Texas Instruments Corp., PO Box 809063, Dallas, TX 75380, USA.
Wizdom Expert System	This program supports proposition and object-oriented programming forward and backward chaining, fuzzy logic, frames, scenario matching and scanning, a semantic definition language and incremental knowledge maintenance. It can address up to 16 Mb of RAM. The program runs on the IBM-PC and PC/AT.	SIL Inc., 1593 Locust Ave., Bohemia, NY 11716, USA.

APPENDIX B-1

KATA'S ALGORITHM

```
;;; File name    : Kcad.lsp
;;;
;;; DESCRIPTION
;;;
;;; The c:kcad function is used to install all essential Autolisp sub-routine programs
;;; and check all essential external programs and all insert block drawings.
;;;
;;; In the Autocad command line, type
;;;
;;; COMMAND: (load "kcad")
;;; COMMAND: kcad
;;;
;;; Remark : before loading KCAD program, please make sure the ACAD.lsp is loaded
first.
;;;
(defun c:kcad(/ fp count ef)
    (setvar "cmdecho" 0)
    (if (findfile "kcad.cfg")    ; find config file
        (progn
          (setq fp (open "kcad.cfg" "r")
            sysdir (read-line fp)
          );setq
          (close fp)
        );progn
        (progn
          (prompt "\nKCAD.CFG file not found!")
          (setq sysdir (strcat (getvar "dwgprefix") "kcad\\"))
        );progn
    );if
    (if (findfile (strcat sysdir "welcome.sld"))
       (command "vslide" (strcat sysdir "welcome")) ; display welcome screen
    );if
    (prompt "\nLoading KCAD.....")

    (setq ef 0)

    ;-------------- Essential Autolisp programs ------------;

    (if (ilload sysdir "error") (setq ef (+ ef 1)) (setq ef 0))   ; 1
    (if (ilload sysdir "setup") (setq ef (+ ef 1)) (setq ef 0))   ; 2
```

```
(if (ilload sysdir "db_tool") (setq ef (+ ef 1)) (setq ef 0)) ; 3
(if (ilload sysdir "getts") (setq ef (+ ef 1)) (setq ef 0))   ; 4
(if (ilload sysdir "att_tool") (setq ef (+ ef 1)) (setq ef 0)) ; 5
(if (ilload sysdir "pplan") (setq ef (+ ef 1)) (setq ef 0))   ; 6

(setq count 0)

;--------------- Knowledge-base-system programs ------------

(ck_file (getvar "dwgprefix") "autosel.exe")   ; 1
(ck_file (getvar "dwgprefix") "bar_sel.exe")   ; 2
(ck_file (getvar "dwgprefix") "mc_upd.exe")    ; 3
(ck_file (getvar "dwgprefix") "tool_upd.exe")  ; 4
(ck_file (getvar "dwgprefix") "proc_upd.exe")  ; 5
(ck_file (getvar "dwgprefix") "p_upd.exe")     ; 6
(ck_file (getvar "dwgprefix") "cut_upd.exe")   ; 7
(ck_file (getvar "dwgprefix") "t_matlup.exe")  ; 8
(ck_file (getvar "dwgprefix") "matl_up.exe")   ; 9
(ck_file (getvar "dwgprefix") "t_f_up.exe")    ; 10
(ck_file (getvar "dwgprefix") "f_up.exe")      ; 11

;---------------- Manufacturing knowledge base ----------

(ck_file (getvar "dwgprefix") "bar.dat")       ; 12
(ck_file (getvar "dwgprefix") "mc_cut.kb")     ; 13
(ck_file (getvar "dwgprefix") "material.kb")   ; 14

;---------------- Tolerance chain program ----------

(ck_file (getvar "dwgprefix") "tolink1.exe")   ; 15

;---------------- Feature Recognition program --------

(ck_file (getvar "dwgprefix") "fr.exe")        ; 16
(ck_file (getvar "dwgprefix") "round.fr")      ; 17
(ck_file (getvar "dwgprefix") "preproc.exe")   ; 18
(ck_file (getvar "dwgprefix") "preproc.hlp")   ; 19

;---------------- slide library --------------

(ck_file (getvar "dwgprefix") "tol.slb")       ; 20
(ck_file (getvar "dwgprefix") "dim.slb")       ; 21
(ck_file (getvar "dwgprefix") "dim1.slb")      ; 22
(ck_file (getvar "dwgprefix") "kb.slb")        ; 23
(ck_file (getvar "dwgprefix") "kcam.slb")      ; 24
(ck_file (getvar "dwgprefix") "kcam1.slb")     ; 25
```

```
(ck_file (getvar "dwgprefix") "fr.slb")         ; 26

;---------------- Lindo program -------------------

(ck_file (getvar "dwgprefix") "lindo.exe")      ; 27

;---------------- Prolog error file ----------------

(ck_file (getvar "dwgprefix") "prolog.err")     ; 28

;--------------- insert block drawing -----------------

(ck_file sysdir "bar.dwg")                 ; 29
(ck_file sysdir "pplanx.dwg")              ; 30
(ck_file sysdir "pplany.dwg")              ; 31
(ck_file sysdir "dim_2.dwg")               ; 32
(ck_file sysdir "dim_h.dwg")               ; 33
(ck_file sysdir "dim_v.dwg")               ; 34
(ck_file sysdir "dim_c.dwg")               ; 35
(ck_file sysdir "dim_h_g.dwg")             ; 36
(ck_file sysdir "dim_v_g.dwg")             ; 37
(ck_file sysdir "gt_pos.dwg")              ; 38
(ck_file sysdir "gt_symx.dwg")             ; 39
(ck_file sysdir "gt_symy.dwg")             ; 40
(ck_file sysdir "abs.dwg")                 ; 41
(ck_file sysdir "dial.dwg")                ; 42
(ck_file sysdir "datum.dwg")               ; 43
(ck_file sysdir "arrow.dwg")               ; 44
(ck_file sysdir "gn_tol.dwg")              ; 45
(ck_file sysdir "surf_no.dwg")             ; 46
(ck_file sysdir "facing.dwg")              ; 47

;------------ template file --------------

(ck_file sysdir "pplanx.txt")              ; 48
(ck_file sysdir "pplany.txt")              ; 49
(ck_file sysdir "dimtempx.txt")            ; 50
(ck_file sysdir "dimtempy.txt")            ; 51

;------------ Kcad menu ------------------

(ck_file sysdir "kcad.mnu")                ; 52

;------------ Kcad Autolisp programs ------

(ck_file sysdir "dim.lsp")                 ; 53
```

```
    (ck_file sysdir "rhf.lsp")              ; 54
    (ck_file sysdir "prepdxf.lsp")          ; 55
    (ck_file sysdir "r_cut.lsp")            ; 56
    (ck_file sysdir "bp_dim_v.lsp")         ; 57
    (ck_file sysdir "gt_pos.lsp")           ; 58
    (ck_file sysdir "dummy.lsp")            ; 59
    (ck_file sysdir "simtool.lsp")          ; 60
    (ck_file sysdir "lhtef.lsp")            ; 61
    (ck_file sysdir "gn_tol.lsp")           ; 62
    (ck_file sysdir "b_dim_vg.lsp")         ; 63
    (ck_file sysdir "gt_sym.lsp")           ; 64
    (ck_file sysdir "mill.lsp")             ; 65
    (ck_file sysdir "bar.lsp")              ; 66
    (ck_file sysdir "tol_opt.lsp")          ; 67
    (ck_file sysdir "rhftf.lsp")            ; 68
    (ck_file sysdir "bp_dim_c.lsp")         ; 69
    (ck_file sysdir "get_gt.lsp")           ; 70
    (ck_file sysdir "dim_v_g.lsp")          ; 71
    (ck_file sysdir "drill.lsp")            ; 72
    (ck_file sysdir "side.lsp")             ; 73
    (ck_file sysdir "rhtf.lsp")             ; 74
    (ck_file sysdir "rhac.lsp")             ; 75
    (ck_file sysdir "b_dim_v1.lsp")         ; 76
    (ck_file sysdir "surf_no.lsp")          ; 77
    (ck_file sysdir "abs_h.lsp")            ; 78
    (ck_file sysdir "abs_v.lsp")            ; 79
    (ck_file sysdir "graphx.lsp")           ; 80
    (ck_file sysdir "graphy.lsp")           ; 81
    (ck_file sysdir "rhtc.lsp")             ; 82
    (ck_file sysdir "bp_dim_h.lsp")         ; 83
    (ck_file sysdir "bp_dim_g.lsp")         ; 84
    (ck_file sysdir "hole.lsp")             ; 85

    (if (and (= count 85) (= ef 6))
        (progn
          (command "menu" (strcat sysdir "kcad"))
          (prompt "\nInstallation OK !")
          (setq ans (getstring "\nPress [Enter] to continue"))
          (command "redraw")
          (setup)
        );progn
        (prompt "\nInstallation not OK !")
    );if
    (setvar "cmdecho" 1)
    (princ)
);defun
```

```
(princ)
(defun ck_file ( fdir ffile / fdir ffile)
    (if (findfile (strcat fdir ffile))
      (setq count (+ count 1))
      (progn
        (setq count 0)
        (prompt "\n[")
        (princ ffile)
        (princ "] not found!")
      );progn
    );if
    (princ)
);defun
(princ)
```

PROCESS PLAN

```
(defun pplan (f_code p_name mc_no pc_max pc_min mc_cost tool_no tw_factor
        t_cost rpm feed dist1 manu_tol dia manu_rad_tol surtext
        d1 nose angl st_x end_x pt1 pt2 hov / hov pt3 ppname ort)
    (setvar "attdia" 0)
    (command "layer" "make" "pplan" "")
    (setq pt3 (getpoint "\nSelect the Cut_dim location:")
    );setq
    (prompt "\nCut No.<")
    (princ #txt)
    (setq txt (getint ">: "));setq txt
    (if txt  (setq #txt txt))
    (if (= (strcase hov) "VER")
        (setq ort "90" ppname "pplany")
        (setq ort "0"  ppname "pplanx")
    );if
    (command "dimsah" "1")
    (command "dim" hov pt1 pt2 pt3 " " cancel)
    (command "insert" (strcat sysdir ppname) pt3 txtsize "" ort
                    #txt
                    f_code
                    p_name
                    mc_no
                    pc_max
                    pc_min
                    mc_cost
                    tool_no
                    tw_factor
```

```
                    t_cost
                    rpm
                    feed
                    dist1
                    manu_tol
                    "??"
                    dia
                    manu_rad_tol
                    surtext
                    d1
                    nose
                    angl
                    st_x
                    end_x
        );command
        (setvar "attdia" 1)
        (princ)
);defun
(princ)
```

TOLERANCE CHAIN ALGORITHM

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int mc_size,bp_size;
float mc_cut[100][7];
float bp-dim[100][5];

void read_file(char *fp1, char *fp2) :

void read_file(char *fp1, char *fp2)
{
   int i, m;
   char data1[11], data3[11], data4[11], data5[11], data6[11], data7[11];
   float no,d1,f_code,manu_tol_i,pc_min,start_x,end_x,dim_no,bp_tol,dim_code;
   FILE *f1; FILE *f2;
   i=0; m=0;
   f1=fopen(fp1,"r");
   while (!feof(f1))   {
      fscanf(f1,"%s %s %s %s %s %s %s \n",data1,data2,data3,data4,data5,data6,data7);
         no=atof(data1);
         d1=atof(data2);
```

```
            f_code=atof(data3);
            manu_tol_i=atof(data4);
            pc_min=atof(data5);
            start_x=atof(data6);
            end_x=atof(data7);
            if ((start_x!=0) & & (end_x!=0))    {
                  mc_cut[i][0]=no;
                  mc_cut[i][1]=f_code;
                  mc_cut[i][2]=d1;
                  mc_cut[i][3]=manu_tol_i;
                  mc_cut[i][4]=pc_min;
                  mc_cut[i][5]=start_x;
                  mc_cut[i][6]=end_x;
                  i=i+1;
            } /* endif */
} /* while */
fclose (f1);
mc_size=i;
f2=fopen(fp2, "r");
while (!feof(f2))        {
        fscanf(f2,"%s %s %s %s %s \n",data1,data2,data3,data4,data5);
             dim_no=atof(data1);
             dim_code=atof(data2);
             bp_tol=atof(data3);
             start_x=atof(data4);
             end_x=atof(data5);
             if ((start_x!=0) & & (end_x!=0))    {
             } /* endif */
} /* while */
fclose (f2);
bp_size=m
}   /* read_file */

main (argc,argv)
int argc;
char *argv[ ];
{

        float sr_cut_link[50] [2];
        float bp_dim_link[50] [2];
        float ignore_cut[50];
        float sr_cut_index[50];
        char data1[8], data2[7], data3[13], data4[3], data5[13];
        int i, z, w, x, x1, end_x1, end, start,
            found_ignore, found _before;
        float sr_cut_no1, start_x1,end_x1, end, start,
```

```
        sr_cut_no, start_x, end_x,
        link_x, max_num, tol,
        total_sr_tol;
      FILE *f3; FILE *f5; FILE *f4, FILE *f6, FILE *f7;
      if(argc!=6) (
        exit (0);
}
f3=open("input.jnk","w");
read_file(argv[1],argv[2]);
num=0;
for(i=0;i<mc_size;i++)  {              /* all final cut  */
      if ((mc_cut[i][2]==0) & & (mc_cut[i][0]!=0))   {
        sr_cut_no1=mc_cut[i][0];
        sr_cut_link[num][0]=mc_cut[i][0];
        sr_cut_link[num][1]=mc_cut[i][0];
        num=num+1;
        start_x1=mc_cut[i][5];
        end_x1=mc_cut[i][6];
        sr_cut_no=sr_cut_no1;
        start_x=start_x1;
        end_x=end_x1;
        close=0;
        max=i;
        found=0;

        for (k=0;k<mc_size;k++)  {   /* check close loop cut */
             if ((mc_cut[k][5]==end_x) &&
                 (mc_cut[k][6]==start_x) &&
                 (mc_cut[k][0]==sr_cut_no-1))   {
                        max=k; found=1; close=1; k=mc_size;
                 }
             } /* for k loop check close loop cut */
        if (close==1)   {
            sr_cut_link[num][1]=mc_cut[max][0];
            sr_cut_link[num][0]=sr_cut_no1;
            sr_cut_no=mc_cut[max][0]
            num=num+1;
            start_x=mc_cut[max][5];
            link x=start_x;

          } /* for k loop  */
          if (found==1)  {
                found_before=0;
                for (k=0;k<mc_size;k++)  {
                        if ((mc_cut[k][6]==link_x)  &&
                           (mc_cut[k][0]<sr_cut_no1)  &&
```

```
                        (mc_cut[k][0]>=max_num))  {
                                max=k; max_num=mc_cut[k][0]; found=1;
                        } /* end if */
} /* while loop */

/*------------- find all sr_cut link to end_x of final_cut i --------------*/

sr_cut_no=sr_cut_no1;
end-x=end_x1;
start_x=start_x1;
link_x=end_x1;
found=1;
while ((found==1  &&  (close==0))   {
        found=0; max_num=0;
        for  (k=0; k<mc_size; k++)   {
                if  ((mc_cut[max][6]==link_x)  &&
                    (mc_cut[k][0]<sr_cut_no)   &&
                    (mc_cut[k][0]>=max_num))   {
                                max=k;  max_num=mc)cut[k][0]; found=1;
        }  /* for k loop*/
        if (found==1)   {
                found_before=0;
                for (s-0; s<num; s++)    {
                        if  ((mc_cut[max][0]==sr_cut_link[s][1])  &&
                            (sr_cut_no1==sr_cut_link[s][0]))  {
                                found_before=1; found=0;
                        } /*  end if found  */
                }   /*  for s loop  */
         } /* end if loop */
         if ((found_before==0)  &&  (found=1))   {
                        sr_cut_link[num][1]=mc_cut[max][0];
                        sr_cut_link[num][0]=sr_cut_no1;
                        sr_cut_no=mc_cut[max][0];
                        num=num+1;
                        start_x=mc_cut[max][5];
                        link_x=start_x;
                } /* end if  */
            } /* while loop */
        } /* end if */
 }  /* for i loop  */

/* -------------- find bp_dim link -------------------- */

/*--------------- check ignore cut ------------------- */

n=0;
```

```
          for (k=0; k<mc_size; k++)  {       /* check ignore cut */
               if (mc_cut[k][2]==0)  {   /*all final cut */
                    for (m=0; m<mc_size, m++)  {  /* all mc_cut */
                         if ((mc_cut[m][5]==mc_cut[k][6])  &&
                              (mc_cut[m][6]==mc_cut[k][5]  &&
                              (mc_cut[m][0]==mc_cut[k][0]-1  &&
                              (mc_cut[m][2]==0  &&
                              (mc_cut[m][1]!=0))  {
                              ignore_cut[n]=mc_cut[m][0];
                         }  /* end if  */
                    }  /* for */
               }  /* for k loop check ignore cut */

/* ------------------- find LHS start ---------------------*/

start=10000;
for (k=0, k<bp_size; k++)  {
          if (bp_dim[k][3]<start)  {
          start=bp_dim[k][3];
          }  /*if  */
}  /*  for k loop  */

/*  ------------------ find RHS -------------------------*/

end=0;
for (k=0, k<bp_size; k++)  {
          if (bp_dim[k][3]>end)  {
               end=bp_dim[k][3];
          } /*  if  */
          if (bp_dim[k][4]>end)  {
               end=bp_dim[k][4];
          } /* if  */
}  /*  for k loop  */

/* ----------------find all sr_cut link to bp_dim i ----------------*/

num1=0;
for (i=0; i<bp_size; i++)  {
          bp_dim_link[num][0]=bp_dim[i][0];
          bp_dim_link[num][1]=0;
num1=num1+1;
start_x1=bp_dim[i][3];
end_x1=bp_dim[i][4];
sr_cut_no=1000;
start_x=start_x1;
end_x=end_x1;
```

```
link_x=start_x1;

/* --------find direct cut --------*/

max_num=0; c;ose=0;
for (k=0; k<mc_size, k++)  {
       if (mc_cut[k][2]==0)  {
                 if (((mc_cut[k][5]==start_x1)  &&  (mc_cut[k][6]==end_x1))  &&
                    ((mc_cut[k][5]==end_x1)  &&  (mc_cut[k][6]==start_x1))  &&
                    (mc_cut[k][0]>max_num))   {
                         close=1;max_num=mc_cut[k][0];
                  } /* if */
                  found_ignore=0;
                  for (z=0; z<n; z++)   {
                          if (mc_cut[k][0]==ignore_cut[z])   {
                                   found_ignore=1;
                          }  /* end if */
                   } /* for */
        } /* if */
} /* for k loop */
if (( close==1) && (close==0))  {
         bp_dim_link[num][1]=mc_cut[max][0];
         bp_dim_link[num][0]=bp_dim[i][0];
         num1=num1+1;
         k=mc_size;
         close=1;
} /* if */

/* ---------- find all sr_cut link to start_x of bp_dim i ----------*/

found=1;
while ((found==1) && (close==0))  {
         found=0; max_num=0;
         for (k=0; k<mc_size, k++)  {
                if ((mc_cut[k][6]==link_x)  &&
                    (mc_cut[k][0]<sr_cut_no)  &&
                    (mc_cut[k][2]==0)  &&
                    (mc_cut[k][0]>=max-num))  {
                            max=k; max_num=mc_cut[k][0]; found=1;
         } /* for k loop */
         if(found==1) {
               found_ignore=0
               for (k=0; k<n; k++)   {
               if (mc_cut[max][0]==ignore_cut[k])   {
               found_ignore=1; found=0;
         } /* for */
```

```
} /* end if */
if(found==1) && (found_ignore==0)) {
                bp_dim_link[num1][1]=mc_cut[max][0];
                bp_dim_link[num1][0]=bp_dim[i][0];
                sr-cut_no=mc_cut[max][0];
                num1=num1+1;
                start_x=mc_cut[max][5];
               link_x=start_x;
               if (( start_x==end) && (start_x==start)) {
                   found=0;
        } /* end if */
} /* while loop */

/* ---------- find all sr_cut link to end_x of bp_dim i ---------*/

sr_cut_no=1000;
end_x=end_x1;
start_x=start_x1;
link_x=end_x1;
found=1;
while ((found==1) && (close==0))  {
          found=0; max_num=0;
          for (k=0; k<mc_size; k++)   {
                if ((mc_cut[k][6]==link_x) &&
                    (mc_cut[k][0]<sr_cut_no)  &&
                    (mc_cut[k][2]==0)  &&
                    (mc_cut[k][0]>=max_num))  {
                         max=k; found=1; max_num=mc_cut[k][0];
                }
          } /* for k loop */
          if(found==1)  {
               found_ignore=0;
               for (k=0; k<n; k++)  {
                     if(mc_cut[max][0]==ignore_cut[k])  {
                            found_ignore=1; found=0;
                     } /* endif */
               } /* for */
          } /* if */
          if((found==1) && (found_ignore==0))  {
               found_before=0;
               for(s=0; s<num1=1; s++)  {
                     if((mc_cut[max][0]==np_dim_link[s][1])  &&
                            (bp_dim[i][0]==bp_dim_link[s][0]))  {
                                   found_before=1; found=0
                            } /* if */
                    } /* for s loop */
```

```
        } /* end if found */
        if((found==1) && (found_before==0) && (found_ignore==0))  {
                        bp_dim_link[num][1]=mc_cut[max][0];
                        bp_dim_link[num][0]=bp_dim[i][0];
                        sr_cut_no=mc_cut[max][0];
                        num1=num1+1;
                        start_x=mc_cut[max][5];
                        link_x=start_x;
                        if((start_x==end) && (start_x==start)) {
                              found=0;
                        } /* endif */
                  } /* end if */
        } /* while loop */
} /* for i loop */

/* -------------write LINDO input.jnk -------------*/

fprintf(f3,"\n");
fprintf(f3,"\npage");
fprintf(f3,"\n");
fprintf(f3,"\n");
w=0; z=0; x1=0;

/* --------- minimize tolrance stackup ------------*/

fprintf(f3,"\nMin");
f6=fopen(argv[3],"w");
f7=fopen(argv[4],"w");
for(n=0; n<mc_size; n++)  {
     s=0;
          for(i=0; i<num; i++) {
               if(sr_cut_link[i][0]==mc_cut[n][0]) {
                    s=s+1;
               } /* if */
          } /* for i loop */
          if (s>1) {
                    w=w+1;
                    sr_cut_index[x1]=mc_cut[n][0];
                    x1=x1+1;
                    if(w==1)
                          fprintf(f3,"w%d",w);
                    else
                          fprintf(f3,"+w%d",w);
          } /* if */
 } /* for n loop */
 for(i=0; i<bp_size; i++)  {
```

```
          z=z+1;
          fprintf(f3,"+z%d",z);
} /* for i loop */
x=z+w;

/* ------------- sr_cut tolerance chain LP model ---------------*/

frpintf(f3,"\nst");
w=1;
for(i=0; i<mc_size; i++) { /* print sr result */
          found=0;
          for(k=o; k<x1; k++)  {
               if(mc_cut[i][0]==sr_cut_index[k]) found=1;
          } /* for k */
          if(found==1) {
               sr_cut_no=mc_cut[i][0];
               total_sr_tol=mc_cut[i][3];
               fprintf(f3,"n\%0.0f ", sr_cut_no);
               fprintf(f6,"0.0f  %0.4f  %0.4f", sr_cut_no, mc_cut[i][5], mc_cut[i][6];
               for(s=0; s<num; s++)  {
                    if(sr_cut_link[s][0]==sr_cut_no)  &&
                         (sr_cut_link[s][1]!=sr_cut_no))  {
                         fprintf(f3,"+t%0.0f", sr_cut_link[s][1];
                         for(n=0; n<mc_size; n++)  {
                              if(mc_cut[n][0]==sr_cut_link[s][1])  {
                                 fprintf(f6," %0.0f  %0.4f  %0.4f", sr_cut_link[s][1],
                                             mc_cut[n][5],mc_cut[n][6]);
                                   total_sr_tol+total_sr_tol+mc_cut[n][3];
                              } /* endif */
                         } /* for n loop */
                    } /* endif */
               } /* for s loop */
               fprintf(f6,"\n");
               fprintf(f3,"w%d=%0.4f", w, total_sr_tol);
               w=w+1;
               } /* endif found */
} /* for i loop */

/* -------- angular cut equation -------*/

for(i=0; i<mc_size; i++)   {
      if((mc_cut[i][2]==0)  &&
           (mc_cut[i][1]==2))  {
                for(k=0; k<mc_size; k++)  {
                     if((mc_cut[k][2]>0)  &&
                          (mc_cut[k][1]==2  &&
```

```
                    (mc_cut[k][6]==mc_cut[i][6])) {
            fprintf(f3,"\n%0.0f+t%0.0f=%0.4f", mc_cut[i][0],mc_cut[k][0],mc_cut[k][3]);
            } /* endif */
        } /* for k loop */
    } /* endif */
} /* for i loop */

/* ------- dummy cut equation --------*/

for(i=0; i<mc_size; i++)  {
    if((mc_cut[i][1]==0)  {
        for(k=0; k<mc_size; k++)  {
            if((mc_cut[k][0]==mc_cut[i][0]-1)  &&  (mc_cut[k][2]==0))   {
            printf(f3,"\n2%0.0f-t&0.0f=0",mc_cut[i][0],mc_cut[k][0]);
            k=mc_size;
          } /* endif */
        } /* for k loop */
    } /* endif */
} /* for i loop */

/* --------- bp_dim tolerance chain LP model ---------*/

z=1;
for(i=0; i<mc_size; i++)  { /* print bp result */
    fprintf(f3,"\nz%d",z);
    fprintf(f7,"%0.0f %0.4f %0.4f", bp_dim[i][0], bp_dim[i][[3], bp_dim[i][4]);
    for(s=0; s<num; s++)  {
        if((bp_dim_link[s][0]==bp_dim_link[s][0])  &&
          (bp_dim_link[s][1]!=0))  {
          fprintf(f3, "+t%0.0f", bp_dim_link[s][1]);
          for(k=0; k<mc_size, k++)  {
              if(bp_dim_link[s][1]==mc_cut[k][0])  {
               fprint(f7, "%0.0f  %0.4f %0.4f", mc_cut[k][0], mc_cut[k][5],mc_cut[k][6]);
              } /* if */
          } /* for k */
    } /* if */
} /* for s loop */
fprintf(f7,"\n");
fprintf(f3, "=%0.4f", bp_dim[i][2]);
z=z+1;
} /* for i loop */

/* -----------Machine PC Constraint -----------*/

for(i=0; i<mc_size; i++)  {
    found=0;
```

```
    for(k=0; k<num; k++) {
          if((mc_cut[i][0]==sr_cut_link[k][1]) &&
               (sr_cut_link[k][0]!=sr_cut_link[k][1])) {
                    found=1;k=num;
           } /*end if */
} /* for k */
for(k=0; k<num; k++)  {
          if(mc_cut[i][0]==bp_dim_link)  {
                    found=1;k=num1;
          } /* end if */
} /* for k */
if (found==0)  {
                    fprintf(f3,"nt%0.0f>=%0.4f",mc_cut[i][0],mc_cut[i][3]);
          }
          else  {
                    fprintf(f3,"\nt%0.0f>=%0.4f",mc_cut[i][0],mc_cut[i][4]);
          } /* end if */
} /* for i loop */
fprintf(f3,"\nend");
fprintf(f3,"\nog");
fprintf(f3,"\ny");
fprintf(f3,"\nsdbc opt_dim.dat");
fprintf(f3,"\nquit\n");
fclose(f3);
fclose(f6);
fclose(f7);
f5=fopen("opt_dim.dat","r");
f4=fopen(argv[5],"w");
system("lindo<input.jnk>output.jnk");
i=1;

/* ----- write tol_ans.dat for AutoCAD -------*/

fscanf(f5,"    %s %s %s %s" data2,data3,data4,data5);
while (!feaoff(f5))  {
        fscanf(f5, "%s %s %s %s %s", data1,data2,data3,data4,data5);
        i=i+1;
        if (i>x+1)  {
             tol=atof(data2);
             fprint(f4, "%s  %0.4f\n", data1,tol);
             } /* endif */
{ /* while eof f5 */
fclose (f5);
fclose (f4);
{ /* main */
```

APPENDIX B-2

MANUFACTURING KNOWLEDGE

MANUFACTURING PROCESS KNOWLEDGE

```
machine_db(2,"lathe_2",[0.05,0.025,350,750],0.05,[1,2,3,4,5],1)
machine_db(3,"cyl_grind_1",[0.02,0.01,250,600],0.1,[6,7],2)
machine_db(5,"ream_mc",[0.05,0.01,100,100],0.5,[9],4)
machine_db(6,"boring_mc",[0.05,0.025,300,300],0.7,[10],5)
machine_db(7,"mill_mc",[0.02,0.01,500,1000],0.5,[12],6)
machine_db(8,"broach_mc",[0.05,0.01,300,300],0.6,[13],7)
machine_db(4,"drill_mc",[0.25,0.05,100,100],0.5,[11,8],3)
machine_db(9,"shape_mc",[0.125,0.025,1000,1000],0.2,[14],8)
machine_db(10,"EDM_mc",[0.1,0.02,100,100],0.5,[19],10)
machine_db(11,"Press_mc",[0.04,0.01,100,100],0.4,[18],11)
machine_db(13,"foundry",[0.8,0.2,1000,1000],0.3,[16],13)
machine_db(12,"foundry_1",[2,0.5,1000,1000],0.1,[15],12)
machine_db(14,"die_cast_mc",[0.5,0.1,1000,1000],0.3,[17],14)
machine_db(15,"forging_mc",[1,0.2,1000,1000],0.3,[20],9)
machine_db(16,"labour",[0.01,0.005,1000,1000],0.5,[21],15)
machine_db(17,"labour",[0.01,0.004,1000,1000],0.6,[22],16)
machine_db(18,"lathe3",[0.001,0.001,350,750],0.01,[1,2,3,4,5],1)
machine_db(1,"lathe_1",[0.05,0.025,350,750],0.01,[1,2,3,4,5],1)
tool_db(1,"HSS_turn_tool","tool1",1,1,1,[1600,2.5,10],0.5,0.01)
tool_db(2,"HSS_cut_off_tool","tool2",1,2,1,[1600,2.5,10],0.5,0.01)
tool_db(3,"HSS_0.5r_nose_turn","tool3",1,3,1,[1600,2.5,10],0.5,0.01)
tool_db(4,"HSS_1.0r_nose_turn","tool4",1,4,1,[1600,2.5,10],0.5,0.01)
tool_db(5,"HSS_1.5r_nose_turn","tool5",1,5,1,[1600,2.5,10],0.5,0.01)
tool_db(6,"0r_nose_g_wheel","tool6",2,1,2,[2000,0.5,5],0.05,0.03)
tool_db(7,"1.0r_nose_g_wheel","tool7",2,4,2,[2000,0.5,5],0.05,0.03)
tool_db(9,"HSS_reamer","ream",4,6,1,[100,0.5,10],0.5,0.5)
tool_db(10,"HSS_boring_tool","bore",5,6,1,[100,0.5,10],0.5,0.5)
tool_db(11,"HSS_center_drill","c_drill",3,8,1,[100,0.5,10],0.5,0.5)
tool_db(12,"HSS_mill_cutter","mill_20",6,7,1,[200,0.5,10],0.5,0.5)
tool_db(13,"HSS_broaching_tool","broach",7,9,1,[100,0.5,10],0.5,0.5)
tool_db(14,"HSS_shaping_tool","shape",8,10,1,[1000,1,10],0.5,0.1)
tool_db(15,"sand_mould","sand",12,11,4,[0,0,0],0,0.2)
tool_db(16,"shell_mould","shell",13,11,7,[0,0,0],0,0.3)
tool_db(17,"die_mould","die",14,11,6,[0,0,0],0,0.1)
tool_db(18,"press_tool_mould","press",11,13,6,[0,0,0],0,0.3)
tool_db(19,"EDM_core","edm",10,12,5,[0,0.5,0.5],0,0.4)
tool_db(21,"honing_stone","honing",15,14,9,[1000,0.3,0.1],0,0.5)
```

```
tool_db(22,"lapping_tool","lap",16,15,8,[1000,0.3,0.1],0,0.6)
tool_db(20,"forging_die","forg",9,16,6,[0,0,10],0,0.3)
tool_db(8,"HSS_twist_drill","drill",3,6,1,[200,0.5,10],0.5,0.1)
process_db(1,1,1,1,[9,12,0.025,0.05],[40,0.2,0.5],1)
process_db(2,1,1,1,[5,9,0.025,0.05],[40,0.1,0.5],1)
process_db(3,1,1,1,[3,6,0.025,0.05],[40,0.1,0.4],1)
process_db(5,1,1,1,[9,12,0.025,0.05],[20,0.1,0],0)
process_db(6,1,1,1,[5,9,0.025,0.05],[20,0.1,0],0)
process_db(8,2,1,2,[2,5,0.01,0.02],[40,0.05,0],0)
process_db(9,3,1,1,[5,10,0.05,0.25],[100,0.5,0],0)
process_db(10,6,1,1,[5,12,0.01,0.02],[100,0.5,0],0)
process_db(11,7,1,1,[5,9,0.01,0.05],[0,0,0],0)
process_db(12,5,1,1,[5,11,0.025,0.05],[100,0.5,0],0)
process_db(13,4,1,1,[5,8,0.01,0.05],[100,0.5,0],0)
process_db(14,1,2,1,[9,12,0.025,0.05],[40,0.2,0.5],1)
process_db(15,1,2,1,[5,9,0.025,0.05],[40,0.1,0.5],1)
process_db(16,1,2,1,[3,6,0.025,0.05],[40,0.1,0.4],1)
process_db(17,1,2,3,[2,5,0.01,0.02],[40,0.1,0.3],1)
process_db(18,1,2,1,[9,12,0.025,0.05],[20,0.1,0],0)
process_db(19,1,2,1,[5,9,0.025,0.05],[20,0.1,0],0)
process_db(20,1,2,3,[3,6,0.025,0.05],[20,0.1,0],0)
process_db(21,2,2,2,[2,5,0.01,0.02],[40,0.05,0],0)
process_db(22,3,2,1,[5,10,0.05,0.25],[100,0.5,0],0)
process_db(23,6,2,1,[5,12,0.01,0.02],[100,0.5,0],0)
process_db(24,7,2,1,[5,9,0.01,0.05],[0,0,0],0)
process_db(25,5,2,1,[5,11,0.025,0.05],[100,0.5,0],0)
process_db(26,4,2,1,[5,8,0.01,0.05],[100,0.5,0],0)
process_db(29,5,1,1,[5,11,0.025,0.05],[100,0.5,0.3],1)
process_db(30,5,2,1,[5,11,0.025,0.05],[100,0.5,0.3],1)
process_db(31,8,1,1,[6,12,0.025,0.125],[100,0.5,0],0)
process_db(33,12,2,4,[9,12,0.5,2],[0,0,0],0)
process_db(35,14,2,6,[5,8,0.1,0.5],[0,0,0],0)
process_db(34,13,2,7,[5,9,0.2,0.8],[0,0,0],0)
process_db(32,10,1,5,[7,9,0.02,0.1],[0,0.1,0],0)
process_db(37,9,2,6,[7,11,0.2,1],[0,0,0],0)
process_db(38,9,2,6,[5,9,0.01,0.05],[0,0,1],1)
process_db(36,15,2,1,[2,7,0.005,0.01],[0,0,0],0)
process_db(39,16,1,8,[1,6,0.004,0.01],[0,0,0],0)
process_db(40,8,1,2,[5,12,0.01,0.02],[100,0.5,1],1)
process_db(41,8,2,1,[5,9,0.01,0.02],[100,0.5,1],1)
process_db(27,3,1,1,[5,10,0.05,0.25],[100,0.5,0.5],1)
process_db(28,3,2,1,[5,10,0.05,0.25],[150,0.5,0.5],1)
process_db(43,1,1,1,[3,6,0.01,0.05],[40,0.1,0],0)
process_db(4,1,1,1,[2,5,0.001,0.02],[40,0.1,0.3],1)
process_db(42,1,1,1,[3,6,0.002,0.01],[40,0.1,0],0)
process_db(7,1,1,1,[3,6,0.003,0.05],[20,0.1,0],0)
```

```
process(1,"turning",1)
process(2,"cylindrical_grinding",1)
process(3,"drilling",2)
process(4,"reaming",2)
process(5,"boring",2)
process(6,"milling",2)
process(7,"broaching",2)
process(8,"shaping",2)
process(10,"EDM",2)
process(11,"punch_&_die",2)
process(12,"sand_casting",2)
process(13,"shell_casting",2)
process(14,"die_casting",2)
process(9,"forging",2)
process(15,"honing",2)
process(16,"lapping",2)
cut_by(1,1,[30,15,0,0],[3,6,5,6],1)
cut_by(2,3,[20,5,0,0],[3,6,5,6],2)
cut_by(3,6,[30,15,0.5,0],[3,6,5,6],3)
cut_by(4,6,[30,15,1,0],[3,6,5,6],4)
cut_by(5,6,[30,15,1.5,0],[3,6,5,6],5)
cut_by(6,4,[30,15,0,0],[3,6,5,6],1)
cut_by(7,5,[30,15,0,0],[6,6,5,6],1)
cut_by(8,2,[30,15,0,0],[6,6,5,6],1)
cut_by(9,7,[0,0,0,0],[6,6,6,6],8)
cut_by(10,8,[0,0,0,0],[6,6,6,6],6)
cut_by(12,10,[0,0,0,0],[6,6,6,6],7)
cut_by(13,11,[0,0,0,0],[6,6,6,6],9)
cut_by(14,12,[0,0,0,0],[6,6,6,6],7)
cut_by(11,9,[0,0,0,0],[6,6,6,6],7)
cut_by(15,10,[0,0,1.5,0],[6,6,5,6],10)
cut_by(17,9,[0,0,0,0],[6,6,6,6],10)
cut_by(18,10,[0,0,0,0],[6,6,6,6],12)
cut_by(19,8,[0,0,0,0],[6,6,6,6],12)
cut_by(16,13,[0,0,0,0],[6,6,6,6],11)
cut_by(20,8,[0,0,0,0],[6,6,6,6],14)
cut_by(21,9,[0,0,0,0],[6,6,6,6],14)
cut_by(22,10,[0,0,0,0],[6,6,6,6],14)
cut_by(23,11,[0,0,0,0],[6,6,6,6],14)
cut_by(24,12,[0,0,0,0],[6,6,6,6],14)
cut_by(25,1,[0,0,0,0],[6,6,6,6],14)
cut_by(26,2,[0,0,0,0],[6,6,6,6],14)
cut_by(27,3,[0,0,0,0],[6,6,6,6],14)
cut_by(28,13,[0,0,0,0],[6,6,6,6],14)
cut_by(29,4,[0,0,0,0],[6,6,6,6],14)
cut_by(30,8,[0,0,0,0],[6,6,6,6],15)
```

```
cut_by(32,14,[0,0,0,0],[6,6,6,6],16)
cut_by(31,15,[0,0,0,0],[6,6,6,6],12)
tool_feature(1,"R0_nose_cutter",1)
tool_feature(2,"cut-off_tool",1)
tool_feature(3,"R0.5_nose_cutter",1)
tool_feature(4,"R1.0_nose_cutter",1)
tool_feature(5,"R1.5_nose_cutter",1)
tool_feature(6,"hole_cutting_tool",2)
tool_feature(7,"mill_cutter",2)
tool_feature(8,"center_hole_cut_tool",2)
tool_feature(9,"keyway_shaping_tool",2)
tool_feature(10,"shaping_tool",2)
tool_feature(11,"moulding",2)
tool_feature(12,"EDM_core",2)
tool_feature(13,"punch_&_die",2)
tool_feature(14,"honing_tool",2)
tool_feature(15,"lapping_tool",2)
tool_feature(16,"forging_die",2)
manu_feature(1,"External_square_shoulder",1)
manu_feature(2,"External_taper",1)
manu_feature(3,"LHS_external_square_shoulder_end",1)
manu_feature(4,"External_face_end",1)
manu_feature(5,"LHS_external_face_end",1)
manu_feature(6,"External_fillet_shoulder",1)
manu_feature(7,"center_hole",2)
manu_feature(8,"hole",2)
manu_feature(9,"side",2)
manu_feature(10,"fillet_pocket",2)
manu_feature(11,"internal_keyway",2)
manu_feature(12,"external_keyway",2)
manu_feature(13,"complex_casting_object",2)
manu_feature(14,"forging_object",2)
manu_feature(15,"shape_corner_pocket",2)
feature_class(1,"Rotational_&_symmetric")
feature_class(2,"Non_symmetric_3D_object")
```

MATERIAL KNOWLEDGE

```
tool_matl(1,"High-speed-steel")
tool_matl(2,"abrasive")
tool_matl(3,"diamond")
tool_matl(4,"green_sand")
tool_matl(5,"brass")
tool_matl(6,"mould_steel")
tool_matl(7,"shell_cast_matl")
tool_matl(8,"lapping_stone")
tool_matl(9,"honing_stone")
```

```
material(2,"aluminum")
material(3,"brass")
material(1,"mild_steel")
```

APPENDIX B-3

FEATURE RECOGNITION

FEATURE RECOGNITION DXF

```
(defun c:prepdxf(/ la cl pt1 pt2 ss dwg file1 file2 cmline1)
   (setvar "cmdecho" 0)
   (setvar "highlight" 1)
   (setq la (getvar "clayer"))
   (command "layer" "make" "0"
                "off" "dim"
                "off" "pplan"
                "off" "rough_cut"
                "off" "final_cut"
                "off" "material" "")
   (setq cl  (entsel "\nSelect the Centre Line: "))
   (prompt "\nSelect the lower-half boundary of the part: ")
   (setq ss  (ssget)
       dwg (getvar "DWGNAME")
       file1 (strcat dwg ".dxf")
       file2 (strcat dwg ".kb1")
       cmline1 (strcat "preproc " file1 " 1 " file2)
   );setq
   (command "dxfout" "" "e" cl ss "" ""
         "shell" cmline1
   );command
   (command "layer" "make" la
                "on" "dim"
                "on" "pplan"
                "on" "rough_cut"
                "on" "final_cut"
                "on" "material" "")
   (prompt "\n DXF preprocessing is OK!")
   (princ)
);defun
(princ)
```

FEATURE RECOGNITION RULES

```
feature_rule_seq([1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25]).
feature_rule(1,"Face","External",1,"line","-",[-1,1000,-1,1000,-1,1,89,91,-0.1,0.1])
```

```
feature_rule(2,"Chamfer","External",1,"line","Right",[0,1000,0,1000,0,5,44,46,-0.1,0.1])
feature_rule(3,"Chamfer","External",1,"line","Left",[0,1000,0,1000,0,5,314,316,-0.1,0.1])
feature_rule(4,"Tapper","External",1,"line","Right",[0,1000,0,1000,0,1000,5,45,-0.1,0.1])
feature_rule(5,"Tapper","External",1,"line","Left",[0,1000,0,1000,0,1000,315,355,-0.1,0.1])
feature_rule(6,"Diameter","External",1,"line","-",[0,1000,0,1000,0,1000,-1,1,-0.1,0.1])
feature_rule(7,"Face","Internal",1,"line","-",[-1,1000,-1,1000,-1,1,89,91,-0.1,0.1])
feature_rule(8,"Chamfer","Internal",1,"line","Right",[0,1000,0,1000,0,5,44,46,-0.1,0.1])
feature_rule(9,"Chamfer","Internal",1,"line","Left",[0,1000,0,1000,0,5,314,316,-0.1,0])
feature_rule(10,"Tapper","Internal",1,"line","Right",[0,1000,0,1000,0,1000,5,45,-0.1,0.1])
feature_rule(11,"Tapper","Internal",1,"line","Left",[0,1000,0,1000,0,1000,315,355,-0.1,0.1])
feature_rule(12,"Diameter","Internal",1,"line","-",[0,1000,0,1000,0,1000,-1,1,-0.1,0.1])
feature_rule(13,"Thread","Thread",1,"line","-",[0,1000,0,1000,0,1000,-1,1,-0.1,0.1])
feature_rule(14,"Fillet","External",1,"arc","Right",[0,1000,0,1000,0,6,44,46,0.1,5])
feature_rule(15,"Fillet","External",1,"arc","Left",[0,1000,0,1000,0,6,314,316,0.1,5])
feature_rule(16,"Round","External",1,"arc","Right",[0,1000,0,1000,0,6,314,316,0.1,6])
feature_rule(17,"Round","External",1,"arc","Left",[0,1000,0,1000,0,6,44,46,0.1,6])
feature_rule(18,"Neck","External",1,"arc","-",[0,1000,0,1000,0,21,-1,1,20,100])
feature_rule(19,"R_groove","External",1,"arc","-",[0,1000,0,1000,0,6,-1,1,0.1,5])
feature_rule(20,"Round","Internal",1,"arc","Left",[0,1000,0,1000,0,6,44,46,0.1,5])
feature_rule(21,"Round","Internal",1,"arc","Right",[0,1000,0,1000,0,6,314,316,0.1,5])
feature_rule(22,"Fillet","Internal",1,"arc","Left",[0,1000,0,1000,0,6,314,316,0.1,6])
feature_rule(23,"Fillet","Internal",1,"arc","Right",[0,1000,0,1000,0,6,44,46,0.1,6])
feature_rule(24,"Neck","Internal",1,"arc","-",[0,1000,0,1000,0,21,-1,1,20,100])
feature_rule(25,"R_groove","Internal",1,"arc","-",[0,1000,0,1000,0,6,-1,1,0.1,5])
dxf_line_type("External",1)
dxf_line_type("Internal",3)
dxf_line_type("Thread",5)
dxf_line_type("Centre",2)
dxf_line_type("Divide",4)
dxf_entire_name("line",1)
dxf_entire_name("arc",2)
dxf_entire_name("circle",3)
```