THE UNIVERSITY OF HULL

Knowledge-based Expert Support in an Assembly-oriented CAD Environment

being a Thesis submitted for the Degree of PhD

in the University of Hull

by

Hong Mei B.Eng., M.Eng.

June 2000

ABSTRACT

Assembly-oriented design offers great potential for product rationalisation, increasing productivity and reducing lead time and cost. It results in simpler and more reliable products which are less expensive to assemble and manufacture. To facilitate assembly-oriented design, an assembly-oriented CAD environment is needed to incorporate Design for Assembly (DFA) evaluation from an early design stage. Assembly planning should also be integrated to support the DFA evaluation.

This thesis reports the results of research towards supporting such an assembly-oriented CAD environment. A novel approach has been used to deploy an *Expert Assembler* to support proactive DFA evaluation and assembly sequence definition. This is particularly useful, as designers are rarely if ever assembly experts. Based on the fact that there are several areas needing expert support in this assembly-oriented CAD environment, but that different areas have very different requirements and different knowledge is involved, the *Expert Assembler* deployed contains several separated modules. Each module is an expert agent devised to tackle a problem area that uses a suitable problem solving strategy, knowledge representation and reasoning method. This brings a number of advantages that are detailed in the thesis.

The thesis presents systematical ideas for support proactive DFA, with the focus on support for part count reduction and assembly sequence generation. This is realised by three elements of the expert agents: *Part Count Advisor*, *Starting Part Advisor*, and *Next Part Advisor*. Part count reduction is usually based on dialogue with the user. There is little computational support for this issue in any of the DFA methodologies and related literature. This research fills the gap: it brings computational support for part count reduction from the early design stage. The work has also made new progress in assembly sequence generation. The *Starting Part Advisor* and the *Next Part Advisor* cooperate with each other and with the user to provide suggestions dynamically and transparently regarding base part and the most suitable next part selection in assembly sequence definition. Case studies were used to test the effectiveness of the *Advisors*.

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Professor K.G. Swift for his guidance and support during the course of the research. Without these, I cannot imagine how I could have achieved what I have.

I would like to express my sincere gratitude to Professor C.A. Brookes for his support on my application of the University Open Scholarship and research bursaries. This brings me the opportunity to carry out my PhD research.

Special thanks should also go to Dr. P.A. Robinson who has provided the answers to many of my technical and computing questions and given me useful suggestions on the thesis writing. Discussing with him is really helpful.

I extend my thanks to my colleagues of the Ophir Project: C.J. Barnes, G.F. Dalgleigh, S.J. Tate who formed a friendly, helpful working environment; and to the industrial collaborators: G. Hird who provided some of the case studies, H. Merryweather who provided the Radan CAD/CAM software.

Lastly, but not the least, I express my special gratitude to my parents for their care and support during the several years of my PhD research.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Knowledge-based Expert Support in an Assembly-oriented CAD Environment

Chapter 1 Introduction

1.1 Obstacles to Assembly-Oriented Design

Today the global market for engineering products is changing rapidly. Many companies have been placed in the difficult situation of having to develop products in a short period of time because of shortening product life cycles and reducing costs. This brings the challenge to design quality into the product and at the same time to reduce product cost and lead time.

An important and often neglected area in product development and manufacturing process is assembly. The assembly operation is often responsible for over 40% of the total manufacturing cost [1], 30% to 40% of the total product cost [2, 3, 4], and 40-60% of total production time [5]. It is often a labour intensive and costly process. In the automotive industry, considered by many to be highly automated, approximately one third of the total workforce is engaged in assembly [4, 6]. To increase efficiency and reduce cost, manufacturing strategies have been directed towards automation since the 1970s, but the target has not been achieved in many product assemblies, most of them are still manually assembled. Even for some automatically assembled products, some assembly operations have to be carried out manually. A survey of 355 companies in the Federal Republic of Germany shows that the most important obstacle against automation in the field of assembly is that product designs are generally not 'assembly-oriented' [7].

Literature review and industrial case studies on products across a broad range of industries from automotive to aerospace, to scientific equipment demonstrate that existing products are being created with at least 50% excess of components and many expensive assembly and manufacturing processes [8, 9]. One of the case studies

"Grating Arm Assembly" (see Figure 1.1) provided by CSC Computer Sciences Ltd (CSC for short), an industrial collaborator of the Ophir Project[1], originally had 39 parts and 59 assembly operations. After redesign, the part count has been reduced to 8, and the number of assembly operations has been reduced to 9 (Table 1). This indicates that the original product design is not assembly-orientated, with too many parts and has to use complex assembly processes.

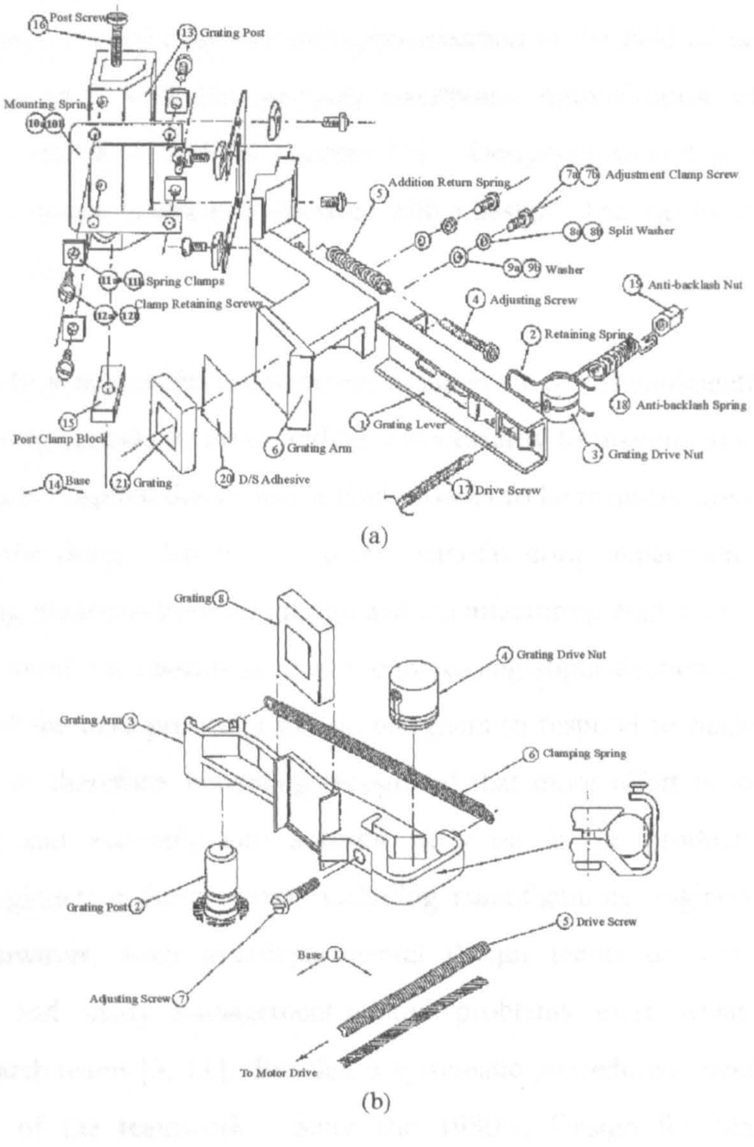


Figure 1.1 Grating Arm Assembly: (a) Original Design, (b) Redesign

[1] Ophir Project is a cooperative project between the University of Hull and Cranfield University. It is funded by EPSRC and supported by three industrial collaborators: Rover Group Ltd., CSC Computer Sciences Ltd., and Radan Computational Ltd..

Table 1: A Comparison of the Original Design and the Redesign

Grating Arm Assembly	**Part Count**	**Assembly Operations**
Original Design	39	59
Redesign	8	9

As much of the production process is implicitly fixed in product design, the designer plays a key role in the development and rationalisation of the field of assembly. If the designs are not good from the assembly standpoint, rationalisation of the chain of assembly will only be of limited success [5]. Designers should give attention to possible manufacturing problems associated with a design. This has been advocated for many years [5, 10].

Traditionally, the idea was that a competent designer should be sufficiently familiar with the manufacturing process to avoid adding unnecessarily to manufacturing costs as the translation of a conceptual design into a final product to be manufactured is sequentially passing from the design department to the manufacturing department with possible time-consuming iterations between design and manufacturing engineers. However, this has been discredited for reasons such as the increasing sophistication of manufacturing techniques, and the time pressures put on designers to respond to market needs more efficiently. It is, therefore, becoming recognised that more effort is required to take manufacturing and assembly into account early on in the product design cycle. Concurrent engineering design teams including manufacturing engineers are a better solution. However, such interdepartmental design teams do not always work harmoniously and many management-related problems exist when building and coordinating such teams [3, 11]. Besides, a systematic procedure is needed to enhance the efficiency of the teamwork. Since the 1980's, Design for Manufacture and Assembly (DFMA) analysis tools [9] have been developed to provide a systematic procedure for analysing a proposed design from assemblability and manufacturability point of view. This reduces the barriers between design and manufacturing and results in simpler and more reliable products that are less expensive to assemble and manufacture. DFMA has been proved to bring manufacturing savings in many companies [8, 10]. It has been found by the user of design for assembly techniques that, typically, twenty to thirty percent of assembly costs can be eliminated when the improved design is compared with a 'traditional design' [12]. There is also often a saving of ten to fifteen percent on manufacturing costs as a result of design for assembly [12].

Three of the well known DFA methods are Boothroyd-Dewhurst DFMA method (BDI method), the Hitachi Assemblability Evaluation Method (AEM), and the CSC Design for Assembly/Manufacturing Analysis technique (CSC technique) – the former Lucas method. They provide a quantitative and systematic procedure to evaluate a product design from the assemblability point of view.

Although DFA procedures are well documented and mechanistic, at present, it is usually carried out as a separate analysis task on a design that is essentially complete. Besides, it requires a large number of judgements to be made by the user, some of which depend upon the component design being substantially complete. Although some of the tedious data manipulation and calculation has been eliminated, and friendly user interfaces have been established, even the advanced DFA software toolkits, DFMA® developed by Boothroyd Dewhurst Inc (BDI for short), and TeamSET™ developed by CSC, require a large number of subjective user inputs to determine the assembly-related part attributes. Another problem is that the importance of generating an appropriate assembly sequence within the DFA analysis has been overlooked [13, 14]. Little if any construction assistance with assembly sequence declaration is available in any of the DFA methodologies, despite the implications for the results if an inappropriate assembly sequence is used. Automatic analysis of assemblability during design especially from the early design stage is still a problem containing many challenging research issues, such as:

1. Product modelling from abstract to detail: to represent assembly information from an early stage in the design process;
2. Assembly planning parallel with the product design: assembly sequence and assembly operations, such as the type of handling/feeding, gripping, and insertion, can drastically influence assemblability [15];

3. Process knowledge representation and application in design environment;
4. Data interrogation and reasoning support.

However, assemblability evaluation in the production stage or towards the end of the design is not required by industries [16], as it is often too late or too expensive to make substantial design changes. Besides, post-analysis tools requiring tedious user input are often viewed as additional burdens and disliked by designers. Minimising user interaction makes DFA evaluation more efficient. This can reduce the time and effort required of the designer, freeing him/her for more productive tasks. Thus assemblability evaluation software should be developed as a design support tool rather than as a reactive tool, to guide designer creating an assembly-oriented design concept. This requires developing a computer-aided design environment incorporating proactive DFA tools and facilitating assembly process planning parallel with the product design process. Such an assembly-oriented design environment should also incorporate assembly-related knowledge and expertise, suitable assembly modelling and reasoning techniques to enhance the efficiency of DFA evaluation by automatic inferring the relevant DFA and sequence data.

1.2 Possible Solutions

The product design process can no longer be viewed as linear, where a design is passed sequentially from the design department to the manufacture department, it should be concurrent: simultaneous consideration of manufacture and assembly-related issues, from conceptual to detail design. To guide a designer in creating an assembly-oriented product, through the entire design process, DFA approaches should take effect proactively rather than reactively. This requires:

- Integrating proactive DFA evaluation and suggestive tools into the design environment;
- Generating an assembly plan concurrently with the product design to facilitate proactive DFA evaluation;
- Knowledge incorporating and application in the design environment, product modelling and reasoning support.

Based on these requirements, an assembly-oriented CAD prototype has been developed out of the Ophir Project to facilitate assembly-oriented design. This prototype environment incorporates proactive DFA evaluation tools, facilitates product/assembly structure [2] definition and assembly sequence construction parallel with the product design process. The architecture of the system implements the data structure of the Four Layer Product Model, which comprises component and assembly models, component interaction data and assembly plans [17].

As assembly-oriented design aimed to bring all the assembly-related issues into the design process, it provides the best opportunities to tackle potential assembly problems as early as possible from the product design stage. However the descriptions of the physical components during the early phase of the product design are often vague and imprecise, and knowledge of all the design requirements and constraints is frequently incomplete, approximate or unknown. Such high-level descriptions of product and imprecise information make quantitative reasoning difficult, if not impossible. As a result, computer-aided tools for conceptual design remain sparse and few. Because a poorly conceived design concept can never be compensated for by a good detailed design, DFA analysis should start at the conceptual design stage. As a result, qualitative DFA conceptual selection methods based on criteria of DFA principles may be realistic and suitable [18]. As the design process goes along, a more quantitative view on the assemblability of the product, in relation to the amount of information available, is more useful. Furthermore, the assembly-oriented CAD environment should not only identify potential assembly problems as early as possible, it should also give suitable advice about remedies. This may require many aspects of knowledge and experience accumulated from human design and manufacturing activities. However, how can the knowledge and experience be effectively represented in an assembly-oriented CAD environment? How can a true intelligent computer-aided design environment be created to highlight problems instantly and generate suitable advice for each particular situation based on the knowledge and experience? These are still

[2] product/assembly structure -- a hierarchy structure to present components and subassemblies in relation to the overall assembly which is referred as product structure in terms of DFA evaluation. It is an assembly structure in the sense that it represents subassembly partitions.

challenging research issues, and they are the subjects of the thesis.

1.3 Research Objectives and Strategies

1.3.1 Research Objectives

This research is concerned with investigating how to incorporate manufacturing and assembly-related knowledge and expertise into the assembly-oriented CAD environment outlined above, and how to apply the knowledge and suitable reasoning techniques to support proactive DFA, and product structure and assembly sequence construction concurrent with product design. The objectives can be summarised as following:

- Identification of the development needs for assembly oriented design
- Selection of suitable approach and integration of appropriate tools for knowledge representation and reasoning support
- Creation of assembly-related expert agents for an assembly-oriented design environment
- Application of the expert agents devised in related areas

These will be achieved through several steps shown in the research strategies.

1.3.2 Research Strategies

The research strategies are:

- Reviewing literature and analysing industrial case studies to identify the need for proactive DFA and concurrent assembly sequence generation. This can consequently validate the Ophir assembly-oriented CAD environment
- Recognising the trends of DFA evaluation and assembly planning from literature review and the results of knowledge engineering with experts in industry. This will help answer the questions of how to support for proactive DFA and concurrent generation of assembly sequence
- Identifying which areas in the assembly oriented design environment need knowledge-based expert support
- Devising appropriate strategies of expert support so that suitable methods and tools can be selected

- Choosing knowledge representation and reasoning methods based on the requirements of support areas and related knowledge resources
- Creation of the knowledge-based expert agents for assembly oriented design and application of the agents devised in related areas
- Finally, using case studies to test the effectiveness of expert support

1.4 Thesis Structure

In the introduction, the need for assembly-oriented design has been specified, and an assembly-oriented CAD environment which can facilitate proactive DFA evaluation has been proposed. After the introduction, **Chapter 2** reviews Design for Assembly methodologies: the well known methods, the latest developments, and needs for further improvement. **Chapter 3** presents a literature review of assembly planning with emphasis on assembly sequence generation. Based on the knowledge engineering results, the gap between industrial requirements and academic research is identified. An assembly planning system which best suits industrial requirements and early implementation of DFA is specified. **Chapter 4** describes the assembly-oriented CAD environment developed by the Ophir Project in which product structure and assembly sequence are constructed concurrently with product design, proactive DFA analysis provides a quantitative view of the assemblability. Knowledge-based and geometric reasoning techniques are required to provide necessary support. It also describes how the knowledge-based expert support is fitted in the environment. **Chapter 5** identifies the main areas and an appropriate strategy for the expert support. It further reviews several AI (Artificial Intelligence) techniques which are useful for knowledge representation and reasoning support. Suitable tools and methods are selected. Expert agents are created and deployed. Implementation issues are discussed. **Chapter 6** to **Chapter 7** reports the detailed work of the expert support in related areas: proactive DFA evaluation and assembly sequence generation. **Chapter 8** presents applications. Case studies are used to test the effectiveness of the expert support. **Chapter 9** discusses the contributions of the work, presents conclusions and outlines areas where further research could be usefully directed.

Chapter 2 Design for Assembly Methodologies

This chapter begins with a review of the topic and goes on to present methods in DFA, and then summarises the latest developments in this domain.

2.1 Introduction

Although as early as the 1960's several companies were developing guidelines for use during the product design process based on the gathered manufacturing data, such as the "Manufacturing Producibility Handbook" published for internal use by General Electric in the USA, the emphasis was on the design of individual parts for "producibility", and little attention was given to the assembly process. In the 1970s, DFMA became a topic when manufacturing strategies were being directed towards automation, however significant benefits from the use of Design for Assembly (DFA) methods were not realised until systematic analysis tools were made available in the late 1970's. Most of the early work in the analysis of assemblability was based on the classification and coding of design attributes in relation to handling and insertion operation. The design attributes of the components, the relation between components and the assembly operations were used to estimate the ease or difficulty of the assembly of components. It was a breakthrough to find these assemblability methods at that time. However, as it is found that the sequence of assembling components has a strong influence on assemblability, more plan-based evaluation systems have been developed recently [15, 19, 20, 21].

The Boothroyd and Dewhurst DFA method (BDI method) [22] was developed in the late 1970's, which grew out of the collaborative research on design for automatic feeding and automatic insertion carried out at the University of Massachusetts, USA, and, the University of Salford Industrial Centre, UK. It was first introduced in handbook form in 1980, and a UK version of the handbook containing similar method was published in 1981 by the University of Salford Industrial Centre [23]. The BDI method is aimed at minimising assembly times and cost by reducing the number of individual parts and optimising the design of the parts for easy handling and insertion.

The pioneering work of Boothroyd has resulted in several automated assembly evaluation and advisory systems [24, 15]. One of the earliest efforts of advisory systems is made by Jakiela and Papalambros [24] which integrated a rule-based system with a commercial CAD environment. This integrated environment restrains the product design to be created using the predefined features. When new features are added to the design, the system makes use of production rules to evaluate the design and offer suggestions for improvement. As the design progresses, the suggestive mode of the system works incrementally, offering advice at every design step. Hence, the design improvement suggestions are strongly influenced by the sequence in which the designer enters various features. Even though the integrated rule-based system is suitable for encoding knowledge in design environment, it is more difficult to use than a stand alone rule-based system. Li and Hwang [1] proposed and partially implemented a framework for automatic DFA evaluation which closely follows the Boothroyd-Dewhurst methodology. The importance of assembly sequence generation and feature recognition in automatic DFA evaluation is specified. The fundamental concept of the framework was in linking DFA evaluation technique with a CAD system. AI, especially expert system approach, is proposed as a feasible way of extracting some non-geometric and operational assembly features and in generating redesign suggestions. The assemblability analysis module and cost estimation module are a direct computer implementation of the BDI method. Limited feature recognition for assembly is performed, while the other information that will affect the evaluation is obtained from the user. The final result is a table which is roughly the same as a manual assembly worksheet. The task of automated redesign is presented as a future goal. Myers et al [25] has presented an approach aimed at the automatic evaluation of the manual handling time for parts in a mechanical assembly. Their software makes use of manual handling data of BDI method, and extracts component-level of data, such as size, section thickness and symmetry, from a CAD database. Because the scope of their research was limited to evaluation of manual handling time, it was restricted to consider the influence of assemblability on component-level factors. The influences from the interactions between components and the assembly process used are not considered. Sturge et al. have developed a semi-automated assembly evaluation system [26] that attempts to overcome some of the limitations of the scheme proposed by Boothroyd and Dewhurst. They differentiated the factors affecting assemblability as *component-level* factors, *system-level* factors and *process-level* factors. They recognised the great influence of the assembly sequence selected on product assemblability. They also highlighted the needs to develop product models that extend beyond current geometric-modelling packages, to represent component interactions, and assembly process information so as to facilitate automatic assembly sequence generation, and automatic assemblability evaluation.

Hsu et al. [15] developed an approach to design-for-assembly that examines and evaluates assembly plans using three criteria: parallelism, assemblability, and redundancy. They evaluate the plan to find the problems with the assembly. When possible, a better assembly plan is created by modifying the existing plan. If a better plan is found, the design is modified by splitting, combining or perturbing various components. Although limited in certain ways, this offers a new plan based approach.

Besides the BDI method, the Hitachi Assemblability Evaluation Method (AEM) [27, 28] has also served as a basis for development of automated assemblability system. This methodology is based on the principle of one motion per part; a symbol mark and a penalty score based on the operation difficulty are assigned for each type of assembly operation. Finally, the method computes an assembly evaluation score and assembly-cost ratio. The methodology is common for manual, automatic and robotic systems.

Another method, the Lucas method (CSC Design for Assembly/Manufacturing Analysis technique) [29, 30, 31] was developed from the collaborative work between the Lucas Engineering & Systems and the University of Hull. In this method, parts are divided into two groups based on functional importance: "category A" parts which are carrying functions vital to the performance of the product, such as drive shafts, insulators etc, and "category B" parts whose purpose is not critical to product function, such as fasteners, spacers etc. The goal is to eliminate as many type B parts as possible through redesign. Analyses of manual handling or automatic feeding, and fitting are carried out on the parts. An assembly sequence flowchart is used to perform fitting analysis. The CSC technique is incorporated into the commercial software

TeamSET™.

Warnecke and Bäβler [7] studied both functional and assembly characteristics. In their method, which they name Assembly-Oriented Product Design, both the assembly difficulty and the functional value are evaluated and a combined rating is given. Parts with low functional value but high assembly difficulty receive low scores, while parts with high functionality and low assembly cost receive high scores. The scoring is used to guide the redesign process. The authors also suggest DFA evaluation should be a part of design process and claim that setting up assembly structure and determination of assembly sequence are necessary steps of DFA evaluation.

Sony Corporation claims to have developed a unique set of rules for increased productivity in the 1980's, involving design for assembly cost effectiveness (DAC) [8]. It reiterates that design for ease of assembly should be conducted from the conceptual design stage and before the detailed design. The improvement of a design at its inception is referred to as the concept of feed forward design.

Kroll et al [32] also emphasise that design for assembly should be considered as early as possible in the design process. They specified that a qualitative approach to guide design for ease of assembly, such as using general rules and guidelines, is too general to be practically applied during design; but that a quantitative approach of DFA requires very specific information which may not be available at the time of analysis. Besides the quantitative approach may involve the time-consuming process of completing standard worksheets. They proposed a knowledge-based expert system approach to implement DFA.

Hsu et al [33] proposed to bring the design for assembly analysis into the conceptual design stage to achieve the best savings by selecting a combination of design concepts such that they can achieve the stated function at the minimum cost for assembly. The problem of selecting the right combination of design concepts is reduced to a well-known 'set covering problem'.

Angermüller and Moritzen [21] suggest bringing assembly planning (not necessarily fully detailed) into the design environment as generally not only product structure and component features, but also assembly process, such as feeding, handling operations influence DFA evaluation. They specified that knowledge-based expert system could be used to support the assembly planning and evaluation in design environment. They concluded that the evaluation of assembly process and assembly sequence could provide feedback information to improve product design.

Furthermore, there are many researchers [24, 1, 25, 26, 21] who believe DFA should be incorporated into CAD systems to facilitate the evaluation of assemblability during the design stage although current CAD systems themselves need to be improved to give more support in design synthesis. Rosario and Knight have studied the problems of extracting feature information from a CAD database and then using this information in DFA analysis [34]. Eversheim and Baumann [35] explained that DFA method has the essential advantage to determine the handling, joining and geometry feature independently of one another. They further specified that DFA should be linked into a CAD system to extract related properties automatically and reduce subjective user input. They developed a design system that integrates functions for assembly-oriented product design, such as the assembly-specific evaluation function, using a commercially available 3D geometry modeller and relational database. Molly, Yang [36] realised one of the weaknesses of the DFA systems is that they do not analyse the design directly, but rely on the designer to correctly reply to the questions concerning the design and its components. They discovered the trends in research on DFM (including DFA) are towards the development of knowledge-based DFM expert systems using object-oriented programming methods, and the interfacing of CAD systems with computer aided assembly planning (CAAP) system using either feature extraction or feature-based design. A methodology and prototype architecture for an integrated CAAP and DFA system based on the use of feature-based design has also been described. The CAAP system accesses the feature-based CAD system using a neutral interface provided by Pro/Engineer. The disassembly sequence generated and the information on the mating features derived from CAD system are used for DFA analysis. Another computer based package on design for assembly [37] has been developed. It is used interactively by the designer at the design stage based on a set of design rules and design guidelines developed for automated and robotic assembly. It is predicated that a CAD system can be developed to incorporate some of the rules and guidelines. Recently Jared et al [17] presented an enhanced product model which comprises component and assembly models, component interaction data and assembly plans, and a DFA system that performs geometric reasoning based on the model. In this way, the DFA system relies less on user input. In this paper, based on the knowledge intensive nature of DFA and the need to update the analysis to take account of technological development, it predicts that it is likely that more future implementation of DFA will take place using knowledge-based expert system techniques.

A summary of the literature review: Design for Assembly has been proposed as a systematic approach to improve product assemblability for more than 20 years. Several formal methods were developed, among which Boothroyd-Dewhurst DFA method, Hitachi Assemblability Evaluation Method and CSC Design for Assembly/Manufacturing Analysis technique (CSC technique) are well known. A number of assembly evaluation and advisory systems have been proposed based on the BDI method. Rule-based DFA advisory systems have been integrated into CAD environment to give redesign suggestion on DFA. The resulted systems did give useful suggestions for redesign, but there are limitations and usage problems. However, many researchers believed that DFA should be incorporated into CAD systems to facilitate the automatic evaluation of assemblability from the early design stage and that the current CAD systems themselves need to be improved to give more support in design synthesis. It is claimed that DFA should be a part of the design process, and it should be considered as early as possible in the conceptual design stage. It is further claimed that functional information should be considered during DFA evaluation. The importance of assembly sequence generation and feature recognition in automatic DFA evaluation is specified. As not only product structure and component features, but also assembly process (sequence and operations) influence assemblability, a suggestion is made to bring assembly planning (not necessary to be fully detailed) into the design environment to facilitate DFA evaluation during design stage. Enhanced product models have been proposed to extend beyond current geometric modelling package to represent component interactions and assembly process information to facilitate the generation of assembly sequence to aid automatic DFA evaluation.

2.2 Methods in DFA and Success Stories

Besides Design for Assembly principles, rules, axioms [38], and some *qualitative* analysis methods, such as Value Analysis [39], Product Design Merit [40], there are three well-known and successful quantitative assemblability evaluation methods. They are the BDI method, the AEM, and the CSC technique.

2.2.1 The BDI Method

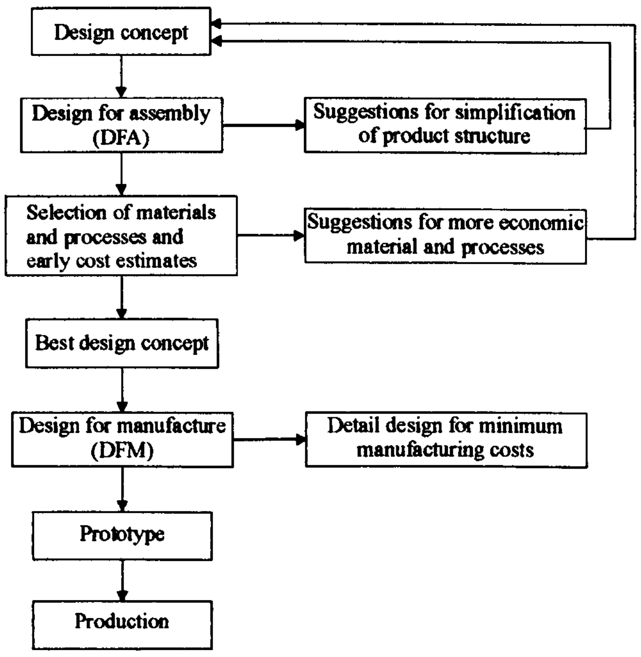

Design concept
Design for assembly (DFA)
Suggestions for simplification of product structure
Selection of materials and processes and early cost estimates
Suggestions for more economic material and processes
Best design concept
Design for manufacture (DFM)
Detail design for minimum manufacturing costs
Prototype
Production

Figure 2.1 Boothroyd-Dewhurst DFMA Methodology

The BDI DFMA method [41] (see Figure 2.1) emphasises that DFA, which helps simplify the product structure through part count reduction, should come at the conceptual design stage as it considers a product as a whole. Next, a means of quantifying the product's assembly time and cost should be provided. Each component in the design is rated for difficulty of assembly on how it is to be moved, grasped, and orientated for insertion, and how it is inserted and/or fastened into the product. After this, selection of materials and processes and early cost estimates should be conducted to make the necessary trade-off decisions between parts consolidation and increased material/manufacturing cost.

The BDI method draws sharp distinction between manual, robotic and automatic assembly (Figure 2.2). In the case of a manual assembly, the assessment is based on the estimation of manual assembly cost using time data, which corresponds to a particular component design classification and operator wage rate. In robotic and automatic assembly, the assessment is determined by the relative cost of the equipment required to process the most simple or ideal design.

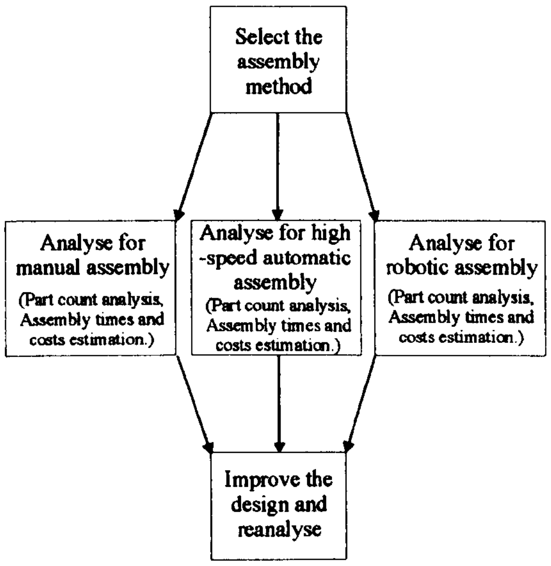

Select the assembly method
Analyse for manual assembly
(Part count analysis, Assembly times and costs estimation.)
Analyse for high -speed automatic assembly
(Part count analysis, Assembly times and costs estimation.)
Analyse for robotic assembly
(Part count analysis, Assembly times and costs estimation.)
Improve the design and reanalyse

Figure 2.2 The Boothroyd-Dewhurst DFA Method

The BDI DFA analysis uses a worksheet. An example of a design for manual assembly worksheet is in Figure 2.3. The procedure of manual assembly evaluation involves identifying a two-digit handling code (column (3)) and insertion code (column (5)) by answering questions about potential handling difficulties or insertion restrictions. From these two digit codes, handling and insertion time (column (4) and column (6)) can be found by referring to a chart of synthetic data. The total operation time in seconds for each part in column (7) is calculated by adding the handling and insertion times in column (4) and (6) and multiplying this sum by the number of repeated operations in column (2); that is (7) = (2) x [(4) + (6)].

The sum of column (7) gives the total estimated manual assembly time TM.

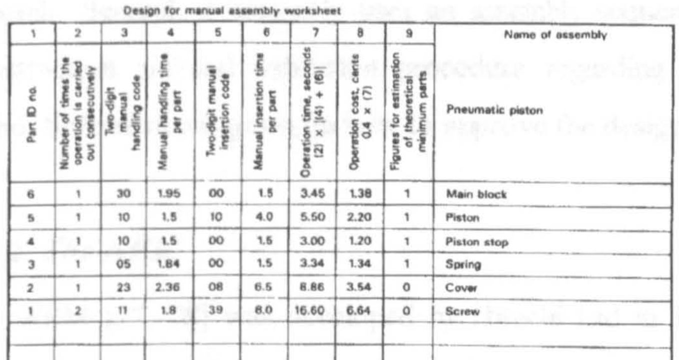
$$TM = \Sigma\,(7)$$

The sum of column (8) gives the total manual assembly cost CM.

$$CM = \Sigma\,(8)$$

The "Design Efficiency" is defined as the ideal assembling time (3NM) divided by the estimated assembling time, where NM represents the theoretical minimum number of parts in column (9), and the number 3 expresses the assumption that an ideal component takes 1.5 second to handle and 1.5 second to insert.

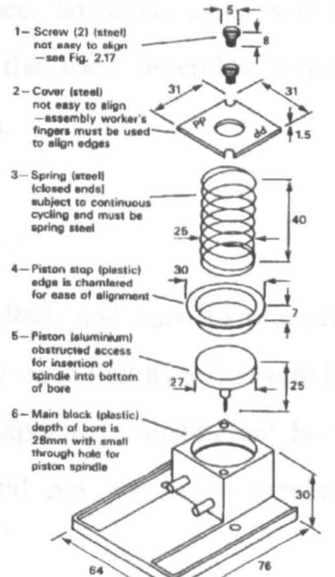
Manual assembly design efficiency EM = 3NM/TM



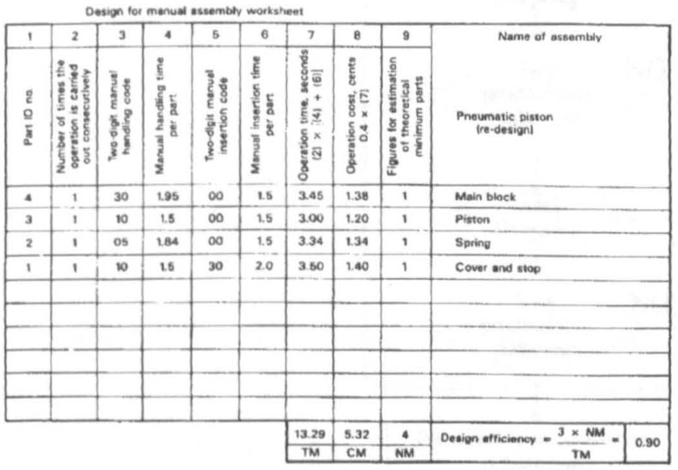
Design for manual assembly worksheet

1	2	3	4	5	6	7	8	9	Name of assembly
Part ID no.	Number of times the operation is carried out consecutively	Two-digit manual handling code	Manual handling time per part	Two-digit manual insertion code	Manual insertion time per part	Operation time, seconds (2) × [(4) + (6)]	Operation cost, cents 0.4 × (7)	Figures for estimation of theoretical minimum parts	Pneumatic piston
6	1	30	1.95	00	1.5	3.45	1.38	1	Main block
5	1	10	1.5	10	4.0	5.50	2.20	1	Piston
4	1	10	1.5	00	1.5	3.00	1.20	1	Piston stop
3	1	05	1.84	00	1.5	3.34	1.34	1	Spring
2	1	23	2.36	08	6.5	8.86	3.54	0	Cover
1	2	11	1.8	39	8.0	16.60	6.64	0	Screw
							40.75 TM	16.30 CM	4 NM

Design efficiency = $\frac{3 \times NM}{TM}$ = 0.29


1– Screw (2) (steel) not easy to align –see Fig. 2.17
2– Cover (steel) not easy to align –assembly worker's fingers must be used to align edges
3– Spring (steel) (closed ends) subject to continuous cycling and must be spring steel
4– Piston stop (plastic) edge is chamfered for ease of alignment
5– Piston (aluminium) obstructed access for insertion of spindle into bottom of bore
6– Main block (plastic) depth of bore is 28mm with small through hole for piston spindle
5
8
31
31
1.5
25
40
30
7
27
25
30
64
76

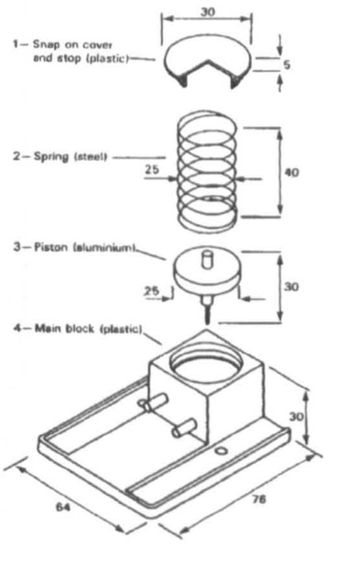
(a)



Design for manual assembly worksheet

1	2	3	4	5	6	7	8	9	Name of assembly
Part ID no.	Number of times the operation is carried out consecutively	Two-digit manual handling code	Manual handling time per part	Two-digit manual insertion code	Manual insertion time per part	Operation time, seconds (2) × [(4) + (6)]	Operation cost, cents 0.4 × (7)	Figures for estimation of theoretical minimum parts	Pneumatic piston (re-design)
4	1	30	1.95	00	1.5	3.45	1.38	1	Main block
3	1	10	1.5	00	1.5	3.00	1.20	1	Piston
2	1	05	1.84	00	1.5	3.34	1.34	1	Spring
1	1	10	1.5	30	2.0	3.50	1.40	1	Cover and stop
							13.29 TM	5.32 CM	4 NM

Design efficiency = $\frac{3 \times NM}{TM}$ = 0.90



(b)

Figure 2.3 An Example of BDI Design for Manual Assembly: (a) Original design, (b) Redesign

The BDI method is documented in a handbook [41], and implemented as a commercial software package [42, 122]. The handbook took form in 1980. The software was first introduced in 1982.

The BDI method is widely recognised. Much of the subsequent work on DFA is based on this approach. However, it has some disadvantages. First, subjective user input is needed. Second, it implicitly uses an assembly sequence, however, there is not any construction aid and validation procedure regarding the used assembly sequence. Third, little advice is given on how to improve the design.

2.2.2 The AEM

The AEM [27, 28] was developed by Hitachi Ltd in 1980, and aims to provide the product designer with some early feedback on the ease or difficulty of assembling a proposed design (see Figure 2.4). It does not make explicit the distinction between manual and automated assembling. This limitation did not appear to weaken its attraction for a number of companies in Japan and USA.

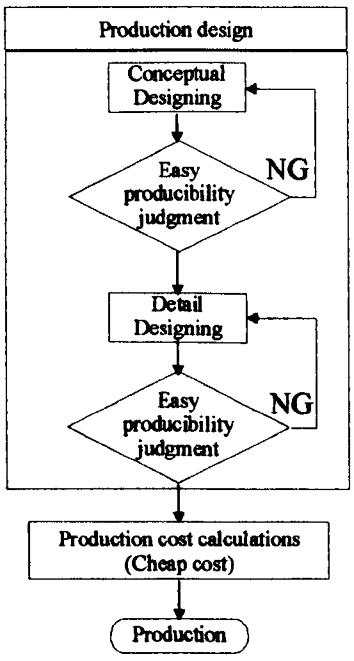

Production design
Conceptual Designing
Easy producibility judgment
NG
Detail Designing
Easy producibility judgment
NG
Production cost calculations (Cheap cost)
Production

Figure 2.4 The Hitachi Approach for Designing Products with Good Producibility

The AEM assesses the assemblability of the proposed design or design concept at the earliest possible stage by making use of two indices: E and K.

E -- the assembly evaluation score used to assess design quality or difficulty of assembly operations.

K -- the estimated assembly cost ratio: an indication of the assembly cost improvements.

Assembly operations are categorised into approximately 20 elemental operations, and each of them is assigned a symbol mark (referred to as an AEM symbol) which clearly indicates the content of the operation. The easiest operation, downward motion for insertion of a part, is chosen as the "idea reference" and given a penalty score of 0. For each of the other more complicated operations, such as operation x, a penalty score of ε_x that depends upon the difficulty of the operation (proportional to the operation cost C_x or operation time T_x) is assigned. That is:

$$\varepsilon_x = f_1(C_x) = f_2(T_x)$$

Example of the elemental operations and the penalty scores are show in Figure 2.5.

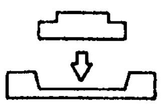
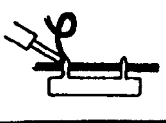
Elemental operation		AEM symbol	Penalty score
	Downward movement	$\downarrow$	0
	Soldering	S	20

Figure 2.5 Examples of AEM Symbols and Penalty Scores

Besides the operation elements, factors which also influence the difficulty of the entire assembling operations are extracted as coefficients, e.g. η: influence of a succession of elemental operations for a part. So the penalty score for a part can be expressed as a function $g(\varepsilon_{ij}, \eta_{ij} \ldots)$ in which "i" represents part "i", and "j" represents operation "j". After completing a worksheet in the same order as the anticipated assembly sequence, the penalty score for each part is used to calculate the assemblability evaluation score for each part, as in Figure 2.6. The higher the penalty score, the lower the E score. The highest E score is 100 with a penalty score of 0. The calculation formula is:

$$E_i = f_3(C_i) = 100 - g(\varepsilon_{ij}, \eta_{ij} \ldots)$$

where $g(\varepsilon_{ij}, \eta_{ij} \ldots)$ is a function which increases when the sum of the elemental operation costs for the part "i" increases.

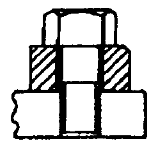
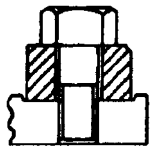
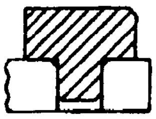
Product structure and assembly operations		*Ei*: Part assemblability evaluation score	*E*: Assembly evaluation score	*K*: Assembly cost ratio	Part to be improved
	1. Set chassis	100	73	1	block
	2. Bring down block and hold it to maintain its orientation	50			
	3. Fasten screw	65			
	1. Set chassis	100	88	Approx. 0.8	screw
	2. Bring down block (orientation is maintained by spot-facing.)	100			
	3. Fasten screw	65			
	1. Set chassis	100	89	Approx. 0.5	block
	2. Bring down and pressfit block	80			

Figure 2.6 Hitachi Assemblability Evaluation and Improvement Examples

The assemblability evaluation score E for a product decreases when its assembly operation cost C increases. The calculation formula is:

$$E = f_4(C) = f_4[\sum_{i=1}^{N}\{f_3^{-1}(Ei)\}] = f_5(E_{i,}, N)$$

where $C = \sum_{i=1}^{N} Ci$

N: Number of parts

$E_{i,}$: Assembly Evaluation Score for part "i"

f_3^{-1} :Inverse function of "f_3"; when $E_i = f_3(C_i)$, $C_i = f_3^{-1}(E_i)$.

If E > 80, it normally indicates that the product can be assembled automatically. The E score is employed to simplify the various operations and not explicitly related to the reduction of part count.

The formula to calculate the Assembly Cost Ratio K is:

$$K=\frac{C}{Cs}=\frac{\sum_{i=1}^{N} Ci}{\sum_{i=1}^{N} Csi}=\frac{\sum_{i=1} f_3^{-1}(Ei)}{\sum_{i=1}^{N} f_3^{-1}(Esi)}=\frac{f_6(\mathrm{Ei, N})}{f_6(\mathrm{Esi, Ns})}=\mathrm{f_7(N, Ns, Ei, Esi)}$$

where C: the assembly cost of the evaluated product

Cs: the assembly cost of the standard product

Ci and Csi: assembly cost of part "i" of the evaluated product and the standard product respectively

Ei and Esi: Assembly Evaluation Score of part "i" for the evaluated product and the standard product respectively

N and Ns: Number of parts of the evaluated product and the standard product

The assembly cost can be calculated from the Assemblability Evaluation Scores of all parts.

$$C=\sum_{i=1}^{N} Ci=\sum_{i=1}^{N} f_3^{-1}(Ei)=\mathrm{f_8(N, Ei)}$$

If K = 0.7, it indicates a 30% saving in assembly cost as a result of modifying the design. The K score provides feedback on the advantages to be gained by reducing the number of parts in the assembly as saving in assembly cost can be achieved by reducing part count in a product and/or simplifying the assembly operations.

The AEM evaluation procedure is as Figure 2.7, in which it is desirable that E is over 80 and K is below 0.7. An automated AEM computing system has been developed [27]. Some AEM evaluation and improvement examples are shown in Figure 2.6.

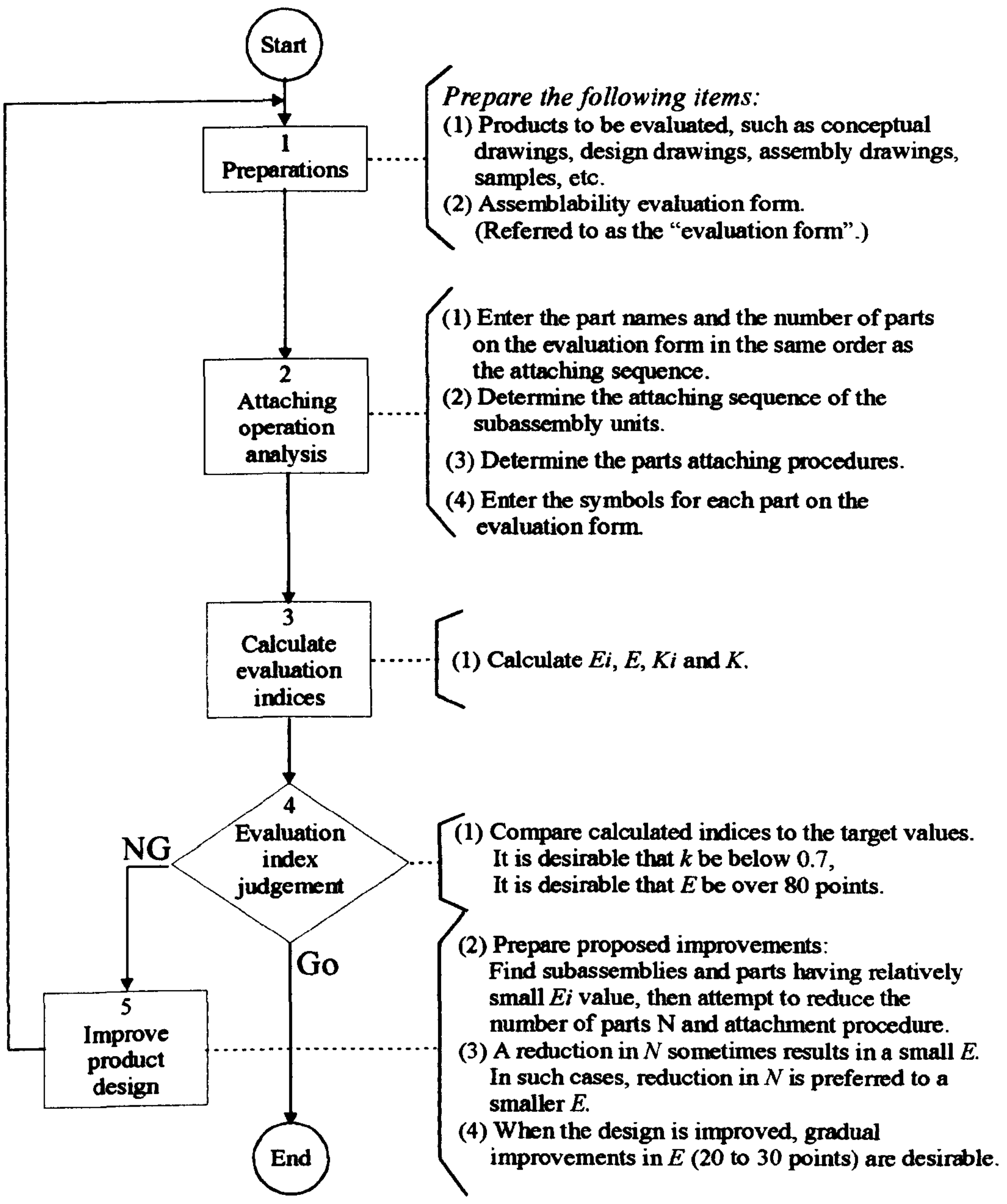

Start
1
Preparations
Prepare the following items:
(1) Products to be evaluated, such as conceptual drawings, design drawings, assembly drawings, samples, etc.
(2) Assemblability evaluation form. (Referred to as the "evaluation form".)
2
Attaching operation analysis
(1) Enter the part names and the number of parts on the evaluation form in the same order as the attaching sequence.
(2) Determine the attaching sequence of the subassembly units.
(3) Determine the parts attaching procedures.
(4) Enter the symbols for each part on the evaluation form.
3
Calculate evaluation indices
(1) Calculate Ei, E, Ki and K.
4
Evaluation index judgement
NG
Go
(1) Compare calculated indices to the target values. It is desirable that k be below 0.7, It is desirable that E be over 80 points.
(2) Prepare proposed improvements: Find subassemblies and parts having relatively small Ei value, then attempt to reduce the number of parts N and attachment procedure.
(3) A reduction in N sometimes results in a small E. In such cases, reduction in N is preferred to a smaller E.
(4) When the design is improved, gradual improvements in E (20 to 30 points) are desirable.
5
Improve product design
End

Figure 2.7 The Hitachi Assemblability Evaluation Procedure

2.2.3 The CSC Technique

The CSC technique (former Lucas method) was developed in the 1980s as a result of collaborative work between the Lucas Engineering & Systems and the University of Hull. Following a period of successful use of the paper-based version, its first commercial computer version was launched in October 1989, and currently forms part of the TeamSET^{TM} commercial software [43]. Unlike the other two methods, the CSC technique is not based on monetary costs, but based on three indices that gives a relative measure of assembling difficulty which makes this method more abstractive and can possibly be applied in an earlier design stage than the other two methods. The CSC technique is carried out in four sequential stages as in Figure 2.8.

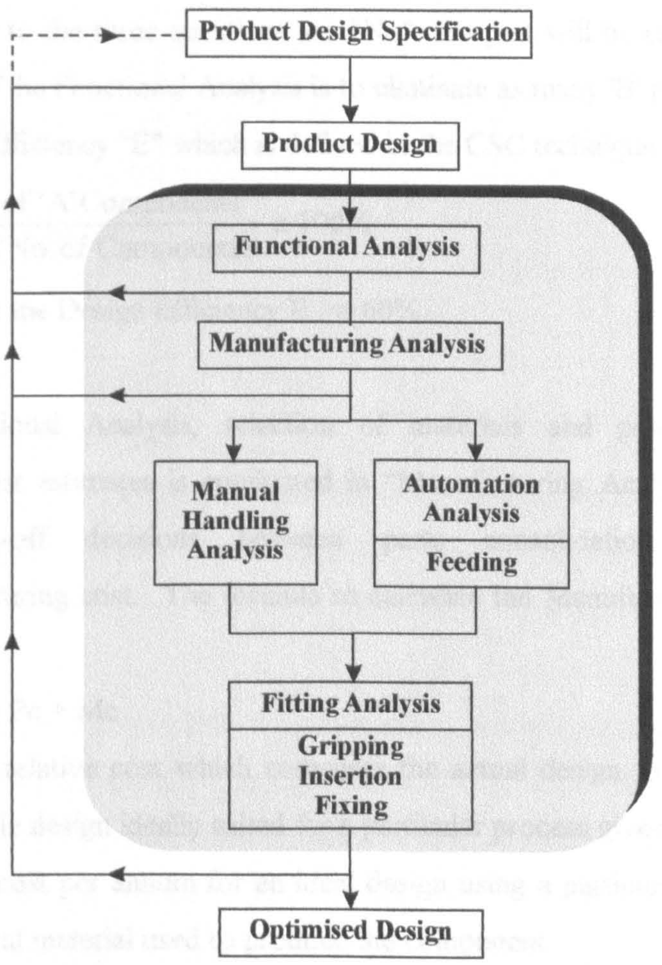


Figure 2.8 The CSC DFA/MA Evaluation Procedure

The "Functional Analysis" is conducted from conceptual design stage to simplify product structure as a whole. Parts are divided into two groups in the Functional Analysis: 'A' parts that perform a primary function and therefore, exist for fundamental reasons; 'B' parts whose purposes are not critical to the product function, which may be eliminated or combined with other parts. The Functional Analysis is mainly based on three pieces of information about parts:

- Does the part being analysed have to move relatively to **<u>all</u>** parts which have already been analysed for the product to function?

- Does the part being analysed have to be made of a different material to __all__ parts already analysed with which there was no relative movement for the product to function?
- Does the part being analysed have to be separated to allow assembling another part or for its inservice adjustment or replacement?

If all the answers to the three questions are "No", the part will be categorised as a 'B' part. The goal of the Functional Analysis is to eliminate as many 'B' parts as possible to increase Design Efficiency "E" which is defined in the CSC technique as:

$$E = \frac{\text{No. of 'A' Components}}{\text{Total No. of Components}} \times 100\%$$

It is desirable that the Design Efficiency E >= 60%.

After the Functional Analysis, selection of materials and processes and early manufacturing cost estimates is conducted in "Manufacturing Analysis" to make the necessary trade-off decisions between parts consolidation and increased material/manufacturing cost. The formula to calculate the Manufacturing Cost Index, Mi is:

$$Mi = Rc \times Pc + Mc$$

Where Rc is the relative cost which compares the actual design to that of the ideal. Therefore, a simple design ideally suited for a particular process gives Rc = 1. Pc is the basic processing cost per annum for an ideal design using a particular process. Mc is the cost of the total material used to produce the component.

Finally, analyses of manual handling or automatic feeding and fitting are carried out on each part. The CSC technique distinguishes between manual and automated assembling but not between various types of automated assembling. Handling analysis is used to assess the preparation (e.g. separation or orientation) of parts for assembly for manual handling or feeding analysis is used if the components are handled by machinery.

$$\text{Manual Handling Ratio} = \frac{\sum_{i=1}^{N} hi}{\text{Number of 'A' Parts}}$$

where N: Total number of parts

hi: Handling Index of part "i" which can be calculated from the Manual

Handling Analysis Tables.

$$\text{Feeding Ratio} = \frac{\sum_{i=1}^{N} \text{fi}}{\text{Number of 'A' Parts}}$$

where N: Total number of parts

fi: Feeding Index of part "i" which can be calculated from the Automatic Feeding Analysis Tables.

The fitting analysis is based on an "assembly sequence flowchart". Of the three well-know DFA methods, only this method has an explicit assembly sequence construction process. The calculation formula is:

$$\text{Fitting Ratio} = \frac{\sum_{i=1}^{N} (\text{Gi} + \text{Fi} + \text{NAi})}{\text{Number of 'A' Parts}}$$

where N: Total number of parts

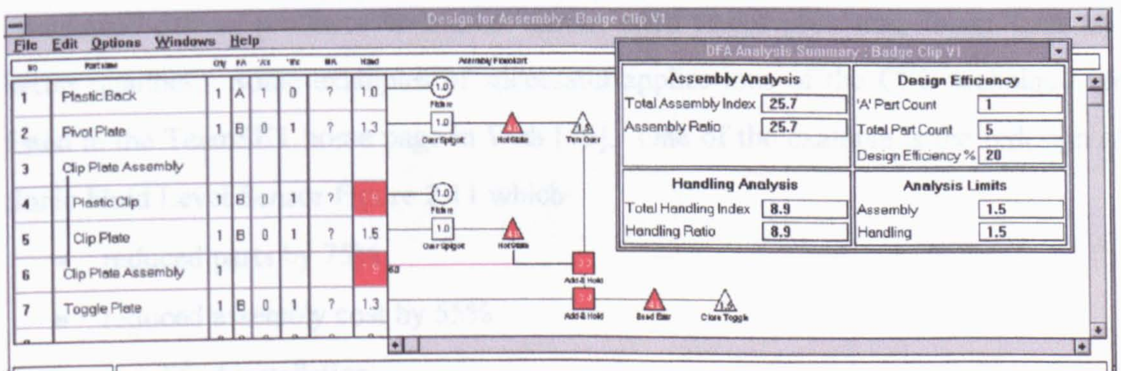
Gi: Gripping Index of part "i"calculated from Gripping Analysis Table;

Fi: Fitting Index of part "i" calculated from Fitting Analysis Table;

NAi: Non-Assembly Index of part "i" calculated from Non Assembly Processes Table.

Each individual index in handling/feeding and fitting analysis should be less than 1.5. Equal or greater than this threshold highlights assembly difficulty for an individual part. The final ratio is used to assess the difficulties of assembly operation on the proposed product. Each final ratio should be less or equal to 2.5. Greater than this threshold highlights assembly difficulty for the product.

An evaluation example using TeamSET^{TM} software is presented in Figure 2.9.



Design for Assembly : Badge Clip V1

No	Part Name	Qty	FA	'A'	'B'	MA	Hand
1	Plastic Back	1	A	1	0	?	1.0
2	Pivot Plate	1	B	0	1	?	1.3
3	Clip Plate Assembly						
4	Plastic Clip	1	B	0	1	?	[illegible]
5	Clip Plate	1	B	0	1	?	1.5
6	Clip Plate Assembly	1					1.9
7	Toggle Plate	1	B	0	1	?	1.3

DFA Analysis Summary : Badge Clip V1

Assembly Analysis

Total Assembly Index	25.7
Assembly Ratio	25.7

Design Efficiency

'A' Part Count	1
Total Part Count	5
Design Efficiency %	20

Handling Analysis

Total Handling Index	8.9
Handling Ratio	8.9

Analysis Limits

Assembly	1.5
Handling	1.5

Figure 2.9 Badge Clip Analysis using the CSC TeamSET^{TM} Software

2.2.4 Success Stories

DFA provides a systematic procedure for analysing proposed designs from the point of view of assembly and manufacture. It results in simpler and more reliable products which are less expensive to assembly and manufacture. It attracts attention on the complete product (or subassembly) as a whole to promote the ideas of part reduction, part standardisation and product modularization. It also highlights problems of assembly operation.

By 1986, more than 1500 engineers at Hitachi had been trained to use AEM, and it was claimed that this method was saving millions of dollars annually [27]. AEM has been widely used by the Hitachi Group and more than 20 well-known companies around the world [28].

There are many successful applications of BDI method in defence, automotive, and electromechanical industries. Some of the detailed case studies and a summary of the results of 43 published case studies from various companies are presented in references [8, 10]. A few more case studies were added to make a summary of 62 applications of BDI [44, page 39].

In previous years, more than 1000 Rover engineers have been trained to use the Lucas method. 40 applications of the Lucas approach of DFA in automotive, aerospace, defence, and scientific equipment industries resulted in an average of 47% assembly cost reduction, 46% part count reduction [29]. For over ten years, the TeamSET™ (software version of the CSC technique) has consistently helped manufacturing companies deliver products to market faster, with fewer problems, lower cost and better qualities. Some examples of successful applications of the CSC technique are listed in the TeamSET home page in Web [45]. One of the example is the redesign of Sonic Fluid Level Sensor Figure 2.11 which

- reduced parts by 75%
- reduced assembly cost by 55%
- simplified installation
- reduced envelope size

- increased the use of standard parts

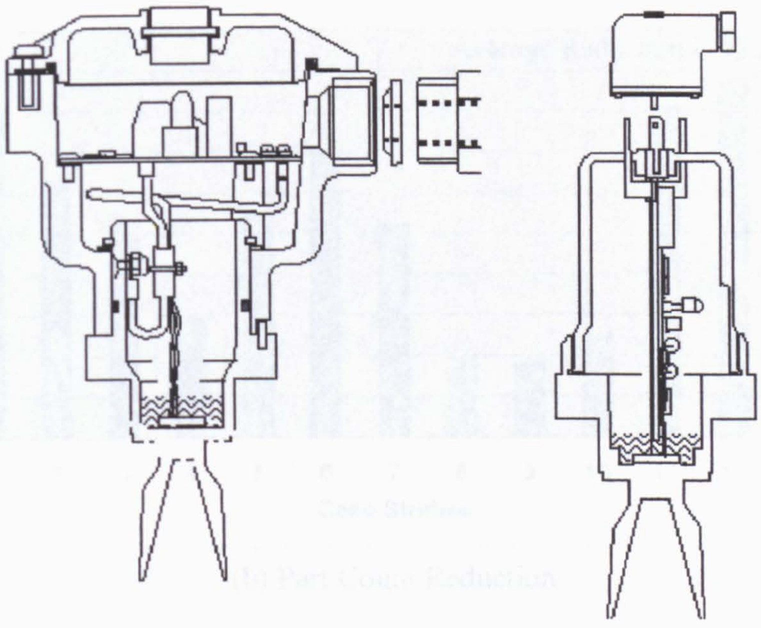
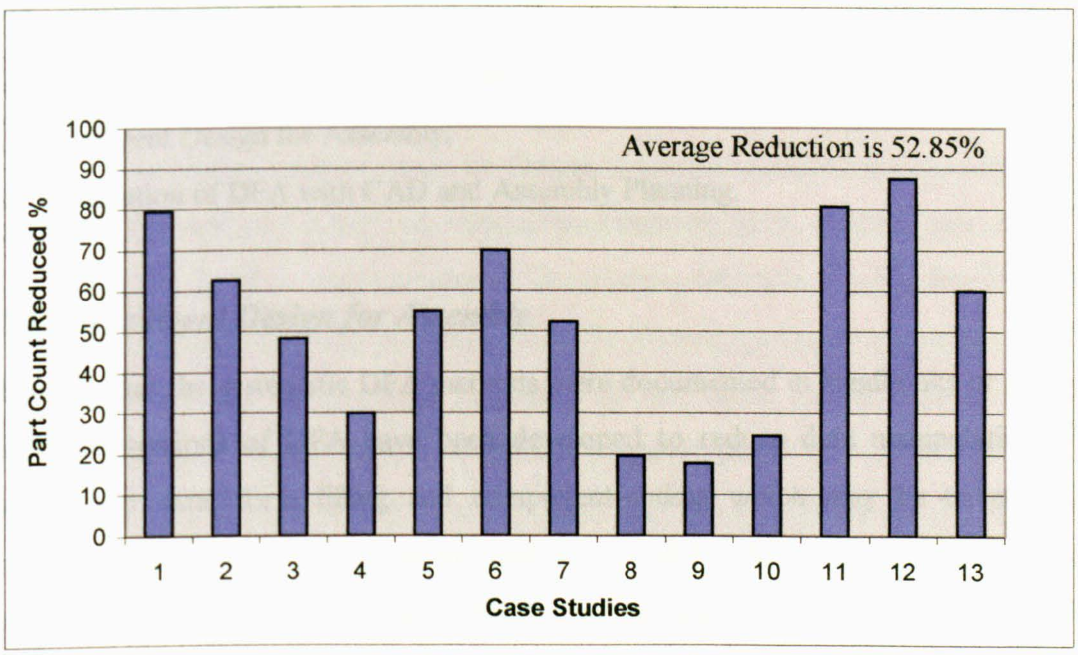


Figure 2.10 Sonic Fluid Level Sensor: (a) before redesign, (b) after redesign

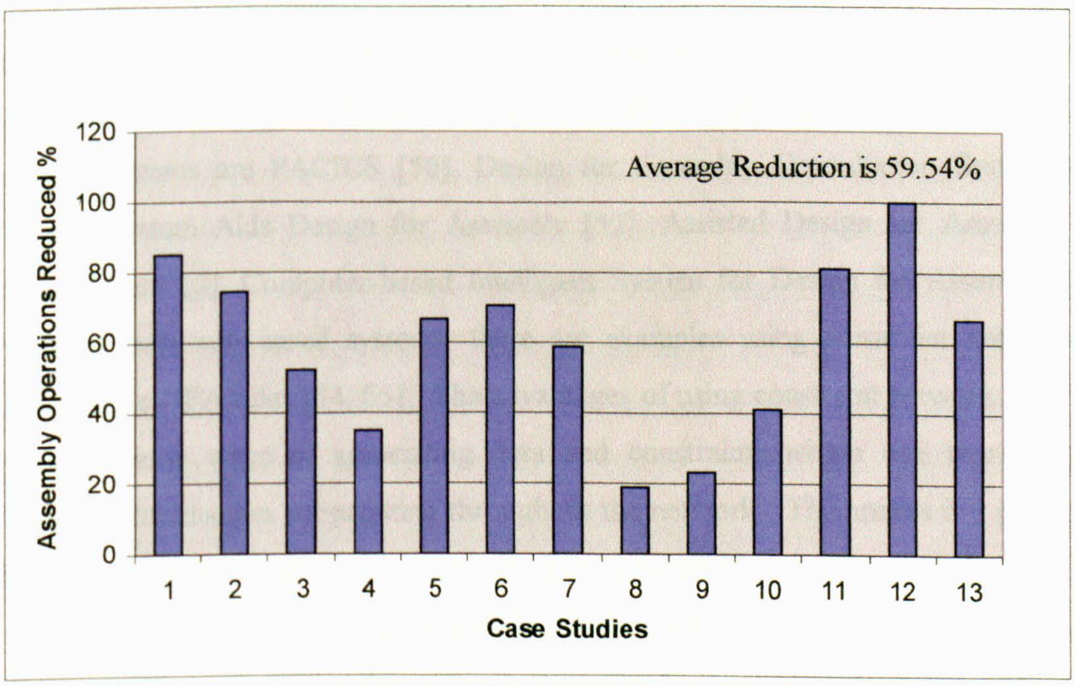
Here, a summary of 13 application case studies are presented in Figure 2.10 to demonstrate the effectiveness of the CSC technique: an average 52.85% part count reduction and 59.54% assembly operation reduction.

DFA Case Study	No Of Parts Original	No Of Parts Redesign	Part Count Reduction (%)	No Of Operations Orinial	No Of Operations Redesign	Operations Reduction (%)
Grating Arm Assembly	39	8	79.49	59	9	84.75
Oil Pump	32	12	62.50	51	13	74.51
Pintle Hook Mounting Assy	56	29	48.21	40	19	52.50
Booster Master Cylinder	80	56	30.00	140	91	35.00
Washer Bottle assembly	31	14	54.84	42	14	66.67
Locker Handle	53	16	69.81	64	19	70.31
Solenoid	42	20	52.38	66	27	59.09
EUI DL	51	41	19.61	84	68	19.05
Pump	181	149	17.68	245	188	23.27
Tandem Booster	78	59	24.36	92	54	41.30
Wiper Motor	31	6	80.65	42	8	80.95
Staple Remover	8	1	87.50	13	0	100.00
Trim Screw	5	2	60.00	9	3	66.67
			52.85			59.54

(a) Case Study Data



(b) Part Count Reduction



(c) Assembly Operation Reduction

Figure 2.11 Case Study Summary - Part Count and Assembly Operation Reduction When CSC Technique Was Used

2.3 State of the Art

In recent years, research on DFA has been focused on the following areas:

1. Intelligent Design for Assembly;
2. Integration of DFA with CAD and Assembly Planning.

2.3.1 Intelligent Design for Assembly

Shortly after the systematic DFA methods were documented in handbooks or manuals, software versions of DFA have been developed to reduce data manipulations, e.g. looking-up data, form filling and component-coding which may be tedious, time consuming and error-prone during analysis [9, 46]. Most of the software uses spreadsheets to quantify the assemblability of products. As the conventional DFA software has the following drawbacks:

- depends heavily on the experience and knowledge of the user,
- is difficult to modify, extend or update,
- has no mechanism to explain the results,

a knowledge-based approach had become the focus and is adopted by some researchers to develop intelligent DFA software [47, 48, 49]. Examples of the developed DFA expert systems are PACIES [50], Design for Assembly Consultation System [51], Expert System Aids Design for Assembly [52], Assisted Design for Assembly and Manufacture [6], Computer-based Intelligent System for Design for Assembly [53]. Besides these rule-based systems, there are examples using constraint networks to incorporate DFA rules [54, 55]. The advantages of using constraint network are that it provides easy ways of associating data and constraints within one representation format, with changes propagating throughout the network. This makes it a potentially powerful tool in modelling the relatively ill-structured DFA problems [56]. Also, one constraint may more easily encapsulate knowledge which would require several rules to express. However, the ability of expert systems to separate knowledge and data is useful when domain experts wish to update models of equipment or DFA knowledge bases without the need to reprogram the constraint network. Within expert systems, the functionality of the system is not so tightly intertwined with the knowledge being applied.

By definition, an expert system should be able to demonstrate a level of expertise. However, DFA expert systems do not always achieve this, giving little more advice than can be deduced from the alternative spreadsheet methods. Only some domain specific DFA expert systems which are limited to a narrow field of engineering, such as in [50], did achieve better or unique performance [57]. They can store and systematically apply a large amount of knowledge, which is in very few hands and normally acquired over long period of time. So knowledge processing is not meant to relieve conventional programming languages of their tasks, but rather to complement and extend them, especially for situations where accurate mathematical models or algorithms can't be established. For problems that can be represented by algorithms, conventional computerised solutions can be used. The advantages of knowledge-based systems lie in the speed of solution, in greater ease of understanding of program structure, and thus in an enhanced comprehensibility as well as simplified operation and maintenance. Most of the DFA systems, spreadsheet style or rule-based, use a three part method comprising collecting data, analysis, and output evaluation results. Most of the DFA expert systems developed require the user to answer numerous questions for elicitation of information required by DFA evaluation, such as the form and functionality of components, and how they interact. This hinders the expert systems' performance. Evidence from tests with both trained and novice DFA users shows that both manual and computer-based methods of analysis (including DFA expert systems) suffer from approximately the same percentage of incorrect data entries [58]. One way of reducing the error, the time and effort required of the designer is to automatically extract information from the design environment i.e. CAD systems. An early work towards the integration of DFA with CAD explored which data could be extracted from a conventional 2 and $2\frac{1}{2}$ dimensional CAD drawings [51].

2.3.2 Integration of DFA with CAD and Assembly Planning

Integrating DFA with CAD environment is also motivated by the following well recognised facts besides the avoidance of tedious and error-prone re-entry of data:

- DFA evaluation should be incorporated into product design process as early as possible;

- DFA evaluation has a great impact on product design.

In an integrated DFA-CAD environment, the product definition in the CAD system can be taken as the input for evaluation, and the improvement can be directly made on the product definition according to the feedback of the evaluation. Thus a proposed design can be evaluated from the earliest possible stage, problems can be highlighted immediately and any changes for improvement can be achieved at a minimum cost.

In terms of application of DFA in early design stage, Timothy et al [18] tried to extend BDI method with abstracted DFA principles to support conceptual design, and Hsu et al [33, 4] tried to select a combination of design concepts from a predefined library to achieve the stated functional requirements at a minimum cost for assembly by calculating DFA index of each design concept. Besides, most research effort is being aimed towards integrating DFA with CAD system. Regarding proposed methods and prototype systems, readers are directed to references [24, 1, 25, 26, 21, 34, 35, 36, 17]. Several systems and methods are discussed below.

An early attempt to integrate DFA with a CAD environment in developing an intelligent CAD system [24] was introduced in section 2.1, in which a production rule program incorporating DFA knowledge was integrated with a commercial CAD system. For this integrated system, effective human-computer interactions are needed to input features for product design and to respond to suggestions of the rule-based system. Ensuring all design constraints, especially functional constraints, is the responsibility of the designer. It was found that design and coding of the rule base was somewhat difficult, but a fairly small number of antecedent and consequent routines are required to address many different problems. The facility to write any antecedent or consequent makes the rule-based approach very flexible. The integrated system is useful for encoding knowledge about design for assembly, but is more difficult to use than a stand alone conventional knowledge-based system.

Molly et al [59] have mentioned that much effort is going into the development of DFM expert systems, which would eventually be fully integrated into an automated design-manufacturing environment. In general these integrated systems should consist of a design tool (CAD), a knowledge-acquisition and storage tool, and an inference tool which applies the DFM knowledge to the design. The main areas of research involved should be knowledge acquisition, product modelling and CAD data exchange, and the application of AI programming methods (particularly the use of object-oriented programming). It specified that if a common application of the integrated system is to be built, it will employ at least some of the following techniques:

- Feature-based design;
- Adapting DFA to different levels of design detail;
- Common data modelling (STEP/EXPRESS);
- AI programming methods, probably using object-oriented methods.

Also an integrated CAD-DFA-CAPP system which uses the above technology was proposed and under development.

In several papers [34, 25, 60] algorithms and computing programs are developed to automatically extract geometric feature information from a CAD system database. Sturges and Kilani [26] further highlighted the need to develop product models that extend beyond current geometric-modelling packages to represent component interactions, and assembly process information to facilitate automatic assembly sequence generation, and automatic assemblability evaluation. Future research is directed to develop algorithms to determine the value of unresolved component-level factors, such as degrees of symmetry, and to investigate qualitative reasoning and other artificial-intelligence techniques for giving recommendation about design. Jared et al [17] presented an enhanced product model which comprises component and assembly models, component interaction data and assembly plans, and a DFA system that performs geometric reasoning based on the product model.

As touched on previously, the result of DFA evaluation is not only influenced by product structure and component features, but also by the assembly operations and tools [21], and the sequence of assembling components has a strong influence on assemblability [13, 14, 61]. In reference [13], an example of an industrial product has been used to specify the influence of assembly sequence on DFA analysis. The influence of assembly sequence and subassembly partitions to complex assemblies is specified in [14]. As a result, more plan-based evaluation systems have been explored in recent years to give re-design suggestions based on the evaluation of an assembly plan [15, 19, 20, 21]. To consider assemblability from early design stage, there are systems proposed to bring assembly planning into the design process, but in reality most of the integrated systems [21, 62, 36, 59] generate an assembly plan towards the end of design process.

There are systems in which assembly planning becomes an essential part of the design exploration process based on replaying existing product designs [63, 20]. In a system [63], design for assembly is more effective and correct as for a particular sequence of assembly, different design alternatives can be created and evaluated. A computer aided tool [62] supports this integrated approach with an internal model of the product, covering both part and operation aspects and two basic views, one tailored to the needs of *a designer*, the other to the needs of *an assembly engineer*. An *Interactive Operation Network Editor* is used to handle the two views of the product. Any manipulation in one view is reflected automatically in the other view. The design of the product and the evaluation of its assemblability are in one view, the assembly techniques and equipment selection, the elaboration of the assembly operations and their precedence relationship are in the other view.

A methodology and prototype architecture [36] for an integrated CAAP and DFA system base on the use of feature based design has been described. The disassembly sequence generated and the information on the mating features derived from CAD system is used for DFA analysis.

Currently the development of integrated DFA-CAD-CAAP systems has become the main focus of research. Such an integrated system should be able to concurrently construct an assembly plan (not necessary to be fully detailed), especially an appropriate assembly sequence needs to be generated in product design process and the system should facilitate DFA evaluation at different levels of abstraction (from concept to detail design) to achieve assembly-oriented design.

2.4 Summary of Findings

The literature review, the current state of DFA in research and applications suggests that DFA software and the basic methodologies need to improve in the following aspects in which the first three are essential:

1. Reduce subjective error prone user input;
2. Take account of the important influence of the assembly process, especially the assembly sequence;
3. Improve DFA to suit the conceptual design stage;
4. Generate suitable redesign suggestions;
5. Connect functional structure with assembly structure and DFA evaluation.

To achieve **the first**, besides suitable geometric data and appropriate assembly knowledge representation methods, automatic reasoning techniques should be applied where possible. This requires:

- Suitable product models to be established to represent not only component geometric information, but also component non-geometric information (e.g. material), component interacting information (e.g. mating information) and assembly plan information (e.g. assembly sequence, assembly operations).
- Related assembly and manufacturing knowledge to be incorporated into DFA evaluation system.
- Suitable algorithms and inference mechanisms to be established to infer the required information automatically from the product model based on the incorporated knowledge.

To achieve **the second**, suitable assembly sequence generation and assembly planning facilities should be provided during DFA analysis. For an early implementation of DFA in the conceptual design stage, DFA evaluation should be incorporated into CAD systems, and concurrent construction of an assembly plan, especially an assembly sequence, with product design is required.

To achieve **the third**, DFA methods should be extended to suit different design stages from concept to detail, from qualitative to quantitative. The related product model should have the ability to represent assembly in abstract without too much reliance on detailed geometry.

To achieve **the fourth**, related expertise should be incorporated into design environment, suitable reasoning techniques should be applied for searching possible solutions to aid decision making.

To achieve **the last**, functional model needs to be constructed, and they must be able to communicate with the assembly model and DFA modules to achieve a compromise between production cost and product performance.

The Ophir Assembly-oriented CAD environment (will be detailed in chapter 4) supported by this research is one of the integrated DFA-CAD-CAAP prototype systems developed to tackle some of the problems mentioned above.

Chapter 3 Assembly Planning

In the last chapter, it was concluded that early implementation of DFA requires an assembly plan to be constructed concurrently with the product design process -- an appropriate assembly sequence needs to be generated from the early design stage. In this chapter we review the current state of computer-aided assembly planning (CAAP), and compare the academic research with industrial assembly planning practice. There are two main objectives. They are:

1. To find out what kind of computer-aided assembly planning system best suits the industrial requirements and the early implementation of DFA evaluation; and
2. To find out what is the best way to apply heuristics [3] to assist computer-aided assembly planning.

3.1 Computer-Aided Assembly Planning Research

A large amount of literature [64, 65] has been published regarding the problem of automatic assembly sequence generation, but no definitive solution has yet been proposed. Some researchers use structured questions to obtain partial precedence relationship from user, and use algorithms to generate all (geometrically) feasible sequences based on the established precedence relationship. They interactively edit the graph representation of assembly sequences generated to prune awkward or unpractical sequences, such as Bourjault, De Fazio and Whitney. Others use a decomposition approach, which assume that the assembly sequence is the reverse of the disassembly sequence, such as Homen de Mello and Sanderson, and use And/Or graph to represent all the geometric and mechanical feasible assembly plans. Generic algorithms such as AO* [4] [66, 67] are used to search for the best plan from the generated plans. These two approaches laid the foundations for the growing field of

[3] heuristic is a "rule of thumb" developed through experience, judgement, intuition, and insight.

[4] AO* is a heuristic search algorithm which provides a way of conducting a best-first search through an "And/Or" graph that is used to represent a problem space. It is a variation of A* algorithm which works on "Or" graphs.

CAAP research. Later on, observing the complexity of automatic assembly plan, many researchers try to narrow down the search space to enhance the efficiency of the planner. Some of them propose to generate good assembly sequences or directly search an optimised sequence instead of enumerating all feasible assembly sequences. Two solutions have mainly been used: constraint-based search and knowledge-based approach. Also connecting the planner with CAD environment is explored to automatically generate geometrically and mechanically feasible plans directly from CAD database [68, 69].

In this section, the state of art of computer-aided assembly planning will be presented, and the trends in its development will be described.

3.1.1 The Liaison Sequence Method

In 1984, Bourjault first proposed a method that generates all assembly sequences of a given product algorithmically using a predefined set of rules about the precedence knowledge of an assembly [70]. This precedence knowledge may have been used intuitively by assembly designers for many years. Bourjault's method begins with a network established from the information contained in the parts list and in the assembly drawing, called liaison diagram (see Figure 3.1), with nodes represent parts and lines between the nodes represent "liaison" [5]. The precedence knowledge came from the answers of the user to a set of structured yes-no questions about whether certain liaisons can be established before or after others. The questions take the single form (L_i is read "the liaison numbered i"):

> Is it true that L_i cannot be established after L_j and L_k have already been established?
>
> Is it true that L_i cannot be established if L_j and L_k are still not established?

The answer "Yes" or "No" represents the ability or inability to assembly a part to a subassembly judged by user from geometric reasoning. Based on the precedence knowledge, rules permit algorithmic generation of all the possible sequences for the

[5] liaison - a close bond or connection. In here it refers to any of certain user-defined relations, generally include physical contact between parts.

given assembly, and an inverted tree was used to document these sequences. As this method involves a large number of questions, it is limited to products with only few parts.

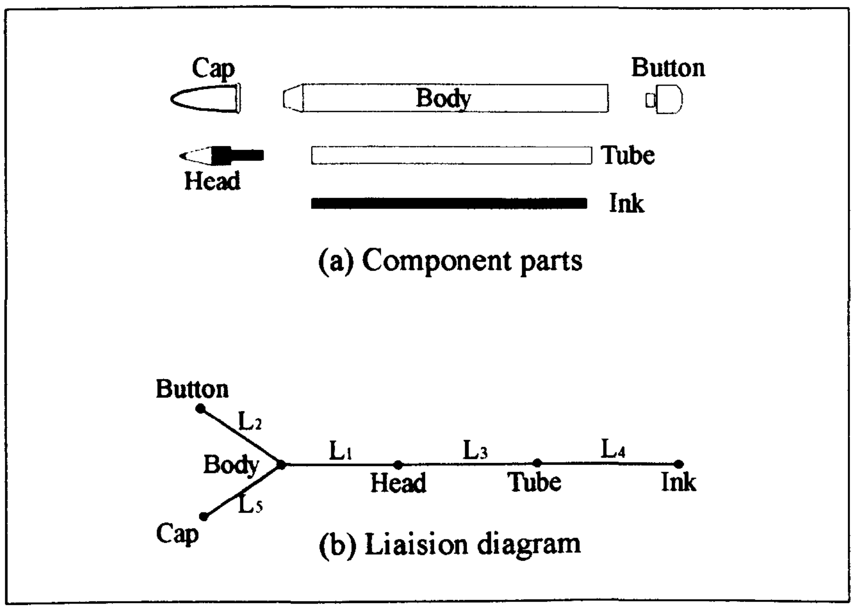


Figure 3.1 Liaison Diagram of A Ballpoint Pen

To solve this problem, De Fazio and Whitney refined Bourjaut's method and reduced the number of questions to be asked drastically from $2L^2$ to 2L (L is the number of liaisons) [71]. However, it increased the complexity of the answers, and requires anticipation from the user. De Fazio and Whitney argue that when a production engineer or assembly mechanic faced with an unfamiliar product to assemble it was asked fewer questions, not necessarily more complicated, but more involved than Bourjault's questions. They changed the question structures to directly evoke liaison relationships. For each liaison i, the following two questions are asked:

Q1: What liaison(s) must be established to allow establishing liaison L_i?

Q2: What liaison(s) must be left unestablished to allow establishing liaison L_i?

The user can answer questions with "nothing" or with a precedence relationship between liaisons or between logical combinations of liaisons. Also, instead of using an inverted tree, a more compact graph, called "Liaison-sequence diagram" or "Diamond Graph" (Figure 3.2), was used to represent all the generated liaison sequences. In "Liaison-sequence diagram", each box represents a state, which contains some cells, each representing a liaison. A blank cell implies that the corresponding liaison is not established, while a marked cell (black cell) implies that the liaison has been established. Empty box in 0th rank to fully marked box (5th Rank) in Figure 3.2 represents beginning (disassembled) state to final (assembled) state respectively. Assembly proceeds from state to state along lines representing available state transitions.

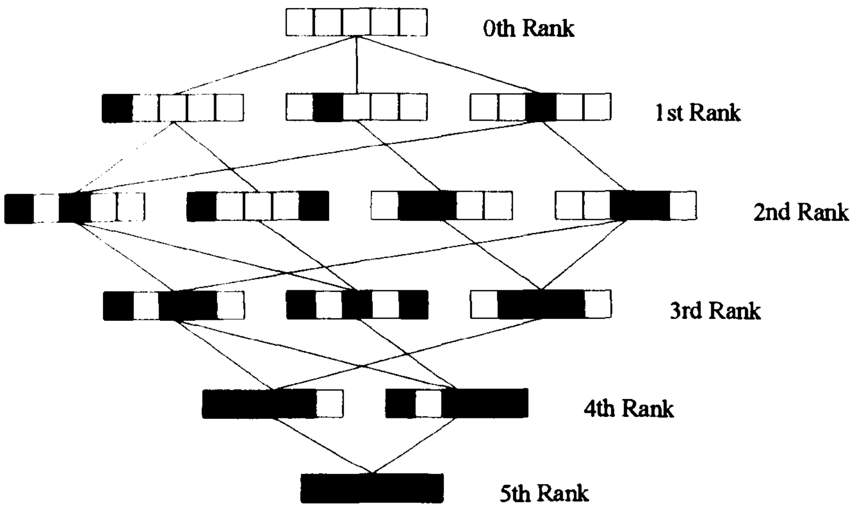
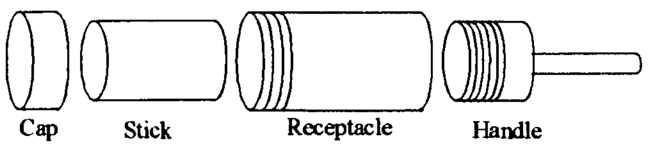


Figure 3.2 Liaison-sequence Graph of the Ballpoint Pen

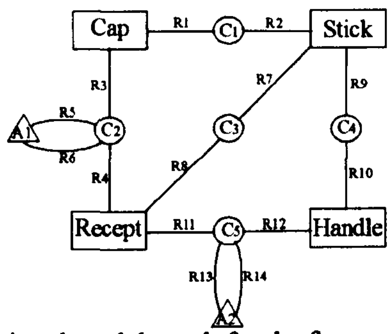
3.1.2 The Decomposition Method

Around the same time, an assembly by disassembly approach (also called decomposition approach) was proposed. It assumed that the assembly sequence is the reverse of the disassembly sequence, therefore the problem of generating assembly sequences becomes the problem of generating disassembly sequences. This seems easier because the disassemblability of a part or a sub-assembly directly implies the satisfaction of precedence relationship, whereas in the forward planning for assembly, the satisfaction of precedence relationship may not be known immediately until an exhaustive search is complete.

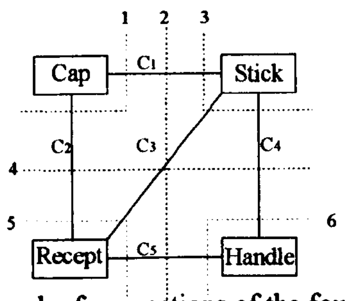
The most general method of decomposition approach is presented by Homem de Mello and Sanderson which takes a description of the assembly and returns an AND/OR graph to represent assembly sequences [72, 73, 74]. Using this method, each decomposition corresponds to a disassembly, and problems can be decomposed into sub-problems. The relational model that includes three types of entities: parts, contacts, and attachments as in Figure 3.3, is used to represent assemblies. Graph connections of the assembly can be generated from the relation model.



(a) A four-part assembly in explored view



(b) Relational model graph for the four-part assembly



(c) Cut sets of the graph of connections of the four-part assembly

Figure 3.3 Relational Model, Cut Sets of Graph of Connections of A Four-part Assembly

The algorithm Homem de Mello and Sanderson used involves generating all cut-sets of the assembly's graph of connections (as in Figure 3.3), and checking which cut-set corresponds to a feasible decomposition. A decomposition of assembly is said to be feasible if it satisfies two predicates:

- *TASK-FESIBILITY* -- feasible to join the two subassemblies to form the assembly. It depends on a number of conditions such as existence of a collision-free path to bring the two subassemblies into contact, the accessibility of fasteners, and the availability of the devices to execute the assembly task.
- *SUBASSEMBLY-STABILITY* -- there is a nonempty set of orientations for which there is a part p such that if p is fixed the other parts maintain their relative

positions and do not break contact spontaneously. The stability of subassemblies depends on a number of conditions such as the gravity and the friction in contacts.

It was assumed that it is possible to decide correctly whether a decomposition is feasible or not, based on geometrical and physical criteria. As the feasibility of assembly operations are considered during planning, the search space is reduced, and this makes the approach more efficient. However, there should be more efficient decomposition and search techniques that are implemented in a partial representation space, rather than searching for all cut-sets of the connections for decomposition of the assembly, as in practice, sequences are not generated exhaustively, rather the search tree for desirable sequences can be pruned during the sequence generation process. For example, it is claimed that an efficient algorithm has been developed to impose geometrical constraints during the generation of feasible decompositions [75].

AND/OR graphs (e.g. Figure 3.4) provide a compact representation of all possible assembly plans in the decomposition approach presented by Homem de Mello and Sanderson. The graph starts from a complete product and ends with a totally unconnected set of parts, where black and white squares represent assembled and unassembled parts respectively. Each feasible decomposition corresponds to a disassembly operation, and corresponds to a hyperarc in the AND/OR graph connecting a node corresponding to the assembly to the other two nodes corresponding to the two subassemblies.

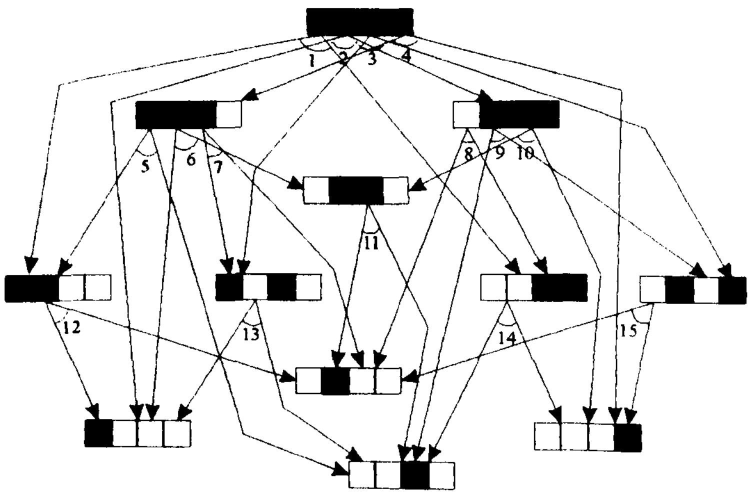


Figure 3.4 AND/OR Graph of the Four-part Assembly

While the decomposition approach is systematic and easy to automate by connecting the planner to the CAD database, it is not very user-friendly. The user has no efficient means of specifying his or her preferred constraints as the algorithm used to detect the feasible decomposition is based merely on the interference arising when a component or subassembly is separated from another subassembly [76]. Conversely, the liaison sequence approach is more user-friendly in specifying the user preferred constraints, but it is not very systematic and depends heavily on user's judgements. Both the liaison sequence method and the AND/OR graph decomposition method generate all possible assembly sequences first, then cut down the number of sequence as in Figure 3.5. The liaison sequence method prunes the sequences manually; and the decomposition method uses genetic algorithm AO* to search for an optimal sequence. These two approaches, whilst being very different, laid the foundations for the growing field of CAAP. Based on the algorithms of these two approaches and the original programs of Lui [77] and Abell [78], a set of user-interactive computer programs for generating and evaluating assembly sequences has been developed by Baldwin et al. [79].

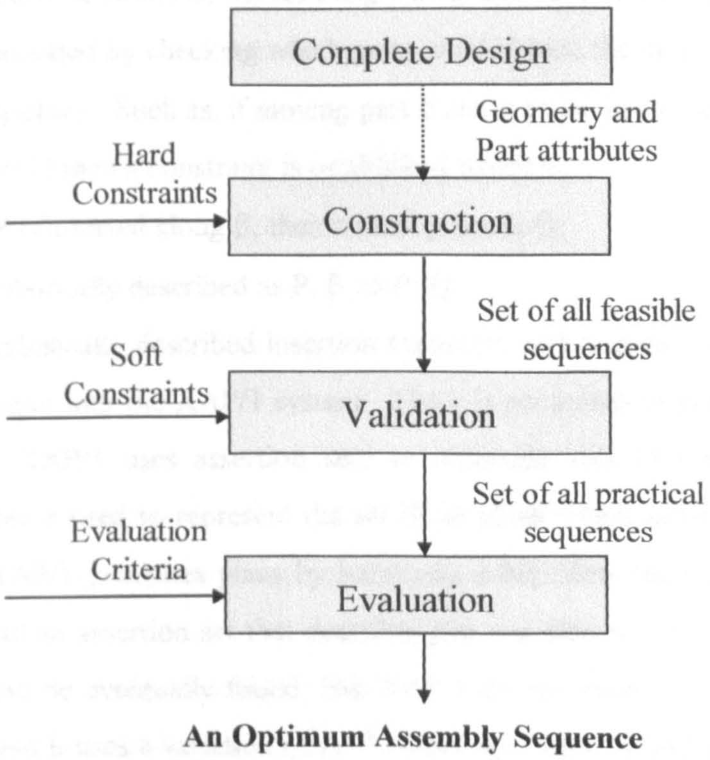


Figure 3.5 Automatic Assembly Sequence Generation Procedure

As, using either of the above methods, the number of feasible assembly sequences generated grows hugely with the increasing of parts in an assembly - the number of sequences generated could be enormous, thus it may be difficult to edit the sequences manually and computationally expensive in using genetic algorithms to search an optimal sequence. Consequently, most research after the above fundamental pieces of work has concentrated on reducing search space to make the planner more efficient. Two solutions are mainly used: constraint-based searching and knowledge-based approach.

3.1.3 Constraint-based Search

A constraint-based planner XAP/1 described by J. Wolter [80] which uses an opportunistic, constraint-posting method to search for a good assembly plan. It generates plan based on insertion operations. A subassembly tree diagram, as Figure 3.6, has been used to represent plan with insertion operations. Each node represents a subassembly or a part with an insertion trajectory. The horizontal links indicate the order of insertion and the other lines connect subassemblies to their belonging parts. The geometric feasibility of the resulting plan is enforced by a single form of constraints that are generated by checking which parts would block the insertion of a given part by a given trajectory. Such as, if moving part P along trajectory β causes P to collide with another part Q then a constraint is established which is:

If P is inserted along β, then it must precede Q,

symbolically described as $P: \beta \Rightarrow P<Q$.

All the symbolically described insertion trajectory and sequencing constraints must be manually input into the XAP/1 system. There is no access to geometric model during planning. XAP/1 uses assertion sets to represent sets of plans. Specifically, an assertion set is used to represent the set of all plans which satisfy all the assertions in the set. XAP/1 generates plans by iteratively subdividing the initial assertion set into subsets until an assertion set that describes just one plan is found. Using this method, all plans can be eventually found. But XAP/1 do not claim to enumerate all possible plans, instead it uses a variation of A* [6] search algorithm to find an optimal linear plan.

[6] A* is a heuristic search algorithm based upon the best first strategy that, under appropriate circumstances, is guarantied to find one optimised solution.

This is achieved by defining an optimal rating function *f(P)* which is a weighted combination of a number of different heuristic criteria. These criteria are provided by a set of plug-in modules. An estimate of the optimal rating is used to decide which plan is the best and worthy of further refinement. The novelty of the approach is the discarding of unpromising sequences during plan generation process as in Figure 3.7. This narrows down the search space.

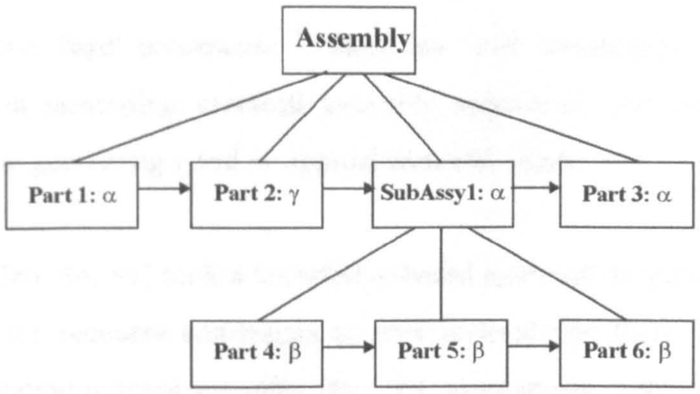


Figure 3.6 Subassembly Tree Diagram

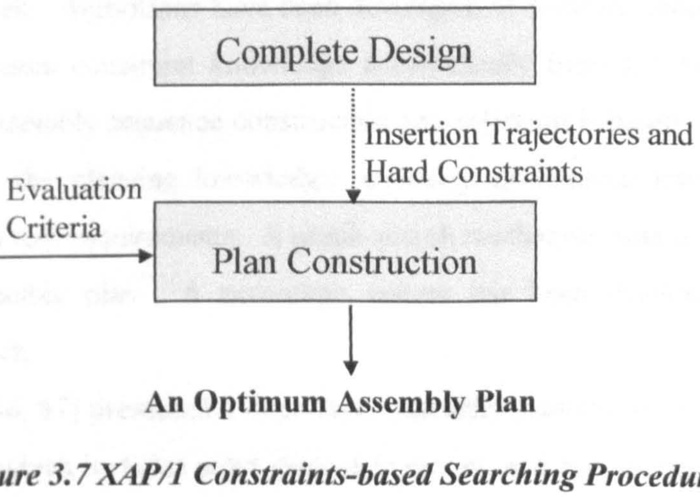


Figure 3.7 XAP/1 Constraints-based Searching Procedure

Three criterion modules are implemented in XAP/1 system. They are:

1. **Directionality module** which supports the insertion of parts as much as possible from a single direction.
2. **Fixture Complexity module** which supports to order the operations so that the partially built assemblies hold themselves together as much as possible.

3. **Manipulability module** which supports performing the most difficult operations with the more easily handled parts.

A major disadvantage of XAP/1 system is that it is limited to linear plans, so it can not generate plans which contains subassemblies. To overcome this, extensions have been made to allow the generation of plans with subassemblies [81, 82].

3.1.4 Heuristic Rules and Knowledge-based Approach

Besides geometric 'hard' constraints [7], there are 'soft' constraints [8] which play an important role in generating practical assembly sequences, and heuristic rules are frequently used in generating good or optimal assembly plans.

Huang and Lee [83, 84, 85] took a knowledge-based approach to generate an assembly plan subject to the recourse constraints as they realised that there is no algorithmic method of generating optimal assembly plan and conventional optimisation techniques are not adequate to handle the complex assembly planning problem. The symbolic presentation (predicate calculus) was used for knowledge representation of the product structure, precedence constraints and resource (such as tools, fixtures) constraints of a given assembly cell. Algorithms have been developed to conduct geometric reasoning to obtain precedence constraint knowledge automatically from a CAD model of the assembly. The assembly sequence construction and selection is based on product rules formulated from the planning knowledge, including precedence knowledge, fixture specification, and tool requirements. A graph search mechanism was used to search for the optimal assembly plan. A prototype system has been developed to test the proposed approach.

Lin and Chang [86, 87] presented a three-layer assembly planning system Figure 3.8 for assembly design which includes solid geometric model, and non-geometric information stored in frames, such as mating joint characteristics, design intents, and properties of mechanical fasteners. Algorithms have been developed to perform geometric reasoning

---

[7] Hard constraints – constraints can not be violated when generating assembly sequence.

[8] Soft constraints – constraints can be violated without causing part collisions and part trappings, but they play a part in determining good assembly sequences.

on solid models of the assembly to generate part precedence order imposed by geometry. A rule-based approach uses a decision tree to derive sequence constraints resulting from non-geometric assembly information. They proposed a two-stage plan generation which considers geometric constraints first to obtain geometrically feasible assembly plans, then considers non-geometric properties to ensure the generated plans realistic.

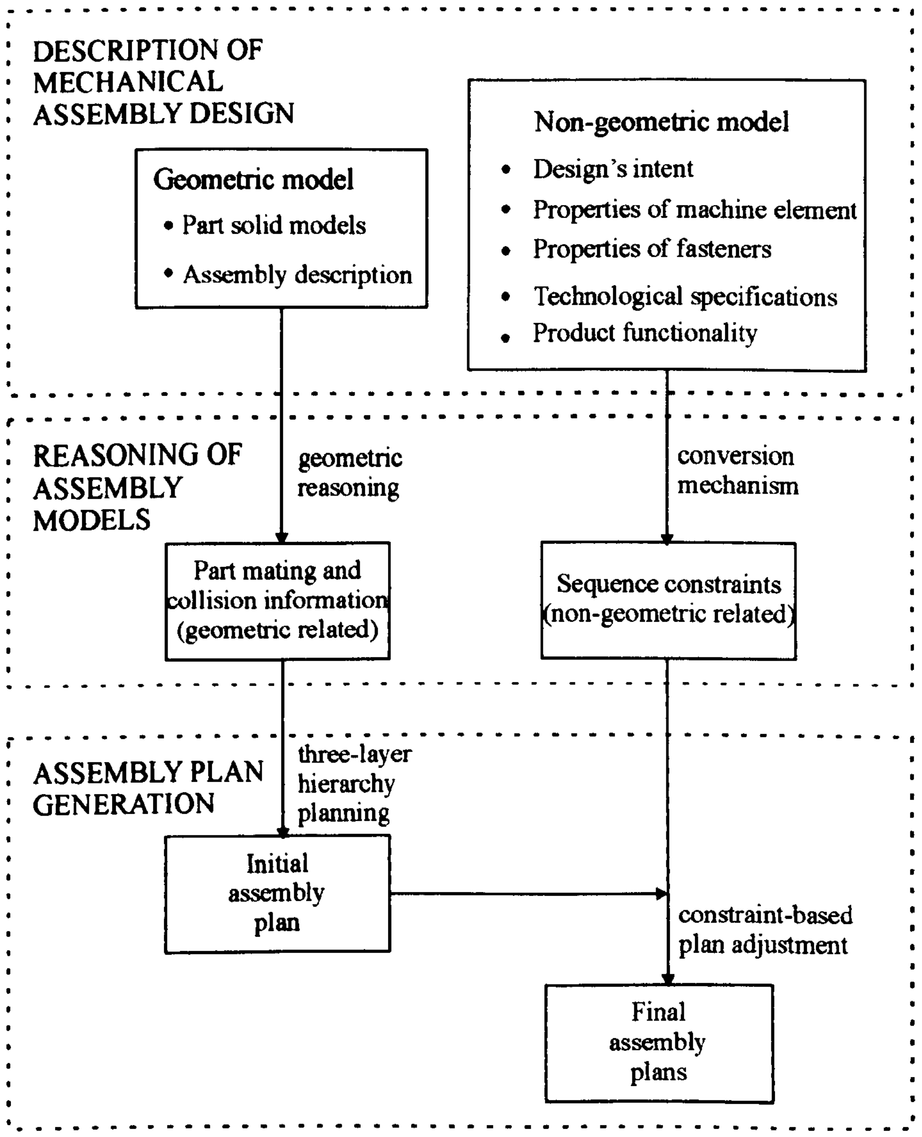

DESCRIPTION OF MECHANICAL ASSEMBLY DESIGN
Geometric model
• Part solid models
• Assembly description
Non-geometric model
• Design's intent
• Properties of machine element
• Properties of fasteners
• Technological specifications
• Product functionality
REASONING OF ASSEMBLY MODELS
geometric reasoning
conversion mechanism
Part mating and collision information (geometric related)
Sequence constraints (non-geometric related)
ASSEMBLY PLAN GENERATION
three-layer hierarchy planning
Initial assembly plan
constraint-based plan adjustment
Final assembly plans

Figure 3.8 Framework of Automated Mechanical Assembly Planning Proposed by Lin and Chang

An expert assembly process planning system has been presented for mechanical assembly around one revolution axis [88]. It generates an optimal sequence directly using the information contained in the initial non-assembled state and in the finial assembled state of the assembly. This idea stems from the observation that, in industry, an engineer does not consider all assemble sequences, instead, he/she tries to find the optimal sequence from the beginning of the sequence planning, and heuristic rules are used to make decisions in eliminating unpromising sequences through the planning stage. The expert system proposed incorporating knowledge acquired from assembly experts to aid the human operator in generating precedence relations. This eases the tedious question and answer procedure as in the liaison sequence method [71]. Tests show that the system results in faster planning and is less memory consuming. Even though it is restricted to product assembly with quite small number of parts around one revolution axis, this approach suggested an idea to use a higher level of abstraction to represent assembly problems, which may offer an opportunity to make computer-planning systems more efficient.

Another proposed method for generating assembly sequences efficiently at the product design stage [89] reduces search space in sequence generation by using a precedence graph generated from heuristics, and existing sequence fragments. The generation of a precedence graph from heuristics is based on product functional hierarchy. This method is especially efficient in generating assembly sequences for design objects modified from existing designs whose assembly sequences have already been generated before.

Heemskerk and van Luttervelt [90] used heuristics to group parts into several clusters to simplify the assembly sequence planning. Part grouping is based on some part characteristics, such as same type and having similar relations to another part. The grouping significantly reduced the combinatorial complexity of possible sequences. The cluster recognition algorithms have been tested on various products which brings a significant reduction of the number of sequences that can be generated. However sometimes, a design feature is considered to be essential for the application even though it may result in technically non-optimal sequences. Because in such cases, some of the implemented cluster recognition algorithms eliminate these sequences, user interference may be necessary in keeping the feature and the sequences. Lee and Shin also described a number of heuristics to group parts into subassemblies in the reference

[91]. Besides part grouping heuristics, accessibility heuristics are also used to eliminate sequences that will result in part/assembly collision; and stability heuristics are used to eliminate sequences containing unstable states in the sequence planning process proposed by Heemskerk and van Luttervelt.

A systematic approach for design and planning of mechanical assembly has been proposed, which captures design concepts, integrates neural network computing, and uses rule-based system to generate a task-level assembly plan automatically [92]. The function of the **neural network module** is to retrieve similar conceptual assembly designs from the design memory using an associative memory type of neural network algorithm. Based on the assembly concept, a designer can retrieve and refine a desired 3D B-rep assembly object. The **preprocessor module** calculates the plane equation for each face of parts in the assembly, and this important piece of information is used throughout the system to determine candidate assembly directions for every components as it includes the outward face normal of components. The **liaison detection module** determines any of the three types of liaisons: *insertion, attachment,* and *contact* between two parts. The output of this module will add the detected physical liaisons between components into the knowledge-base fact-list in CLIPS expert system shell. The **obstruction detection module** employs a collision detection algorithm for every component in every candidate assembly direction. The objective is to determine the set of components which would obstruct the assembly operation if they were already in their final position. Construction of assembly plan uses a disassembly approach under the condition that the reverse of a disassembly sequence is a valid assembly sequence and this is performed by the **plan formulation module**. The knowledge base contains two criteria to select the 'best' component for removal:

1. Minimising the number of tool changes;
2. Uniformity in the directionality of successive disassembly actions.

The system is integrated with a solid modeller which provides a graphical user-interface to design and modify the assembly. The model data can be translated into format of CLIPS facts. Also a simulation facility is available to graphically demonstrate the assembly plan steps. System input can be either conceptual assembly, solid model of the product or CLIPS facts.

Besides knowledge-based systems as mentioned above, other AI techniques, such as neural-network [93, 92], case-based reasoning [94, 95], fuzzy logic [96] have also been used in assembly sequence planning. For example, case-based reasoning [94, 95] increases planning efficiency from 'plan reuse' which divides the assembly into a number of constituent configurations and retrieves plans for each sub-configurations and then fuses these sub-plans into a set of complete plans.

3.1.5 Connecting Assembly Planner with CAD Database or CAD System

In addition to reducing search space in assembly planning, attempts have been made to connect assembly planners with CAD to extract precedence knowledge directly from a product model. The method [86, 87] mentioned previously using geometric reasoning instead of user question answer to derive precedence relationships of parts directly from a CAD system is also suggested in references [97, 98].

A knowledge-based architecture for extracting interface information between part components from product models with minimal user interaction is presented in [99] where heuristic rules have been used for base part selection and part order determination. In the architecture, a database interpreter accepts the stored product model as input and writes the information about features in frames. A knowledge-acquisition module parses component frames to filter out all the assembly-related features and stores them as predicates with their geometric, manufacturing and orientation parameters. The knowledge analysis module Figure 3.9 uses this set of assembly features to generate a list of feasible interface relationships (feature liaisons) by checking feature type, feature geometric dimension and feature orientation. The knowledge representation procedure uses feature liaison information generated to update the location of features in the final assembly state. Thus the entire assembly is represented in terms of spatial relationship predicates and updated feature predicates. Based on these, an undirected connectivity graph representing all feasible configurations of the assembly is generated. A concept of articulation points (Artpts [9])

[9] an Artpt is a vertex in connectivity graph where allows the graph structure to break down into two or more pieces of complete connections or single node with one edge.

is first time used to decompose the connectivity graph to small pieces for subassembly generation. Procedure are developed and a spanning tree is used for searching all Artpts (dash boxed vertices in Figure 3.10), and subassemblies are then formed by grouping parts around the Artpts. After subassemblies are identified, the sequence generation process uses heuristic rules to determine the base part and the ordering preferences of the other parts. Then a directed connectivity graph which contains assembly sequence information is generated. Finally, algorithms are used to transfer this directed connectivity graph into a triangle connectivity matrix to represent one feasible assembly sequence.

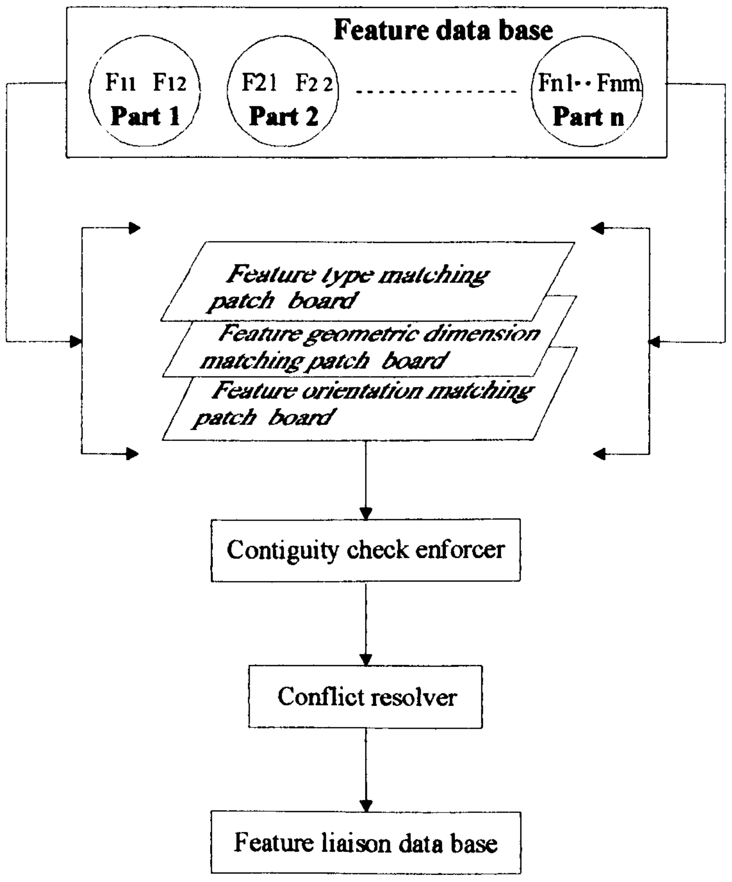

Feature data base
F11 F12
Part 1
F21 F22
Part 2
Fn1··Fnm
Part n
Feature type matching patch board
Feature geometric dimension matching patch board
Feature orientation matching patch board
Contiguity check enforcer
Conflict resolver
Feature liaison data base

Figure 3.9 The Knowledge Analysis Module in Liaison Determination Procedure

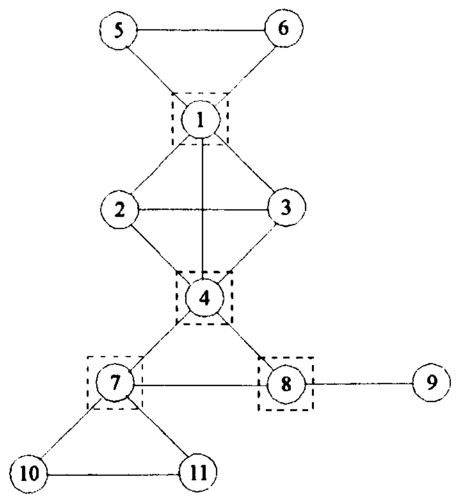


Figure 3.10 Articulation Points in Undirected Connectivity Graph

A computer integrated assembly planning system (CIAPS) that extracts precedence knowledge directly from an assembly model has been proposed [100, 101]. It implements a typical three phase assembly sequence planner, see Figure 3.11.

In the first phase, it starts with the design and creates solid models of the assembly. Then the precedence knowledge is automatically extracted from the solid model of the design in its assembled configuration in terms of contact (C) and translation (T) functions for each pair of components. The C-function of any pair of parts can be extracted from the geometry and topology by testing if there is any interference occurring between the two parts in any of the six parallel and anti-parallel [10] co-ordinate directions by moving one part over a small incremental distance. The T-function of any pair of parts can be extracted by checking in which parallel and anti-parallel co-ordinate direction there is a collision-free path to disassembly one of the parts from the other. The extracted precedence information is stored as a relational model and this mode is flexible enough to include any additional information.

In the second phase, an assembly sequence generation algorithm takes contact and precedence information from the relational model as input and generates all the

[10] anti-parallel -- parallel but opposite in directions

geometrically feasible sequences. The generated sequences are then represented by two schemes: Assembly Sequence Graph (ASG) and Assembly Sequence Table (AST).

In the third phase, sequence selection uses various criteria to filter out the unpractical sequences. DFA toolkit software is used to evaluate the feasible and practical sequences to obtain an optimal sequence.

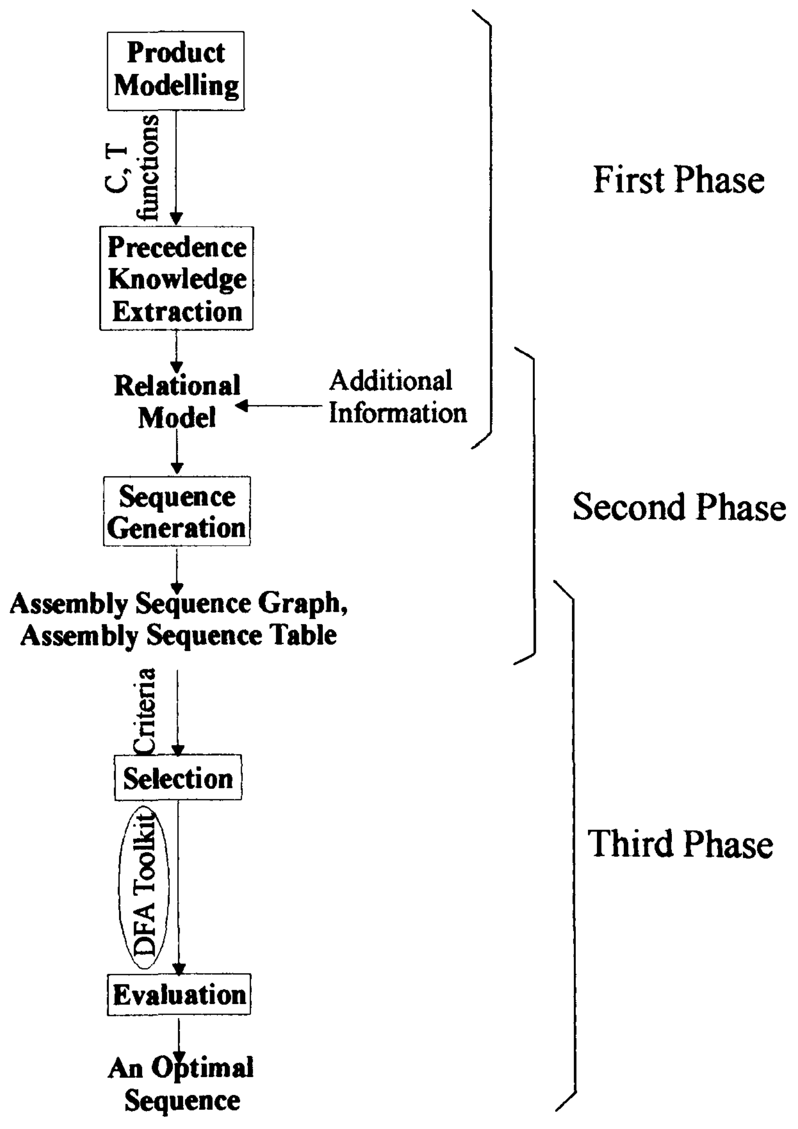


Figure 3.11 The CIAPS Assembly Sequence Planner

Some later approaches allow the inclusion of a more detailed level of operation to increase the practicality and usefulness of the resultant plan, such as the FLAPS (Flexible Assembly Planning System) [102].

FLAPS is an integrated assembly planning system. It uses a CAD model as input to generate assembly sequences and it can be connected to a computer network to access process planning systems, and plant simulation. Interestingly, besides assembly sequences, it can also plan the assembly operations with trajectories and select gripping tools to generate off-line robot programs. These and the additional ability of simulating assembly with plant layout make FLAPS potentially capable of creating actual plans, see Figure 3.12. A decomposition approach was used in sequence generation. A file containing part joining information in the design stage was used for contact recognition between two parts by translating parts along six main directions to test the intersections and thus to find the direction of disassembly. A "table of contacts" for every disassembly direction is then built. Sometimes the disassembly directions can not be detected automatically, e.g. the part can be disassembled only through an elastic deformation or the part has a disassembly direction different from the six main directions. During this time, human intervention is needed. In FLAPS, a compromise between the fully automation and human interaction has been reached to reduce the computationally complexity and planning time. By searching the table of contacts and related incidence matrix of the contact graph, the system extracts feasible subassemblies based on certain rules, such as the stable configuration rule, etc. Tests show that user intervention is also needed to solve multiple equivalent solutions and to confirm the choices taken by the system in grouping subassemblies. The sequence generation process is such that after the base part is selected by the user, two algorithms are used to generating all geometric feasible sequences. Then precedence rules and accessibility and stability criteria are used for selection of good sequences semi-automatically.

The system plans the operations and the trajectories that the robot performs based on the "assembly rules" structured in knowledge base. A module for gripping surface and gripping tool (including gripper fingers) selection can make suggestions regarding grasping surface design as well as new gripper finger design. The assembly plant simulation module verifies the feasibility of the assembly sequences through a simulation process. The minimum execution time has been used to identify the best assembly plan.

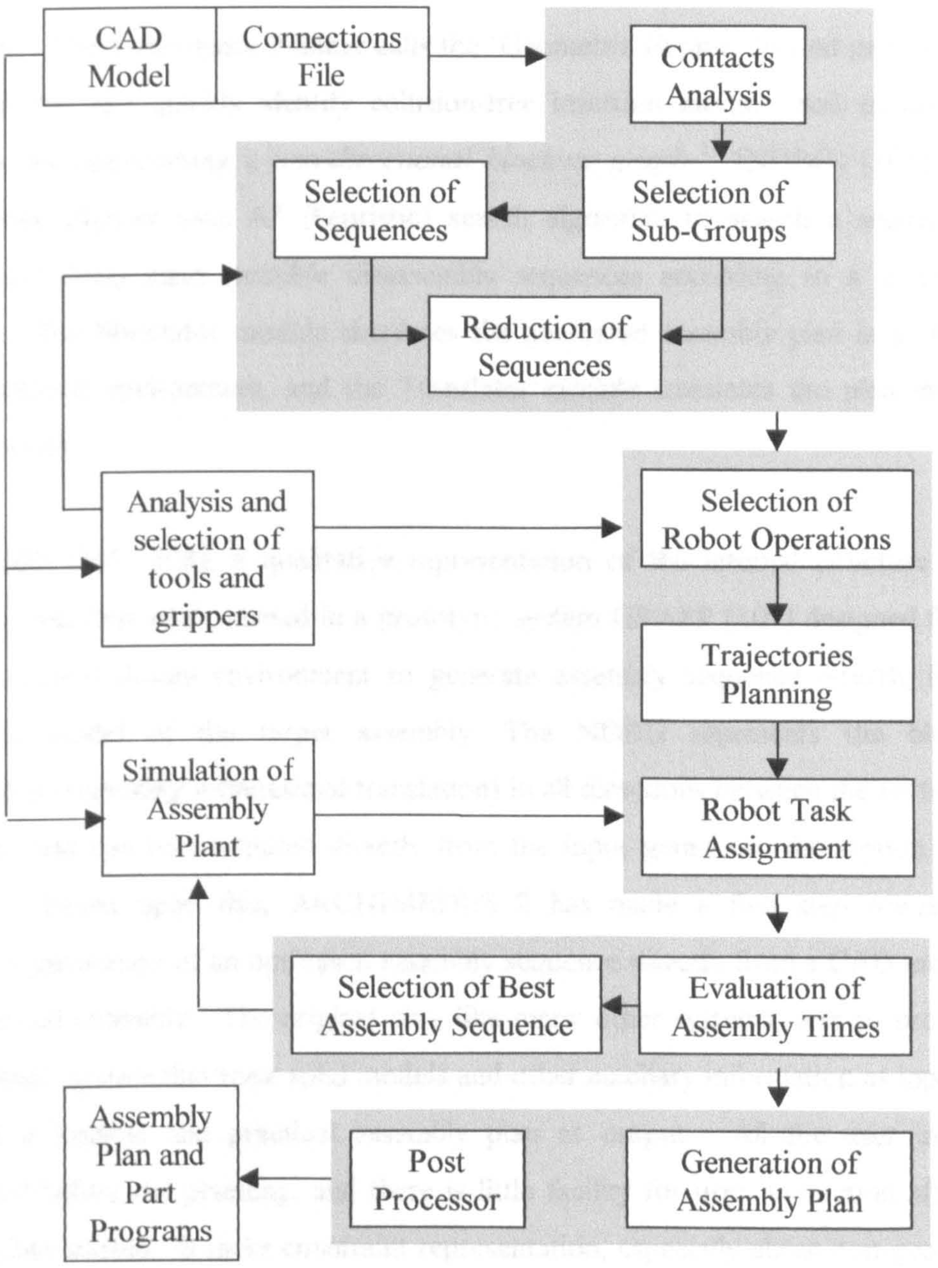

CAD Model
Connections File
Contacts Analysis
Selection of Sequences
Selection of Sub-Groups
Reduction of Sequences
Analysis and selection of tools and grippers
Selection of Robot Operations
Trajectories Planning
Simulation of Assembly Plant
Robot Task Assignment
Selection of Best Assembly Sequence
Evaluation of Assembly Times
Assembly Plan and Part Programs
Post Processor
Generation of Assembly Plan

Figure 3.12 Overview of the FLAPS

ARCHIMEDES [103, 104, 105] is a comprehensive assembly planning system which takes product model and user-defined constraints to automatically generate (with user interacting facilities) assembly order. After ten years of development, there are several versions of software. ARCHIMEDES 2 [103] can accept translated data from commercial CAD systems and generate an optimized assembly plan which can then be translated automatically to robotic code for a robot workcell. The system architecture Figure 3.13 requires inputs of geometric solid models (ACIS models) in assembled configuration, joining methods, recommended subassemblies, and suggested insertion directions. The State-Space Planner calls the 'Geometric Engine' to find part contacts automatically, and quickly identify collision-free insertion motions and subassembly partitions by constructing a *non-directional blocking graph* [11] (NDBG) [106]. The State-Space Planner uses A* (heuristic) search algorithm to search a sequence of lowest cost from many possible disassembly sequences according to a given cost function. The Illustrator module simulates the generated assembly plan in a realistic robot workcell environment, and the Translator module translates the plan into V+ robotics code.

The NDBG [107, 108], a qualitative representation of the internal structure of an assembly, was first implemented in a prototype system GRASP [109] designed for use in a concurrent design environment to generate assembly sequence directly from a geometric model of the target assembly. The NDBG represents the blocking relationships (blocking infinitesimal translation) in all directions between the parts of an assembly, and can be computed directly from the input geometric description of the product. Based upon this, ARCHIMEDES 2 has made a first step towards an automatic generation of an optimised assembly sequence directly from a CAD model of a mechanical assembly. The original aim, like many other systems, was to provide a "Black Box" system that took solid models and other auxiliary information as input and returned a feasible and practical assembly plan as output. All the user input is performed before the planning, and there is little facility for user interaction after the planning has started. It lacks constraint representation, especially about non-geometric data representation. This contradicted the preference in industry which requires a computer-aided environment to allow an engineer to richly and fully participate during the planning process [110, 111]. It is apparent that during the realisation of the difficulty of the task and considering the preference of industry, the aspirations towards totally automatic assembly sequence generation has changed to allow increasing amounts of user interaction. Later revisions of the ARCHIMEDES software aim to

[11] non-directional blocking graph (NDBG) represents the blocking structure of an assembly for infinitesimal translations in all directions. The arrangement of points and intervals on an unit circle and the associated *directional blocking graph* (DBG) form the NDBG, see references [107, 108]

find a balance between user interaction and the use of automation.

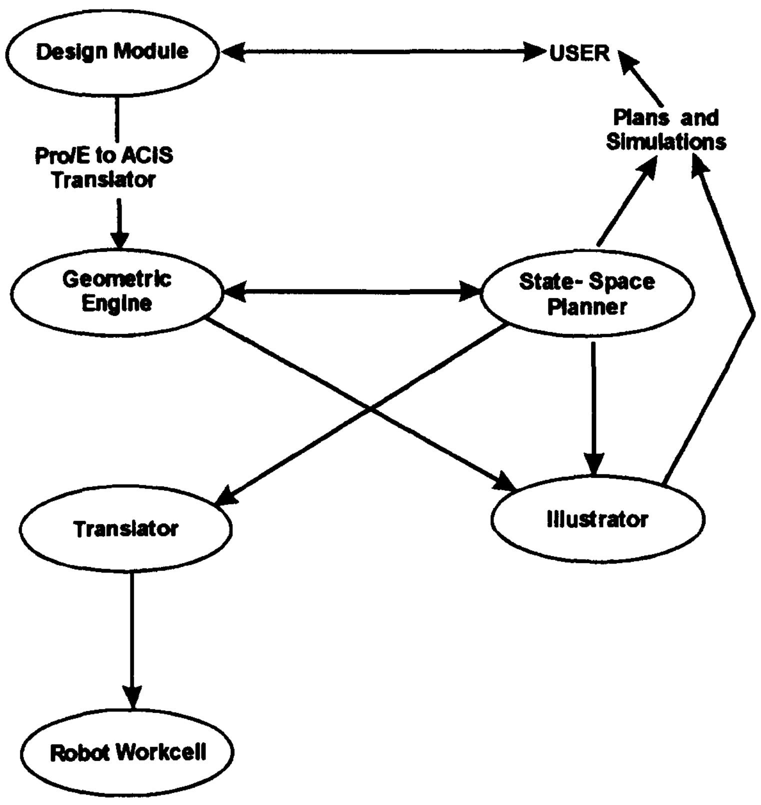


Figure 3.13 The Architecture of the ARCHIMEDES 2

A comprehensive survey of assembly planning constraints was conducted [112] to gather all possible constraints used in a real assembly environment. A substantial body of constraints has been formalised and categorised, and a library of constraint types has been integrated into ARCHIMEDES 4.0 (Figure 3.14) to interactively input or edit a rich variety of useful constraints [111]. The search strategy is carefully tuned to generate a first plan as quickly as possible in the domain of mechanical assembly. Then it uses the view-constrain-replan cycle to interactively generate a feasible and practical plan (see Figure 3.15). Systematic exploration of the space of possible assembly sequences is based on a user's knowledge of the application-specific assembly process requirements. This reduces the search space and makes the system very effective. But the underlining data structure is still unchanged, and the separate application of geometric and technological constraints do not help to narrow down the search space early in the planning process. Currently ARCHIMEDES has increased its scope to include consideration of lifecycle design factors and implemented cost analysis in the planning process [105, 113].

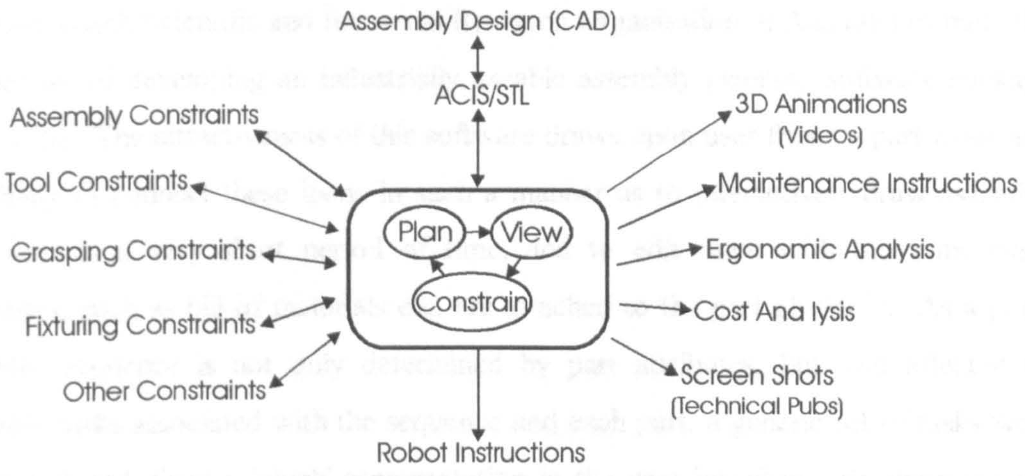


Figure 3.14 The Archimedes 4.0 Assembly Analysis and Planning Software System

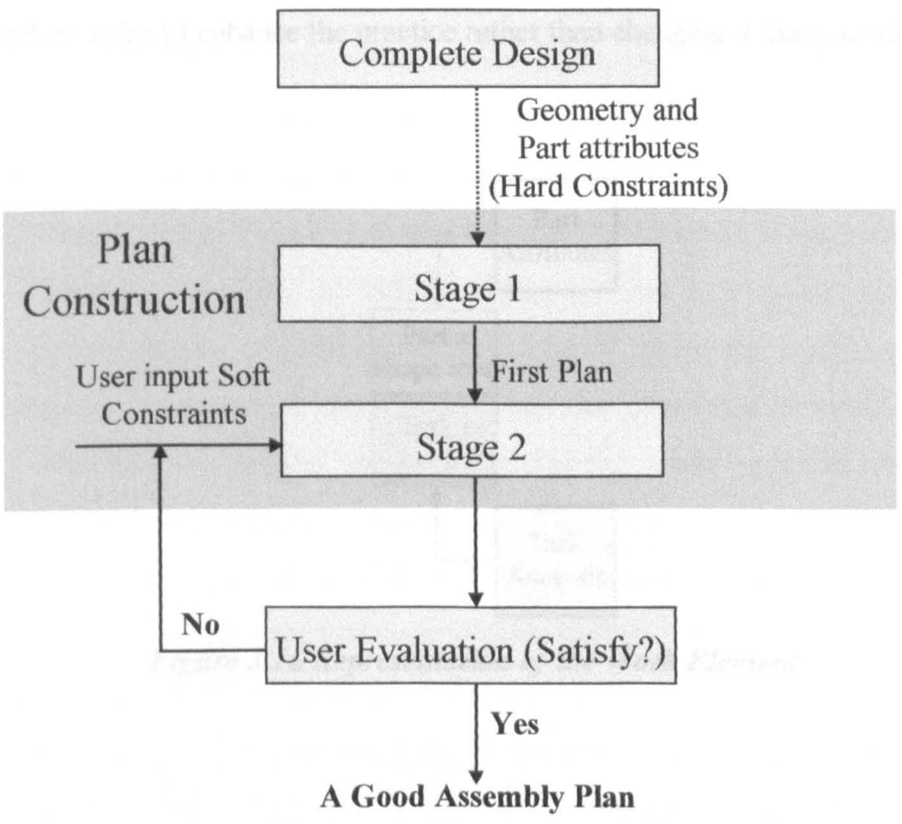

Complete Design
Geometry and Part attributes (Hard Constraints)
Plan Construction
Stage 1
First Plan
User input Soft Constraints
Stage 2
No
User Evaluation (Satisfy?)
Yes
A Good Assembly Plan

Figure 3.15 Archimedes 4.0 Interactive Assembly Planning Procedure

3.1.6 Interactively Draw Assembly Sequence

An interesting assembly planning software package has been developed by CSIRO, the Commonwealth Scientific and Industrial Research Organisation of Australia to meet the perspective of developing an industrially useable assembly planning software package [114, 115]. The attractiveness of this software draws upon user friendly part icons and the ability to connect these icons in such a manner as to interactively draw assembly sequences in a very short period of time, and to edit them with the same ease. Attributes, such as bill of materials etc, are attached to the part glyphs [12]. As a good assembly sequence is not only determined by part attributes, but also affected by assembly tasks associated with the sequence and each part, a generic set of tasks were established and given a 'glyph' representation in the user interface. By marrying the tasks and part dataglyphs [13] to form the work element (Figure 3.16) and giving the necessary precedence, it becomes possible to interactively draw an assembly sequence as in Figure 3.17. The industrial feedback shows that this software is very user friendly, mirrors the practices that the assembly engineers have been following for a long time, and therefore helps to enhance the practice rather than changing it dramatically.

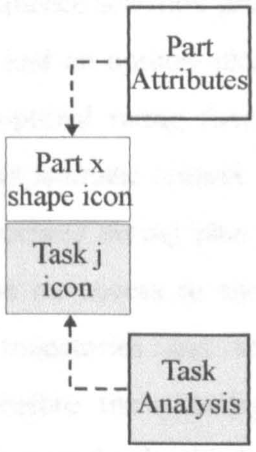


Figure 3.16 Representation of the Work Element

[12] part glyphs -- part shape icons.

[13] dataglyphs refer shape icons associated with subsequent data.

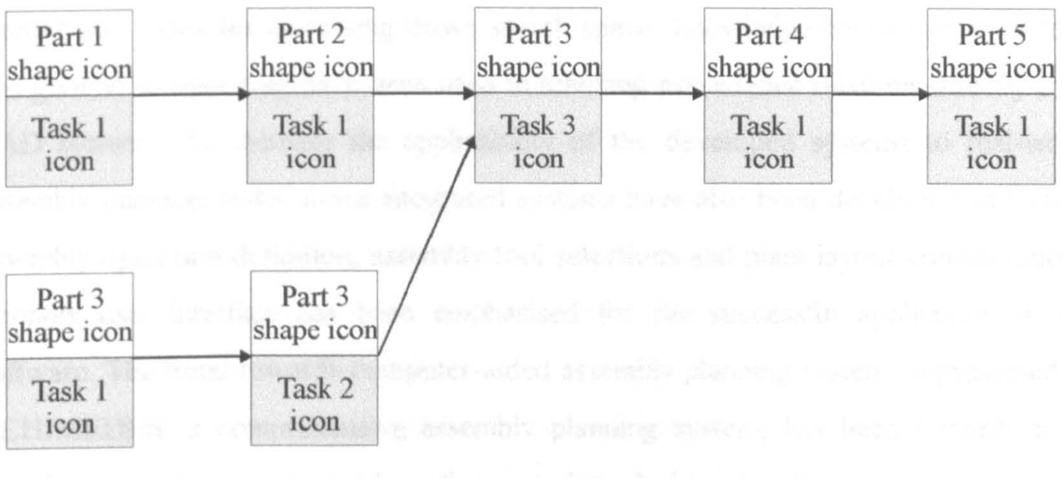


Figure 3.17 Interactive Assembly Sequence Generation in CSIRO Planning Software

3.1.7 Summary of Computer-aided Assembly Planning Research

In the reviewed literature, most of the computer-aided assembly planning methods and systems developed apply geometric hard constraints to generates all feasible sequences first, and then apply heuristic soft constraints manually or using heuristic search algorithms to prune the awkward or unpractical sequences to obtain optimal sequences. This is computationally expensive in the feasible sequence generation process and it results in a huge search space in sequence selection process. To avoid these problems, constraint-based search is used to find an optimal plan directly in sequence planning. This is achieved by defining an optimal rating function *f(P)* which is a weighted combination of a number of different heuristic criteria. The novelty of the approach is the discarding of unpromising sequences during plan generation process. However, currently the reviewed method has no access to the geometric model, and all the symbolically described insertion trajectories and sequencing constraints must be manually input into the system before the planning takes place. So any other constraints which could be much more easily identified at certain planning stage can not be input into the system to justify the optimal plan. An alternative to the constraint-based search is to use a knowledge-based approach which uses heuristic rules to narrow down the search space during the sequence planning, such as clustering parts into groups to simplify the planning, eliminating awkward or unpractical sequences to reduce search space throughout assembly sequence generation processes. Reviewed papers addressed this, but some of the knowledge-based systems are still used in a limited way. Besides narrowing down search space, knowledge-based expert systems and geometric reasoning have been used in inferring precedence relations directly from CAD system. To increase the applicability of the developed systems to real-world assembly planning tasks, more integrated systems have also been developed to include assembly operation definition, assembly tool selections and plant layout considerations. Friendly user interface has been emphasised for the successful application of the software. The trend towards computer-aided assembly planning systems represented by ACHIMEDES, a comprehensive assembly planning system, has been towards more user intervention, easy to apply soft constraints during planning process, instead of completely automated assembly planning.

3.2 Industrial Assembly Planning Practice and Requirements

From the above it can be seen that the area of assembly sequence generation has attracted much academic interest over the last decade. However, few of the proposed methods and implemented systems have developed into commercial products and industrial applications. Three main reasons have been cited for this in the foreword of a book [116] which collected a number of important papers regarding computer-aided mechanical assembly planning. The three reasons are:

- *Computational Efficiency* – The planning systems can only analyse assemblies with relatively few components before computation time becomes unacceptable and consequently the user input is required to save expensive computing.
- *Inadequate Evaluation Criteria* – Appropriate evaluation criteria of assembly sequences have not been defined comprehensively in existing planners. Cost and ease of assembly operations are commonly used evaluation criteria, but tasks related criteria such as fixturing and tooling requirements or even more practical criteria as assembly line layout varies widely, and they are difficult to be defined comprehensively.
- *Industry Conservatism* – There is an inevitable delay in technology transfer from academia to industry.

To improve the relevance of computer-aide assembly-planning systems, it is necessary to define the industrial working practices and hence identify how the computer can aid the process. In addition, discussions with potential end users can bring many issues to light and provide useful input for software development. For these reasons, an investigation of the assembly planning practice in industry has been conducted by the Ophir Project [16]. The results were compared with other published surveys and studies to define a general industrial assembly planning practice and industrial requirements for a computer-aided assembly planning system.

Ten diverse companies were visited, covering a wide spectrum of British industry. This ranged from one operator bench building precision electronics for defence use to assembly line mass-producing consumer goods. It was felt that this was a representative perspective on the assembly issues encountered by industry today. The investigation used structured interviews with relevant employees and observed real assembly processes.

It was found that, in half the companies visited, the sequence is determined after the design process has been completed, once the product attributes are fixed. This means that the consideration of the assembly sequence cannot be used to improve the design. Only one of the ten companies employed sequence optimisation techniques to improve assembly times. In three of the businesses, the sequence was never formally defined. It was left to the assembly operators to decide.

The observed processes of design and assembly planning in industry are that designers do generally consider how an assembly may be built and try to create suitable subassemblies. However, because this thought process is rarely documented, this bears little relation to the final product assembly partition and sequence. Typically, once the design is complete, it is signed off and sent to the assembly planning engineers. Their task is to collate all relevant data to enable product assembly. This data includes parts and assembly drawings, any prototypes or first production samples, similar products and factory capability information. Some information, often not explicitly available, comes from the planner's knowledge and expertise gained in the course of generating many plans. It is evident that two categories of data are used to build the plan; product specific data such as geometry and materials; and knowledge such as factory capability, available jigs and possible joining processes. This knowledge imposes constraints upon the sequence variations possible throughout the planning process. The planner first identifies the most suitable subassemblies and, in fact, defines the assembly structure. Once this has been completed satisfactorily, each subassembly in turn is considered. Then the whole plan is composed to establish the sequence of component assembly and the associated tasks. This approach can be considered as a *breadth-first, depth-second* search for feasible sequence configurations. The validation of each sub-plan is achieved by actually disassembling the product as defined. Once each subassembly is planned, the overall plan is generally validated by a trial run down the assembly line or on the workbench.

The industrial investigation has shown that there is the lack of understanding of the influence of assembly within businesses today, and assembly is still a neglected topic. If greater emphasis was placed upon improving this area, fewer production issues would occur. However it is not enough just to focus upon the actual assembly, the design has to be right in the first place. Designers do their best to create easy to assemble product, but poor communication and a lack of understanding make this less than effective.

Despite the negativity of these findings, in companies where sequences were generated, a common process became apparent. It has found that heuristic rules have been applied to constraint the sequence configurations. The hard and soft constraints were interwoven within the process and not separately considered. This is coincident with an experiment conducted with production engineers in reference [117]. It has also evidenced that the planner uses a *breadth-first, depth-second* approach that is consistent with a knowledge engineering study in reference [118] which identified a three stage planning strategy commonly used by engineers:

1. Decompose the assembly into subassemblies.
2. Select the most suitable base part for each subassembly.
3. Complete the assembly sequence for each subassembly.

In addition, it appears that validation and evaluation of the generated sequence are hardly considered in industrial situations as identified in an industrial survey [119].

3.3 Summary of Findings

By visiting a number of companies and comparing the findings to some published industrial surveys, a generic assembly planning process has been found (Figure 3.18) [120]. That is the planner uses a ***breadth-first*** and ***depth-second*** approach to identify the best subassembly partitioning and develop a rough global plan first, then plans each individual subassemblies in turn. It is evident that there is a strong heuristic base in generating good assembly sequences and optimal assembly plan. The hard and soft constraints were interwoven throughout the planning process and not separately applied.

Comparing these with the computer-aided assembly planning systems developed to day, differences have been found. That is most automatic assembly planning systems ignore the soft constraints during sequence generation process. Even when some developed systems apply soft constraints, they are separately considered after all the hard constraints are applied. The ignorance of soft constraints during sequence generation process results in enormous sequences to be generated, this may overwhelm the engineer during selecting a good sequence. Even if heuristic algorithms can be used to search an optimal sequence, the search space is huge, thus computationally expensive. Much improvement has been achieved for systems applying soft constraints during sequence generating process, such as ARCHIMEDES, but the efficiency of sequence generation process is still defected, as they do not narrow down the search space in the early stage of sequence generation. It is evident that appropriate application of soft constraints determines the performance of sequence planning [117]. But many soft constraints are hard to quantify or computerise, so their applications rely heavily on human judgement, knowledge and decision making. Therefore, computer-aided assembly planning systems need to facilitate the user collaborating with artificial intelligence in generating good assembly sequences. An interactive planning system with decision support rather than completely automated assembly planning system may be more suitable for industrial application. This has been suggested in several papers [117, 110, 102]. It has also been suggested in industrial feedback [114] which shows that users prefer interactive planning which mimics their practice and users prefer the data manipulation transparent. In summary, some of the main reasons of poor applicability of the developed assembly planning systems are:

1. The separation in applying soft and hard constraints
2. Difficulty to apply soft constraints properly during the assembly sequence generation stage.

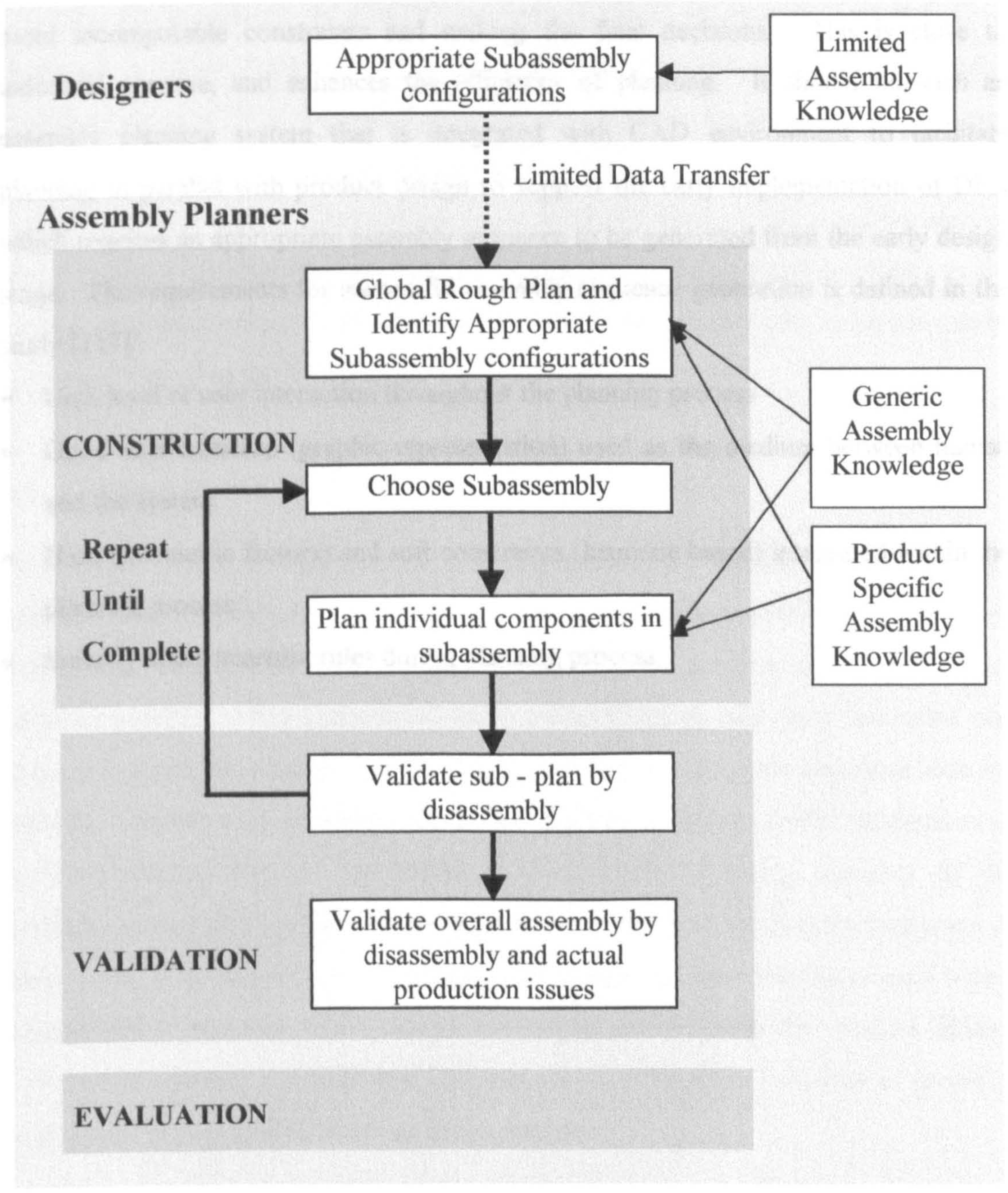

Designers
Appropriate Subassembly configurations
Limited Assembly Knowledge
Limited Data Transfer
Assembly Planners
Global Rough Plan and Identify Appropriate Subassembly configurations
Generic Assembly Knowledge
CONSTRUCTION
Choose Subassembly
Repeat Until Complete
Product Specific Assembly Knowledge
Plan individual components in subassembly
Validate sub - plan by disassembly
VALIDATION
Validate overall assembly by disassembly and actual production issues
EVALUATION

Figure 3.18 Manual Assembly Planning Practice

To accommodate the difference between the requirements of assembly engineers and the automated assembly planning systems, it is suggested here that an interactive planning system with decision support rather than a completely automated assembly planning system should be developed. In such an interactive system, geometric hard constraints and heuristic based soft constraints should be integrally applied, and the user should be able to collaborate with artificial intelligence throughout the planing process with the system focusing on computable constraints, and the user focusing on more incomputable constraints and making the final decisions. This is close to industrial practice, and enhances the efficiency of planning. It should be such an assembly planning system that is integrated with CAD environment to facilitate planning in parallel with product design to support the early implementation of DFA which requires an appropriate assembly sequence to be generated from the early design stage. The requirements for successful assembly sequence generation is defined in the study [117]:

- High level of user interaction throughout the planning process
- Good user interface (graphic representation) used as the medium between human and the system
- Hard (geometric factors) and soft constraints (heuristic based) integrated within the planning process
- Suitably apply heuristic rules during planning process.

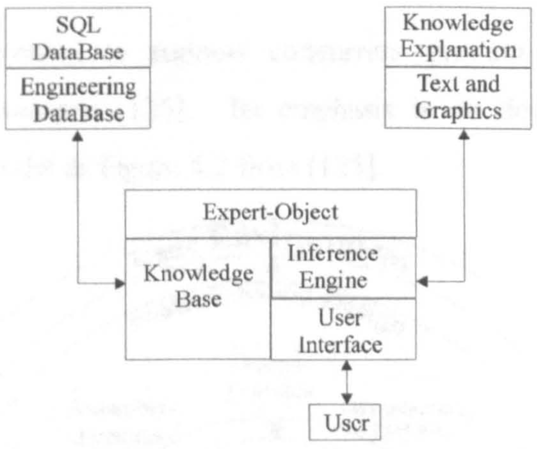
Chapter 4 Assembly-oriented CAD Environment

Conventional CAD systems have made significant contributions towards the detailed modelling and analysis of geometric design, but are generally incapable of supporting the early conceptualisation and synthesis stages of design. To overcome this and facilitate the generation of assembly-oriented product, an assembly-oriented CAD prototype has been developed out of the Ophir Project which encourages the earlier implementation of DFA evaluation in a proactive manner by providing the designer with tools for creation of a product structure and an assembly sequence concurrently with the product design. Then the sequence forms the basis of the proactive analysis that provides a quantitative view on the assemblability of the product in relation to the amount of available data. Consequently the product design, assembly sequence can be optimised according to DFA criteria.

4.1 Related Work

Bäßler et al. [121] has suggested that assembly-oriented product design means that the products should be designed in such a way that the assemble expenditure is reduced to a minimum. At the same time, the production cost of the product must be reduced as much as possible. After analysing different kinds of design aids, they concluded that <u>design rules</u> and <u>assemblability evaluation procedures</u> are the most important aids for assembly-oriented product design. They specified that in order to design products in an assembly-oriented manner, the technical requirements regarding assembly of the product, the assembly system information must be available at the development stage of the product. The assembly process must be taken into account when the product is first created and it must be analysed and planned in parallel with the product design. However, Eversheim and Baumann [35] had not considered the influence of assembly sequence in their assembly-oriented design process.

Several other papers addressed the issue of integrating design for assembly and assembly planning with CAD environment [122,123, 21, 15, 124, 36, 7, 59], but most of the methods proposed are toward the end of design process. Little work has addressed the problem of realising assembly-oriented design in a real concurrent manner. For example, a knowledge-based system [122] that integrates product design, assembly modelling, process planning and assemblability evaluation has been developed, but the assemblability evaluation is considered when detailed design is complete. An expert system [123] for concurrent product design and assembly planning integrating design, design for assembly evaluation, assembly system design, assembly process planning and simulation, and technical-economics justification has been described, see Figure 4.1. In this system, the DFA analysis is conducted toward the end of conceptual design and the assembly planning is not started until detailed design has been created.



(a) CE-ES Expert System

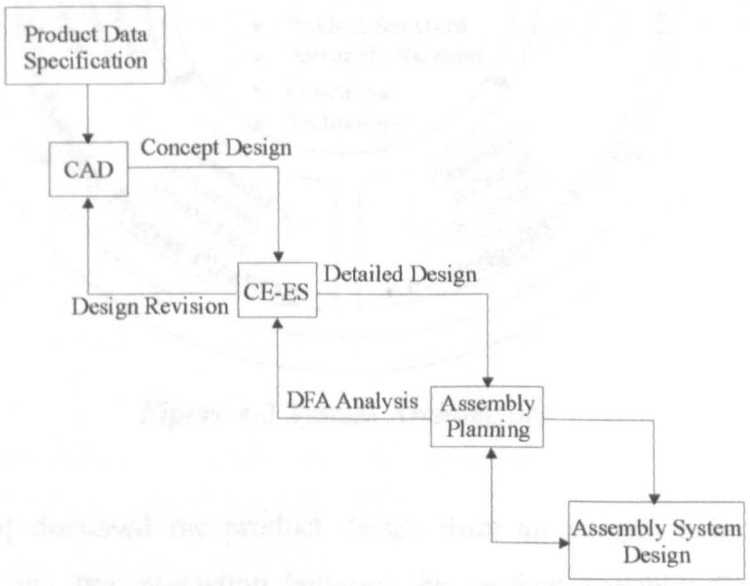


Figure 4.1 CAD-DFA-AP Integration System (CE-ES Is Detailed In (a))

Compared to the traditional assembly process planning practice which is sequential, and work-separated, simultaneous (a truly concurrent) product design and assembly process planning requires a parallel, interactive, and integrated approach. Both design and assembly process knowledge is necessary. Assembly process planning knowledge and operation-specific knowledge are the most important. New computer-aided tools supporting a simultaneous approach must be developed to improve the cooperation and communication. Intelligent decision support must be provided. A suitable product model has to be established to represent component interactions, product and process data.

A knowledge-based system to support concurrent product design and assembly planning has been developed [125]. Its emphasis is on developing an integrated product and process model as Figure 4.2 from [125].

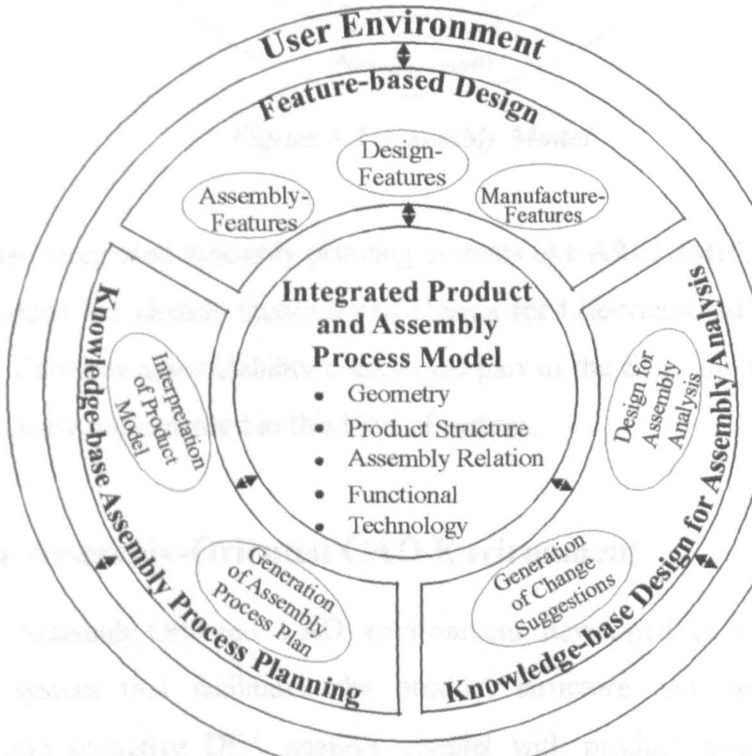

User Environment
Feature-based Design
Design-Features
Assembly-Features
Manufacture-Features
Integrated Product and Assembly Process Model
• Geometry
• Product Structure
• Assembly Relation
• Functional
• Technology
Knowledge-base Assembly Process Planning
Interpretation of Product Model
Generation of Assembly Process Plan
Knowledge-base Design for Assembly Analysis
Design for Assembly Analysis
Generation of Change Suggestions

Figure 4.2 System Architecture

Rampersad [126] discussed the product design from an integral point of view to improve insight into the interaction between the product variables for developing product design for ease of robotic assembly. An assembly model [126] is presented to represent the interactions between a collection of assembly variables Figure 4.3.

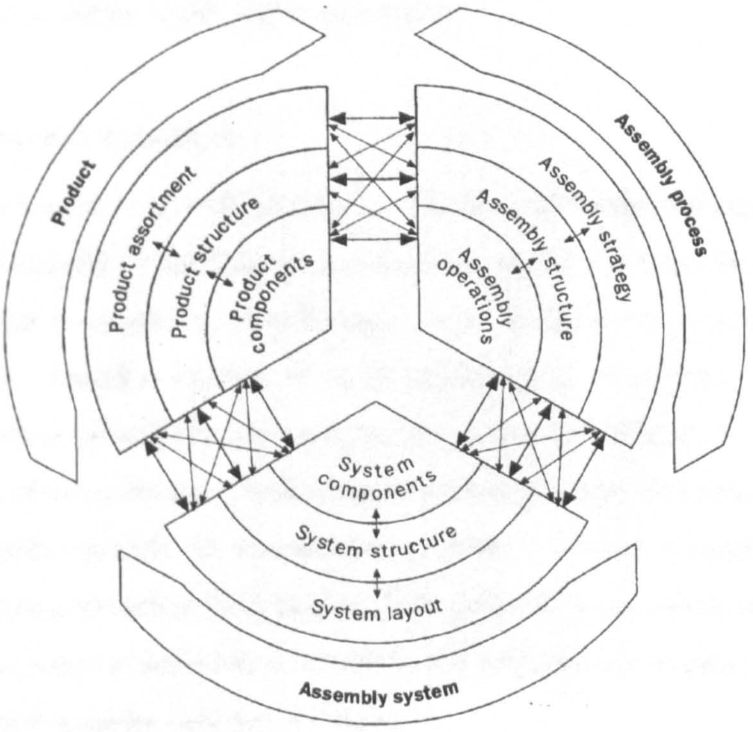


Figure 4.3 Assembly Model

There are also integrated assembly planning systems like ARCHIMEDES mentioned in chapter 3, which has already incorporated Design-for-Lifecycle-Cost Analysis module [105, 113]. Certainly assemblability contributes part of the cost, and DFA analysis can be more explicitly implemented in this kind of system.

4.2 Ophir Assembly-Oriented CAD Environment

The Ophir Assembly-Oriented CAD environment developed is a truly integrated concurrent system that facilitates the product structure and assembly sequence generation and proactive DFA analysis parallel with product design process. As product modelling and reasoning support are required for automatic inference of DFA and sequence data, research in the definition of 'Assembly-oriented CAD' in the Ophir Project focuses on the following four areas:

- Proactive DFA analysis
- Assembly Sequence Generation

- Product Modelling and Geometric Reasoning Support
- Knowledge-based Expert Support

Each of the above will be briefly introduced below.

4.2.1 Proactive DFA Analysis

Proactive DFA is concerned with integrating process knowledge and experience within the design environment to facilitate the generation of design solutions that are amenable to assemble and economic to manufacture. It is in some ways analogous to the automatic spell correction systems in some current word processors. Here, as you type, each word is proactively analysed and any mistakes highlighted. Compared to earlier version of the software, which required periodic or final checking, the proactive method highlights instantly an encountered problem. Instead of applying language grammar in writing, proactive DFA applies DFA grammar in the design process to alert the designer to potential problems in assembly and manufacture as product designs are being created and suggests redesign solutions.

The proactive approach is adopted to improve the application of current DFA methodologies. This is because even though there are many proven success of industrial applications over a number of years [8, 9, 10] as reviewed in chapter 2, there is one of the criticisms of DFA that it is essentially a reactive tool, requiring much detailed information and only provides useful results when applied to existing products or completed design [127].

In the Ophir environment, the proactive DFA evaluation is based on an assembly sequence. When the sequence is concurrently generated with product design, a proactive DFA analysis is complete which provides a quantitative view on the assemblability of the product in relation to the amount of available data. In this process, problems are highlighted instantly. Suitable actions can be taken immediately to improve the design and sequence from the earliest stage of product design. The proactive DFA evaluation [127] operates at three levels in the Ophir environment as in Figure 4.4:

- Product Group Support

- Product Structure Support
- Component Detailed Design Support

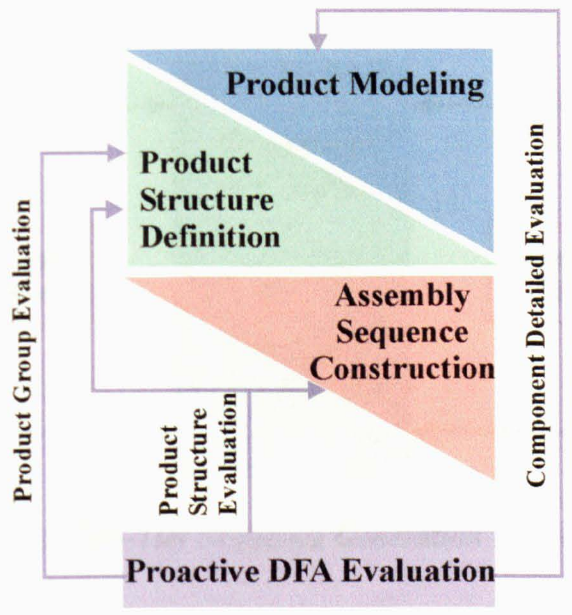

Product Modeling
Product Structure Definition
Assembly Sequence Construction
Proactive DFA Evaluation
Product Group Evaluation
Product Structure Evaluation
Component Detailed Evaluation

Figure 4.4 Proactive DFA Support

4.2.2 Product Structure Definition and Assembly Sequence Generation

A two-tier methodology, as Figure 4.5, has been implemented for concurrent generation of assembly sequence [128].

The *Structure Definition* tier is the top tier of this methodology to provide support for the early definition of the product structure and partitioning of subassemblies based upon the function structure of the product. An assembly hierarchy is used to document the components and subassembly partitions. Product structure optimisation can be achieved in this tier. It is believed that it is necessary to consider product structure optimisation before considering individual component optimisation because only limited assemblability improvements can be realised by optimising individual components [5]. This top-down manner [14] is essentially required to support for design process [129]. The product structure definition can be completed following defined procedure.
The lower tier, *Sequence Construction* tier, comprises the necessary tools to build the

[14] From assembly to individual components

sequence. In this tier, all the parts in assembly hierarchy are listed in a holding area to wait for the interactive construction of assembly sequence. *Mating Joint Type, Joining Process* and *Assembly Action* can be added to the sequence liaison to explore more detailed assembly plan.

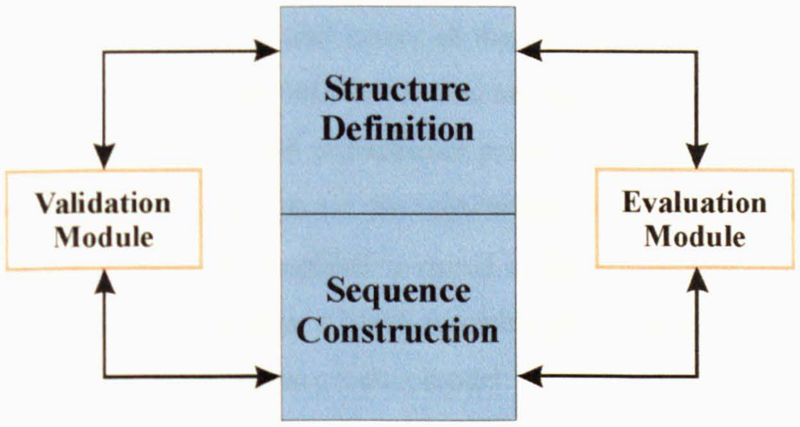


Figure 4.5 Two-Tier Sequence Generation Methodology

The validation and evaluation modules guarantee the sequence generated feasible and practical based on integrated hard and soft constraints, and sequence evaluation criteria. An interactive generation procedure Figure 4.6 as that recommended in chapter 3 is used. Decision support is provided, and part icons are used to ease of the interactive product structure and sequence definition process.

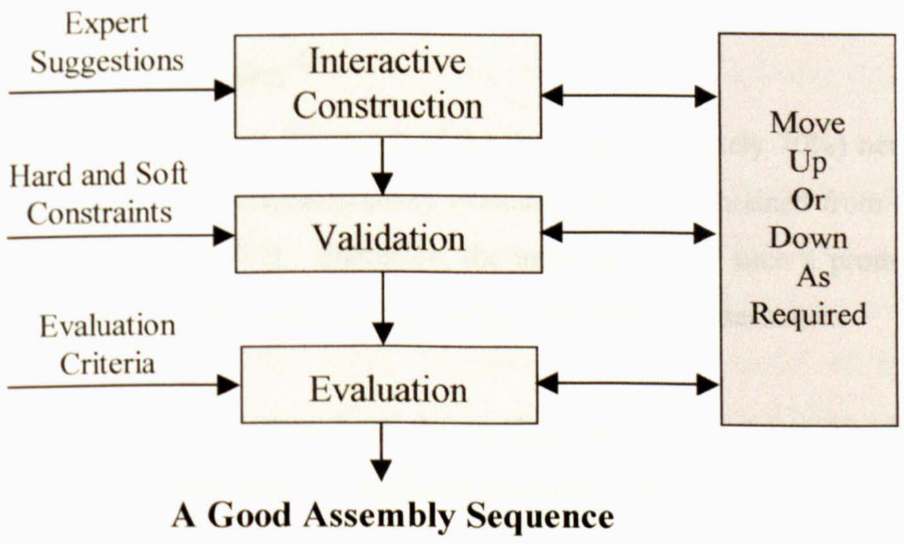


Figure 4.6 Interactive Sequence Generation Procedure

4.2.3 Product Model

To support concurrent sequence generation and proactive DFA evaluation, a product model is required to extend beyond CAD model to represent product data including assembly relations. In the Ophir environment, the underlying data structure – Four Layer Product Model, represents and stores all the relevant product data: component and assembly models, component interaction data, and assembly plans. Some part-level attributes, such as part material and manufacture process, probably need to be included in the component model. Data about the adjacencies between the components and details of how the product is assembled is stored in the component interaction layer. For the purpose of DFA evaluation, a rough assembly plan may also need to be stored. There are four different layers in the product model:

- *Component Model* - solid model (ACIS 'sat' file), enhanced with component attribute information such as surface finish, material, process, etc.
- *Final Assembly Model-* position and orientation information for each instantiated component within the assembled product.
- *Component Interaction Model* – details of the component mating faces, their method of assembly and other liaison attributes.
- *Assembly Plan Model* – temporal data of sequence of component assembly operations including non-assembly processes.

4.2.4 Geometric Reasoning [15]

Previous research has shown that most of the data (approximately 70%) necessary to drive assemblability and manufacturability evaluation may be obtained from a suitably enhanced product model [17]. Therefore, the interrogation of such a product model represents a highly efficient means of supporting a DFA/DFM assessment.

The extraction of relevant data from the product model may be a straightforward matter if the required information is explicitly presented. Where it is not presented or is

[15] Geometric Reasoning is the process by which application requests, related to geometric properties and attributes, are satisfied from a geometric model using both inferential and algorithmic methods [17].

in an implicit form some type of Geometric Reasoning is required to answer the model interrogation.

Opportunities exist to identify assembly-related issues, such as a collision-free path, subassembly stability, insertion trajectories and component characteristics related to DFA analysis such as shape complexity, cross section properties, symmetry, and major and minor axis. Detailed work on detection of symmetry and primary axes in support of proactive DFA analysis has been presented in a paper [130].

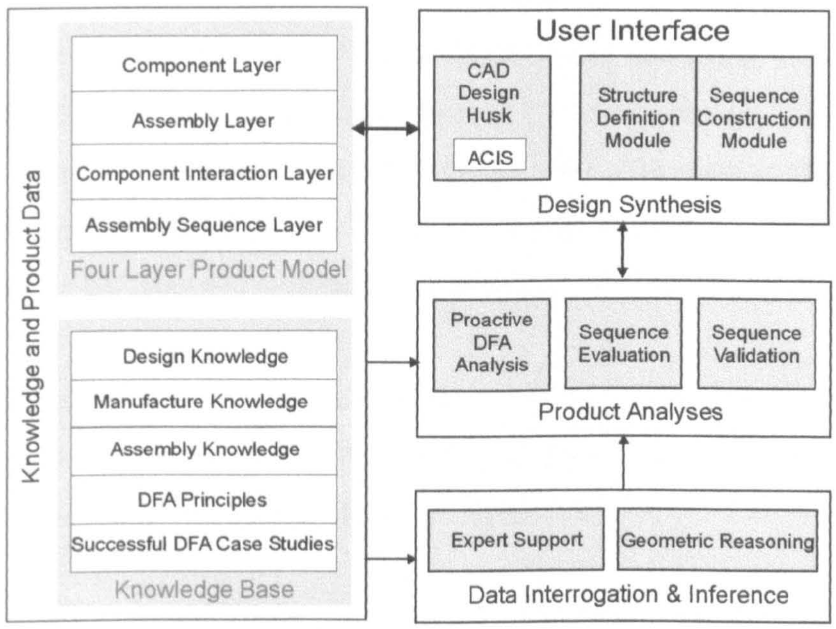


Figure 4.7 System Architecture of the Ophir Assembly-oriented CAD

As it is the intention of the approach to provide advice at an early stage of the design process there will be situations where the necessary geometric information is uncertain or not available. Expert support may be used to deploy heuristic knowledge about components to provide 'best guess' answers for product model interrogations. Heuristic methods can also replace geometric reasoning algorithms where an approximate answer provided quickly is more useful than a precise answer produced by lengthy computation. Most importantly, it is always useful to provide the designer with decision support based on relevant best practice, as design is a creative activity built on established knowledge and experience. Figure 4.7 shows how the expert support fits into the Ophir Assembly-oriented CAD environment to automatically infer DFA and sequence data from product model based on the knowledge incorporated. The expert support is the focus of the thesis. It will be detailed from the next chapter.

Chapter 5 Expert Support for the Ophir Environment

In this chapter, an appropriate strategy for the expert support will be choosen. Several AI techniques which are useful for the implementation of the support will be reviewed. Suitable knowledge representation methods and reasoning tools will be selected. Finally, an '*Expert Assembler*' will be created and deployed in the Ophir environment.

5.1 A Strategy for Expert Support

Industrial assembly planning investigation shows that generating a good assembly sequence has a strong heuristic nature [16], and proactive DFA analysis is a knowledge intensive activity. The Ophir assembly-oriented CAD environment facilitates the definition of an assembly sequence and the proactive DFA evaluation concurrent with the product design process. However, designers are rarely assembly experts, and they often need support for assembly sequence definition and proactive DFA evaluation.

There are many aspects in this assembly-oriented CAD environment that can benefit from a knowledge-based approach. After detailed analysis, some main areas requiring expert support have been identified as shown in Figure 5.1.

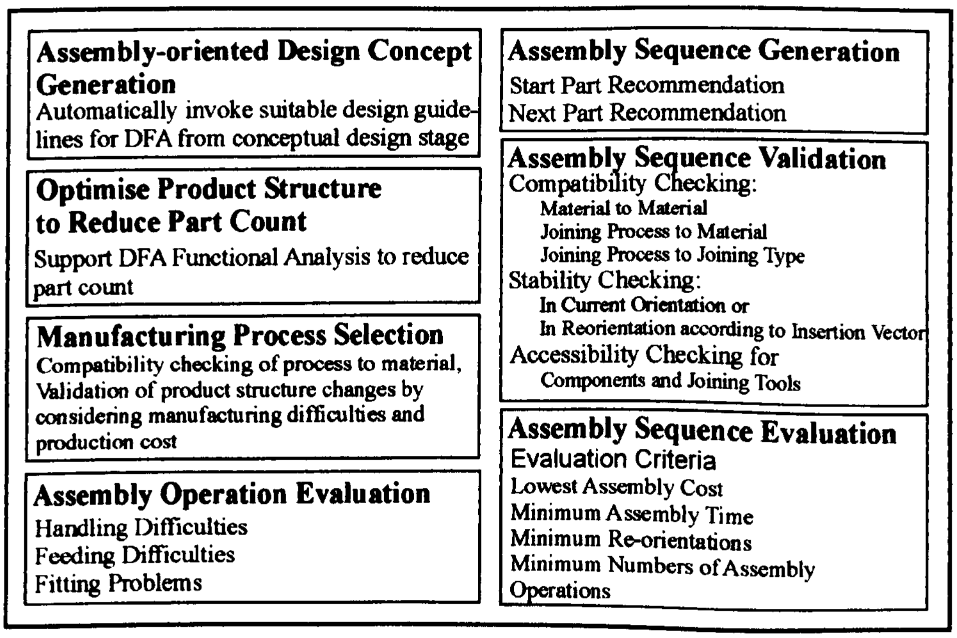

Assembly-oriented Design Concept Generation
Automatically invoke suitable design guidelines for DFA from conceptual design stage
Optimise Product Structure to Reduce Part Count
Support DFA Functional Analysis to reduce part count
Manufacturing Process Selection
Compatibility checking of process to material, Validation of product structure changes by considering manufacturing difficulties and production cost
Assembly Operation Evaluation
Handling Difficulties
Feeding Difficulties
Fitting Problems
Assembly Sequence Generation
Start Part Recommendation
Next Part Recommendation
Assembly Sequence Validation
Compatibility Checking:
Material to Material
Joining Process to Material
Joining Process to Joining Type
Stability Checking:
In Current Orientation or
In Reorientation according to Insertion Vector
Accessibility Checking for
Components and Joining Tools
Assembly Sequence Evaluation
Evaluation Criteria
Lowest Assembly Cost
Minimum Assembly Time
Minimum Re-orientations
Minimum Numbers of Assembly Operations

Figure 5.1 The Areas Requiring Expert Support

In this figure, there are several areas regarding DFA and the assembly sequence definition requiring expert support, but different areas have very different requirements. The knowledge involved is also different, which is mainly:

- the CSC technique - the basis of the proactive DFA evaluation,
- the sequence generation heuristics extracted from knowledge engineering and industrial case study analysis - the knowledge source for the assembly sequence definition,
- successful DFA case studies and
- process selection knowledge [131].

Changes made in one area may influence another area. In many cases, the support is required concurrently - the nature of the Ophir environment is concurrent. For these reasons, it is very difficult to adopt a single approach, such as a dominant knowledge-based expert system to support the Ophir environment. It may be more appropriate to implement several separate expert modules: each addressing a particular problem area and using a suitable problem solving strategy and knowledge representation. The rule-based approach can provide a convenient means to represent process knowledge, assembly expertise, and new knowledge can be appended easily to the knowledge base. The case-based approach is helpful in situations where a number of case studies can be employed to support the design decisions. Procedural and combined implementations may also be useful to deploy heuristics. Thus the strategy of expert support proposed [132] is to deploy an '*Expert Assembler*' which contains several expert modules and each tackles a problem area. Three expert modules are mainly required to support for the proactive DFA evaluation and the assembly sequence definition in the Ophir environment. They are:

- *Product Structure Expert* - to give guidance on minimising variants and part count, and fully utilising the benefits of modules and standard parts.
- *Assembly Sequence Expert* - to give guidance on assembly sequence definition.
- *Assembly Process Expert* - to provide checks on assembly operations, for example, to identify insertion problems.

We call an expert module an expert, and expert modules experts. Each of the expert modules may contain several elements (expert elements) to fulfil closely related tasks.

Furthermore, to increase the efficiency and reduce subjective user inputs, each of the expert modules should be self-contained and be able to detect and update relevant information automatically. It is also desirable that the expert modules can cooperate with each other and with the user. All these characteristics indicate that each expert module behaves like an **'intelligent agent'** in some ways except that it may not take actions directly to the product design. Instead, the *Expert* provides suggestions and the final actions are left to the user. This is because the focus of the approach adopted by the Ophir Project is to deploy the knowledge-based experts as decision support tools rather than as decision-makers.

5.2 Intelligent Agents

Intelligent agents are one of the most important developments in computer science in the 1990s. Agents are of interest in many important application areas, ranging from human-computer interaction, information retrieval and filtering, to industrial process control and air traffic control. In this section, the concept of the *intelligent agent*, the agent theories, architectures, and languages will be introduced. The differences between *agents* and *objects* in object-oriented programming, and the differences between *intelligent agents* and *expert systems* will be specified. The emphasis of this section is on what are the intelligent agent properties required of the expert modules, and how the properties could be implemented.

5.2.1 The Concept of the Intelligent Agent

The concept of an *agent* is defined in reference [133]:

> an *agent* is a computer system that is *situated* in an *environment*, and that is capable of *autonomous action* in this environment in order to meet its design objectives.

The autonomy here means that the agent is able to act without the intervention of humans or other systems: they have control both over their own internal states, and over their behaviours.

The concept of the *agent* has become important in both mainstream computer science and in artificial intelligence. An *intelligent agent* is one that is capable of *flexible* autonomous action in order to meet its design objectives, and flexible implies three properties [133]:

- *reactivity*: intelligent agents are able to perceive their environment, and respond in a timely fashion to changes that occur in it in order to satisfy their design objectives;
- *pro-activeness*: intelligent agents are able to exhibit goal-directed behaviour by *taking the initiative* in order to satisfy their design objective;
- *social ability*: intelligent agents are capable of interacting with other agents (and possibly humans) in order to satisfy their design objectives.

These properties are more demanding than they might at first appear. Consider *pro-activeness*: goal directed behaviour, it is not hard to build a system that exhibits goal directed behaviour by writing a procedure in PASCAL, a function in C, or a method in JAVA. But for some systems, the goal directed programming is not acceptable, as it makes some assumptions. In particular, it assumes that the environment *does not change* while the procedure is executing.

In many environments, neither of these assumptions is valid. In particular, in domains that are *too complex* for an *agent* to observe completely, that are multi-agents (i.e., they are populated with more than one agent that can change the environment), or where there is *uncertainty* in the environment, these assumptions are not reasonable. Thus, in complex dynamic environment, blindly executing a procedure without regard to whether the assumptions underpinning the procedure are valid is a poor strategy; an *agent* must be *reactive*, it must be responsive to events that occur in its environment. So it is required to build a system that achieves an effective *balance* between the goal-directed and the reactive behaviours.

We want the *agents* to attempt to achieve their goals systematically, but we don't want the *agents* to continue when the procedure will not work, or when the goal is for some reason no longer valid. We want the *agent* to react to the new situation, but we do not want the *agent* to be continually reacting, and hence never focusing on a goal long enough to try to achieve it. The balance is hard for a computer system, but humans are very good at this. If these two properties are required for the expert modules in the

Ophir environment, an advantage is that the balance can be achieved by human/computer cooperation.

For some researchers -- particularly those working in AI, the term "intelligent agent" has a stronger and more specific meaning [134]. That is besides the properties identified above, the *agent* is either conceptualised or implemented using concepts that are more usually applied to humans, such as using mentalistic notions (knowledge, belief, intention, and obligation). But this is not the emphasis in this project as the target is not to try to create the human like *intelligent agents* but to deploy the *Expert Assembler* mentioned previously that contains several expert modules: each has some of the properties associated with an *intelligent agent*. The most concerned issues are what properties each expert module should have, and how to implement them.

5.2.2 Agents and Objects, Agents and Expert Systems

In object-oriented programming, *objects* are defined as computational entities that encapsulate some state, are able to perform actions or methods on this state, and communicate by message passing. While there are obvious similarities, there are also significant differences between *agents* and *objects*. Agents are different from objects in that they have:

- stronger kind of autonomy - no public functions,
- flexible autonomous behaviours (the standard object model has not this type of behaviour),
- multi-threaded programming (in the standard object model, there is a single thread of control in the system).

Expert systems were the most important AI technology of the 1980's, and they were capable of solving problems or giving advice in some knowledge-rich domain. The important distinction between *agents* and *expert systems* is that *expert systems* normally do not interact directly with any environment: they use humans as a media to get information about their environment, and to take action to their environment. This will become clearer after the detailed review of the expert system technique later.

5.2.3 Required Intelligent Agent Properties of the Experts

Before analysing the required properties of the experts in the Ophir environment, it should be made clear that the focus of the approach adopted is that all the final decisions regarding product structure, assembly sequence, or assembly operations will be left to the user, but the expert support should provide intelligent assistance as much as possible. The role of the experts is to guide the assembly sequence definition, to automatically infer DFA related properties, to highlight assembly problems instantly, and to provide improvement suggestions if possible. So the properties required of the experts are: detecting relevant information automatically, reasoning about the detected information, updating the environment data, feeding back the reasoning results in user friendly ways. These clearly include perceiving environment and goal-directed behaviour.

The environment information perceived by the experts will change under two circumstances:

1. When actions have been taken upon the Ophir environment by the user;
2. When environment data has been updated by an expert.

In each situation, notification of changes should be sent to the related experts automatically. Otherwise automatic-change-detecting is required of the experts. These imply that co-operation is required between the experts themselves and the user.

In summary, the properties required of the experts are perceiving environment, goal-directed behaviour, good communication and co-operation between the experts themselves and the user.

Here it is worth mentioning that the most important advantage over a traditional knowledge-based expert system to deploy the agent-related expert modules is in their autonomy, in their perceiving environment property. Because of this, inference can be made continuously 'in the background' of the Ophir environment from available information Figure 5.2. Thus the user is relieved from the extra burden of responding to many assembly-related questions concerning the design and its components.

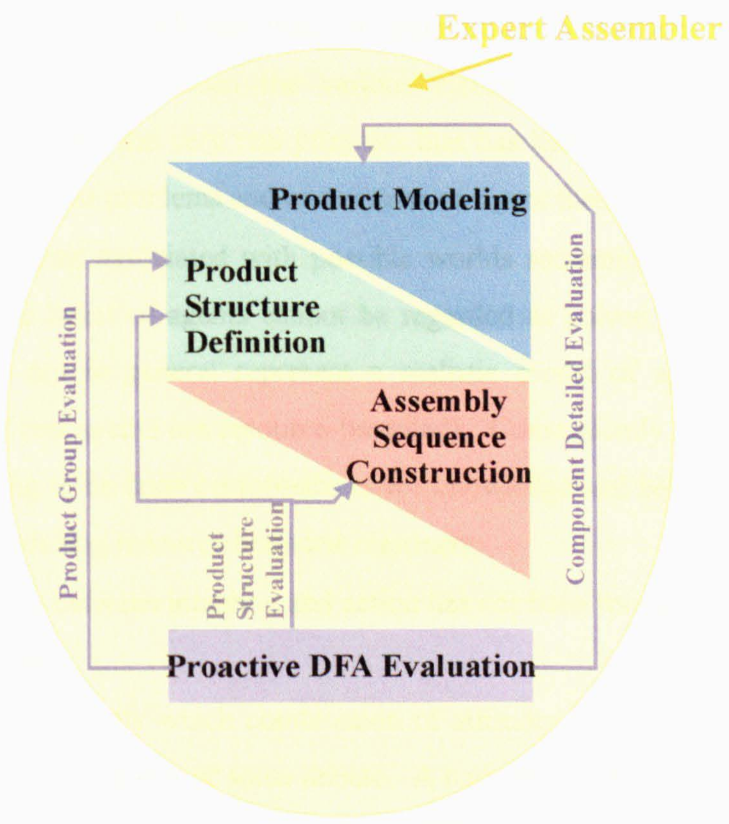

Expert Assembler
Product Modeling
Product Structure Definition
Assembly Sequence Construction
Proactive DFA Evaluation
Product Group Evaluation
Product Structure Evaluation
Component Detailed Evaluation

Figure 5.2 The Expert Assembler Working in the Background

5.2.4 Agent Theories, Architectures, Languages and Other Issues

Intelligent agent brings benefit in that

- Stand-alone applications can be made to provide "value-added" services by enhancing them in order to participate and interoperate in cooperative heterogeneous set-ups;
- The legacy software problem may be ameliorated.

Agent-based software engineering provides a radical new approach to software designs.

There are clear explanations and discussions about agent theories, agent architectures, agent types, agent languages, and agent applications in references [133, 134, 135]. Here, some of the topics will be briefly introduced.

5.2.4.1. Agent Theories

Formalisms for reasoning about agents have come a long way since Hintikka's pioneering work on logics of knowledge and belief [136]. Within AI, perhaps the main emphasis of subsequent work has been on attempting to develop formalisms that capture the relationship between the various elements that comprise an agent's cognitive state. Despite the very real progress that has been made, there still remain many fairly fundamental problems and open issues on agent theories.

- First, the problems associated with possible worlds semantics [16] for logics of the knowledge and belief of agents cannot be regarded as solved, as possible worlds semantics do not in general represent a realistic model of agents with limited resources (all real agents are resource-bounded). Consequently, there is still much work remaining to be done on formalisms for knowledge and belief, in particular in the area of modelling resource bounded reasoners.
- The relationship between intention and action has not been formally represented in a satisfactory way.
- The question of exactly which combination of attitudes is required to characterise an agent is also the subject of some debate. A current popular approach is to use a combination of beliefs, desire and intentions (BDI architectures [137])

5.2.4.2. Agent Architectures

There are three approaches in agent architecture: deliberative, reactive and hybrid [134].

1. Classical approaches: deliberative architectures

A deliberative agent or agent architecture is defined to be one that contains an explicitly represented, symbolic model of the world, and in which decisions are made via logical reasoning, based on pattern matching and symbolic manipulation.

Since the early 1970s, the AI planning community has been closely concerned with the design of artificial agents; various attempts have been made to construct agents whose primary component is a planner. Much effort has been devoted to developing effective techniques, such as *hierarchical* and *non-linear* planning. However, in the mid 1980s,

[16] The possible worlds model for logics of knowledge and belief was originally proposed by Hintikka [136].

Chapman established some theoretical results which indicate that even such refined techniques will ultimately turn out to be unusable in any time-constrained system [138]. Researchers like Agre and Chapman [139] have challenged the usefulness of having elaborate plans in a dynamic environment, they may more than others have led to the work on alternative approaches.

2. Alternative approaches: reactive architectures

The reactive architecture is defined to be one that does not include any kind of central symbolic world model, and does not use complex symbolic reasoning. Reactive agents represent a special category of agents, they act/respond in a stimulus-response manner to the present state of the environment in which they are embedded.

The work of reactive agents dates back to research such as Brookes [140] and Agre and Chapman [139]. Brooks has built a number of robots, based on the subsumption architecture. A subsumption architecture is hierarchy of task-accomplishing *behaviours*. Each behaviour "competes" with others to exercise control over the robot. Lower layers represent more primitive kinds of behaviour (such as avoiding obstacles), and have precedence over layers further up the hierarchy. The resulting systems are, in terms of the amount of computation involved, extremely simple, with no explicit reasoning of the kind found in symbolic AI systems. But despite this simplicity, Brookes has demonstrated the robots doing tasks that are impressive even if they were accomplished by symbolic AI systems. Similar work has been reported by Steel [141]. Their work demonstrates that intelligence can derive from the emergent behaviour of the interactions of the various modules.

Compare the two approaches mentioned above, symbolic AI agents have an attractive proposition as reactive systems have as yet little associated methodology. It is not obvious how to design reactive systems so that the intended behaviour emerges from the set-up of agents. Some researchers have suggested that techniques from the domain of genetic algorithms or machine learning might be used to get around these development problems. There are relatively few number of reactive software agent-based applications; currently they are mainly limited to games and simulations.

3. Hybrid architectures

Many researchers suggested that neither a completely deliberative nor a completely reactive approach is suitable for building agents. They have argued the case for hybrid systems, which attempt to marry classical and alternative approaches.

An obvious approach is to build an agent out of two (or more) subsystems: a deliberative one, containing a symbolic world model, which develops plans and makes decisions in the way proposed by mainstream symbolic AI, and a reactive one, which is capable of reacting to events that occur in the environment without engaging in complex reasoning. This kind of structuring leads naturally to the idea of a layered architecture, of which TouringMachines [142] and InteRRaP [143] are good examples. Layered architecture is currently the most popular agent architecture. A key problem in such architecture is what kind of control framework to embed the agent's subsystems in, to manage the interactions between the various layers.

5.2.4.3. Agent Languages

The need of software support tools for the design and construction of agent-based systems was identified as long ago as the mid-1980s [144]. Rule-based systems have a number of features which appeared to make them particularly natural for constructing intelligent agent:

- Rules can fire in response to patterns of data and can therefore respond directly to observed states or events in the environment. -- capability for adaptive behaviour.
- The clearly separating control mechanisms from task knowledge--a fixed architecture for selecting and firing rules, provides an adequate control structure for applying a variable body of knowledge expressed in condition-action form.

Different agents could be implemented using different programming languages and techniques.

The emergence of a number of prototypical agent languages is one sign that agent technology is becoming more widely used, and that many more agent-based applications are likely to be developed in the near future. The release of TELESCRIPT, a commercial agent language is particularly important, as it potentially makes agent technology available to a user base that is industrially oriented.

5.3 AI Techniques for Expert Support

From the properties required of the expert modules, the knowledge involved and the requirements of the support areas, it is felt that besides procedural programming, some AI techniques, such as knowledge-based expert systems, case-based reasoning, heuristic search, and constraint satisfaction evaluation, are useful for the implementation of the experts in the Ophir environment. These techniques will be reviewed below.

5.3.1 Knowledge-based Expert Systems

Knowledge-based approach has been generally adopted in many DFA evaluation framework and systems [24, 36, 37, 17, 47, 49, 50, 51, 52, 6, 53, 145], in assembly sequence generation [88, 85, 92], assembly-oriented design [122, 123, 125] and other concurrent design system [146, 147]. Knowledge-based expert systems have a wide range of applications which include interpretation, prediction, diagnosis [148], planning [149], scheduling [150], debugging, control [151] and design [152, 153]. They have received rather obvious and quite prominent acceptance during the 1980s and the early 1990s as a way to apply large amount of knowledge to solve problems emulating human experts.

Professor Edward Feigenbaum of Stanford University, an early pioneer of expert systems technology has defined an expert system as:

> *An intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution [154]*

That is, an expert system is a computer system which emulates the decision-making ability of a human expert.

Artificial intelligence scientists are generally agreed that knowledge is the essential ingredient in intelligence even some sophisticated reasoning techniques are very useful.

As a computer, despite its enormous capacity for data storage, simply could still not store all of the knowledge necessary to simulate all facets of human intelligence, it might be possible to at least store that knowledge associated with a narrow domain – and then exhibit abilities on the computer similar to that of a human for this restricted area. So they give name to the more focused, directed toward a narrow sector of expertise – rather than general, overall intelligence, the subfield of AI "expert system".

The first generation of expert systems were rule-based. This was based on the discovery that human problem solving or cognition that could be expressed by IF THEN type rules demonstrated in the General Problem Solver program created by Newell and Simon in 1960s. The rule takes the form of:

```
IF:
        condition 1, ...condition n
THEN:
        conclusion 1, ...conclusion n
        action 1, ...action n
```

It infers conclusions from conditions, and it may take actions.

For example:

```
IF:
        Mary and Lisa have the same parents and
        Mary is female and
        Lisa is female
THEN:
        Mary and Lisa are sisters
```

Rules like these are called *production rules*. The Newell and Simon model of human problem-solving in terms of long-term memory (rules), short-tem memory (working memory), and a cognitive processor (inference engine) is the basis of rule-based expert systems, which are also called *production systems*. The term "expert system" is often used synonymously with "knowledge-based system" or "knowledge-based expert system". The "expert" part of the name is an aspiration as even the performance of the system may be able to compete with a human expert, but performance is only one dimension of human behavior. Other behaviors of the human expert, such as learning, breaking rules, reformulating, graceful degradation are still very difficult to attain for a computer program. "Knowledge-based System" is a more technically relevant term. There is a point of view which considers expert system as a subset of knowledge-based system [155, page 238].

The basic concept of the knowledge-based expert system can be illustrated as Figure 5.3: it contains *Knowledge Based*, *Working Memory* and an *Inference Engine*. When requirements and information are input into the Expert System, the *Inference Engine* reasons about the asserted facts in the *Working Memory* based on the knowledge stored in the *Knowledge Base*, and expert advice is fed back. The *Knowledge Base* can be updated easily.

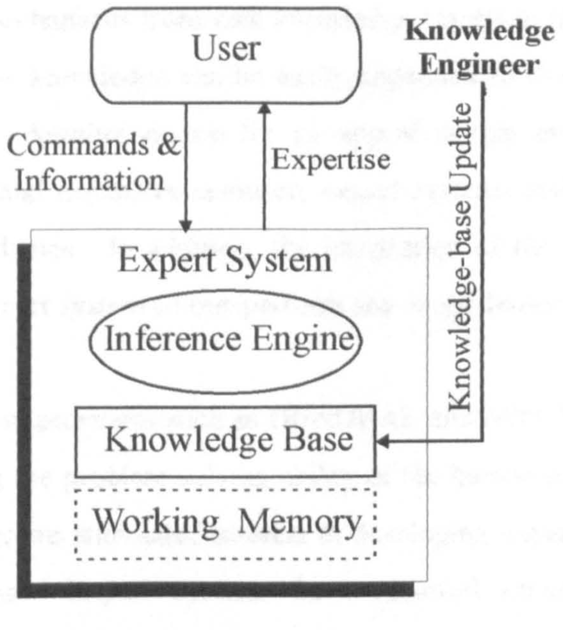


Figure 5.3 Knowledge-based Expert System

Before the development of the expert systems, heuristic search that deploys heuristics in a search procedure has already been used for a long time. But the primary enhancements brought to a heuristic search to form an expert system are not trivial. The separation of the inference process (the solution procedure) from the knowledge base (the heuristics) is a concept which at first glance might seem minor, but this separation has resulted in the ability to focus the implementation efforts much more intensely toward the development of the knowledge base model – rather than have to spend an inordinate (and generally unnecessary) amount of time and resources in the development and implementation of the entire solution, and in particular, the inference process. Since it is the knowledge that determines the outcome, whether one uses algorithms, heuristic search, or expert systems, it should be apparent that the knowledge is the area in which the bulk of our efforts we should be devoted. So expert systems can not be simply considered as another format for the implementation of heuristic search. A comparison among algorithmic procedure, heuristic search and expert system in general has been presented in reference [156, page 44].

Expert systems are a relatively new concept [157], with the earliest such system developed in the late 1960s. They are attractive because of their conceptual simplicity and the ease with which the rule-interpreter can be implemented. Also the clear separation of control mechanisms from task knowledge results in the modularity of the knowledge base, so new knowledge can be easily appended to the knowledge base as condition-action rules. Another reason for its appeal is that even though "expert" knowledge is a scarce and expensive resource, expert systems make expert behaviour available to a large audience. In addition, the integration of the expertise of several experts may lead the expert system to out-perform any single human expert.

Some early expert system programs such as DENDRAL and MYCIN demonstrated the possibility of simulating the problem-solving ability of the human experts. The success of these and other programs stimulated interest in developing expert systems for a vast number of applications. Expert systems have received obvious and prominent acceptance during the 1980s. They have been brought out of the university laboratories and have produced commercial products.

As the experience with expert systems increased, it quickly became apparent that production rules were inappropriate for some types of knowledge representation, thus *schemata*, known as *frames*, have also been used to represent knowledge, and *object-oriented* and *access-oriented* programming are used to perform inference.

A *frame* is a set of information relating to a particular subject, and the information is stored in slots that associate methods. *Frames* are often represented in a semantic network, which is widely used for representing knowledge of relationships (Figure 5.4).

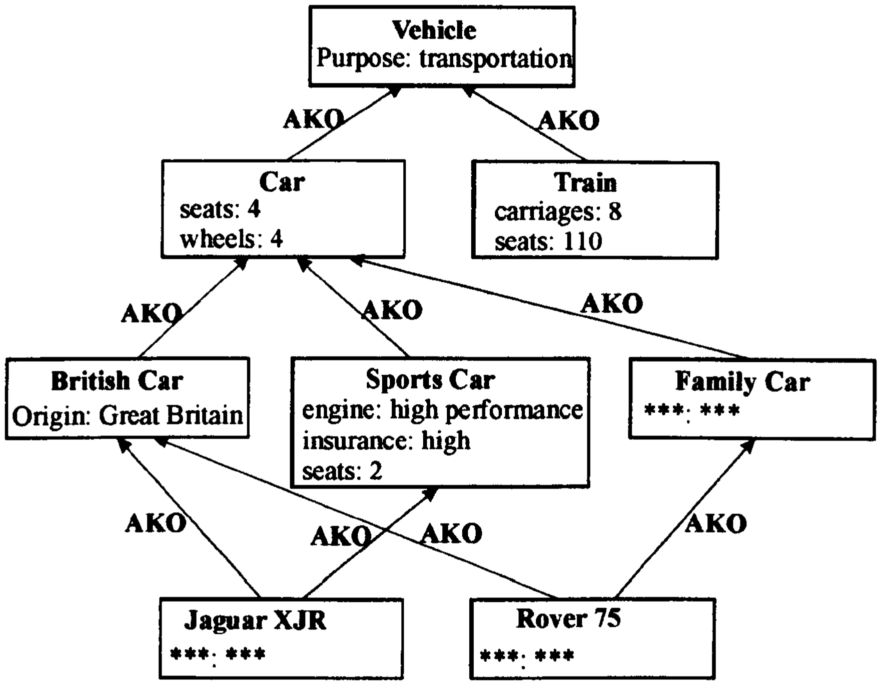


Figure 5.4 Frames in Semantic Network with A-Kind-Of (AKO) Links

Object-oriented programming used in expert systems is a method of drawing inferences based on a schema-based representation of knowledge. An *Object* is a *schema*, some of whose slots contain pointers to procedures (known as *methods*) written in a programming language. This allows methods which are most relevant to a particular schema to be linked to the slots within that schema. The object-oriented programming is useful for creating a tidy and modular knowledge base. Inference occurs by sending a message to the slot to which a method is attached, this causes that method to be activated.

Access-oriented programming refers to that some of the knowledge-based system toolkits allow the user to define functions which are executed whenever certain data is fetched or stored.

Besides these powerful abilities, friendly user interfaces have been built into some of the commerce expert system shells to make the development of expert systems quick and easy. This makes expert systems more appealing to some of the application developers.

After a period of optimism, some of the shortcomings of expert systems were exposed, which are:

1. The symbolic pattern matching used in most of the expert systems does not investigate the process of cognition, but rather describes the process of how to handle and manipulate descriptions of a reality.
2. The difficulty of knowledge explication; this becomes the bottleneck in the development of expert systems.
3. Expert systems do not remember the problems they solved as human experts do, so they are simply unable to reuse the solutions directly for the same problems encountered in future.

Because of these shortcomings and the big promise made as implied by the name, the assessments of expert systems have often been overly pessimistic since 1990. But the true status of an expert system should lie somewhere between -- that is expert system is a powerful, practical and eminently useful tool – but only when employed on the right problems, and in the right manner. It suggests that other forms of reasoning should be investigated to overcome the shortcomings. Also it is probably better to call each developed system a more technically relevant name, such as call it a knowledge-based system if there is not significant expertise involved.

5.3.2 Case-based Reasoning

Case-based Reasoning (**CBR**) is a technique that solves new problems by adapting solutions that were used to solve old problems. It reasons from experience. It is a fresh reasoning paradigm and an alternative for the design of expert systems in domains that may not be appropriate for other reasoning paradigms such as model-based reasoning. As a result, and because of its intuitive nature and close resemblance to human reasoning, **CBR** has attracted increasing interest both from those experienced in developing expert systems and from novices. In contrast to knowledge-based system that depends on an explicit model of the domain,

- **CBR** does not require an explicit domain model, so knowledge elicitation becomes a task of gathering case histories
- implementation is reduced to identifying significant features that describe a case; an easier task than creating an explicit model

- large volumes of case information can be managed in a database
- **CBR** can learn by acquiring new knowledge as cases, thus makes maintenance easier

Advantages for using case-based reasoning include:

1. Increasing efficiency in solving new problems (because relevant reasoning can be re-used rather than having to be re-derived)
2. Improving the quality of solutions (because prior cases can guide the reasoner towards successful alternatives and warn of prior problems to avoid)
3. Simplifying knowledge acquisition. **CBR** facilitates the knowledge acquisition process because cases can be stored and used without having a perfect domain theory; case acquisition can be achieved without analysing the interactions between individual factors in a case. New cases can be added easily. Also pre-existing case library may be available for many tasks to provide a starting point when building **CBR** systems.

Depending upon the task being performed, a case-based reasoning system may be able to function successfully using cases that have little internal structure. Some tasks may require cases that include more complete representations, or require cases can be manipulated at varying levels of granularity. The effectiveness of **CBR** systems depends upon their ability to retrieve the right case at the right time. Learning by storing new cases is fundamental to the case-based reasoning process: **CBR** systems learn by adding the results of current processing to their case libraries as cases for future use.

The work of Schank and Abelson in 1977 is widely held to be the origins of **CBR** [158]. Whilst the philosophical roots of **CBR** could perhaps be claimed by many, it was the work of Roger Schank's group at Yale University in the early 1980's that produced both a cognitive model for **CBR** and the first **CBR** applications based upon this model. Janet Koloner developed the first **CBR** system called CYRUS which was an implementation of Schank's dynamic memory model. The case memory model of CYRUS later served as the basis for several other **CBR** systems, including

MEDIATOR, CHEF, PERSUADER, CASEY and JULIA.

CBR tools have made the **CBR** theory practically feasible. Two examples are given. CBR-Express, produced by Inference Corp., is perhaps the most successful **CBR** product. It is specially tailored to a vertical market, that of help desks. CBR-Express has a simple case structure and uses nearest neighbour matching to retrieve cases. The key feature of CBR-Express is its ability to handle free-form text. The use of trigrams means that CBR-Express is very tolerant of spelling mistakes and typing errors such as letter transpositions. CBR-Express examines a user's free form text entry and matches this against case titles and descriptions. This results in the retrieval of a set of cases. A list of ranked solutions with likelihood values is generated from the cases, and the user is offered these along with a set of *questions*. Answers to these questions help narrow the number of cases that matched, leading to a more accurate solution that is presented to the user. In the event of a solution not being reached, CBR-Express closes the CBR cycle by using the concept of an *unsolved case*. If after searching on a problem description and answering several questions a successful match is not obtained, an entire transcript of the consultation can be saved as an unresolved case. The administrator (human expert) of the case base subsequently can find out what that case's solution was and modify the unresolved case to create a new case. Another example is the Easy Reasoner supplied with Eclipse by Haley Enterprise. It is a C library that provides CBR functionality. The Easy Reasoner provides nearest neighbour and inductive retrieval of records in a database. Once records have been retrieved they can be asserted as Eclipse objects for adaptation by its rule-base. Eclipse offers fully compatible C++ objects, and optimised forward chaining using the Rete algorithm.

CBR technique provides problem solving the solutions directly or a good starting point if there are relevant cases documented. In addition to the large amount of documentation work that may be involved at the beginning of case base development, such as the identification of valid cases, the organising and indexing of cases, a suitable mechanism has to be established to retrieve relevant cases. Where these may not be very difficult as there are already commercial **CBR** tools that support the case retrieval step, the adaptation of old problems to new problems to find solutions is generally not easy. A review of industrial case-based reasoning tools shows that commercial **CBR** tools can support the retrieval step, but are not well suited to perform the adaptation process [159]. Nevertheless, case-base reasoning has already been used in mechanical design, DFA evaluation and assembly sequence generation [160, 161, 162, 94, 95].

5.3.3 Heuristic Search

Heuristics [89, 90, 91, 85, 86], and heuristic search such as A* and AO* algorithm have been widely used in generation of good assembly sequences. Heuristics have been explained in foot note 3 of chapter 3. Now let us have a look what is heuristic search.

In search of a problem space or its graphic representation, different control strategies can be used. These include:

- Generate-and-test
- Hill climbing
- Breadth-first search
- Depth-first search
- Best-first search
- Problem reduction
- Constraint satisfaction
- Means-ends analysis

The uninformed search methods, such as breadth-first search and depth-first search, conduct exhaustive search which is not efficient. For many tasks it is possible to use task-dependent information to help reduce search space. Information of this sort is usually called heuristic information, and search procedures using the information are called ***heuristic search methods*** [66].

Heuristic search is a powerful tool for the solution of difficult problems, often used in cases for which more formal analytical methods (in particular, methods that develop optimal solutions) would prove less effective. In these cases, heuristics are then intended for use in obtaining acceptable solutions quickly instead of pursuing optimal solutions. Heuristic search may lack the rigor or credibility of algorithmic approaches, but in most cases heuristic programming does solve difficult problems (such as some

NP-complete problem [17]) that are far beyond the capabilities of the analytical techniques. As a more general-purpose method compared with expert systems, heuristic search uses less domain specific knowledge. So it does not have powerful reasoning ability as expert systems do. It belongs to so called ***weak methods***.

There are basically two methods of incorporating heuristic information about a state-space problem into a search procedure designed to solve that problem; these methods correspond to the use of "evaluation functions" and "generator functions". Evaluation functions can be used to evaluate and order the candidates in each search step to guide the search expands along those sectors of the frontier thought to be most promising. Generator functions can be used to generate the most promising candidate for expanding in each search step, but they are not used as often as the evaluation functions.

Here I give some simple explanations of the two most interesting heuristic search strategies: Best-first search and Means-ends analysis.
Best-first search is a way of combining the advantages of both depth-first and breadth-first search into a single method. At each step of the search process, the most promising candidate is chosen to generate its successors. Then the most promising successor is chosen to generate successors again until a solution is reached. A* and AO* algorithm used in automatic assembly sequence generation are best-first search.

Means-end is a valuable heuristic that may be employed in a variety of ways. The basic approach is to select operators that seem to provide a means to some end, most often the goal. The objective is to choose operators whose action moves the current system state closer (in some measure) to the goal state. This involves carefully choosing *state evaluation function*.

[17] NP-complete problem – for which nondeterministic polynomial-time algorithms are known, all known deterministic algorithms are exponential.

5.3.4 Constraint Satisfaction

Constraint satisfaction approach has considerable attraction both in AI and in other areas of computer science. Many problems can be viewed as problems of *constraint satisfaction* in which the goal is to discover some problem state that satisfies a set of constraints. These include product design activities - they must be within fixed time limits, available material and equipment; or the assembly sequence generation process - besides geometric constraints, it is restricted by current assembly technology and available equipment. For a constraint satisfaction problem, it is useful to augment the description of the problem state with a list of constraints that dynamically changes as pieces of the problem are solved, and to augment the search mechanism to manipulate that list of constraints.

A variety of techniques have been developed for finding partial or complete solutions for different kinds of constraint expressions, such as backtracking, constraint propagation and cooperative algorithms. These have been successfully applied to diverse tasks such as design, diagnosis, truth maintenance, scheduling, logic programming, and user interface. Constraint satisfaction has also been used in generation of good assembly sequence [72, 103, 80]. Constraint networks are graphical representations used to guide strategies for solving constraint satisfaction problems (CSPs).

Constraint satisfaction problems can mainly be classified as two kinds [163]. One kind of the problems called *Boolean constraint satisfaction problems* is those in which one has a set of variables, each to be instantiated in an associated domain and a set of Boolean constraints limiting the set of allowed values for specified subjects of variables. This general formulation has a wide variety of incarnations in various applications. It is generally a search problem.

The standard solution procedure for solving this kind of constraint-satisfaction problems is backtracking search. The algorithm typically considers the variables in order and, starting with the first, assigns a provisional value to each successive variable in turn as long as the assigned values are consistent with those assigned in the past.

When in the process, a variable is encountered such that none of its domain values is consistent with previous assignments (a situation referred to as a dead-end), backtracking takes place. That is, the value assigned to the immediately preceding variable is replaced, and the search continues in a systematic way until either a solution is found or until it may be concluded that no such solution exists.

Improving backtracking efficiency to reduce the size of its expanded space depends on the way the constraints are represented, the instantiation order of the variables, and how values are assigned to each variable when one solution suffices. Complementary algorithms called *consistency-enforcing algorithms* (also called *consistency algorithms*) have been developed to employ in advance of performing the backtracking search. These algorithms are most easily described in the network method of the constraint satisfaction problems. The best framework for understanding these algorithms is to see them as removing local inconsistencies from the network which can never be part of any global solution. When those inconsistencies are moved, they may cause inconsistencies in neighbouring arcs that were previously consistent. Those inconsistencies are in turn removed, so the algorithm eventually arrives, monotonically, at a fixed-point consistent network and halts. These transform a given constraint network into an equivalent, yet more explicit network. Intuitively, a consistency-enforcing algorithm will make any partial solution of a small subnetwork extensible to some surrounding network, and this is called constraint propagation.

Another kind of problems called *constraint optimisation problems* are the numerical optimisation problems that arise when one is designing a system to maximise the extent to which the solutions it provides satisfy a large number of local constraints. This is because in computational vision and other AI domains one is often not just satisfying a set of Boolean constraints but rather optimising the degree to which a solution satisfies a variety of conflicting continuous constraints. Algorithms that are variously known as *cooperative* or *probabilistic relation algorithms* for the solutions of this kind of problems are based on generalisations of *the consistency algorithms*. The main ideas in *cooperative algorithms* are that compatible values in neighbouring domains can cooperatively reinforce each other by increasing each other's weight. Simultaneously, incompatible values compete, trying to suppress each other. Each value in a domain is competing with each of the other values in that domain. The attraction of the *cooperative algorithms* is that they are highly parallel, requiring only local neighbourhood communication between uniform processors that need only simple arithmetic operations and limited memory.

As the expert support for proactive DFA and assembly sequence generation starts from the early design stage when information of the constraints involved is still vague, it is impractical to search a good or optimal solution based on these imprecise and incomplete constraints. It is more appropriate to evaluate whether a generated solution (or partial solution) satisfies any available constraints in a timely fashion. Thus constraints will be used in the validation of the generated assembly sequence instead of searching for a good or optimal sequence in the Ophir environment.

5.4 Selection of Tools and Methods

An expert module, or a module element with intelligent agent properties can be developed by a procedural language, an expert system language, such as OPS5, PROLOG, or even an expert system building tool (also called expert system shell) for convenience, maintainability, efficiency and speed.

An expert system language is a translator of commands written in a specific syntax, it will also provide an inference engine to execute the statements of the language [164, Page 24]. Depending on the implementation, the inference engine may provide forward chaining, backward chaining or both. The primary functional difference between expert system languages and procedural languages is the focus of representation. Procedural languages like C focus on providing flexible and robust techniques to represent data. For example, data structures such as arrays, records, linked lists, stacks, queues, and trees are easily created and manipulated. The data and methods to manipulate the data are tightly interwoven. In contrast, expert system languages focus on providing flexible and robust ways to represent knowledge. The expert system paradigm allows two level of abstraction: data abstraction and knowledge abstraction. Expert system languages specifically separate the data from the methods of manipulating the data. An example of this separation is that of facts (data abstraction) and rules (knowledge abstraction) in a rule-based expert system language. This difference in focus also leads to a difference in program design methodology. Because of the tight interweaving of data and knowledge in procedural languages, the programmer must carefully describe the sequence of execution. However, the explicit separation of data from knowledge in expert system languages requires considerably less rigid control of an execution sequence. Typically, an entirely separate piece of code, the inference engine, is used to apply the knowledge to the data. This separation of knowledge and data allows a higher degree of parallelism and modularity.

An expert system building tool is defined as:

> *A language plus associated utility programs to facilitate the development, debugging, and delivery of application programs* [164].

Utility programs may include text and graphics editors, debuggers, file managers, and even code generators. Cross assemblers may also be provided to port the developed code to different hardware. Some tools may even allow the use of different paradigms such as forward and backward chaining in one application.

For the convenience, speed, efficiency and maintainability in development, it was decided to integrate an expert system shell in the Ophir environment. Four available expert system shells have been investigated for their suitability. They are CLIPS from NASA, KAPPA-PC from IntelliCorp, Eclipse and Rete++ from Haley Enterprise. The investigation is based on the preconditions that the Ophir environment will be developed in VC++, and Microsoft Access will be used to develop the database.

5.4.1 Tool Investigation

CLIPS (the C Language Integrated Production System) expert system shell was developed by the artificial intelligence section in NASA/Johnson Space Center [165, 166]. The specific purpose for developing CLIPS is to provide high portability, low cost, and easy integration with external systems. It provides a complete environment for construction of rule and/or object based expert systems as well as effective forward chaining ability based on the use of Rete algorithm. CLIPS comes with all source code which can be easily modified or tailored to meet a user's specific needs, such as to integrate with external functions or applications. CLIPS can be installed on many different computers without making modifications to its source code. Another advantage is its low cost. Its simple user interface (interactive, text-oriented) does not defect its suitability, as the expert support implemented will work in background in the Ophir Project. The only weak point of CLIPS is that it does not provide support for backward chaining. But forward chaining can fulfil most of the tasks required for the expert support.

Further investigation tested the integration of CLIPS with VC++. It is easy to integrate them through CLIPSDLL [167]. The application developed with the integrated environment, is able to load and run *.clp files from VC++ environment. However, CLIPSDLL is a freeware, it is still in development, so not all the functions in CLIPS are wrapped in the CLIPSDLL.

KAPPA-PC is a powerful and user friendly expert system shell [168, 169] with a very good graphical user interface. Its powerful object-oriented programming, rule-based reasoning, and tools for developing and customising user interface are well integrated. Coding of class hierarchy, rules, functions, methods is quite straight forward in KAPPA-PC. It provides forward chaining and backward chaining reasoning. It communicates with dBase, and the new version of KAPPA-PC, version 2.4, has new database access interface based on ODBC and supports Windows NT. The documentation shows that KAPPA-PC can interface some CAD packages or conventional programming languages. KAPPA-PC is fully supported as 16 bit windows application, but not 32 bit. Its source code is not available. Despite many advantages and its integrated development environment for developing a stand-alone or a dominant expert system in an integrated environment, KAPPA-PC may not be flexible enough to develop several embedded expert modules required in the Ophir environment. Therefore, further testing of the integration of KAPPA-PC with VC++ has not conducted.

Eclipse is syntactically similar to CLIPS, but it is more powerful than CLIPS in some ways [170]: more constructs, *e.g. defrelation (father Jerry Tom);* more inheritances provided e.g. a template can inherit properties from its super templates; and its backward chaining ability. But the backward chaining in Eclipse is not as powerful as that in KAPPA-PC. It can automatically generate *goals* and allow *goals* to drive pattern matching in combination with facts in a normal data-driven manner. *Goals* can exist simultaneously in Eclipse and allows rules to do strategic reasoning. If dependencies on *goals* are used, *goals* will only persist as long as they are relevant, and the *facts* which are dependent on these *goals* will automatically retract when the *goals* do not exist any more. However, during backward chaining, if any information (e.g. slot value) to achieve a *goal* does not exist, Eclipse will not reason further. It does not pop up a dialogue box automatically to query the user about the missing information and then act on user's response as in KAPPA-PC.

Rete++ fully encapsulates Eclipse within C++ [171]. It also has modules for developing Case-Based Reasoning. So, from performance point of view, Rete++ is the most suitable expert system building tool. But it is quite expensive and the source code is not available. Furthermore, run-time license is required for the distribution of any developed applications from Rete++. An industrial collaborator of the Ophir Project did not accept this.

Finally, CLIPS is considered to be the most suitable shell for its flexibility, easy integration, reasonable performance, and low cost. It has been integrated into the Ophir environment.

5.4.2 Complementary C++ Functions

In situations that require to extend the functionality from CLIPS (this is more necessary as the CLIPSDLL has not implemented all the CLIPS functions), C++ functions will be created to complement the ability of the integrated CLIPS. Besides complementary of the rule-based reasoning, C++ functions will also be used in connecting the experts with the database, in implementing heuristic search, constraints satisfaction evaluation, and case-based reasoning.

5.5 Creation of the Expert Assembler

5.5.1 Expert Assembler

The *Expert Assembler* Figure 5.5, has been created and deployed into the Ophir environment according to the strategy specified in the beginning of this chapter. It has three expert modules:

- *Product Structure Expert*
- *Assembly Sequence Expert.*
- *Assembly Process Expert*

As each expert module has intelligent agent properties, we consider it as an ***'expert agent'***, so an expert refers to an expert agent.

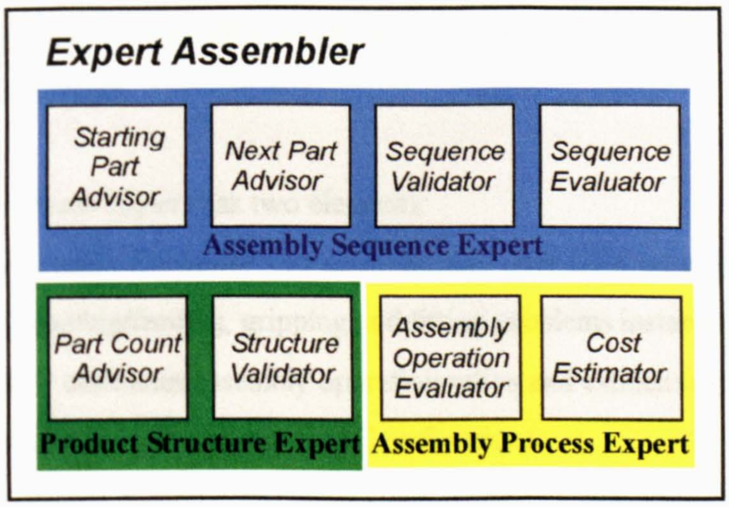


Figure 5.5 The Expert Assembler

In each expert agent, there are several expert elements to fulfil closely related tasks. The *Assembly Sequence Expert* has four elements:

- *Starting Part Advisor* - to recommend the base part for assembly sequence definition
- *Next Part Advisor* - to suggest the best next part when sequence construction is commenced
- *Sequence Validator* - to highlight sequence problems instantly
- *Sequence Evaluator* - to identify a good assembly sequence for proactive DFA evaluation

Currently three elements: the *Starting Part Advisor*, the *Next Part Advisor*, and the *Sequence Validator* have been implemented. The *Starting Part Advisor* uses rule-base approach, the *Next Part Advisor* uses a heuristic search. The *Sequence Validator* uses constraint satisfaction evaluation.

The *Product Structure Expert* has two elements:

- *Part Count Advisor* - to recommend candidate parts for elimination and combination
- *Structure Validator* - to make trade-off decision between part consolidation and part manufacturing cost and product performance

Currently one element, *Part Count Advisor* has been implemented. The implementation is mainly procedural C++ functions, but involves rule-based kinematic reasoning. Case-base Reasoning is explored to retrieve relevant DFA examples and successful DFA case studies to assist redesign.

The *Assembly Process Expert* has two elements:

- *Assembly Operation Evaluator* - to automatically infer DFA related part properties, to highlight handling/feeding, gripping and fitting problems instantly
- *Cost Estimator* calculates assembly operation ratios and estimates assembly cost

They are both implemented by procedural C++ functions.

5.5.2 Knowledge Base

The experts and expert elements in the *Expert Assembler* infer sequence and DFA related data using the knowledge stored in the knowledge base. The implemented knowledge base is as Figure 5.6. It has two separated parts: a CLIPS Knowledge Base, and a General Knowledge Base which is developed using Microsoft Access. Production rules, for example the set of rules regarding whether a part is suitable to be the base part of an assembly sequence, are stored in the CLIPS Knowledge Base. Other knowledge such as manufacture and assembly processes, material data, DFA criteria, design catalogue, and several successful DFA case studies are stored in the General Knowledge Base. All the product information required by expert reasoning is stored in the Four Layer Product Model Database specified next.

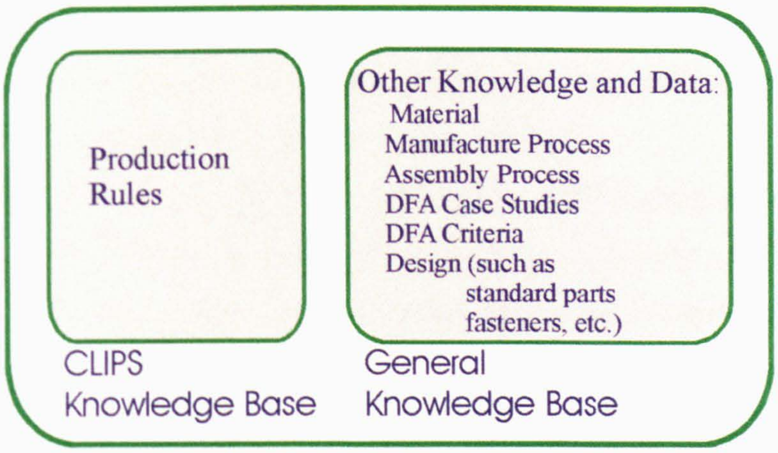


Figure 5.6 The Ophir Knowledge Base

5.5.3 *Product Model*

The underlining data structure – Four Layer Product Model [17] represents and stores all the relevant product data. To charter three possible kinds of design activities [172]:

- *Original Design*: providing an original solution principle for a given system,
- *Adaptive Design*: the adaptation of a known solution principle to a changed task,
- *Variant Design:* The variation of size or arrangement of an existing system,

four disparate types of components have to be handled. As the Four Layer Product Model is implemented using an object oriented approach, different component classes can be derived from the *Component Model* to hold data and methods for each type of components. For instance a data member of the New Subassembly class describes which other components are members of this particular subassembly.

The four component classes are:

1. ***New Component*:**

   Definition: Component designed specifically for a product and can only be a member of that product.

   Methods: Full access to the component model is available.

2. ***Existing Component*:**

   Definition: Component that can be a member of many products. It could be either a bought-in item or part of a standard in-house range.

   Methods: Only the assembly position and orientation can be edited.

3. ***New Subassembly*:**

   Definition: A collection of two or more components that only exist within a particular product. The incorporated components can be either *New Component* or *Existing Component* types.

   Methods: Full access is given to the product model to make changes. However, any changes to individual components can only alter the respective product model if the components are the *New Component* types.

4. ***Existing Subassembly*:**

   Definition: A collection of components that together form a subassembly that can be a member of many products. Again, this may be a bought-in assembly, part of a standard in-house range or a standard module.

   Methods: Only the assembly position and orientation can be edited.

Each type of component class is instantiated as many times as the number of each type of components in the assembly. The *Final Assembly Model* and *Component Interaction Model* are only ever instantiated once for a particular assembly. Although there should only be one *Assembly Plan Model* for each assembly, the possibility of exploring a number of sequences means this can be instantiated for comparison purposes. However, the final design can only have one instance of *Assembly Plan Model.* Data persistence is achieved by exporting the data stored in the four layers of the model to an Access database, which facilitates data searches and ease of retrieval.

5.5.4 Expert Reasoning

The experts in the *Expert Assembler* collect data from the Product Model or Product Model database. For rule-based implementation, the collected data is asserted as CLIPS facts to trigger the rules in the CLIPS Knowledge Base. This results in new facts to be asserted and some old facts to be retreated from CLIPS. There are several routes to pass results from CLIPS into C++ environment.

For other implementations, which include heuristic research, constraint satisfaction evaluation, and cases retrieving, algorithms have been developed and C++ functions have been created. After data is collected by the experts, reasoning is conducted in a

C++ environment based on the knowledge stored in the General Knowledge Base. Some of the variables in the C++ environment or data in the Product Model database will be updated as a result of the expert reasoning. Recommendations or warnings from the experts will be fed back to the user through a user-friendly interface. Depending on the user's response, each expert acts further.

5.6 Implementation

The Ophir assembly-oriented CAD environment is written in Visual C++, integrating the ACIS solid modelling kernel, the CLIPS expert system toolkit and Microsoft Access Database. It implements the data structure of Four Layer Product Model. The knowledge base incorporates assembly, manufacturing expertise, and successful DFA case studies. The proactive DFA analysis module provides a quantitative view on the assemblability of the product in relation to the amount of available data. The software integration can be described as follows.

5.6.1 Embed CLIPS in C++ Environment

Files (*.cpp and *.h files) that provide the CLIPSWrap class which dynamically loads the CLIPSDLL to encapsulate the core CLIPS functions are inserted into the C++ project of the Ophir application, and the main header file is included in a class AssemblyDoc where the CLIPWrap will be instantiated. An instance of the CLIPSWrap is created, and initialised in the constructor of the AssemblyDoc class which holds all the document data about product structure and assembly sequence, such as the data of *Assembly Object*, the *Component Interaction Object*, and the *Assembly Sequence Object*. Several routes to return information from CLIPS to C++ environment are established before the initialisation. The CLIPSWrap class dynamically loads the CLIPSDLL which encapsulates the core CLIPS actions like *Load*, *Reset* and *Run* to allow a developer to embed CLIPS into a C++ program. To make this work, the CLIPSDLL has to be placed in the local project directory or in the compiler searchable directories.

5.6.2 Integrate ACIS with C++

Files that provide the CAcisApp, CAcisDoc and CAcisView base classes needed to build the basic acismfc application, are inserted into the C++ project. An instance of CAcisApp is created in the constructor of the application class COphirApp in the project and it is initialised. This starts the ACIS modeler and initialises each of its core libraries in C++ environment. An instance of the CAcisGLDoc class, which is derived from the CAcisDoc, is created in the document class CDesignDoc of the project. The CAcisGLDoc has a function to create a *rendering context,* which is an object that manages a scene of entities and the views to define different eye points (cameras) used to look at the scene. An instance of the CAcisView class is created in one of the view classes of the C++ project, CDesignView. This makes the ACIS geometric model can be viewed in the Ophir Assembly-oriented CAD. It is also necessary to let the compiler know the directories of all the ACIS header files and tell the builder any relevant ACIS libraries and their directories.

5.6.3 Access and Manipulate the Database from C++ Environment

Accessing and manipulating the Four Layer Product Model Database and the General Knowledge Base from C++ environment is achieved by deriving classes from the CDaoRecordset class in Microsoft MFC. A CDaoRecordset object, a kind of recordsets, represents a set of records selected from a data source that you can use to examine, add, change, or delete records from an underlying database table. This connects a derived recordset class with a data source which is a table or a query in the Database or in the General Knowledge Base. In the C++ environment, from the derived recordset class, we can access and manipulate its connected table or query to:

- scroll through the records,
- add new records,
- update the records,
- filter certain records from those available,
- sort the records.

We can also parameterise the recordset to customise its selection with information not known until run time.

5.7 The Ophir Environment and Its Supporting Experts

The Ophir environment comprises a number of workspaces:

- The *Structure Builder* – a window to develop product/assembly structure.
- The *Sequence Builder* – a window to build assembly sequences.
- The *CAD Solid Modeller* – a set of CAD windows for designing each component, subassembly and overall product assembly.

A typical screen of this environment shows in Figure 5.7. Three windows are visible: at the bottom right of the screen is the full solid model of the assembly, its background colour indicates the access and control abilities authorised. Currently the model shows in white background that indicates only position and orientation of each part can be manipulated, as it is an assembly view. For an existing component/subassembly, the background colour will be blue which means that the part can only be viewed. For a new component, it will be shown in a black background which indicates that the user has full access to its model to create or edit it freely.

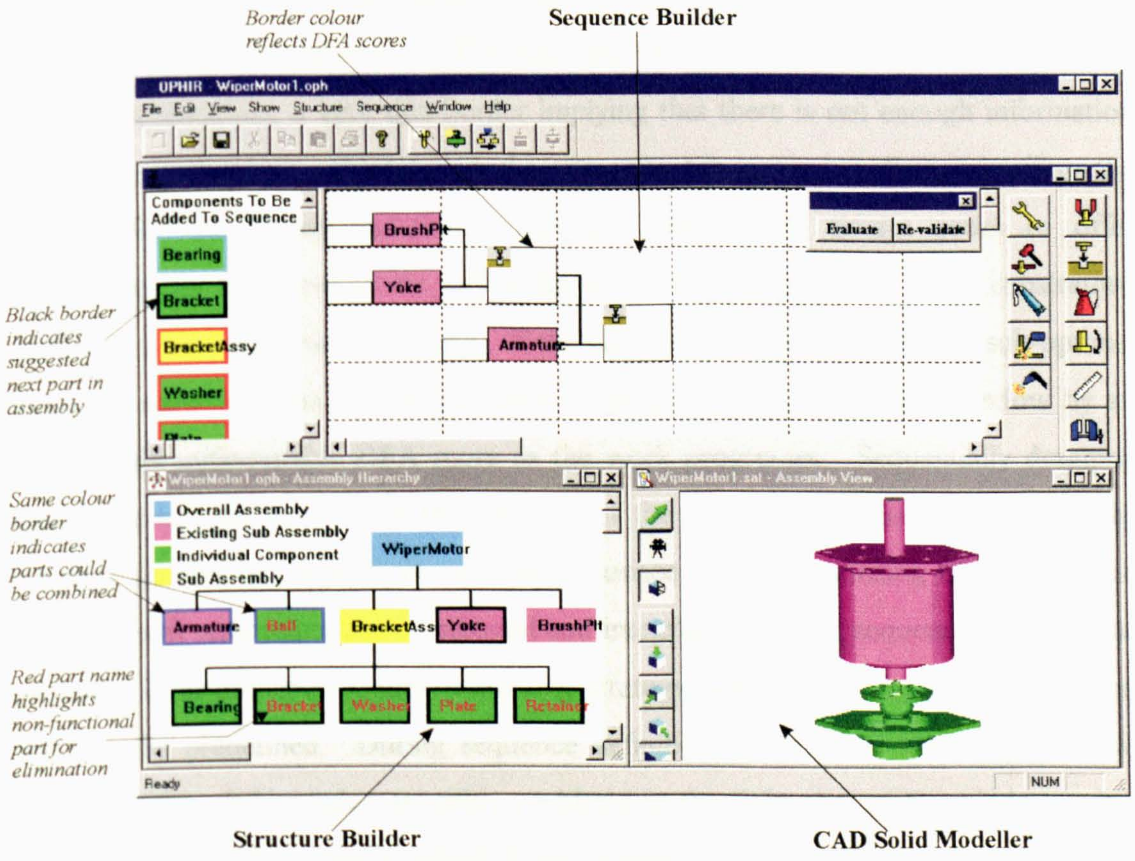

Border colour reflects DFA scores
Sequence Builder
OPHIR - WiperMotor1.oph
File Edit View Show Structure Sequence Window Help
Components To Be Added To Sequence
Bearing
Bracket
BracketAssy
Washer
BrushPlt
Yoke
Armature
Evaluate
Re-validate
Black border indicates suggested next part in assembly
Same colour border indicates parts could be combined
Red part name highlights non-functional part for elimination
WiperMotor1.oph - Assembly Hierarchy
Overall Assembly
Existing Sub Assembly
Individual Component
Sub Assembly
WiperMotor
Armature
Ball
BracketAssy
Yoke
BrushPlt
Bearing
Bracket
Washer
Plate
Retainer
WiperMotor1.sat - Assembly View
Ready
NUM
Structure Builder
CAD Solid Modeller

Figure 5.7 The Experts Operate in the Ophir Environment

At the bottom left of the screen is the product/assembly structure, showing components and subassemblies in relation to the finished assembly. Here, each component is displayed with its name in a *coloured rectangular background* (part icon), and the colour indicates its type (subassembly or individual component). It is in this hierarchical structure, the *Product Structure Expert* works. Group of parts whose combination is suggested are highlighted with the same coloured borders, such as the *armature* and the *ball* in the figure. Parts which are not essential for the functionality of the finished assembly, such as the *bracket*, have their names displayed in red.

At the top of the screen is a window split into two; it is here that the *Assembly Sequence Expert* works. It gives guidance to the interactively assembly sequence definition; it highlights sequence problems instantly. In the left-hand part of the window, a list of the components is displayed in coloured icon with a border whose colour dynamically shows recommendation for the starting/next part in the assembly sequence. A black border indicates that the part is recommended as the starting/next part, a red border implies that the part may not be suitable and a blue border means that there is no indication either way. The thickness of the border indicates the strength of the recommendation, a very thin border implying that there is not enough information to make a judgement. Clicking on the component brings up a dialog detailing the reasons for the recommendation. Although the system always seeks to offer suggestions for the starting/next part in the sequence, the designer is never constrained to follow a particular assembly path. If the designer persists in defining a sub-optimal assembly sequence, this will be indicated in the right-hand part of the window by an increasingly unfavourable DFA score as the work progresses. Sequentially dragging and dropping part icons from the left-part of the window into the right-part of the window interactively defines an assembly sequence. At the beginning of this process, hard and soft constraints can be selected and integrated into the sequence construction process. User-preferred sequence building strategy, such as bottom up or inside out, can also be predefined. During sequence definition process, the *Assembly Process Expert* works. It highlights assembly problems in the right part of the window. Any unfavourable DFA score resulted from assembly operations: handling, feeding, gripping, will be highlighted with red border of the part icon. Fitting problem or sequence problem will be highlighted with red border of the sequence liaison. Double clicking the part icon or right clicking the sequence liaison can bring up a dialog detailing the problems. Control of the sequence is always the designer's, but as much support as possible is offered from the experts during its definition.

To implement all the experts, a large amount of work is required. My work is focused on the areas to support for product structure optimisation in proactive DFA and support for assembly sequence generation, and this will be detailed in the next two chapters.

Chapter 6 Expert Support for Proactive DFA

In the Ophir Assembly-oriented CAD environment, the proactive DFA analysis is based on the CSC technique. The expert support for the proactive DFA evaluation provided by the *Expert Assembler* includes four areas:

1. Providing general DFA guidelines when necessary;
2. Support for part count reduction to optimise product structure;
3. Support for material and process selection and early manufacturing cost estimation to validate part consolidations;
4. Assisting detailed assembly operation evaluation.

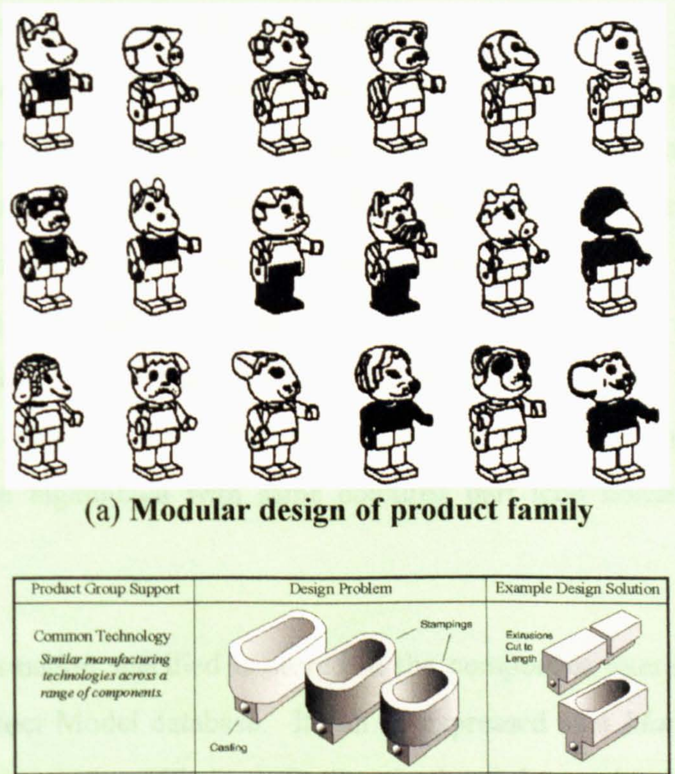
The *Expert Assembler* should guide the designer to generate an assembly-oriented design concept, product structure, and component features. It should highlight DFA problems instantly. For a limited time only, the second area (support for part count reduction) is the focus of the implementation. Ideas of how to support for the other areas will be briefly described.

6.1 Providing General DFA Guidelines

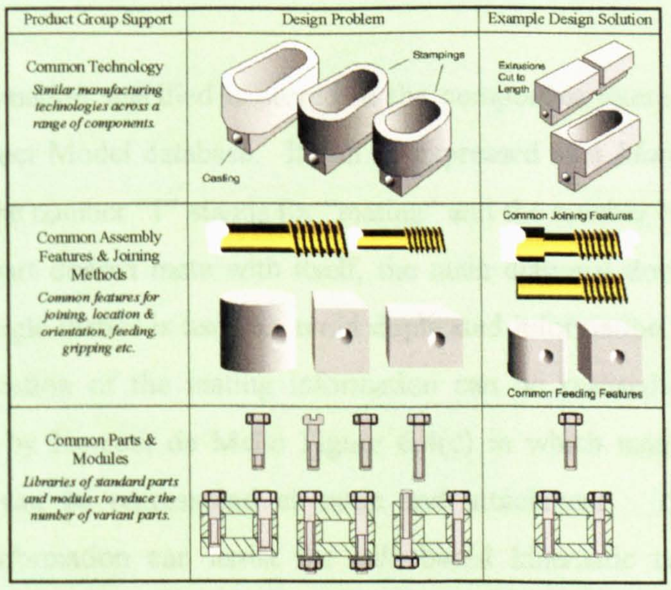
In early design stage, the descriptions of the physical components of a product are often vague and imprecise, and knowledge of all the design requirements and constraints are usually approximate or unknown. High-level descriptions of product and imprecise information make quantitative reasoning difficult in this stage, if not impossible. But a poor design concept can never be compensated for by a good detailed design. So DFA analysis should start from conceptual design stage. As a result, general DFA guidelines and some qualitative DFA evaluation methods may be realised to guide the generation of assembly-oriented design concept.

To provide general DFA guidelines, a "DFA Principle Library" can be established in design environment. Guidelines based on these principles can be arranged in categories and sub-categories according to their applicable areas, and applicable stages (from concept to detail). Each guideline should be either connected to some key words, or specific design features, or certain actions performed in design environment. The

Expert Assembler can then check any actions made by the designer that may relate to a key word, a kind of design features or is directly linked to a DFA guideline. Any connections should trigger certain guidelines, and the designer can assess if the related guidelines are appropriate to be applied. For example, when a design feature violates DFA criteria, warnings should be given, so the designer can make decisions to change the design if necessary. Hints and tips could be the suitable forms for delivery of guidelines, but they are better with examples and graphic representations for ease of understanding like the two guidelines in Figure 6.1 ((a) is from [5], (b) is from [127]).



(a) Modular design of product family



Product Group Support	Design Problem	Example Design Solution
Common Technology *Similar manufacturing technologies across a range of components.*	Stampings Casting	Extrusions Cut to Length
Common Assembly Features & Joining Methods *Common features for joining, location & orientation, feeding, gripping etc.*		Common Joining Features Common Feeding Features
Common Parts & Modules *Libraries of standard parts and modules to reduce the number of variant parts.*		

(b)Use parts with similar manufacturing technology; shared assembly characteristic; Repeated use of similar parts.

Figure 6.1 DFA Guidelines with Examples

6.2 Support for Part Count Reduction

Based on the CSC technique, Functional Analysis is carried out to identify 'B' parts. 'B' parts are candidates for elimination and one way of elimination is to combine a part with another part. The support for part elimination and part combination falls to the *Part Count Advisor* (**PCA**), one of the expert elements introduced in chapter 5. With it, 'B' Parts and pairs or groups of parts that could be combined will be identified and highlighted in the product structure.

6.2.1 Procedures of the Part Count Advisor

The search of parts for combination is based on a working flowchart in Figure 6.2, which uses information from a number of sources: component mating information entered by the user and the assessments of functional parameters such as the necessity of relative movement between parts, the requirement that parts be made of different materials and the likelihood of separation of parts for adjustment or replacement. The assessments of functional parameters will identify all the 'B' parts. 'B' parts will be highlighted with names in red. Eventually pairs or groups of parts that could be combined will be highlighted with same coloured part icon borders in the product structure.

The mating information specified is stored in the component interaction layer of the Four Layer Product Model database. It can be expressed as a *Mating Matrix* Figure 6.4(b) in which the number "1" stands for "mating" and the number "0" stands for "not mating". As a part doesn't mate with itself, the main diagonal does not contain any numbers. A triangle matrix is used to avoid duplicated information. A more detailed graphic representation of the mating information can be realised by the *relational model* proposed by Homem de Mello Figure 6.4(c) in which mating joint type and joining process can be represented as node and attachment. The more detailed mating/joining information can assist the rule based kinematic reasoning as some kinematic information has been stated implicitly in mating joint types and joining processes. For example, if the mating joint type between two parts is force-fit, then the mated two parts will not move relatively. If the joining process is fastening, then the fasteners and the parts to be fastened will not move relatively. Lin et al [173] proposed a component mating joints hierarchy graph to represent the conceptual assembly design Figure 6.5. Based on this graph, certain mating joint types with which the mated parts are unlikely to move relatively Table 2(a) or very likely to move relatively Table 2(b) have been identified and stored in the CLIPS Knowledge Base.

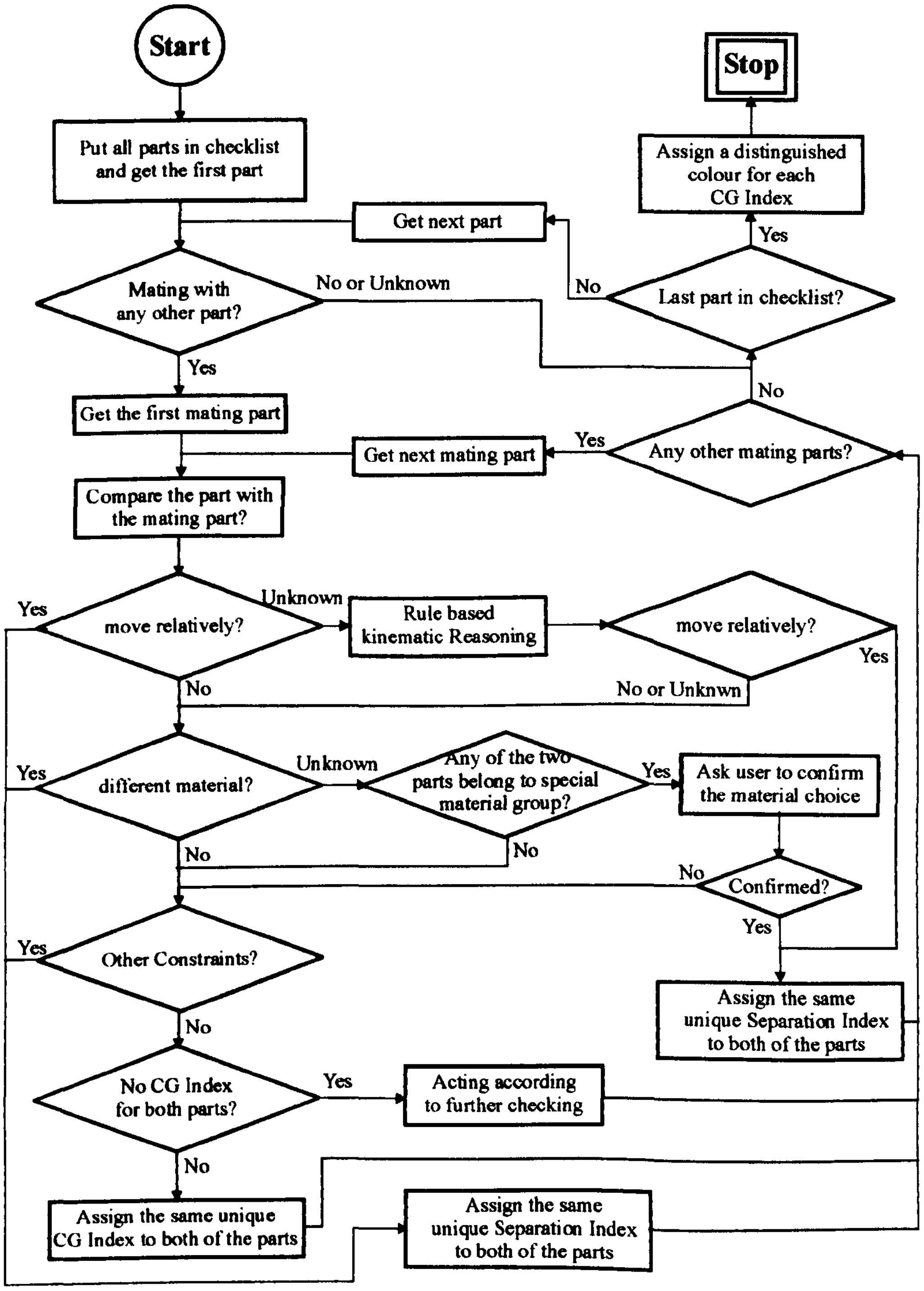

Start
Stop
Put all parts in checklist and get the first part
Assign a distinguished colour for each CG Index
Get next part
Yes
Mating with any other part?
No or Unknown
No
Last part in checklist?
Yes
No
Get the first mating part
Yes
Get next mating part
Any other mating parts?
Compare the part with the mating part?
Yes
move relatively?
Unknown
Rule based kinematic Reasoning
move relatively?
Yes
No
No or Unknwn
Yes
different material?
Unknown
Any of the two parts belong to special material group?
Yes
Ask user to confirm the material choice
No
No
No
Confirmed?
Yes
Yes
Other Constraints?
Assign the same unique Separation Index to both of the parts
No
No CG Index for both parts?
Yes
Acting according to further checking
No
Assign the same unique CG Index to both of the parts
Assign the same unique Separation Index to both of the parts

Figure 6.2 Part Count Advisor Working Flowchart

(CG Index – Combination Group Index)

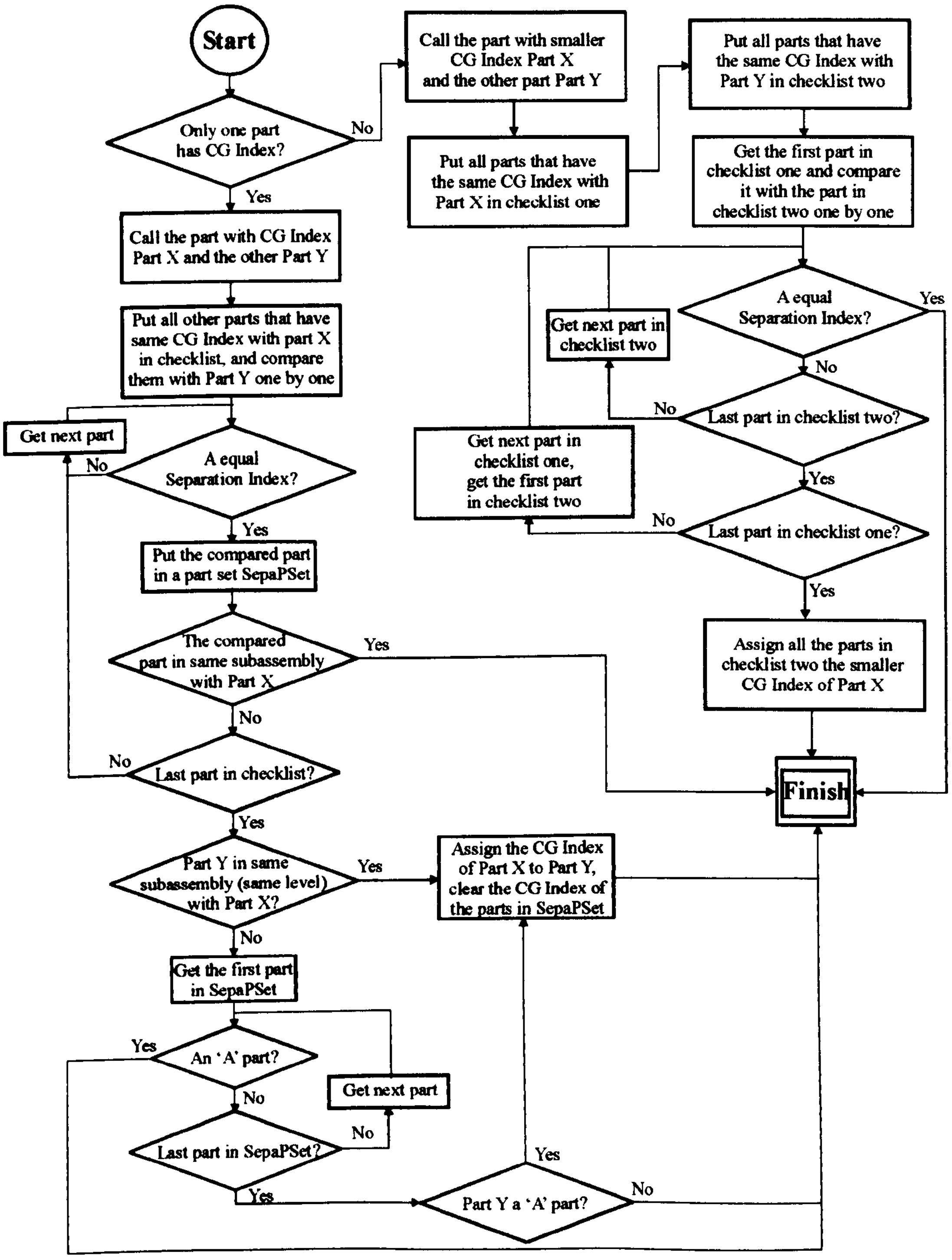
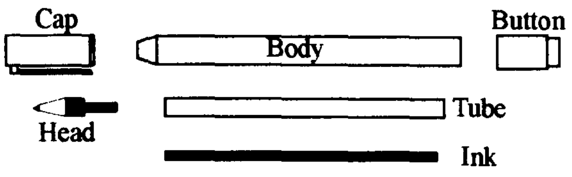

Start
Only one part has CG Index?
No
Yes
Call the part with smaller CG Index Part X and the other part Part Y
Put all parts that have the same CG Index with Part X in checklist one
Put all parts that have the same CG Index with Part Y in checklist two
Get the first part in checklist one and compare it with the part in checklist two one by one
A equal Separation Index?
Yes
No
Get next part in checklist two
Last part in checklist two?
No
Yes
Get next part in checklist one, get the first part in checklist two
Last part in checklist one?
No
Yes
Assign all the parts in checklist two the smaller CG Index of Part X
Finish
Call the part with CG Index Part X and the other Part Y
Put all other parts that have same CG Index with part X in checklist, and compare them with Part Y one by one
Get next part
A equal Separation Index?
No
Yes
Put the compared part in a part set SepaPSet
The compared part in same subassembly with Part X
Yes
No
Last part in checklist?
No
Yes
Part Y in same subassembly (same level) with Part X?
Yes
No
Assign the CG Index of Part X to Part Y, clear the CG Index of the parts in SepaPSet
Get the first part in SepaPSet
An 'A' part?
Yes
No
Get next part
Last part in SepaPSet?
No
Yes
Part Y a 'A' part?
Yes
No

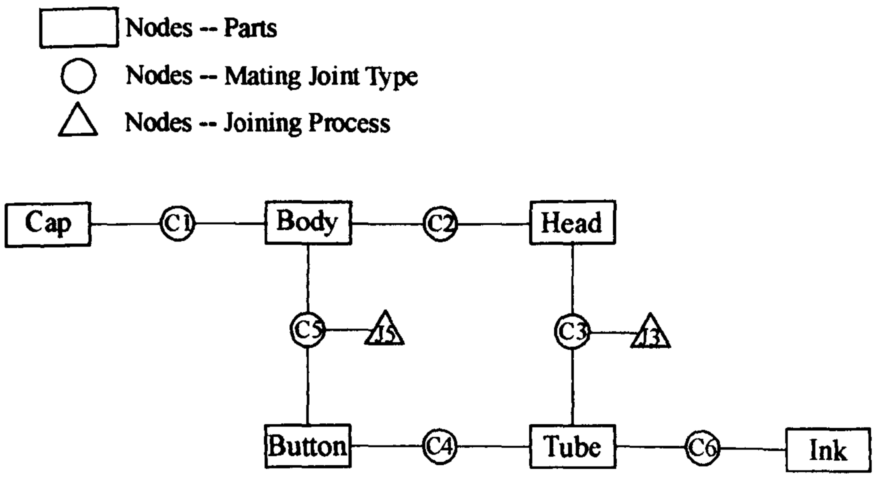
Figure 6.3 The Procedure of Further Checking in Part Count Advisor Flowchart



(a)

	Cap	Body	Head	Button	Tube	Ink
Cap		×	×	×	×	×
Body	1		×	×	×	×
Head	0	1		×	×	×
Button	0	1	0		×	×
Tube	0	0	1	1		×
Ink	0	0	0	0	1	

(b)



(c)

Figure 6.4 (a) Component Parts of Ball-point Pen,

(b) Mating Matrix of Ball-point Pen,

(c) Relational Model Graph of Ball-point Pen

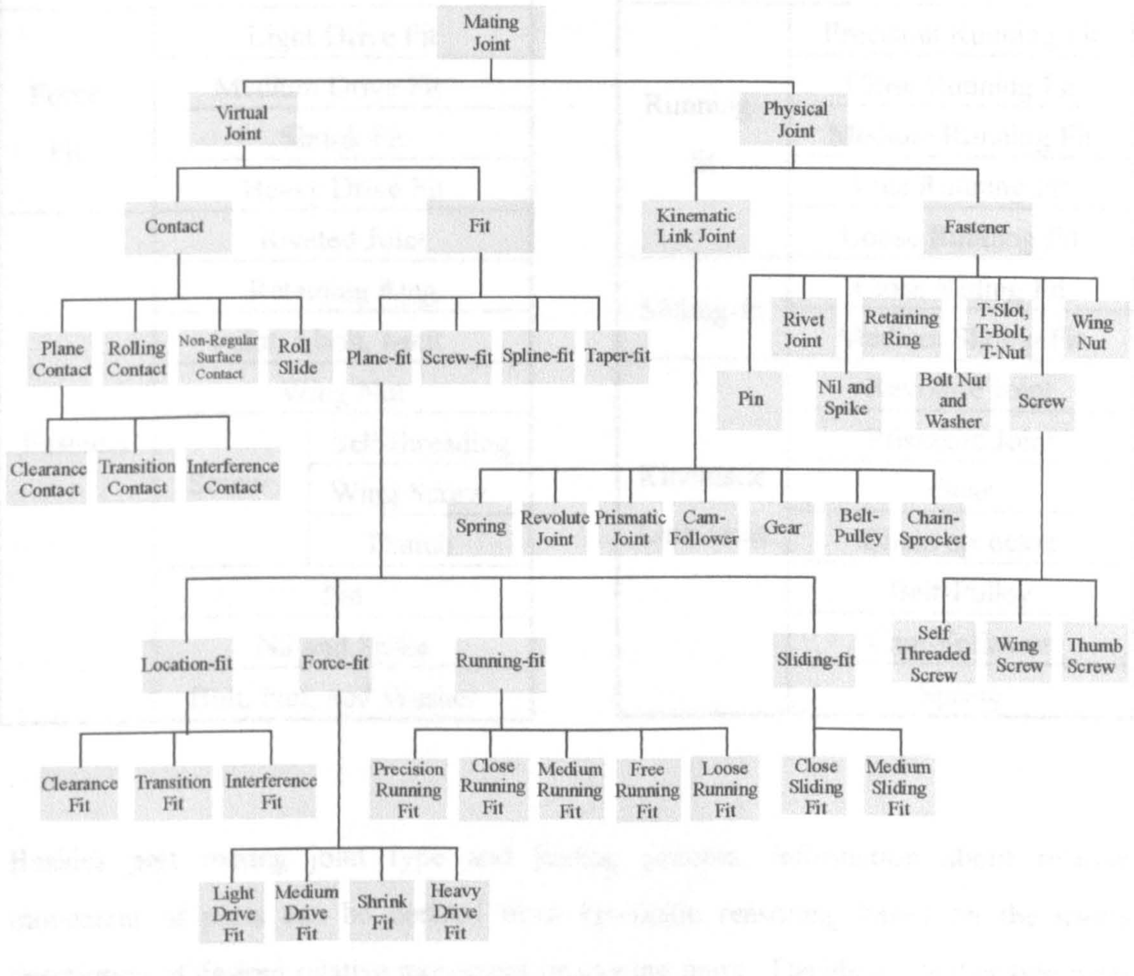


Figure 6.5 Hierarchy of Mating Joints Proposed by Lin and Chang

Table 2: Identified Mating Joint Types

(a) Unlikely to move relatively

Mating Joint Types		
Interference Contact		
Non-regular Surface Contact		
Interference Fit		
Force Fit	Light Drive Fit	
	Medium Drive Fit	
	Shrink Fit	
	Heavy Drive Fit	
Fastener	Riveted Joint	
	Retaining Ring	
	t-slot, t-bolt, t-nut	
	Wing Nut	
	Screw	Self-threading
		Wing Screw
		Thumb
	Pin	
	Nil and Spike	
	Bolt, Nut, and Washer	

(b) Very likely to move relatively

Mating Joint Types	
Roll-slide Contact	
Rolling Contact	
Clearance Fit	
Running-fit	Precision Running Fit
	Close Running Fit
	Medium Running Fit
	Free Running Fit
	Loose Running Fit
Sliding-fit	Close Sliding Fit
	Medium Sliding Fit
Kinematic Link Joint	Revolute Joint
	Prismatic Joint
	Gear
	Chain-sprocket
	Belt-Pulley
	Cam-Follower
	Spring

Besides part mating joint type and joining process, information about relative movement of parts can be derived from kinematic reasoning based on the user's description of desired relative movement or moving parts. Deciding whether two parts must be capable of relative motion is clearly the responsibility of the designer, since this is a question of design intent, but the number of relative motion judgements required may be as many as the square of the number of the parts, which can make for a tedious task. Using information (explicit and implicit) as the designer enters, rule-based inferences can be made about parts; if part A is immobile relative to part B, and part B is immobile relative to part C, then it infers that parts A and C do not move relative to each other. This relative movement information is perhaps the hardest kind of information to extract from the CAD model, since the model is a snapshot of the parts in a particular configuration. The rule-based expert making simple inferences such as this can drastically reduce the time and effort required of the designer, freeing him for more productive tasks. With the addition of suggestions for improvement based on best practice to be integrated, the **PCA** will be of benefit to the designer from the conceptual design stage. If the designer specifies names of components, such as "bolt", "nut" and "washer", the potential immediately exists to ask if the washer is necessary, or indeed if the nut could be replaced by the use of a threaded hole. If the user is also queried as to whether disassembly is important, then it may be possible to suggest replacing three components with a permanent fixing – and all without a single component having reached the detail design stage.

The assessment of the necessity of parts to be made of different materials is based on material information in the database in the first instance. Special groups of materials are being established based on high or low values of particular material properties such as magnetic permeability, electrical resistivity, corrosion resistance and thermal conductivity. If the material for any component belongs to an established special group, it is very likely that this material has been specified for a reason, and is therefore not a good candidate for combining with others. If the part could otherwise be combined with another, then the user is asked for confirmation of the material choice. The adjustment and replacement assessment is mainly conducted by the user and is based on the product's functional specification. The **PCA** immediately identifies and highlights 'B' parts, and pairs or groups of parts that could be combined from the earliest design stage. Holbrook [174] proposed a similar method to give suggestions on part combination, but this was based on the detailed geometry, so it could only be applied to a finished design.

The **PCA** also allows the designer to enter additional criteria such as geometric constraints. A typical geometric constraint is the requirement that two parts have to be separated to allow the addition of other parts into the assembly (for example, two halves of a gearbox casing must be separable to allow the gears to be fitted). Some geometric constraints can be identified automatically [175] by a reasoning method used for checking the mobility (global translation) of each part in a target assembly, if the related product model data is available, although the computation involved is significant and costly in processor time. Finally, parts suggested for elimination and combination are highlighted on screen, thus helping the designer recognise those which may not be confined for geometric reasons. This allows the designer dynamically to identify more constraints to put into the system, so the accuracy and feasibility of the suggestions from the **PCA** will be continually increased.

If design efficiency $(\frac{\text{No. of 'A' Parts}}{\text{Total No.of Parts}} * 100\%)$ does not reach 40%, warning will be issued by the system to suggest reducing part count before going further [18]. Sometimes it will not be possible to achieve an efficient design with a set of components defined, such as in situation that the designer can not improve the design efficiency up to 15%, in this case, the system will suggest the consideration of a thorough redesign from scratch.

The user can always enter extra information or edit any data from the reasoning and analysis, as the experts are decision support tools. Ultimately, the assembly is still the responsibility of the designer, who may choose to ignore or accept any of the suggestions offered. But problems will be further highlighted.

6.2.2 An Example to Demonstrate How the Part Count Advisor Works

A 20w rear screen wiper motor, exploded view as Figure 6.6, can be analysed to demonstrate how the **PCA** works in the Ophir environment.

This wiper motor has five subassemblies as in its product structure Figure 6.7. To simplify the demonstration, only one subassembly *Bracket Assembly* will be analysed. The other subassemblies will be defined as 'Existing Subassembly' type which will be treated as individual components. The simplified product structure is presented in Figure 6.8.

[18] In this work, 60% (recommended by the CSC technique) is still the desired design efficiency target. But 40% is used as a criterion to alert the designer to improve the product design

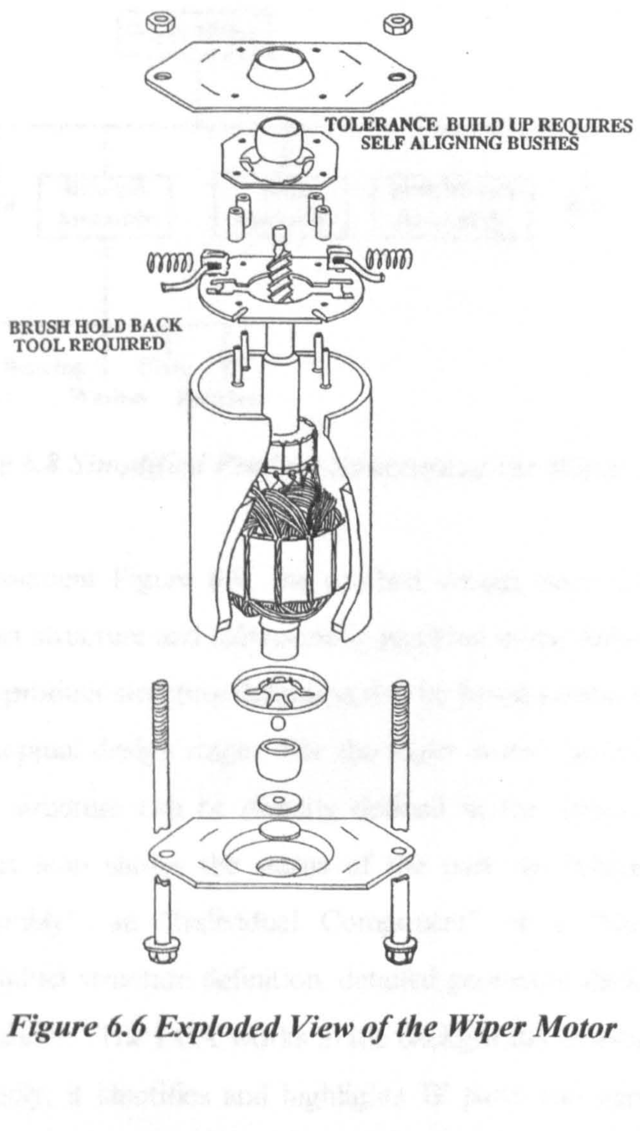

TOLERANCE BUILD UP REQUIRES SELF ALIGNING BUSHES
BRUSH HOLD BACK TOOL REQUIRED

Figure 6.6 Exploded View of the Wiper Motor

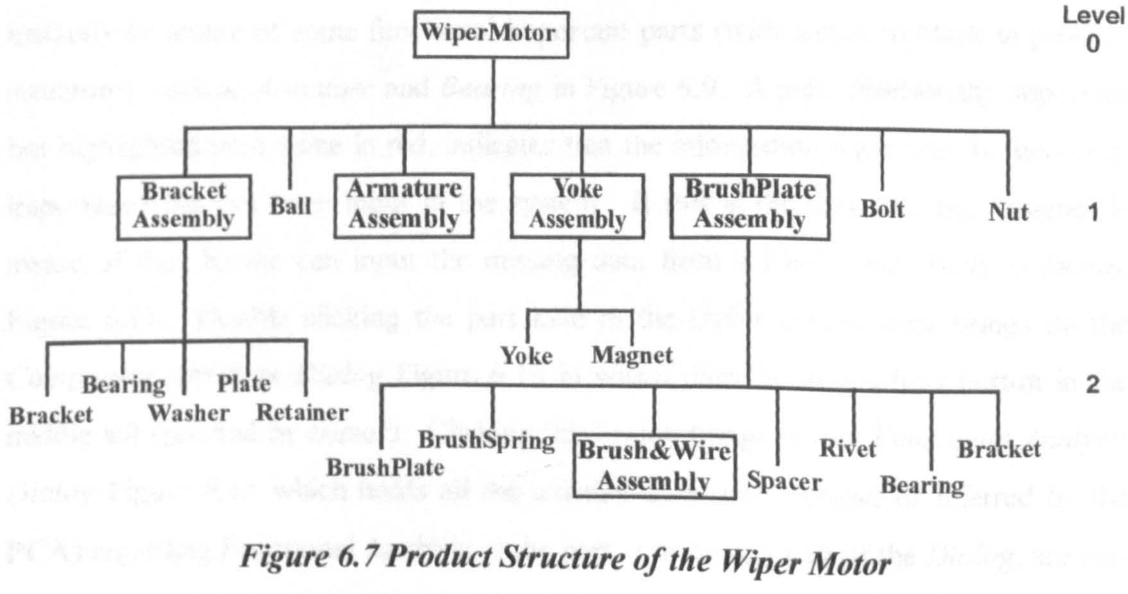


Figure 6.7 Product Structure of the Wiper Motor

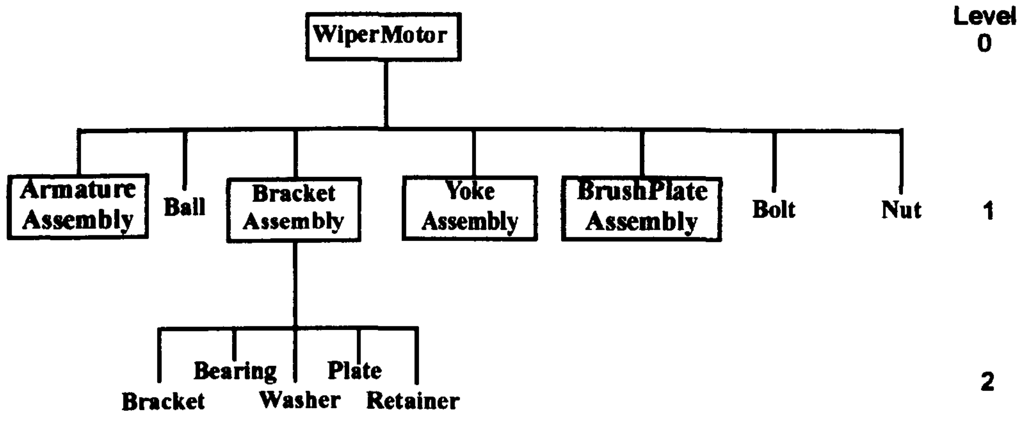


Figure 6.8 Simplified Product Structure of the Wiper Motor

In the Ophir environment Figure 6.9, the product design starts from the interactive definition of product structure and subassembly partition in the *Structure Builder*. For a new product, the product structure definition will be based on the functional structure formed in the conceptual design stage. For the *wiper motor*, an existing product, the simplified product structure can be directly defined in the *Structure Builder*. The colour of each part icon shows the status of the part: an "Overall Assembly", an "Existing SubAssembly", an "Individual Component" or a "New SubAssembly". Parallel with the product structure definition, detailed geometric design can carry out in the *CAD Solid Modeller*. The **PCA** works in the background to evaluate relevant data continuously. Initially, it identifies and highlights 'B' parts and parts which have not explicit information regarding their functional importance. This makes the designer instantly be aware of some functional important parts (with names in black in product structure), such as *Armature* and *Bearing* in Figure 6.9. A part, functionally important but highlighted with name in red, indicates that the information regarding its functional importance has not been input in the system. If this is the case and the designer is aware of this, he/she can input the missing data from a *Functional Analysis Dialog* Figure 6.11. Double clicking the part icon in the Ophir environment brings up the *Component Attribute Dialog* Figure 6.10 in which there is an *Analysis* button in the middle left (pointed by cursor). Clicking this button brings up the *Functional Analysis Dialog* Figure 6.11 which holds all the existing data (user defined or inferred by the **PCA**) regarding Functional Analysis of the part. On the left part of the *Dialog*, are part name, part number, and part functional descriptions. The functional descriptions can be edited by the user. On the right part of the *Dialog*, parts that have to move relatively with the analysed part are listed, such as the *Amature*. Also parts that have to be made of different material and parts that are very likely to be separated from the analysed part - *Bearing*, are listed in different tabs. The user can edit these part lists from different tab selections. He/she can ignore any information already existed to change the functional assessment result by clicking 'A' part or 'B' part radio box directly from the bottom left of the *dialog*. If 'B part' is clicked, all the parts listed in the part lists regarding relative movement, different material and separation for adjustments and services will be cleared. At any time, if the *OK* button is clicked, information regarding functional assessment of the part in the database will be updated.

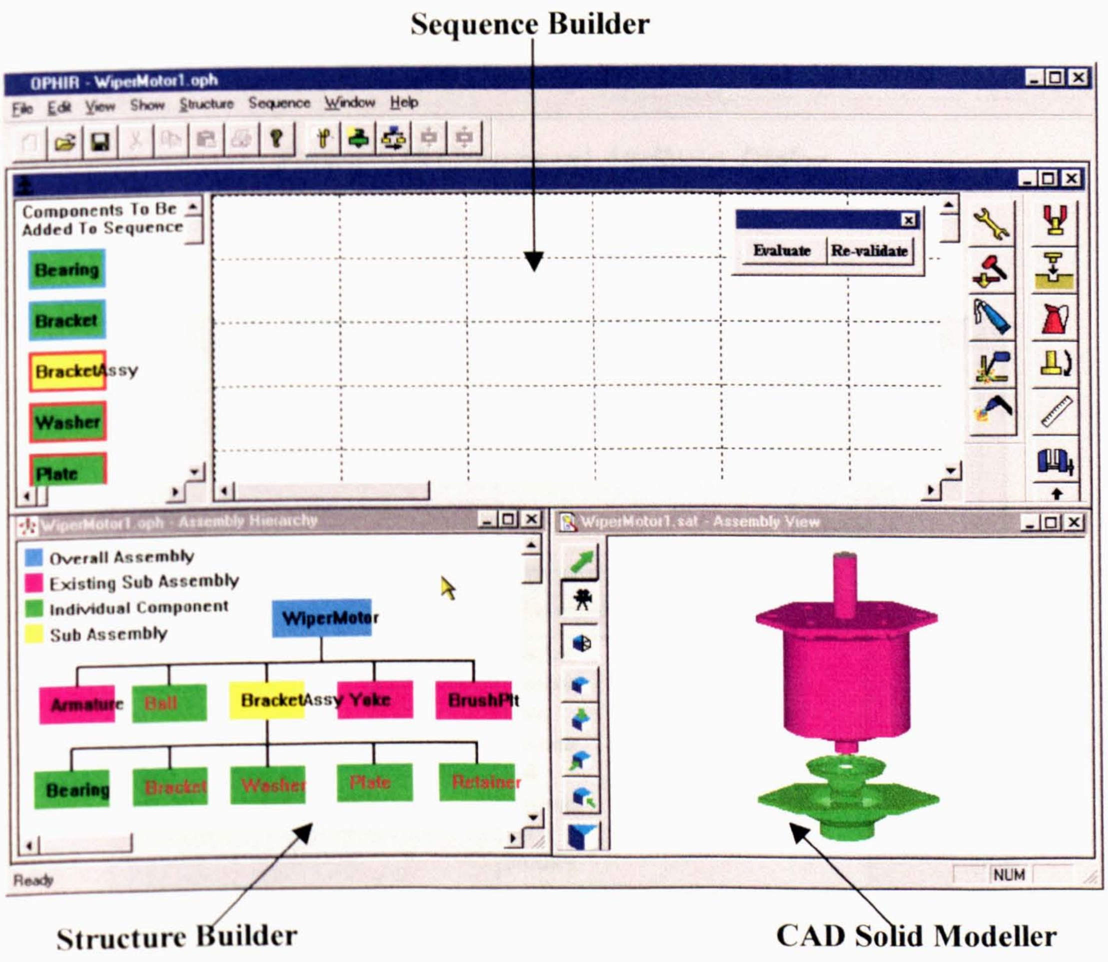
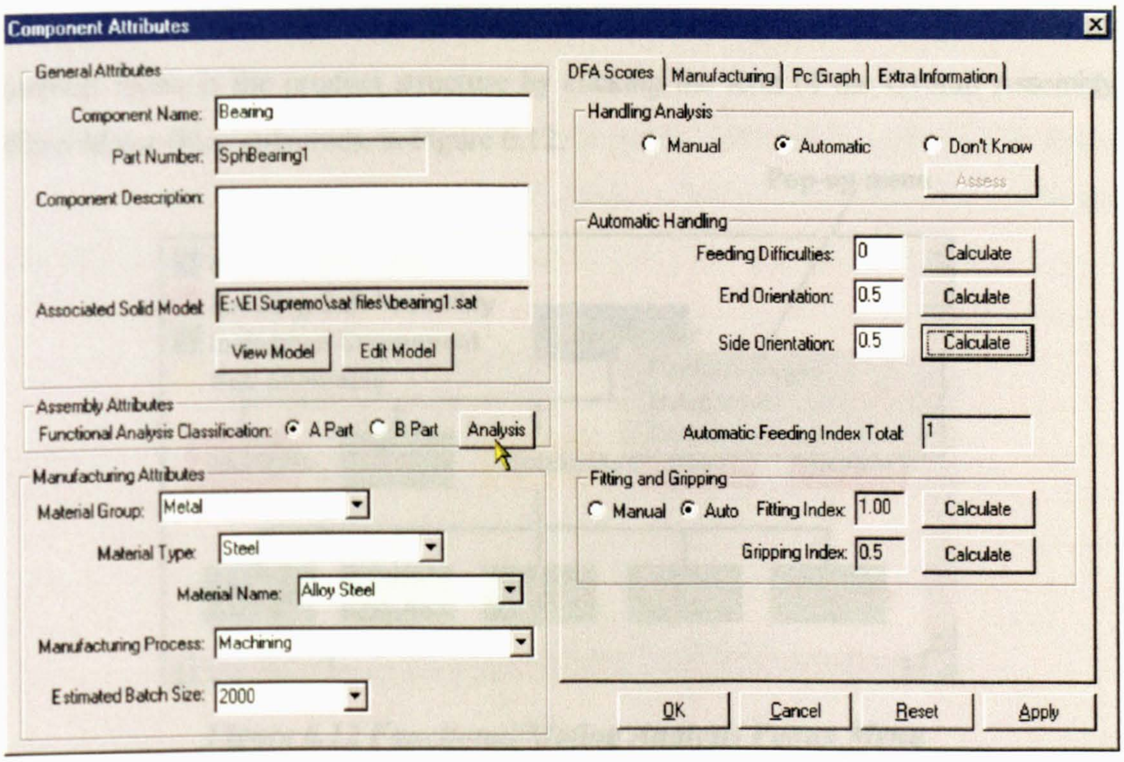


Figure 6.9 The Ophir Environment


Component Attributes

General Attributes
Component Name: Bearing
Part Number: SphBearing1
Component Description:
Associated Solid Model: E:\EI Supremo\sat files\bearing1.sat
View Model | Edit Model

Assembly Attributes
Functional Analysis Classification: [✓ A Part] B Part | Analysis

Manufacturing Attributes
Material Group: Metal
Material Type: Steel
Material Name: Alloy Steel
Manufacturing Process: Machining
Estimated Batch Size: 2000

DFA Scores | Manufacturing | Pc Graph | Extra Information

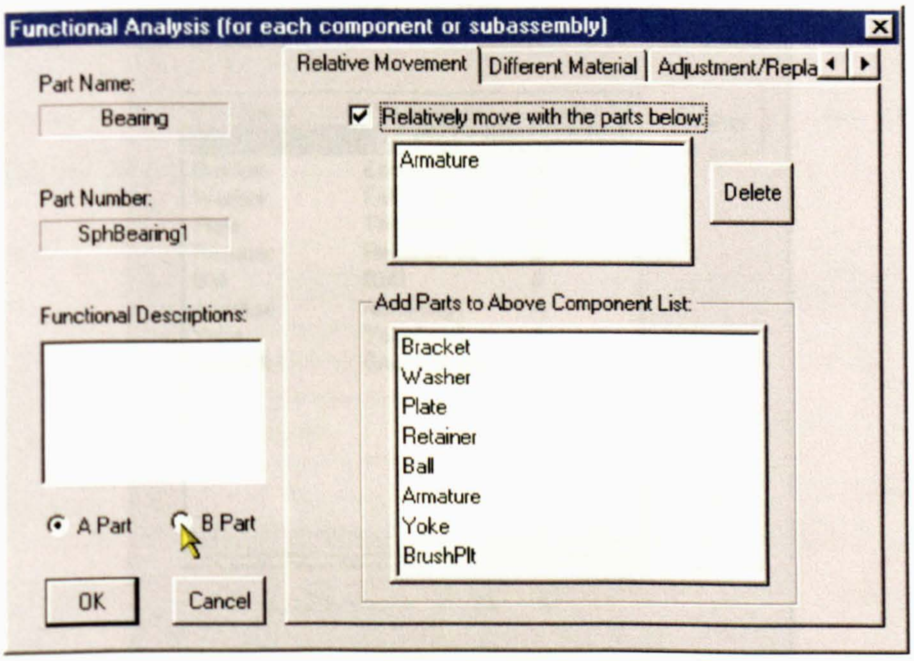
Handling Analysis
Manual [✓ Automatic] Don't Know | Assess

Automatic Handling
Feeding Difficulties: 0 | Calculate
End Orientation: 0.5 | Calculate
Side Orientation: 0.5 | Calculate
Automatic Feeding Index Total: 1

Fitting and Gripping
Manual [✓ Auto] Fitting Index: 1.00 | Calculate
Gripping Index: 0.5 | Calculate

OK | Cancel | Reset | Apply

Figure 6.10 Component Attributes Dialog


Functional Analysis (for each component or subassembly)

Part Name: Bearing
Part Number: SphBearing1
Functional Descriptions:
[✓ A Part] B Part
OK | Cancel

Relative Movement | Different Material | Adjustment/Repla

[✓] Relatively move with the parts below:
Armature
Delete

Add Parts to Above Component List:
Bracket
Washer
Plate
Retainer
Ball
Armature
Yoke
BrushPlt

Figure 6.11 Functional Analysis Box

Besides functional assessment for each individual component, the assessment for all parts in product structure can be viewed and edited from the *Functional Analysis (for* *parts in overall assembly) Dialog* Figure 6.13. This *dialog* can be brought out from a 'pop-up' menu in the product structure by clicking the icon of the Overall Assembly, *WiperMotor* (blue coloured), in Figure 6.12.

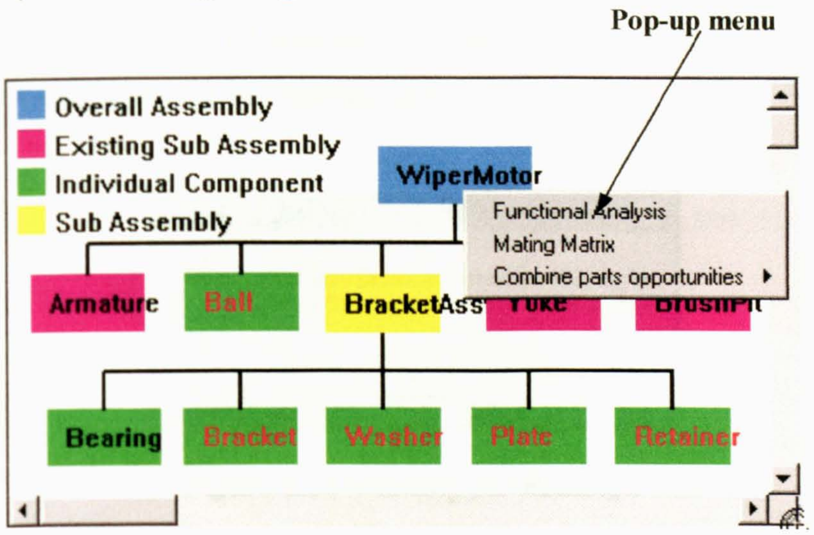
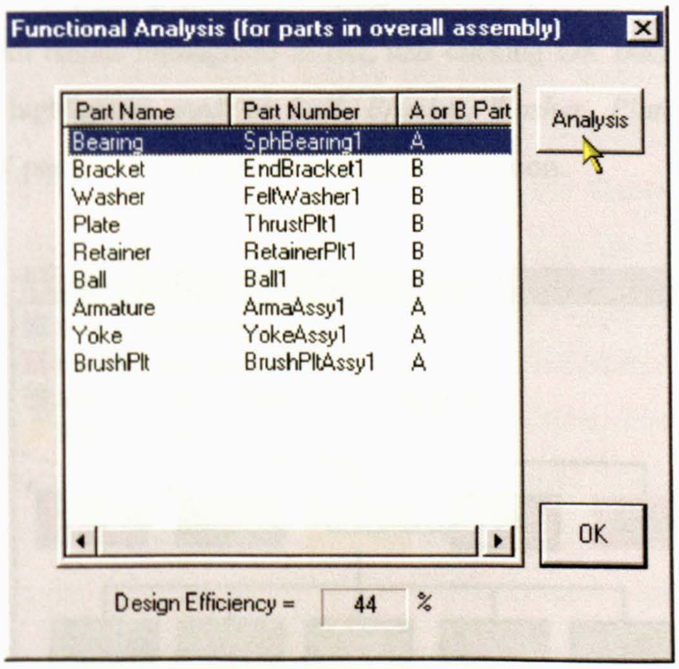


Figure 6.12 Functional/Mating Analysis Popup Menu


Functional Analysis (for parts in overall assembly)

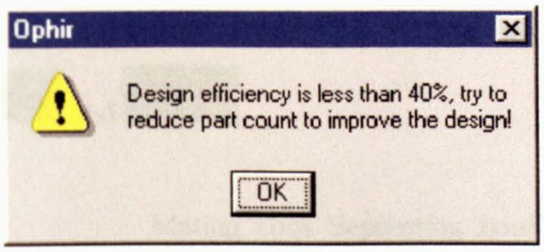
Part Name	Part Number	A or B Part
Bearing	SphBearing1	A
Bracket	EndBracket1	B
Washer	FeltWasher1	B
Plate	ThrustPlt1	B
Retainer	RetainerPlt1	B
Ball	Ball1	B
Armature	ArmaAssy1	A
Yoke	YokeAssy1	A
BrushPlt	BrushPltAssy1	A

Design Efficiency = 44 %

Figure 6.13 Functional Analysis for Overall Assembly Box

In the *Functional Analysis (for parts in overall assembly) Dialog*, the results of functional assessment ('A' part or 'B' part) of all parts in the product structure are listed.

The design efficiency of the product is shown in the bottom of the *dialog*. If the design efficiency is less than 40%, warning is issued as Figure 6.14. Selecting any part in the list, such as *Bearing*, then clicking *Analysis* button (pointed by cursor) will further bring up previously described *Functional Analysis Dialog* Figure 6.11 for conducting the functional assessment for the selected part.



Design efficiency is less than 40%, try to reduce part count to improve the design!

Figure 6.14 Warning for Redesign

Any changes confirmed from the *Functional Analysis Dialogs* or from the **PCA** reasoning will show immediately on the product structure Figure 6.15 in colour of part names and in colour of part icon borders. Thus the designer can instantly be aware of the changes. After inputting possible missing data for parts which are functionally important, but with names highlighted in red, and clicking *OK* button to confirm the inputs, the parts highlighted, such as *Ball*, *Bracket*, *Washer*, *Plate* and *Retainer* in Figure 6.15 are 'B' parts. They are candidates for elimination.

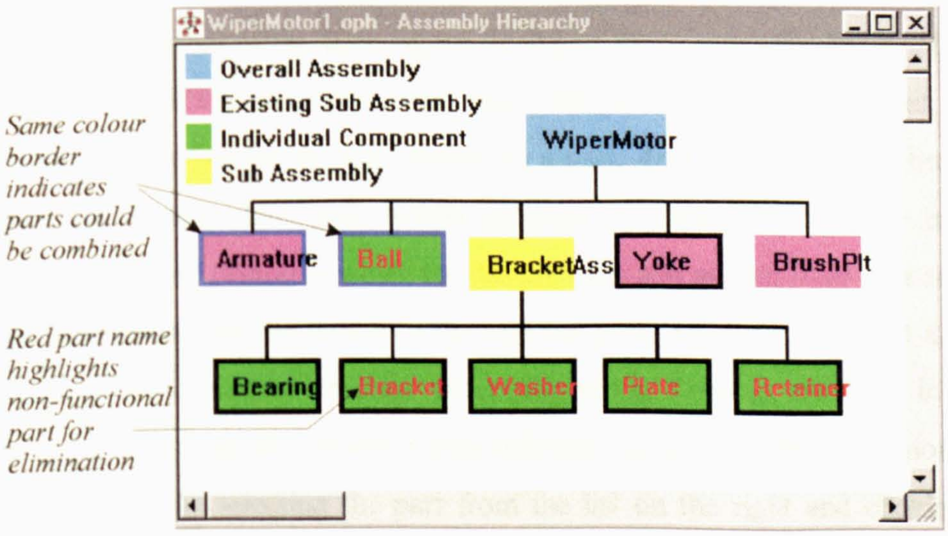

WiperMotor1.oph - Assembly Hierarchy
Overall Assembly
Existing Sub Assembly
Individual Component
Sub Assembly
WiperMotor
Armature
Ball
BracketAss
Yoke
BrushPlt
Bearing
Bracket
Washer
Plate
Retainer
Same colour border indicates parts could be combined
Red part name highlights non-functional part for elimination

Figure 6.15 Highlighted Parts for Combination and Elimination

Besides highlighting 'B' parts with names in red, the **PCA** reasons candidate parts for combination based on parts mating information entered by designer and the functional parameters mentioned above. In the Ophir environment, in addition to enter mating information by clicking relevant faces of components as in a conventional CAD system, the designer can interactively input desirable mating information earlier before detailed geometry has been created. This is achieved by clicking *Mating Tool* from the *Product Structure Tool Bar* Figure 6.16 and then clicking the relevant part icons in product structure, such as **Bearing** and Bracket.

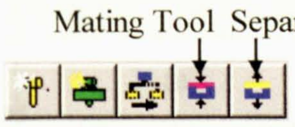
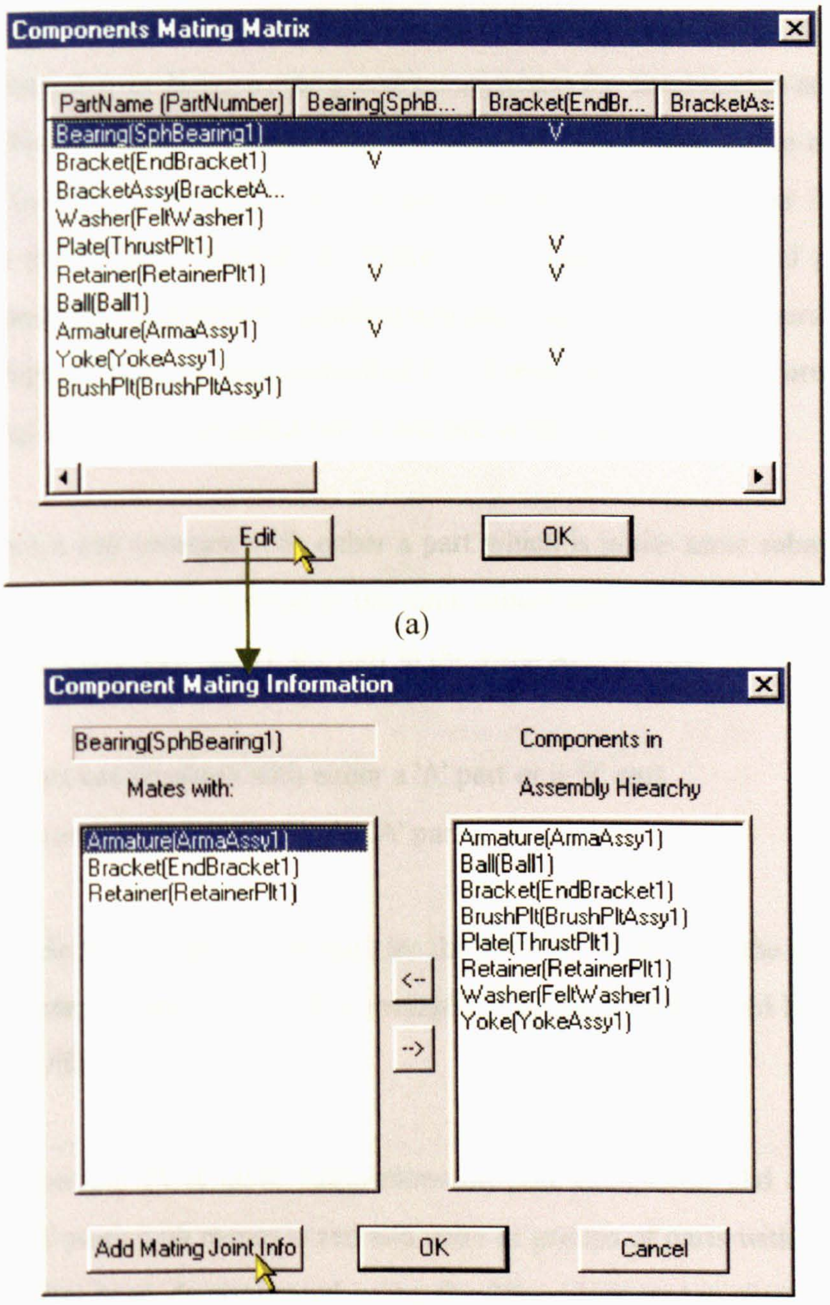


Figure 6.16 Product Structure Tool Bar

Click *Separating Tool* and the relevant part icons will put "To be separated" constraint on the parts be clicked. All the current existing mating information can be viewed from *Components Mating Matrix* Figure 6.17 (a). In the matrix, each tick corresponds two parts of the relevant row and column, such as the first tick corresponds *Bracket* and *Bearing*. These two corresponding parts are mating parts. Selecting any part from the *Component Mating Matrix* and clicking *Edit* button from the *Matrix* box will bring up the *Component Mating Information Dialog* Figure 6.17 (b). In this dialog, the part name and part number of the selected part (call part x), *Bearing(SphBearing1)*, appears on top left, and all the other parts that mate with part x: *Amature*, *Bracket*, *Retainer*, are shown in a list box on the left. Selecting a part, *Armature*, from this list box, and clicking *Add Mating Joint Info* button (pointed by cursor) will further bring up a *Mating Joint Detail Dialog* in which the designer can add *mating joint type* and *joining process* information for the liaison of the selected part, *Amature*, with part x, *Bearing*. All the parts in the product structure except for part x, *Bearing*, are listed in a list box on the right. The designer can edit mating information, e.g. to add an additional mating part of *Bearing*, by selecting the part from the list on the right and clicking the *left pointer arrow* in the middle. He can also delete any undesirable mating part by selecting it from the list on the left and clicking the *right pointer arrow*.


Components Mating Matrix

PartName (PartNumber)	Bearing(SphB...	Bracket(EndBr...	BracketAs:
Bearing(SphBearing1)		V	
Bracket(EndBracket1)	V		
BracketAssy(BracketA...			
Washer(FeltWasher1)			
Plate(ThrustPlt1)		V	
Retainer(RetainerPlt1)	V	V	
Ball(Ball1)			
Armature(ArmaAssy1)	V		
Yoke(YokeAssy1)		V	
BrushPlt(BrushPltAssy1)			

(a)

Component Mating Information

Bearing(SphBearing1)

Mates with:
Armature(ArmaAssy1)
Bracket(EndBracket1)
Retainer(RetainerPlt1)

Components in Assembly Hiearchy
Armature(ArmaAssy1)
Ball(Ball1)
Bracket(EndBracket1)
BrushPlt(BrushPltAssy1)
Plate(ThrustPlt1)
Retainer(RetainerPlt1)
Washer(FeltWasher1)
Yoke(YokeAssy1)

(b)

Figure 6.17 (a) Component Mating Matrix,
(b) Component Mating Information Dialog

For a new product design, part mating information can be entered by the designer. To redesign an existing product, the mating information may be inferred by geometric reasoning as detailed geometry may already be available. However, the involved computing could be expensive. Several references have detailed the research on automatic inference of the mating information from CAD model [176, 108, 87, 177].

Based on all the available mating information, functional parameters and additional constraints imposed by designer, the possible candidates for combination are inferred by the **PCA**. Finally, pairs or groups of parts which are very likely to be combined are highlighted in same coloured borders of part icon, such as blue border for *Armature* and *Ball* in the product structure in Figure 6.15. The algorithms and procedure of **PCA** to reason the candidates for combination and assign different colours for different pairs or groups of parts are implemented as C++ functions. Some rules are also used in the reasoning. For example, listed below are two rules used:

1. **If** part x can combine with either a part which is in the same subassembly with part x or a part which is not in the same subassembly with part x
   then combine part x with the part in the same subassembly

2. **If** a part can combine with either a 'A' part or a 'B' part
   then combine the part with the 'A' part

The second rule has actually been used by the **PCA** in suggesting the combination of *ball* and *Armature* in the wiper motor example. Otherwise *ball* could be suggested to combine with *Plate*.

Until now, how the **PCA** gives suggestions on part elimination and combination by highlighting 'B' parts with names in red and pairs or groups of parts with same coloured icon borders has been demonstrated using the 20w wiper motor example. Actually, DFA evaluation for this wiper motor had been conducted manually before, and there is a proposed redesign solution. The exploded view of the redesign solution is as Figure 6.18, and its product structure is as Figure 6.19. The proposed redesign uses one piece formed *Can* with a *Bush* instead of the original *Yoke* and the other five parts: B*earing*, *Bracket*, *Washer*, *Plate*, and *Retainer* in *Bracket Assembly*. The suggestions for combination from the **PCA** actually match the redesign solution. The redesign also uses one piece moulded "End Cap and Bearing" with combined "Brush Spring and Connectors" to replace 'BrushPlate' Assembly and 2 bolts and nuts. The elimination of 2 bolts and 2 nuts resulted from using snap fit to assemble the *Cap* on the top of the

Can. As the material of the proposed *Cap* is plastic, it is dielectric, and it is also suitable for snap fit assembly. In the demonstration using the simplified wiper motor, the 'BrushPlate' Assembly has not been analysed in detail.

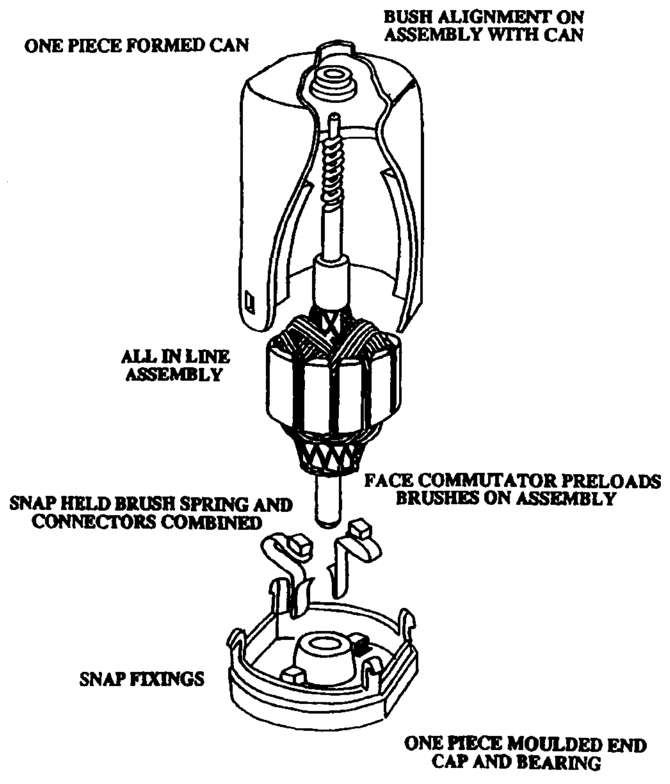

BUSH ALIGNMENT ON ASSEMBLY WITH CAN
ONE PIECE FORMED CAN
ALL IN LINE ASSEMBLY
FACE COMMUTATOR PRELOADS BRUSHES ON ASSEMBLY
SNAP HELD BRUSH SPRING AND CONNECTORS COMBINED
SNAP FIXINGS
ONE PIECE MOULDED END CAP AND BEARING

Figure 6.18 Explored View of the Wiper Motor Redesign

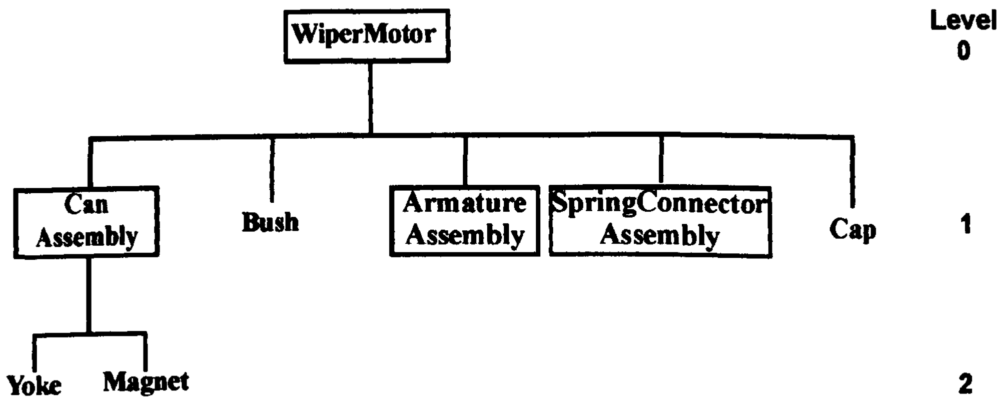
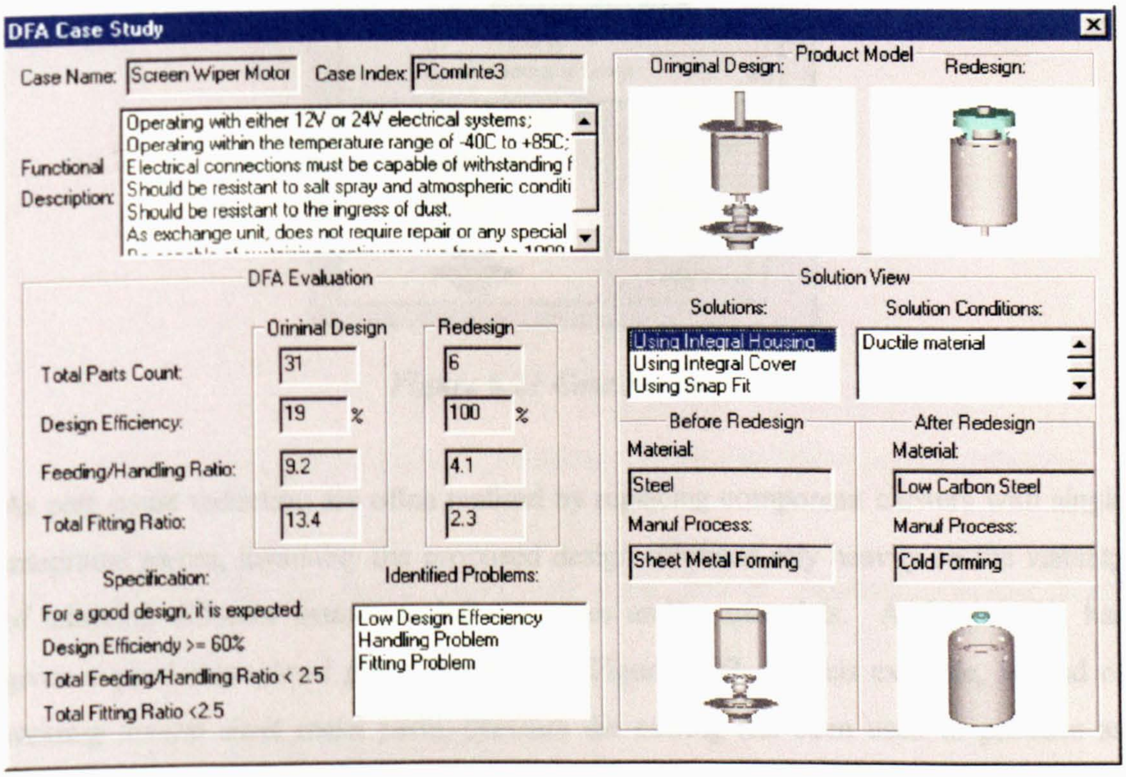


Figure 6.19 Assembly Structure of the Wiper Motor Redesign

This redesign was not actually put into production because it was really too late to change the product design when the DFA analysis was conducted. It is felt that if the redesign solution could be proved to be practical, it will be a useful experience for future product design in realising part consolidation. This is helpful as it is normally more difficult to generate redesign solution than to highlight opportunities for improvement. With large varieties of product, the solution space may be huge, and much effort in searching the solution may achieve nothing. A suitable solution can be found or can be found easily or not very much depends on the command of the advanced manufacturing technologies and the experience accumulated from the past. So apart from manufacturing and assembly knowledge, successful DFA case studies are useful to provide redesign solutions for creating assembly-oriented design.

There are about 40 to 60 DFA case studies mentioned in [8, 10, 44] and some 40 DFA case studies mention in [29]. Many of the redesigns have been used in industries and brought significant savings [8, 10]. The integration of case study solutions in design environment is useful to give suggestions for a similar situation that could be encountered in the future. Actually, from a historical point of view, the human design process is often based on past experience, established products and structures. But the question is how to represent the case studies, the past experience in design environment? How to retrieve the relative information and apply it when necessary? Case-based reasoning is certainly the most suitable means to incorporate the successful DFA case studies into design environment. As currently the DFA case studies are not numerous, it is not sure that it is worth it or not to put a large effort on the integration work. But for demonstration purposes, a case base has been established in the Ophir environment with a few successful DFA case studies to support the generation of redesign solutions. Case studies including the wiper motor and other efficient design examples have been categorised, indexed and stored in the case base. Pattern-matching algorithms have been developed to search the case base for successful and efficient designs that are relevant to a new design. According to the degree of relevance to the new design, cases and examples are retrieved and a few of the most relevant cases are presented to the designer. The measurement of relevance is based on the similarity of a stored design with the current design: the trigram distance of strings contained in the name and functional descriptions of the two designs. The case retrieved normally contains the original and redesign product models, the identified problems of the original design by DFA evaluation, and the solutions used to solve the problems. In the wiper motor example Figure 6.20, the redesign used *integral housing, integral cover* and *snap fit*. If the designer highlights one of these partial solutions, more detailed information will be provided, such as in the highlighted partial solution "using integral housing", the redesign replaces the *Bracket Assembly* and the *Yoke* with an one piece *Can* and a *Bush* that is shown graphically on the bottom right of the figure. But the *Can* has to be made of ductile material, such as low carbon steel for cold forming.


DFA Case Study

Case Name: Screen Wiper Motor
Case Index: PComInte3

Functional Description:
Operating with either 12V or 24V electrical systems;
Operating within the temperature range of -40C to +85C;
Electrical connections must be capable of withstanding f
Should be resistant to salt spray and atmospheric conditi
Should be resistant to the ingress of dust.
As exchange unit, does not require repair or any special

DFA Evaluation

	Orininal Design	Redesign
Total Parts Count:	31	6
Design Efficiency:	19 %	100 %
Feeding/Handling Ratio:	9.2	4.1
Total Fitting Ratio:	13.4	2.3

Specification:
For a good design, it is expected:
Design Efficiendy >= 60%
Total Feeding/Handling Ratio < 2.5
Total Fitting Ratio <2.5

Identified Problems:
Low Design Effeciency
Handling Problem
Fitting Problem

Product Model
Oringinal Design:
Redesign:

Solution View
Solutions:
Using Integral Housing
Using Integral Cover
Using Snap Fit

Solution Conditions:
Ductile material

Before Redesign	After Redesign
Material: Steel	Material: Low Carbon Steel
Manuf Process: Sheet Metal Forming	Manuf Process: Cold Forming

Figure 6.20 DFA Case Study Dialog

DFA case studies can also be browsed in the Ophir environment from the *Case Browser* Figure 6.21. The *Browser* can be brought out at any time from the design menu. It stays and keeps active until the designer closes it. It lists all the categorised case studies and DFA examples. Selecting any one from the list, such as *Screen Wiper Motor*, and clicking *Case View* button will bring up the *DFA Case Study Dialog*. In this case, it is the wiper motor case study as in Figure 6.20. Clicking *Model View* button from the *Browser*, product models of the original design and redesign (if available) will be presented in the design environment.

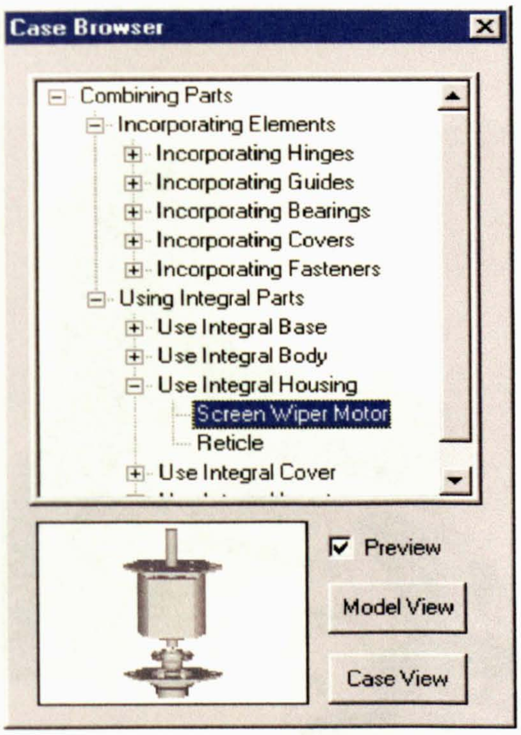


Figure 6.21 Case Browser

As part count reduction are often realised by replacing component clusters with single integrated pieces, invariably the proposed design solutions rely heavily on the viability of adopting different manufacturing processes and/or materials. Andreasen [5] has given a good example of part consolidation Figure 6.22. In this example, instead of welding several sheet metal parts, pressure die casting has been used to produce an integrated single piece corner element. This leads part count reduction from 3 to 1 and the elimination of welding assembly operation. There are many such successful examples. However, selection of the right process is not always simple and obvious, in some cases there are several processes that can be used and the selection depends on a large number of factors [178]. In such circumstances, expert support may be inevitable needed for decision support as a designer is often not a manufacturing expert. There are other occasions when the integrated single piece component may be too complicated to manufacture economically by current manufacturing technology, such as an example given by Anderson [5, page 101]. So manufacturing cost estimation is needed to make the trade-off between part consolidation and production cost. This is one of the steps in the BDI method and the CSC technique. Besides production cost, functional requirements and assembly operation difficulties should also be considered carefully in part consolidation [179].

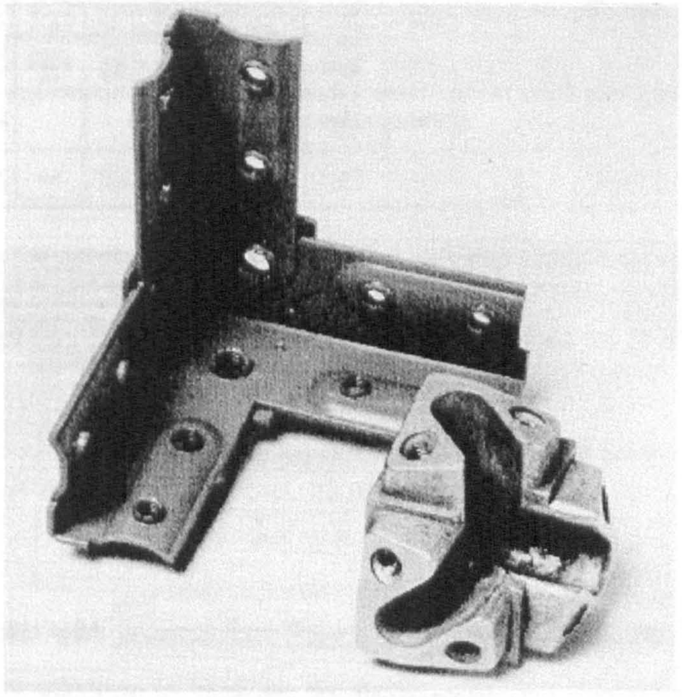
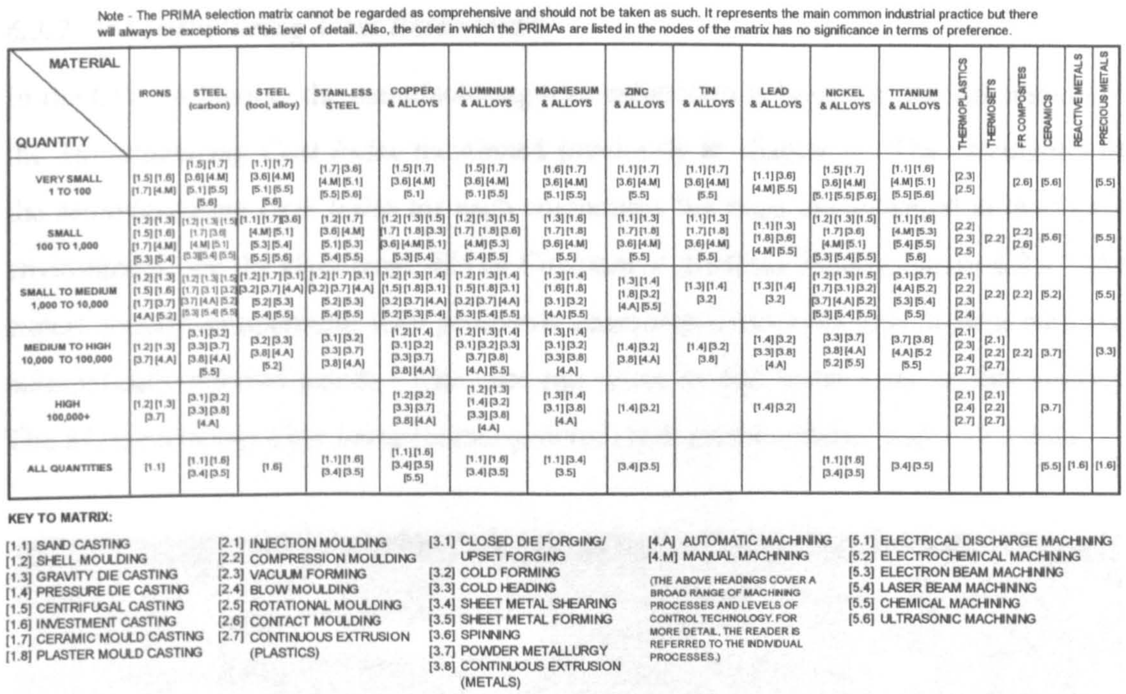


Figure 6.22 Part Consolidation by Adopting Different Manufacturing Process

6.3 Support for Process Selection and Early Manufacturing Cost Estimation

6.3.1 Manufacturing Process Selection

The manufacturing process information maps (PRIMAs) [131] can be used to guide the manufacturing process selection. A PRIMAs selection matrix Figure 6.23 can be stored in the General Knowledge Base of the Ophir environment to provide the knowledge for the compatibility checking of process to material.



Note - The PRIMA selection matrix cannot be regarded as comprehensive and should not be taken as such. It represents the main common industrial practice but there will always be exceptions at this level of detail. Also, the order in which the PRIMAs are listed in the nodes of the matrix has no significance in terms of preference.

MATERIAL / QUANTITY	IRONS	STEEL (carbon)	STEEL (tool, alloy)	STAINLESS STEEL	COPPER & ALLOYS	ALUMINIUM & ALLOYS	MAGNESIUM & ALLOYS	ZINC & ALLOYS	TIN & ALLOYS	LEAD & ALLOYS	NICKEL & ALLOYS	TITANIUM & ALLOYS	THERMOPLASTICS	THERMOSETS	FR COMPOSITES	CERAMICS	REACTIVE METALS	PRECIOUS METALS
VERY SMALL 1 TO 100	[1.5] [1.6] [1.7] [4.M]	[1.5] [1.7] [3.6] [4.M] [5.1] [5.5] [5.6]	[1.1] [1.7] [3.6] [4.M] [5.1] [5.5] [5.6]	[1.7] [3.6] [4.M] [5.1] [5.5] [5.6]	[1.5] [1.7] [3.6] [4.M] [5.1]	[1.5] [1.7] [3.6] [4.M] [5.1] [5.5]	[1.6] [1.7] [3.6] [4.M] [5.1] [5.5]	[1.1] [1.7] [3.6] [4.M] [5.5]	[1.1] [1.7] [3.6] [4.M] [5.5]	[1.1] [3.6] [4.M] [5.5]	[1.5] [1.7] [3.6] [4.M] [5.1] [5.5] [5.6]	[1.1] [1.6] [4.M] [5.1] [5.5] [5.6]	[2.3] [2.5]		[2.6]	[5.6]		[5.5]
SMALL 100 TO 1,000	[1.2] [1.3] [1.5] [1.6] [1.7] [4.M] [5.3] [5.4]	[1.2] [1.3] [1.5] [1.7] [3.6] [4.M] [5.1] [5.3] [5.4] [5.5]	[1.1] [1.7] [3.6] [4.M] [5.1] [5.3] [5.4] [5.5] [5.6]	[1.2] [1.7] [3.6] [4.M] [5.1] [5.3] [5.4] [5.5]	[1.2] [1.3] [1.5] [1.7] [1.8] [3.3] [3.6] [4.M] [5.1] [5.3] [5.4]	[1.2] [1.3] [1.5] [1.7] [1.8] [3.6] [4.M] [5.3] [5.4] [5.5]	[1.3] [1.6] [1.7] [1.8] [3.6] [4.M] [5.5]	[1.1] [1.3] [1.7] [1.8] [3.6] [4.M] [5.5]	[1.1] [1.3] [1.7] [1.8] [3.6] [4.M] [5.5]	[1.1] [1.3] [1.8] [3.6] [4.M] [5.5]	[1.2] [1.3] [1.5] [1.7] [3.6] [4.M] [5.1] [5.3] [5.4] [5.5]	[1.1] [1.6] [4.M] [5.3] [5.4] [5.5] [5.6]	[2.2] [2.3] [2.5]	[2.2]	[2.2] [2.6]	[5.6]		[5.5]
SMALL TO MEDIUM 1,000 TO 10,000	[1.2] [1.3] [1.5] [1.6] [1.7] [3.7] [4.A] [5.2]	[1.2] [1.3] [1.5] [1.7] [3.1] [3.2] [3.7] [4.A] [5.2] [5.3] [5.4] [5.5]	[1.2] [1.7] [3.1] [3.2] [3.7] [4.A] [5.2] [5.3] [5.4] [5.5]	[1.2] [1.7] [3.1] [3.2] [3.7] [4.A] [5.2] [5.3] [5.4] [5.5]	[1.2] [1.3] [1.4] [1.5] [1.8] [3.1] [3.2] [3.7] [4.A] [5.2] [5.3] [5.4]	[1.2] [1.3] [1.4] [1.5] [1.8] [3.1] [3.2] [3.7] [4.A] [5.3] [5.4] [5.5]	[1.3] [1.4] [1.6] [1.8] [3.1] [3.2] [4.A] [5.5]	[1.3] [1.4] [1.8] [3.2] [4.A] [5.5]	[1.3] [1.4] [3.2]	[1.3] [1.4] [3.2]	[1.2] [1.3] [1.5] [1.7] [3.1] [3.2] [3.7] [4.A] [5.2] [5.3] [5.4] [5.5]	[3.1] [3.7] [4.A] [5.2] [5.3] [5.4] [5.5]	[2.1] [2.2] [2.3] [2.4]	[2.2]	[2.2]	[5.2]		[5.5]
MEDIUM TO HIGH 10,000 TO 100,000	[1.2] [1.3] [3.7] [4.A]	[3.1] [3.2] [3.3] [3.7] [3.8] [4.A] [5.5]	[3.2] [3.3] [3.8] [4.A] [5.2]	[3.1] [3.2] [3.3] [3.7] [3.8] [4.A]	[1.2] [1.4] [3.1] [3.2] [3.3] [3.7] [3.8] [4.A]	[1.2] [1.3] [1.4] [3.1] [3.2] [3.3] [3.7] [3.8] [4.A] [5.5]	[1.3] [1.4] [3.1] [3.2] [3.3] [3.8] [4.A]	[1.4] [3.2] [3.8] [4.A]	[1.4] [3.2] [3.8]	[1.4] [3.2] [3.3] [3.8] [4.A]	[3.3] [3.7] [3.8] [4.A] [5.2] [5.5]	[3.7] [3.8] [4.A] [5.2 [5.5]	[2.1] [2.3] [2.4] [2.7]	[2.1] [2.2] [2.7]	[2.2]	[3.7]		[3.3]
HIGH 100,000+	[1.2] [1.3] [3.7]	[3.1] [3.2] [3.3] [3.8] [4.A]			[1.2] [3.2] [3.3] [3.7] [3.8] [4.A]	[1.2] [1.3] [1.4] [3.2] [3.3] [3.8] [4.A]	[1.3] [1.4] [3.1] [3.8] [4.A]	[1.4] [3.2]		[1.4] [3.2]			[2.1] [2.4] [2.7]	[2.1] [2.2] [2.7]		[3.7]		
ALL QUANTITIES	[1.1]	[1.1] [1.6] [3.4] [3.5]	[1.6]	[1.1] [1.6] [3.4] [3.5]	[1.1] [1.6] [3.4] [3.5] [5.5]	[1.1] [1.6] [3.4] [3.5]	[1.1] [3.4] [3.5]	[3.4] [3.5]			[1.1] [1.6] [3.4] [3.5]	[3.4] [3.5]				[5.5]	[1.6]	[1.6]

KEY TO MATRIX:

[1.1] SAND CASTING
[1.2] SHELL MOULDING
[1.3] GRAVITY DIE CASTING
[1.4] PRESSURE DIE CASTING
[1.5] CENTRIFUGAL CASTING
[1.6] INVESTMENT CASTING
[1.7] CERAMIC MOULD CASTING
[1.8] PLASTER MOULD CASTING

[2.1] INJECTION MOULDING
[2.2] COMPRESSION MOULDING
[2.3] VACUUM FORMING
[2.4] BLOW MOULDING
[2.5] ROTATIONAL MOULDING
[2.6] CONTACT MOULDING
[2.7] CONTINUOUS EXTRUSION (PLASTICS)

[3.1] CLOSED DIE FORGING/ UPSET FORGING
[3.2] COLD FORMING
[3.3] COLD HEADING
[3.4] SHEET METAL SHEARING
[3.5] SHEET METAL FORMING
[3.6] SPINNING
[3.7] POWDER METALLURGY
[3.8] CONTINUOUS EXTRUSION (METALS)

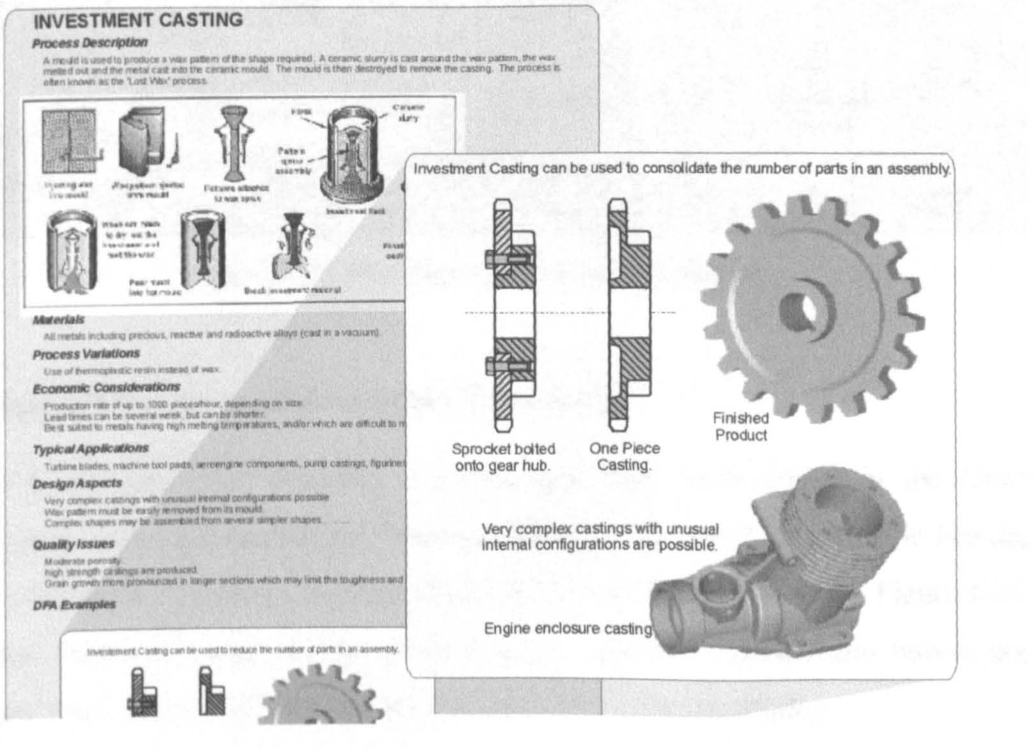
[4.A] AUTOMATIC MACHINING
[4.M] MANUAL MACHINING

(THE ABOVE HEADINGS COVER A BROAD RANGE OF MACHINING PROCESSES AND LEVELS OF CONTROL TECHNOLOGY. FOR MORE DETAIL, THE READER IS REFERRED TO THE INDIVIDUAL PROCESSES.)

[5.1] ELECTRICAL DISCHARGE MACHINING
[5.2] ELECTROCHEMICAL MACHINING
[5.3] ELECTRON BEAM MACHINING
[5.4] LASER BEAM MACHINING
[5.5] CHEMICAL MACHINING
[5.6] ULTRASONIC MACHINING

Figure 6.23 PRIMA Selection Matrix

Also DFA examples with process data Figure 6.24 stored in the case base can help the designer in process selection to facilitate the design solutions in terms of DFA.



INVESTMENT CASTING

Process Description

A mould is used to produce a wax pattern of the shape required. A ceramic slurry is cast around the wax pattern, the wax melted out and the metal cast into the ceramic mould. The mould is then destroyed to remove the casting. The process is often known as the 'Lost Wax' process.

Materials

All metals including precious, reactive and radioactive alloys (cast in a vacuum).

Process Variations

Use of thermoplastic resin instead of wax.

Economic Considerations

Production rate of up to 1000 pieces/hour, depending on size.
Lead times can be several week, but can be shorter.
Best suited to metals having high melting temperatures, and/or which are difficult to m

Typical Applications

Turbine blades, machine tool pads, aeroengine components, pump castings, figurines

Design Aspects

Very complex castings with unusual internal configurations possible.
Wax pattern must be easily removed from its mould.
Complex shapes may be assembled from several simpler shapes.

Quality Issues

Moderate porosity.
high strength castings are produced.
Grain growth more pronounced in longer sections which may limit the toughness and

DFA Examples

Investement Casting can be used to reduce the number of parts in an assembly.

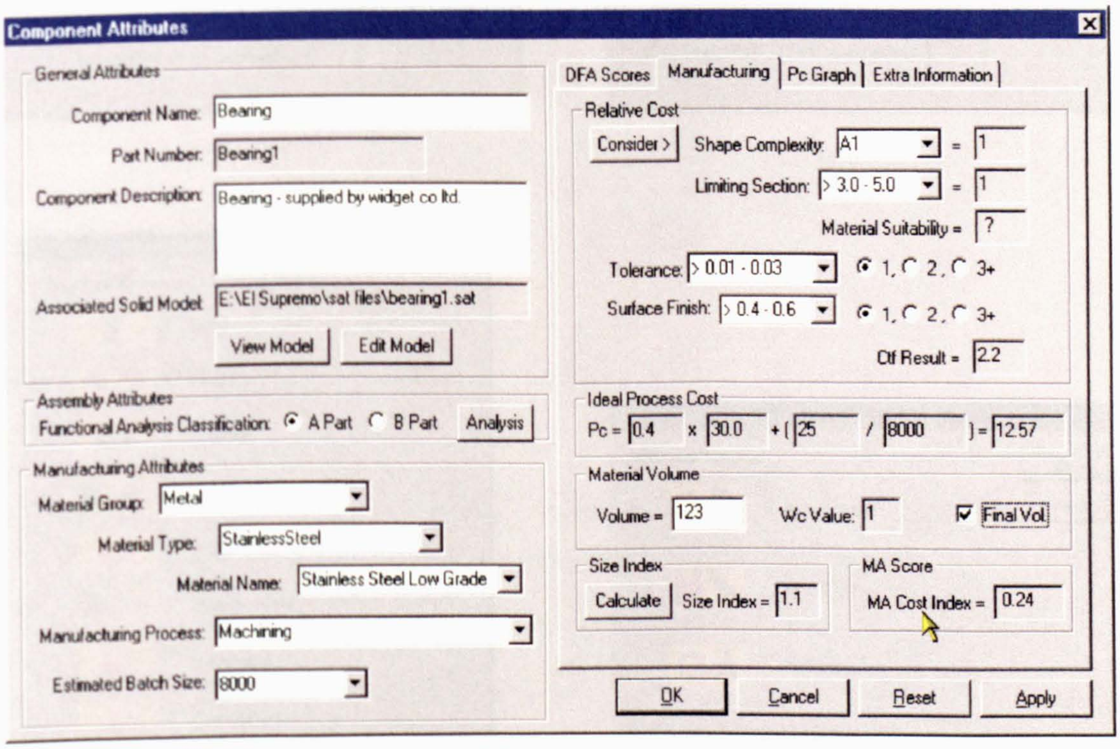
Investment Casting can be used to consolidate the number of parts in an assembly.

Very complex castings with unusual internal configurations are possible.

Figure 6.24 DFA Examples with Process Data

6.3.2 Manufacturing Cost Estimation

In the CSC technique, the manufacturing cost estimation is based on the calculation of the *Manufacturing Cost Index* mentioned previously in chapter 2. The calculation of the *Manufacturing Cost Index* for each component has been implemented in the Ophir environment, see the right hand side of *Component Attribute Dialog* Figure 6.25. The expert support cooperating with geometric reasoning overwrites default data with the automatically inferred results. The user can select or edit some items in this *Dialog*. The *Manufacturing Cost Index* (cursor pointed) reflects the inferred and edited data.



Component Attributes

General Attributes
Component Name: Bearing
Part Number: Bearing1
Component Description: Bearing - supplied by widget co ltd
Associated Solid Model: E:\EI Supremo\sat files\bearing1.sat

Assembly Attributes
Functional Analysis Classification: [✓ A Part] B Part

Manufacturing Attributes
Material Group: Metal
Material Type: StainlessSteel
Material Name: Stainless Steel Low Grade
Manufacturing Process: Machining
Estimated Batch Size: 8000

Relative Cost
Shape Complexity: A1 = 1
Limiting Section: > 3.0 - 5.0 = 1
Material Suitability = ?
Tolerance: > 0.01 - 0.03 [✓ 1], 2, 3+
Surface Finish: > 0.4 - 0.6 [✓ 1], 2, 3+
Cf Result = 2.2

Ideal Process Cost
Pc = 0.4 x 30.0 + (25 / 8000) = 12.57

Material Volume
Volume = 123 Wc Value: 1 [✓ Final Vol]

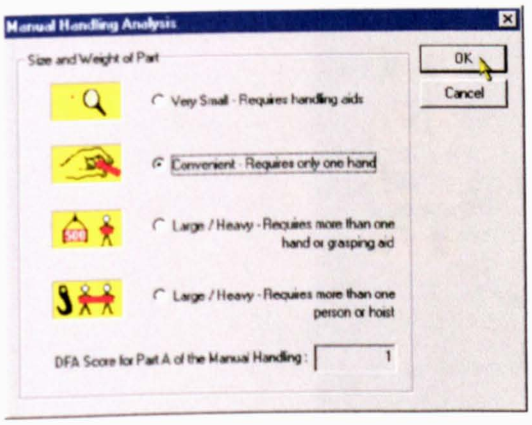
Size Index
Size Index = 1.1

MA Score
MA Cost Index = 0.24

Figure 6.25 Manufacturing Cost Estimation

6.4 Support for Ease of Assembly Operation

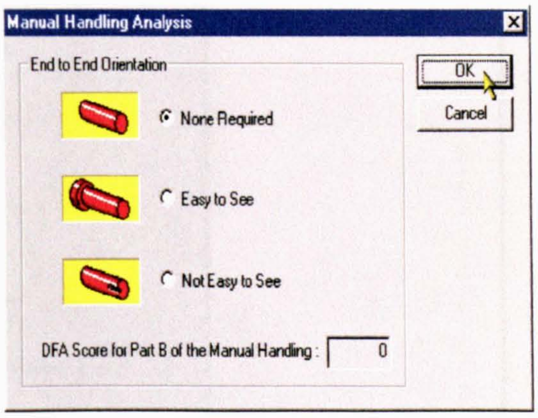
The DFA evaluation for assembly operation has been implemented in the Ophir environment. This includes Manual Handling Analysis Figure 6.27, Automatic Feeding Analysis Figure 6.27, Fitting Analysis Figure 6.28 and Gripping Analysis Figure 6.29. All these *Analysis Dialogs* can be brought up by clicking related *radio* button and 'calculate' button from *DFA Scores* tab in *Component Attribute Dialog*.


Manual Handling Analysis

Size and Weight of Part

( ) Very Small - Requires handling aids
[✓ Convenient - Requires only one hand]
( ) Large / Heavy - Requires more than one hand or grasping aid
( ) Large / Heavy - Requires more than one person or hoist

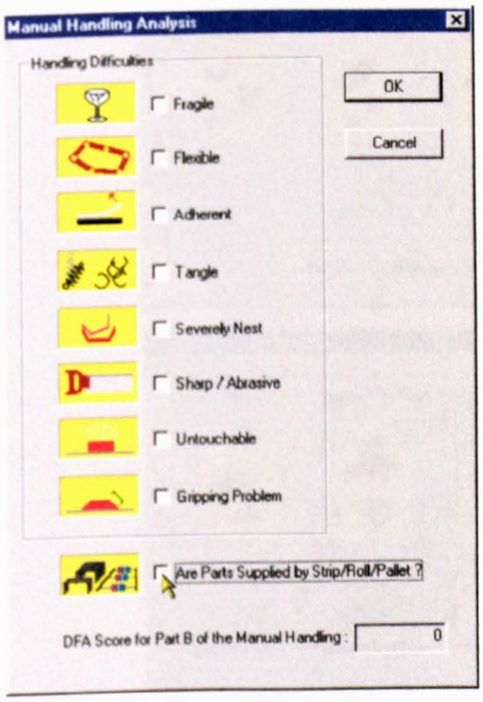
DFA Score for Part A of the Manual Handling : 1


Manual Handling Analysis

End to End Orientation

[✓ None Required]
( ) Easy to See
( ) Not Easy to See

DFA Score for Part B of the Manual Handling : 0

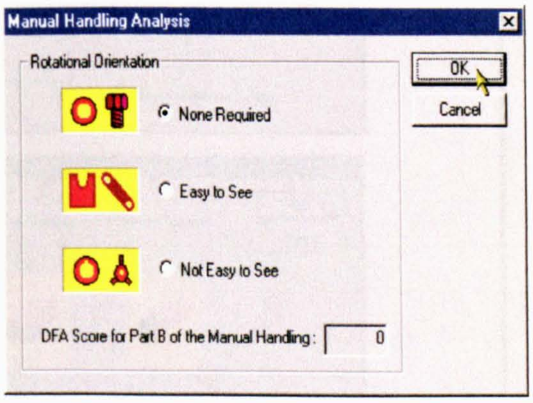

Manual Handling Analysis

Handling Difficulties

☐ Fragile
☐ Flexible
☐ Adherent
☐ Tangle
☐ Severely Nest
☐ Sharp / Abrasive
☐ Untouchable
☐ Gripping Problem

☐ Are Parts Supplied by Strip/Roll/Pallet ?

DFA Score for Part B of the Manual Handling : 0

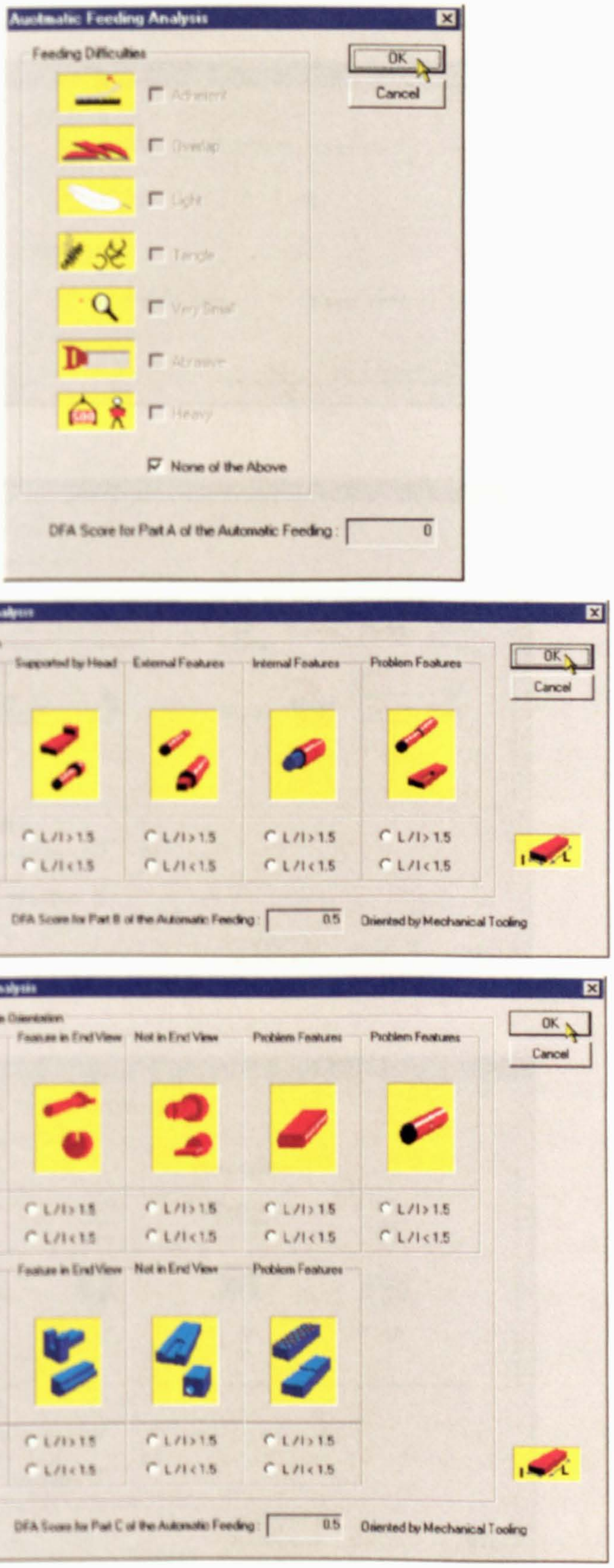

Manual Handling Analysis

Rotational Orientation

[✓ None Required]
( ) Easy to See
( ) Not Easy to See

DFA Score for Part B of the Manual Handling : 0

Figure 6.26 Manual Handling Analysis



[✓ None of the Above]

DFA Score for Part A of the Automatic Feeding : 0

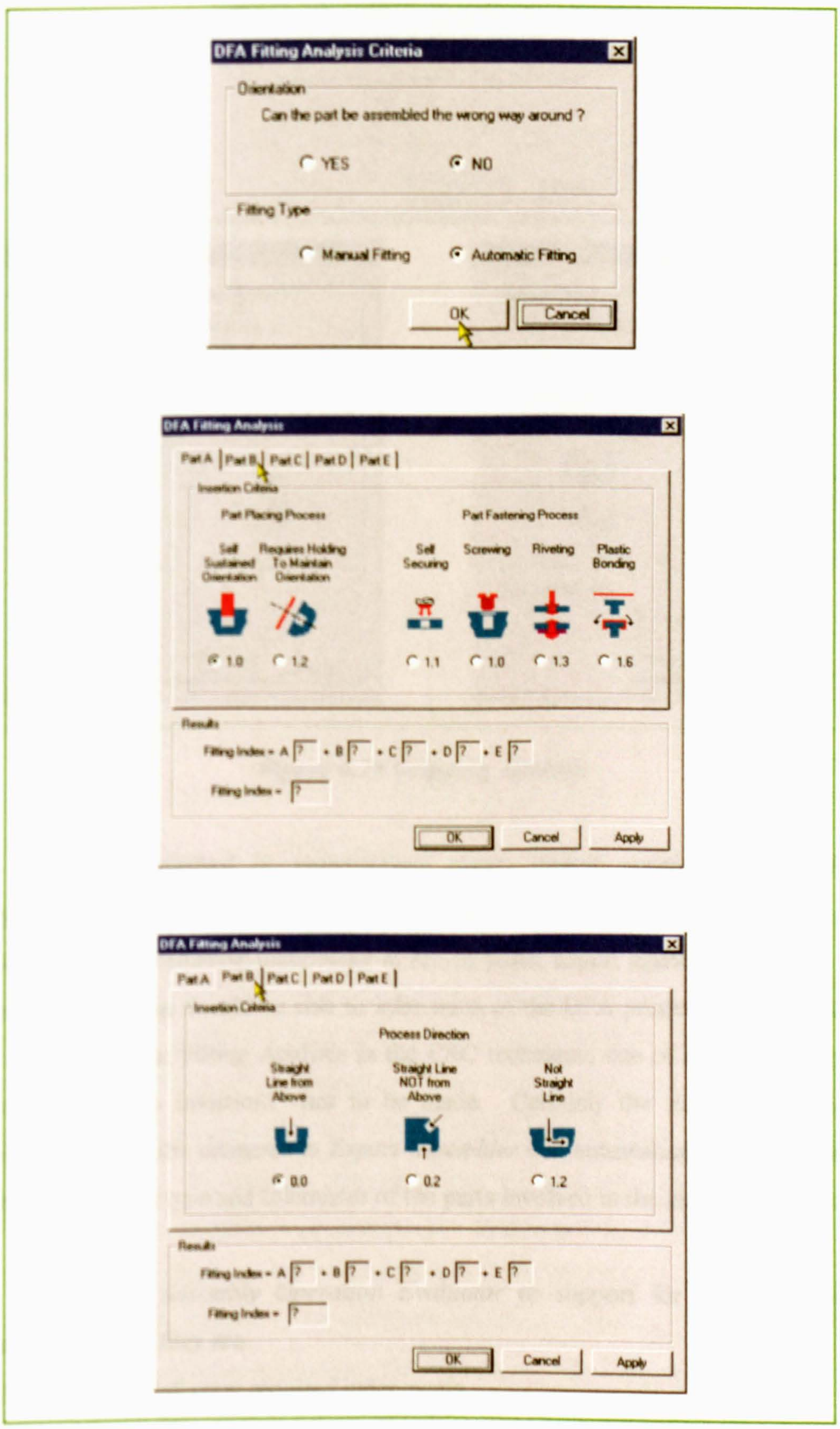
End to End Orientation — None Required: L/l>1.5 [✓ L/l<1.5]

DFA Score for Part B of the Automatic Feeding : 0.5 Oriented by Mechanical Tooling

Rotational Side to Side Orientation — None Required: L/l>1.5 [✓ L/l<1.5]

DFA Score for Part C of the Automatic Feeding : 0.5 Oriented by Mechanical Tooling

Figure 6.27 Automatic Feeding Analysis



Can the part be assembled the wrong way around ?

YES [✓ NO]

Manual Fitting [✓ Automatic Fitting]

Self Sustained Orientation [✓ 1.0]

Requires Holding To Maintain Orientation 1.2

Self Securing 1.1

Screwing 1.0

Riveting 1.3

Plastic Bonding 1.6

Fitting Index = A ? + B ? + C ? + D ? + E ?

Fitting Index = ?

Straight Line from Above [✓ 0.0]

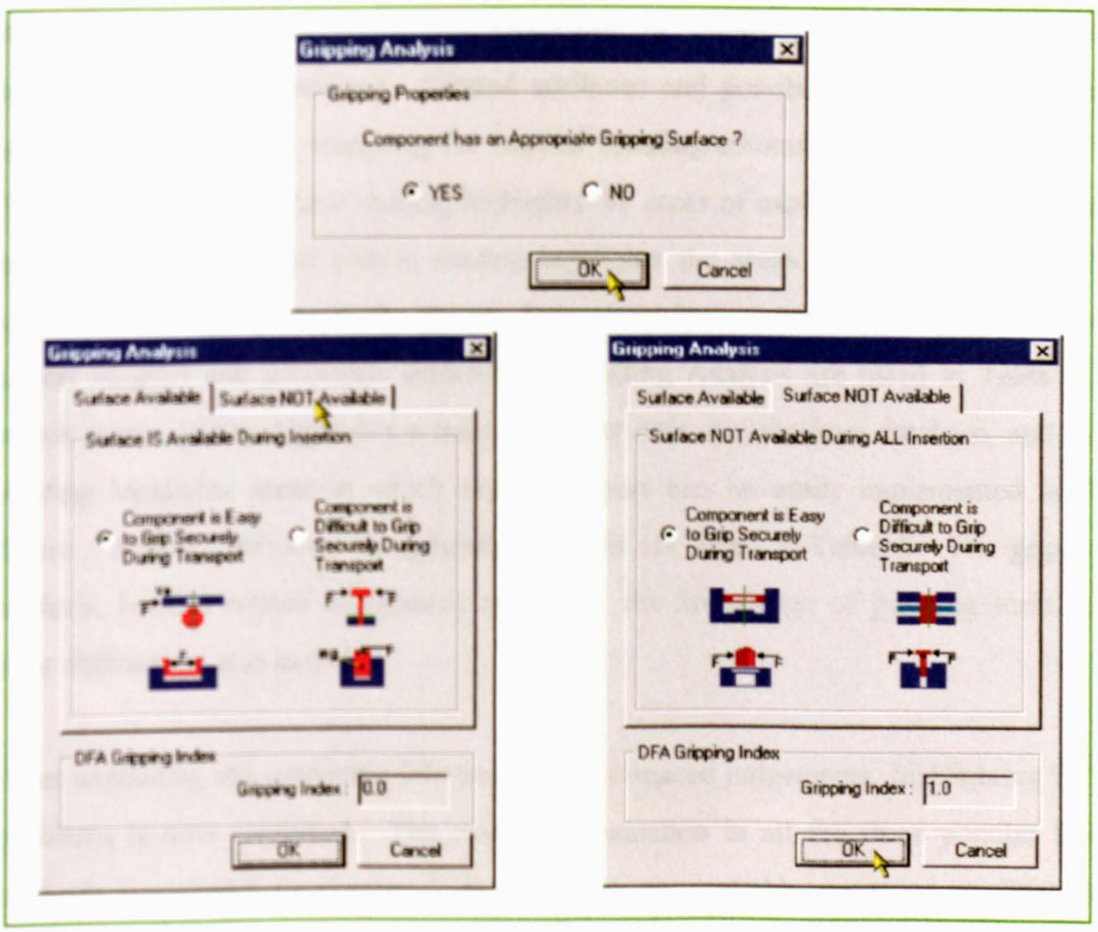
Straight Line NOT from Above 0.2

Not Straight Line 1.2

Fitting Index = A ? + B ? + C ? + D ? + E ?

Fitting Index = ?

Figure 6.28 Fitting Analysis



Gripping Properties

Component has an Appropriate Gripping Surface ?

[✓ YES] NO

Gripping Analysis

Surface Available | Surface NOT Available

Surface IS Available During Insertion

[✓ Component is Easy to Grip Securely During Transport] Component is Difficult to Grip Securely During Transport

DFA Gripping Index

Gripping Index : 0.0

Gripping Analysis

Surface Available | Surface NOT Available

Surface NOT Available During ALL Insertion

[✓ Component is Easy to Grip Securely During Transport] Component is Difficult to Grip Securely During Transport

DFA Gripping Index

Gripping Index : 1.0

Figure 6.29 Gripping Analysis

For spreadsheet method or conventional expert system approach, most of the judgements required in the analyses have to be made by the user. With the advances of computing and information technology in recent years, expert agents cooperating with geometric reasoning should be able to infer most of the DFA properties automatically. For example, in the Fitting Analysis in the CSC technique, one of the judgements "Is there resistance to insertion?" has to be made. Certainly the *Assembly Operation Evaluator*, one of the elements in *Expert Assembler* can automatically infer this based on the mating joint type and tolerances of the parts involved in the insertion operation.

The roles of the *Assembly Operation Evaluator* to support for ease of assembly operation are two; they are:

- Automatically inferring required judgements
- Highlighting handling/feeding, gripping and fitting problems

We explain them one by one, firstly the automatic inference of the required judgements.

All the judgements needed for the evaluation of assembly operation in the CSC technique have been analysed. Related attributes and possible solutions from expert support and geometric reasoning for manual handling/automatic feeding are listed in Table 3. In this table, blue shading highlights the areas of expert support implemented or in implementing, and yellow shading highlights the areas that are possible to be supported by the experts in the future. Related attributes and possible solutions from expert support and geometric reasoning for Fitting Analysis are listed in Table 4 in which green shading highlights areas easy to get data directly from database, and blue shading highlights areas in which expert support can be easily implemented in the future. Related attributes for gripping analysis are listed in Table 5. For gripping analysis, besides related component attributes, the knowledge of gripping tools and their abilities are also useful.

After explaining the automatic inference of the required judgements, highlighting DFA problems is now explained. The assembly evaluation in all the three popular DFA methods introduced in chapter 2 is based on an assembly sequence explicitly or implicitly. In the Ophir environment, facilities for the sequence construction have been provided, and joining methods and assembly actions can be defined in the sequence construction process. The definition of the joining method and assembly action is achieved by dragging a representative icon into the *Pre Process Box* or the *Insertion Box*, see Figure 6.30. In an early design stage, not all the detailed assembly operation data is available, so only a rough evaluation is possible. But the evaluation result will become more and more accurate when the process definition gets more and more detailed. Any DFA problems relating to a part, such as handling/feeding and gripping problems, will be highlighted by the *Assembly Operation Evaluator* with a red border of the part icon in the assembly sequence. For a fitting problem, liaison of the two parts involved in the insertion operation – the *Insertion Box*, will be highlighted by a red border. An example of the evaluation of a *Butterfly Valve Assembly* in the Ophir environment has highlighted fitting problems: restricted access and requiring holding to maintain orientation when assembling the *Plate* on to the *Shaft*; and fitting problem: restricted access when screwing the *Plate* on to the *Shaft*. Right clicking a highlighted *Insertion Box* will detail the fitting problems.

Table 3: Manual Handling/Automatic Feeding Analysis

	Judgements Needed	Criterion	Judgements Possibly Inferred By	Related Attributes
Size and Weight of Part	Very Small	Require handling aids	Expert Support	Part volume (can be calculated from part *mass properties*)
	Light			Part volume & Material density
	Large and/or Heavy	1. Require more than one hand or grasping aid 2. Require more than one person or hoist		Part volume & Material density
Handling Difficulties	Delicate (Fragile)	Ease of inflicted damage	Expert Support & Geometric Reasoning	Material property (ultimate elongation e<5%) Shape Dimension (minimum section thickness)
	Flexible	Component flexibility	Expert Support & Geometric Reasoning	Material property (Modulus of elasticity) Shape dimension (minimum Section thickness)
	Adherent	Components stick together	Expert Support	Material properties
	Tangle	Components Interlinking		Geometric Shape
	Severely Nest	Components Cupping and difficult to separate		Geometric Shape
	Sharp/Abrasive		Expert Support & Geometric Reasoning	Geometric shape, Material properties & Surface finish
	Untouchable		Expert Support	Material & Process properties
	Gripping Problem		Expert Support & Geometric Reasoning	Suitable gripping surfaces Surface finish & Material properties
	Overlap			Geometric Shape
End to End Orientation Along the Axis of Insertion	Symmetrical, None Required		Geometric Reasoning	Symmetrical
	End to End Orientation Easy to See			Unsymmetrical
	End to End Orientation Not Easy to See			Partial Symmetrical
Rotational Orientation About the Axis of Insertion	Symmetry, None Required			Symmetrical
	Rotational Orientation Easy To See			Unsymmetrical
	Rotational Orientation Not Easy To See			Partial Symmetrical

🟦 Implemented or are in implementing 🟨 Likely to be implemented in the future

Table 4: Fitting Analysis

Judgements Needed	Criterion	Judgements Possibly Inferred By	Related Attributes
Can Assemble Wrong way	Symmetry, or Asymmetry Additional features	Geometric Reasoning	Geometric shape and feature
Stability	Self sustained orientation or Requires holding to maintain orientation	Expert Support & Geometric Reasoning	Centre of Gravity (CG) and Supporting area (CG can be calculated from *mass properties)*
Part Fastening Process			Fastening processes
Process Direction	Straight line from above or Straight line not from above or Not straight line	Geometric Reasoning	Insertion trajectory Insertion axes
Number of Insertions each time	Single or Multiple or Simultaneous	Geometric Reasoning	
Restricted Access and/or Vision	Tooling access space Clear line of vision	Geometric Reasoning	Feature dimensions Tooling & Access space requirements
Alignment Difficulties		Expert Support & Geometric Reasoning	Dimensions Tolerances Chamfers
Resistance to Insertion	Insertion resistance force	Expert Support	Mating joint type Tolerances
Non Assembly Processes			Non Assembly Processes

🟩 Can directly get it from database if available 🟨 Can be implemented easily

Table 5: Gripping Analysis

Judgements Needed			Judgements Possibly Inferred By	Related Attributes
Component has an appropriate gripping surface	Surface is available during insertion process	Component is easy to grip securely enough during transport	Expert Support & Geometric Reasoning	**Geometric shape & dimension** **Weight** **Suitable gripping face** **Flexibility** **Material Properties** (such as easy to scratch) **Surface finish**
		Component is difficult to grip securely enough during transport		
	Surface is not available during all the insertion	Component is easy to grip securely enough during transport		
		Component is difficult to grip securely enough during transport		
Component has no suitable gripping surface				

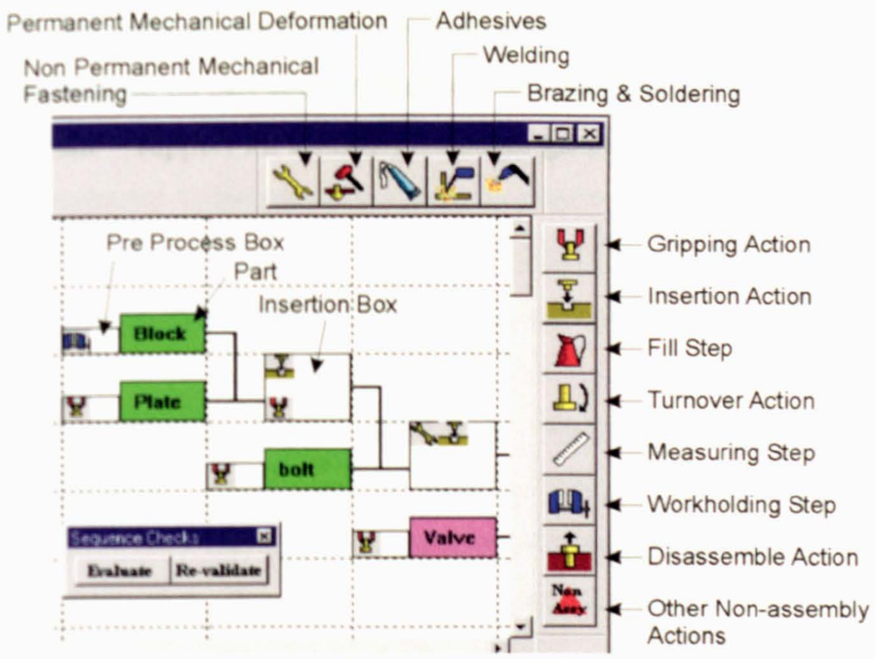


Figure 6.30 Assembly Process Definition

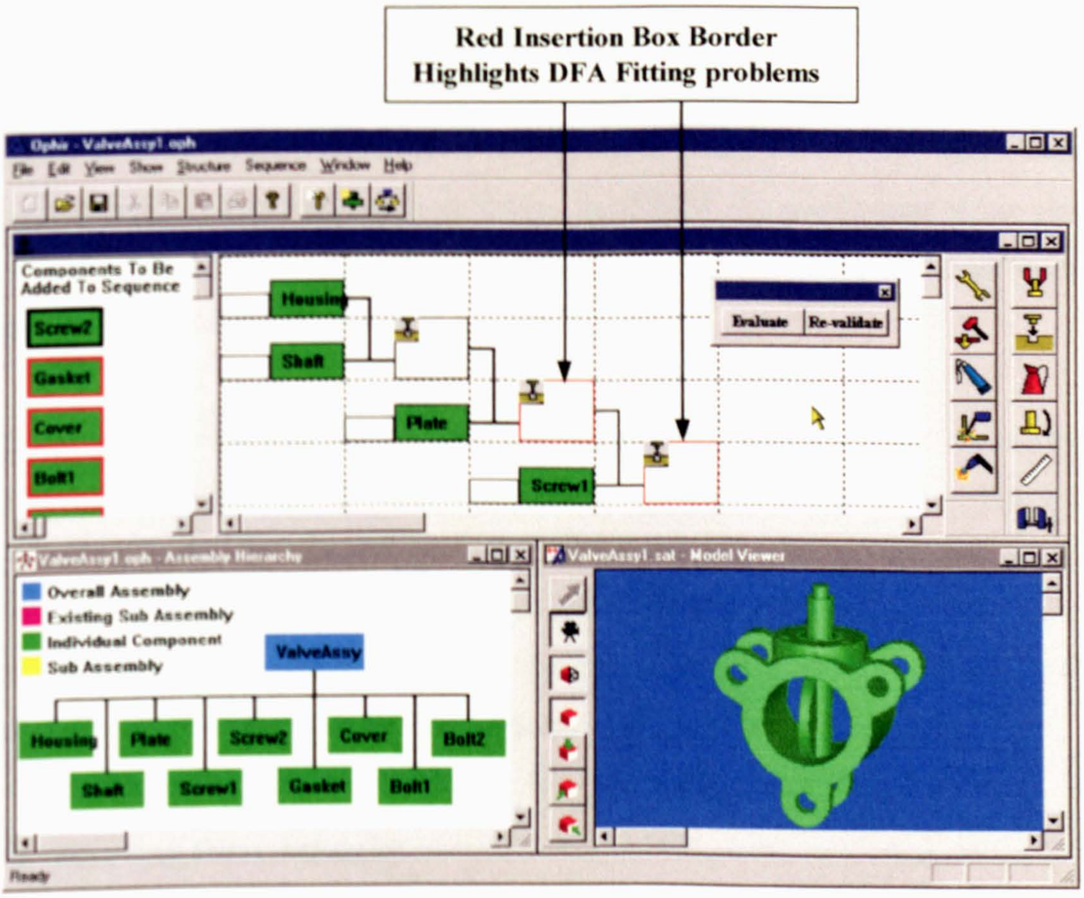


Figure 6.31 Highlighting DFA Problems in Sequence Builder

In this chapter we detailed the work of expert support for proactive DFA with the implementation focused on part count reduction. Another area of expert support in the Ophir environment -- support for assembly sequence generation, will be detailed in the next chapter.

Chapter 7 Expert Support for Assembly Sequence Generation

In chapter 2 and chapter 3, the following conclusions have been reached:

1. Proactive DFA analysis of a product design requires concurrent generation of an assembly plan -- an appropriate assembly sequence needs to be generated from the early design stage;
2. Assembly planning has a strong heuristic base. An interactive assembly sequence generation integrating hard and soft constraints, collaborating the user and artificial intelligence, most suits industrial requirements and proactive DFA evaluation.

However, as a designer is often not an assembly engineer, he/she may not have the necessary heuristics to facilitate the assembly planning process. Geometric reasoning techniques are useful in detecting collision free path and stable subassemblies to support the application of hard constraints. But, this normally involves lengthy computing and requires detailed geometry. As stated previously, to deploy heuristics and assist the application of soft constraints, knowledge-based approach can be useful. The expert support is especially important when the proactive DFA evaluation and current assembly sequence generation starts from the early design stage as this means that detailed geometry may not be available. In addition, the expert support can play a key role in assisting the generation of a feasible and practical assembly sequence that can be used in industrial practice.

The support provided by the *Expert Assembler* for the two-tier concurrent sequence generation methodology is mainly realised as three expert elements. The **PCA** mentioned in the last chapter supports part count reduction to optimise product structure. The *Starting Part Advisor* (**SPA**) and the *Next Part Advisor* (**NPA**) which will be described in this chapter support assembly sequence generation. There are also other expert elements that support for sequence validation and evaluation.

7.1 Starting Part Advisor

In deciding upon an assembly sequence, the first problem is that of choosing the part with which to start. This is an important decision that will influence all other selections regarding the sequence. An unsuitable base component can result in the whole assembly plan becoming impractical. It is for this reason that the **SPA** has been developed to provide relevant and timely suggestions to eliminate the possibility of an inappropriate choice. As parts of an assembly are added to the assembly structure interactively, the **SPA** automatically collates and makes decisions upon information germane to the definition of an assembly sequence. Its decision is made on the basis of a group of general heuristics derived from a combination of knowledge engineering and assembly sequence case study analyses.

7.1.1 Identification of Starting Part Rules

The general rules illustrating the decision-making process involved in the operation of the **SPA** are shown in Figure 7.1. They are extracted from the analysis of thirty-four automotive electromechanical assembly sequence case studies and knowledge engineering with experts in industry. These rules are generally suitable for all electromechanical products, and suitable from an early design stage, as they are not heavily relying on detailed geometry. They have been tested in six different manufacturing companies for their relevance and applicability.

The starting part must not be flexible.
The starting part must not be fragile.
The starting part must not be a free moving/loose part.
The starting part should be large and heavy in relation to other parts.
The starting part must not be a fastener.
The starting part must not be small in relation to other parts.
The starting part must not be light in relation to other parts.

Figure 7.1 General Heuristics Extracted for Starting Part Advisor

The test result Table 6 shows how often the experienced assembly planning engineers use these rules in planning process, and it is generally consistence with the result of case study analysis of the rule usage.

Table 6: Frequency of Usage in Practice – Starting Part Rules

Rule Used To Find Base Part	**Case Study % Times Used**	**Industrial Survey % Times Used**
Use a heavy and large part	46%	100%
Not a light part	83%	81%
Not a small part	83%	95%
Not a free moving/loose part	97%	100%
Not a flexible part	100%	100%
Not a fragile part	100%	100%
Not an expensive part	0%*	0%
Part positioned relative to a datum	8%	5%
Part with mating faces only on one side	14%	52%
Most mating faces	38%	62%
Not a fastener	94%	100%

* Insufficient Data to Determine

7.1.2 Implementation of Starting Part Rules

These rules have been structured and realised in the CLIPS Knowledge Base of the *Expert Assembler*, and rule-based reasoning has been used to identify the base part. It is acknowledged that a single candidate for the base part may not be identified using this set of rules, especially if the underlying product model is incomplete. Rather a shortlist of parts is highlighted for consideration. All the identified components in this list are potential base parts, but it is left to the user's discretion to select the most suitable part.

Notation:

V_P	part volume;
M_P	part mass;
V_{Max}	maximum volume of all the parts in assembly;
M_{Max}	maximum mass of all the parts in assembly;
V_{Ave}	average volume of all the parts in assembly;
M_{Ave}	average mass of all the parts in assembly;
t	part minimum section thickness;
E	modulus of elasticity of part material;
C_L	large criterion, initially $C_L = 1$;
C_H	heavy criterion, initially $C_H = 1$;
C_S	small criterion;
C_{Li}	light criterion;
C_F	flexural criterion;
C_T	thickness criterion of flexibility;
C_{Fr}	fragility criterion;
C_{Tf}	thickness criterion of fragility;
WF_{LH}	weighting factor of "large and heavy" criterion;
WF_{Fm}	weighting factor of "free moving" criterion;
WF_{Fl}	weighting factor of "flexible" criterion;
WF_{Fr}	weighting factor of "fragile" criterion;
CF_{LH}	confidence factor, $CF_{LH} = 1$ if part is large and heavy;
CF_{Fm}	confidence factor, $CF_{Fm} = 1$ if part is not free moving;
CF_{Fl}	confidence factor, $CF_{Fl} = 1$ if part is not flexible;
CF_{Fr}	confidence factor, $CF_{Fr} = 1$ if part is not fragile;
CF	final confidence factor, $CF=CF_{LH}*WF_{LH}+CF_{Fm}*WF_{Fm}+CF_{Fl}*WF_{Fl}+CF_{Fr}*WF_{Fr}$.

Sample Rules:

If: $\frac{V_P}{V_{Max}} >= C_L$ **And:** $\frac{M_P}{M_{Max}} >= C_H$
Then: Part Large and Heavy (1)

If: $\frac{V_P}{V_{Ave}} =< C_S$
Then: Part Small (2)

If: $\frac{M_P}{M_{Ave}} =< C_{Li}$
Then: Part Light (3)

If: $f(Et^3) < C_F$
Then: Part Flexible (4)

If: Part is manufactured from sheet metal
And: $t < C_T$
Then: Part Flexible (5)

If: $e < 5\%$
And: $f(\sigma_u t^2) < C_{Fr}$
Then: Part fragile (6)

If: $e < 5\%$
And: Part is thin shell part
And: $t < C_{Tf}$
Then: Part fragile (7)

If: Part has kinematic linkage
Then: Part Free Moving (8)

If: Part is defined as a fastener
Then: Part Fastener (9)

If: Part Small
Or: Part Light
Or: Part Flexible
Or: Part Fragile
Or: Part Free Moving
Or: Part Fastener
Then: Not Starting Part (10)

If: Part Large and Heavy
And: not Part Flexible
And: not Part Fragile
And: not Part Free Moving
And: not Part Fastener
Then: Starting Part (11)

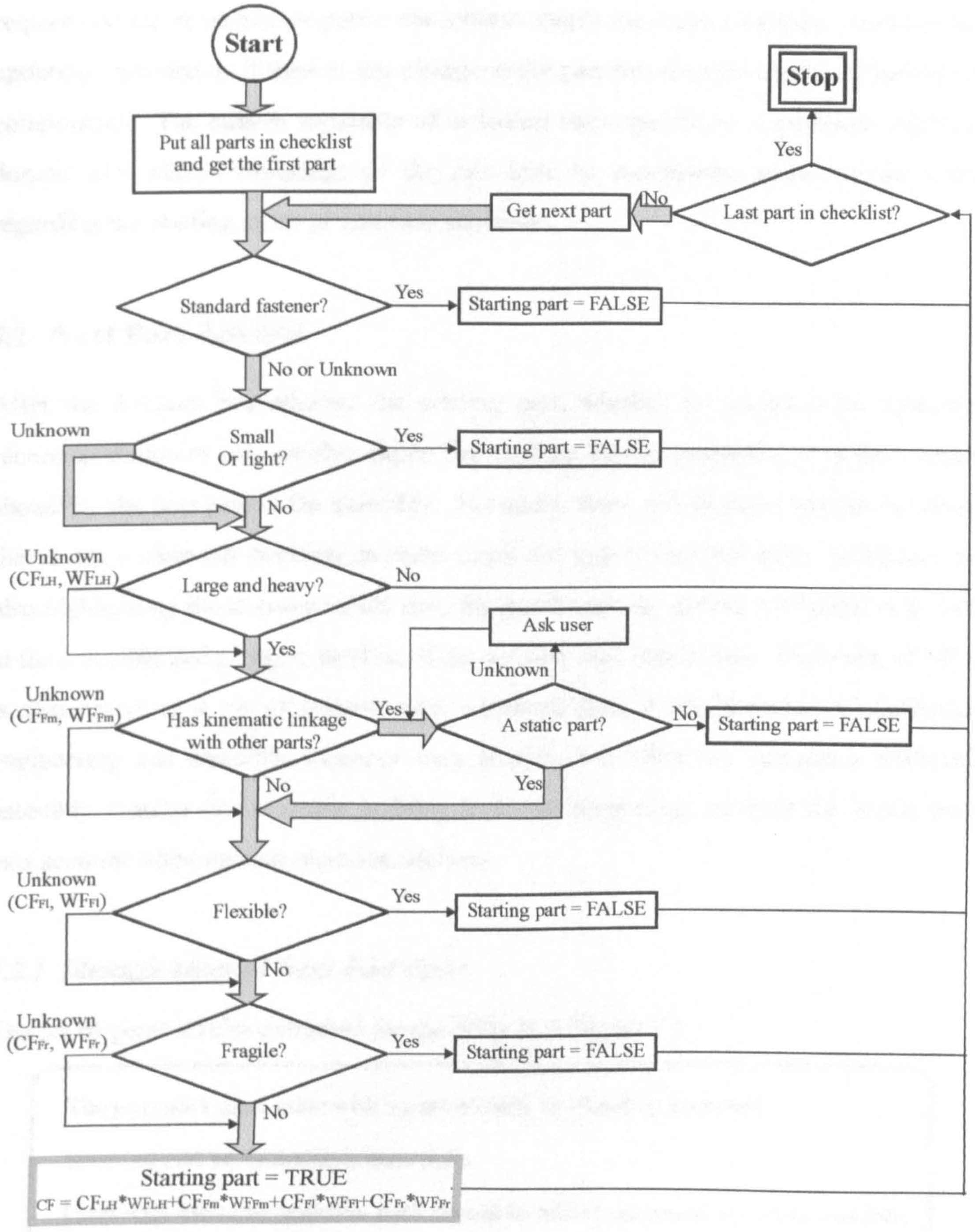
If: $CF > 75\%$
Then: Starting Part (12)

Part Large and Heavy, rule (1) uses volume and mass data stored or calculated from the solid model. If this is not available, an "unknown" answer is returned with an estimated confidence factor CF_{LH}, and a predefined weighting factor WF_{LH} is involved. It is possible that the largest part is not the heaviest part, and vice versa and no parts are found to be suitable. If this rule does not identify at least one base part candidate, C_L and C_H are decremented by 0.1 and this rule is trigged again until at least one part is recommended relatively large and heavy. *Part Small,* rule (2) and *Part Light*, rule (3) are realised by comparing volume and mass of each part with the part's average. *Part Flexible*, rule (4) calculates the flexibility of the component and compares with a predetermined threshold value. Rule (5) defines a specific application domain implementation for the rule, *Part Flexible*. This considerably reduces the computation necessary. *Part Fragile*, rule (6) calculates the fragility of the component and compares with a predetermined threshold value. Rule (7) defines a specific application to one category of components for the rule, *Part Fragile*. This is also for reducing computation. Kinematic part attributes are found by searching all mating components with kinematic linkage, Rule (8). However, an exception occurs when the part moves relative to others but it is static itself. Thus if this rule is TRUE, confirmation of static status is requested. The *Part Fastener*, rule (9) excludes many standard fasteners which may exist in assembly, this reduces the work and increases the efficiency of the **SPA**. However if the precondition for rule (9) is unknown, the utilisation of *Part Small,* rule (2) and *Part Light*, rule (3) would exclude most fasteners, as they are generally small and light. This is convenient given the difficulty of defining fasteners from attributes. To reduce computation, fasteners could be identified on declaration. *Starting Part FALSE*, rule (10) will be fired if one of the conditions is satisfied, and the condition triggering the rule is captured to enable an explanatory warning to be issued. Conversely, to fire *Starting Part TRUE*, rule (11), the part must fulfil all the conditions specified. If any of the data required is undefined, then a confidence factor and a weighting factor are involved. If the total confidence factor is greater than 75%, it leads to a conclusion of *Starting Part TRUE*, Rule (12). If no suitable base part is found, options are offered to modify a part, add new part into the assembly or find the most suitable part in the existing list. The last option identifies a base component from the existing parts using the following rule precedence:

1. Part Fragile, Part Flexible, Part Free Moving.
2. Part Small, Part Light.
3. Part Standard Fastener.

As the least important, **SPA** ignores *Part Fastener*, rule (7) and fires the other rules. Reasoning stops if a base part is found. Otherwise *Part Small,* rule (2) and *Part Light*, rule (3) are ignored and reasoning continues. This process is repeated until a base part is found.

The implementation of the rules from (1) to (12) can be illustrated as a flowchart Figure 7.2.



Notation: The final confidence factor CF expresses the degree of confidence that the part should be suggested as the start of the assembly.

Figure 7.2 Starting Part Advisor Working Flowchart

As components are added to the assembly structure, the **SPA** checks all the parts, indicating parts that are not suitable as base parts and recommending the best candidate of the starting point of assembly sequence. This process is invisible to the user, and requires no effort on his/her part – the system simply indicates candidate components, updating immediately if there is any change in the part list or in the design of individual components. The built-in capability of including rules specific to a particular problem domain also allows expansion of the rule base to incorporate more specific rules regarding the starting point of assembly sequence.

7.2 Next Part Advisor

After the designer has selected the starting part, whether he accepted the system's recommendation or not, another expert **NPA** of the *Expert Assembler* considers which should be the next part in the assembly. Naturally, there will be many occasions where this is not a clear-cut decision; in these cases the system can still offer assistance, by also highlighting those components that, for good reasons, should not be the next part in the assembly and giving a shortlist of the possible best candidates. The work of **NPA** is also based on a set of general rules extracted from a combination of knowledge engineering and assembly sequence case studies, and takes the designer's preferred assembly strategy (for example building from the bottom up, or from the inside out) into account when making recommendations.

7.2.1 Identification of Next Part Rules

The set of general rules extracted for the **NPA** is in Figure 7.3.

The next part must mate with a part already in assembly sequence.

There should be no multiple insertions.

Parts with the same insertion path should be added consecutively when possible.

Parts with similar final locations should be inserted consecutively when possible.

Unstable, small and light parts should be secured immediately when possible.

Identical parts should be inserted consecutively when possible.

Separate dirty and clean jobs when possible.

Similar joining processes to be completed consecutively when possible.

Figure 7.3 General Heuristics for Next Part Advisor

The test result of their relevance and applicability from the thirty-four automotive electromechanical case studies analysed and the six manufacturing companies visited is shown in Table 7.

Table 7: Frequency of Usage in Practice -- Next Part Rules

Next Component Rule	**Case Study Results**	**Industrial Survey**
Identical parts inserted consecutively	30%	86%
Secure unstable part immediately	65%	81%
Flexible parts late in sequence	0%	0%
Fragile parts late in sequence	9%	57%
Expensive parts late in sequence	0%	24%
Free moving / loose parts late in sequence	8%	19%
Access same tools consecutively	5%	62%
Complex subassemblies early in sequence	10%	62%
Work to the riveting side	0%	33%
Install parts individually	100%	100%
Light parts secured immediately	4%	95%
Small parts secured immediately	4%	95%
Work bottom up	50%	76%
Work inside out	41%	52%
Work outside in	0%	48%
Similar joining processes consecutively	4%	81%
Same insertion direction consecutively	95%	100%
Insert parts in same location consecutively	95%	81%
Separate dirty and clean jobs	8%	71%

It can be seen that the analysis of the case studies proved inconclusive. But the discussions and observations of industrial practice discovered an implicit decision that was normally taken before building the sequence -- the overall assembly strategy, which decides the main build direction of a product. These strategies are defined as:

- Top Down
- Bottom Up
- Inside Out
- Outside In.

7.2.2 *Implementation of Next Part Rules*

The rules for the **NPA** are deployed into a heuristic search algorithm which is implemented as C++ functions. A working flowchart Figure 7.4 shows the heuristic search process of the **NPA**. The search by the **NPA** for the next part is commenced from the attributes of the last part added to the sequence. The process can be explained using set theory as follows. A set of components in an assembly of *m* components is defined, Aset = $\{p_1, p_2, \ldots, p_m\}$ and a set of all suitable next parts, NextPset, such that NextPset $\subset$ Aset. $Lset_i$ is a set of all parts on level i ($i>0$) of the assembly hierarchy where $Lset_i \subset$ Aset. Let x be the last part added to the assembly sequence such that $x \in Lset_i$. Nset is the set that contains all those parts which have not been added to the developing assembly sequence, where Nset $\subset$ Aset. All parts not in the assembly sequence, which are from the same level, i in the assembly hierarchy are added to NextPset, thus:

NextPset = $Lset_i \cap$ Nset;

If NextPset $=\phi$ and $i > 1$,

NextPset = $Lset_{i-1} \cap$ Nset;

Else If NextPset = ϕ, Stop.

Two rules are implemented as options which to be used must be selected prior to commencement of sequence construction:

(1) Rule: *Cluster similar joining processes*

JPset is the set of all parts with the same joining process as x and not added to the sequence, JPset $\subset$ Nset. If this option is selected, this set of components is added to NextPset:

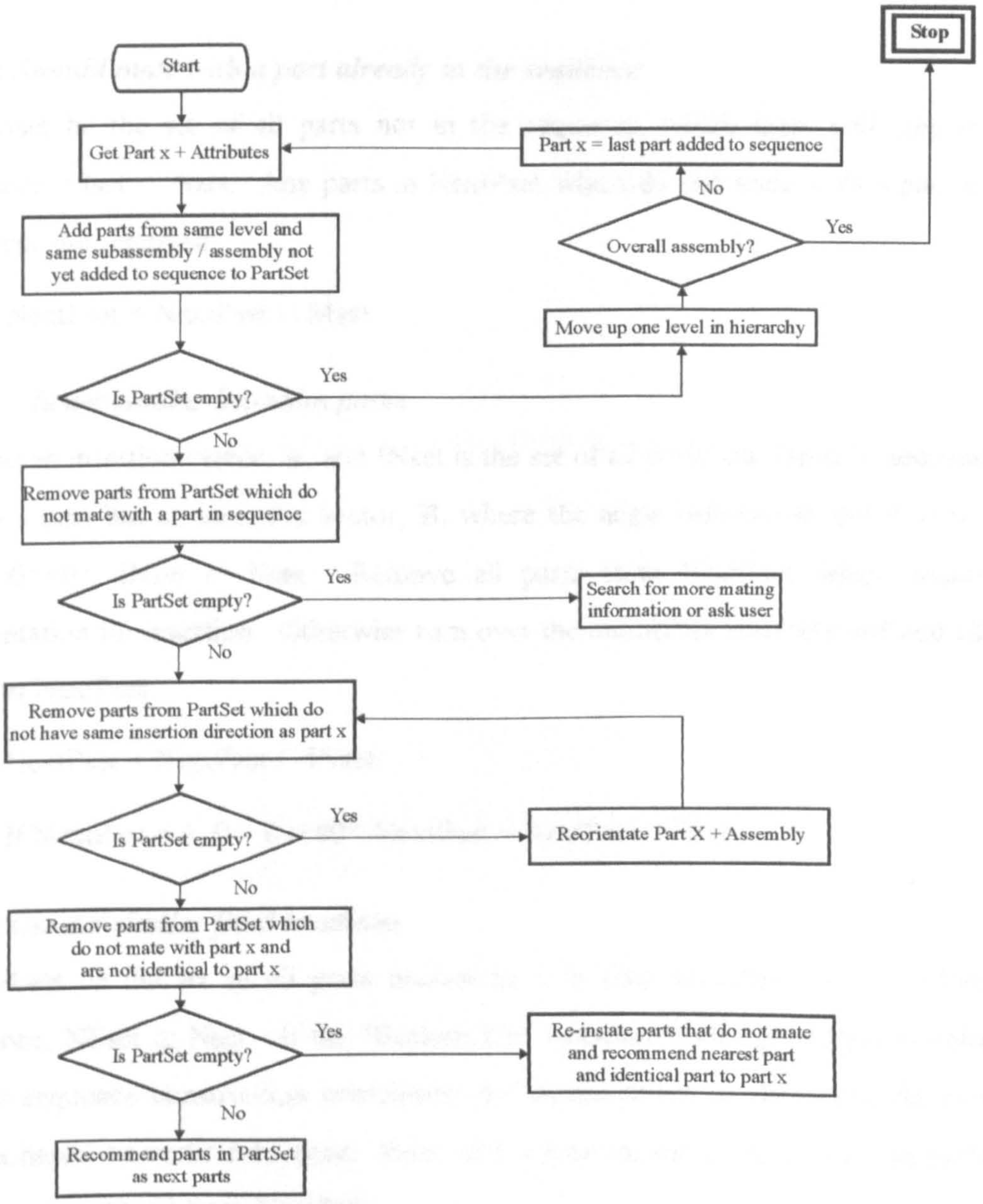
NextPset = NextPset U Jpset.

(2) Rule: *Separate dirty and clean jobs*

Let Dset be the set of parts with dirty processes and not yet added to the sequence, Dset $\subset$ Nset. If this option is selected and x has a dirty process attribute, then all other parts with a dirty process attribute are added to NextPset:

If x $\in$ Dset, NextPset = (NextPset - (Nset - Dset)) U Dset;

Else NextPset = (NextPset - Dset) U (Nset - Dset).


Stop
Start
Get Part x + Attributes
Part x = last part added to sequence
No
Yes
Overall assembly?
Add parts from same level and same subassembly / assembly not yet added to sequence to PartSet
Move up one level in hierarchy
Yes
Is PartSet empty?
No
Remove parts from PartSet which do not mate with a part in sequence
Yes
Is PartSet empty?
Search for more mating information or ask user
No
Remove parts from PartSet which do not have same insertion direction as part x
Yes
Is PartSet empty?
Reorientate Part X + Assembly
No
Remove parts from PartSet which do not mate with part x and are not identical to part x
Yes
Is PartSet empty?
Re-instate parts that do not mate and recommend nearest part and identical part to part x
No
Recommend parts in PartSet as next parts

Some rules are realised as options which can be chosen prior to construction of the assembly sequence. Examples of optional rules are:

1. Parts with similar joining process to be completed consecutively
2. Separate dirty and clean jobs

Figure 7.4 Next Part Advisor Working Flowchart

Once these optional rules have been applied, the mandatory selection of components commences,

Rule: ***Cluster similar parts***

Let IDset be the set of all parts identical to x and not added to the sequence, Idset $\subset$ Nset. All these identical parts are added to NextPset:

NextPset = NextPset U IDset.

Rule: ***Should mate with a part already in the sequence***

Let Mset be the set of all parts not in the sequence, which mate with any in the sequence, Mset $\subset$ Nset. Any parts in NextPset which do not mate with a part in the sequence are removed:

NextPset = NextPset $\cap$ Mset.

Rule: ***Cluster similar insertion paths***

If x has an insertion vector, **A**, and INset is the set of all parts not added to sequence in which a part has an insertion vector, **B**, where the angle between **B** and **A** is θ, and $90° >= \theta >= 0°$, INset $\subset$ Nset. Remove all parts from NextPset which require a reorientation for insertion. Otherwise turn over the unfinished assembly and add all the parts to NextPset.

NextPset = NextPset $\cap$ INset;

If NextPset = ϕ, $\theta = \theta + 180°$, NextPset = NextPset $\cap$ INset.

Rule: ***Cluster similar final locations***

Let NEset be the set of all parts nearest to x in final assembly and not added to sequence, NEset $\subset$ Nset. If the "Bottom Up" sequence building strategy is selected before sequence construction commence, the measurement to determine the nearest part is based on vertical distance. Parts with a final location which is a long distance from x are removed from NextPset:

NextPset = NextPset $\cap$ NEset.

NextPset now contains those parts, which according to the defined rules, are suitable to be added to the sequence as the next component. If any of these recommended parts are "New Subassembly" type, the **NPA** triggers the **SPA** to determine which is the best part for commencing the subassembly. If the user did not accept the recommendation and chose another part, the recommended part is kept in the NextPset as it should always be a suitable choice to continue the unfinished part of sequence definition.

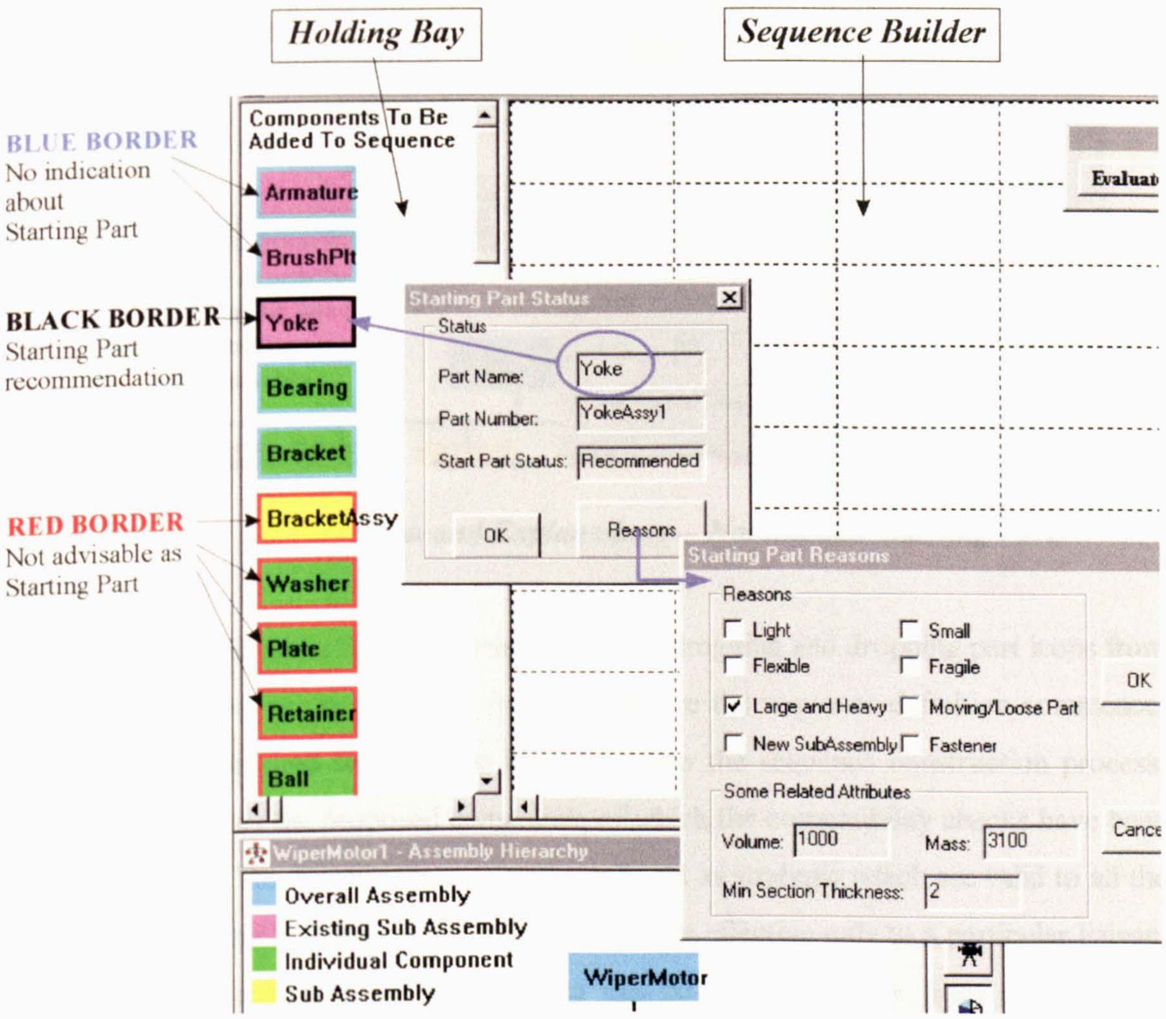
The **NPA** continues to consider which part should be next until the assembly is complete (which, as a useful side effect, means that it becomes impossible to forget to include a part in the assembly). The great advantage of such a continuous, background approach whereby inference is automatically made 'in the background' based on available information is that the designer can immediately see the effects of design changes upon the assembly sequence.

As the sequence construction is in parallel with the assembly structure definition, if adding another part to the assembly structure, the search of the **NPA** will be interrupted. In stead, the **SPA** will be active to check if the new added part is the best starting point of the overall assembly or a subassembly that has already started to be built. The **SPA** recommends this part if it is suitable. After the user made decision, the **NPA** renews its work.

7.3 An Example

The example of the wiper motor used in the last chapter can also be used to illustrate how the *Starting Part Advisor* and the *Next Part Advisor* works in the Ophir environment.

When the designer interactively adds components, existing subassembly or new subassembly of the wiper motor into the assembly structure, all these parts appear and wait in the *Holding Bay* Figure 7.5. The **SPA** checks all the parts for their suitability as the base part whenever a part attribute changes or a new part is added. The result is indicated as the border of part icon in the *Holding Bay*. A black border indicates the recommendation of the best candidate for the starting part, such as *Yoke* in the figure. A red border highlights the part that may not be suitable as the base part of the assembly sequence, such as *BracketAssy*, *Washer*, *Plate*, *Retainer* and *Ball*. A blue border has no indication either way, such as the border around *Amature*, *BrushPl*t, *Bearing* and *Bracket*. The thickness of the border shows the confidence of the recommendation. A very thin border indicates that there is not enough information to infer the starting part. Right clicking a part icon in the *Holding Bay*, brings up a dialog -- the *Starting Part Status Dialog*, which shows the suitability of this part as the starting part. Clicking the *Reasons* button from the *Dialog* will bring up the *Starting Part Reasons Dialog* which details the recommendation Figure 7.5 or advice of not suitable Figure 7.6.



Status

Part Name: Yoke

Part Number: YokeAssy1

Start Part Status: Recommended

Reasons

- [ ] Light
- [ ] Small
- [ ] Flexible
- [ ] Fragile
- [x] Large and Heavy
- [ ] Moving/Loose Part
- [ ] New SubAssembly
- [ ] Fastener

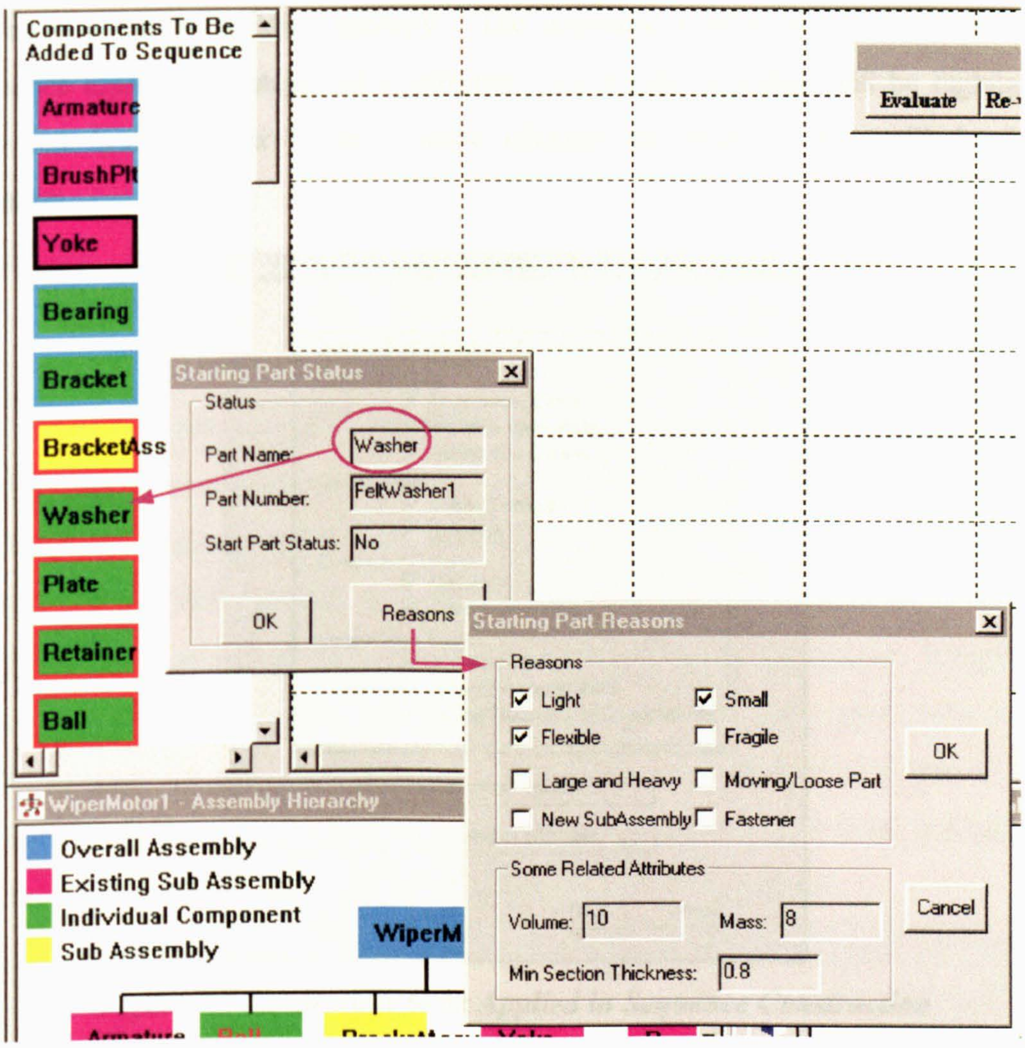
Some Related Attributes

Volume: 1000 Mass: 3100

Min Section Thickness: 2

🟦 Overall Assembly
🟪 Existing Sub Assembly
🟩 Individual Component
🟨 Sub Assembly

Figure 7.5 Recommendation and Explanations -- Suitable Starting Part



Starting Part Status

Status

Part Name: Washer

Part Number: FeltWasher1

Start Part Status: No

Starting Part Reasons

Reasons

☑ Light	☑ Small
☑ Flexible	☐ Fragile
☐ Large and Heavy	☐ Moving/Loose Part
☐ New SubAssembly	☐ Fastener

Some Related Attributes

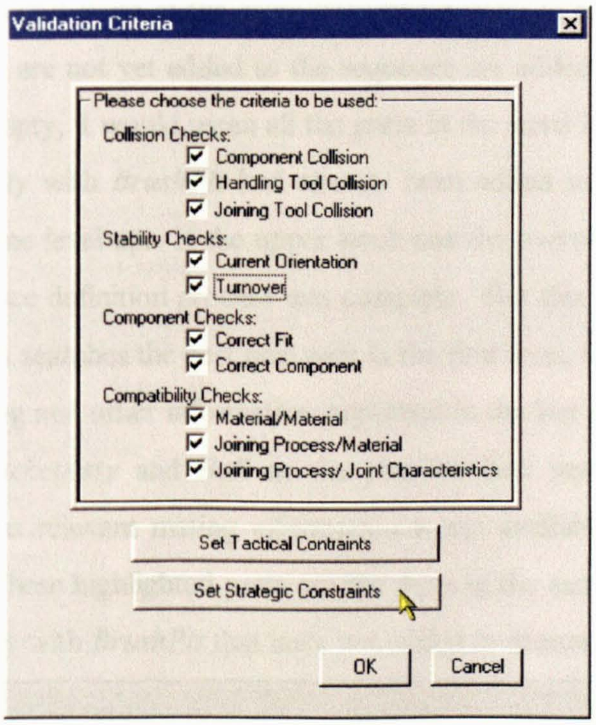
Volume: 10 Mass: 8

Min Section Thickness: 0.8

🟦 Overall Assembly
🟥 Existing Sub Assembly
🟩 Individual Component
🟨 Sub Assembly

Figure 7.6 Indication and Explanations -- Not Suitable Starting Parts

The sequence construction is achieved by simply dragging and dropping part icons from the *Holding Bay* to the *Sequence Builder*. Before the sequence definition commence, constraints (hard and soft) can be integrated into the sequence construction process. Figure 7.7 shows the proposed constraints of which the compatibility checks have been implemented. The constraints proposed can be set as strategic which are valid to all the liaisons in assembly sequence, or tactical which are effective only to a particular liaison. The tactical constraints for each liaison can be changed easily from a *Validation Criteria Dialog* that can be brought up by right clicking the liaison -- the *Insertion Box*. Also the preferred assembly construction strategy has to be chosen from a *Sequence Strategy Dialog* Figure 7.8 before the sequence construction commences. The **NPA** reasons the next part candidate based on the next part rules extracted and the user preferred assembly building strategy. The sequence validation process checks the sequence against the integrated constraints. Constraint violation will be highlighted with red *Insertion Box* border. Right clicking the *Insertion Box* will detail the violation.


Validation Criteria

Please choose the criteria to be used:

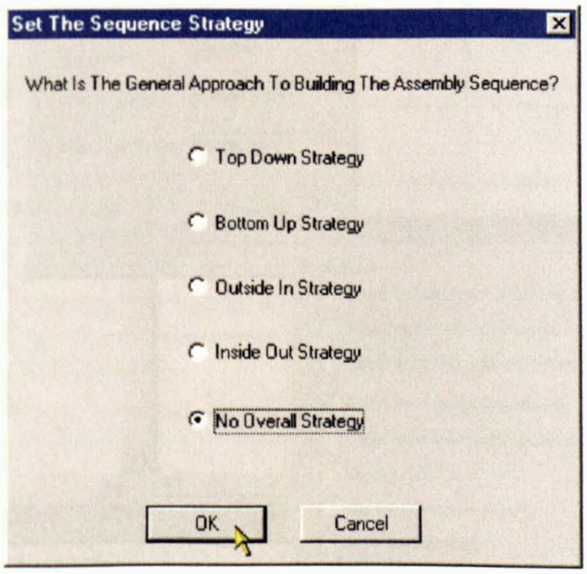
Collision Checks:
- [✓] Component Collision
- [✓] Handling Tool Collision
- [✓] Joining Tool Collision

Stability Checks:
- [✓] Current Orientation
- [✓] Turnover

Component Checks:
- [✓] Correct Fit
- [✓] Correct Component

Compatibility Checks:
- [✓] Material/Material
- [✓] Joining Process/Material
- [✓] Joining Process/ Joint Characteristics

Figure 7.7 Constraints to be Applied in Sequence Construction


Set The Sequence Strategy

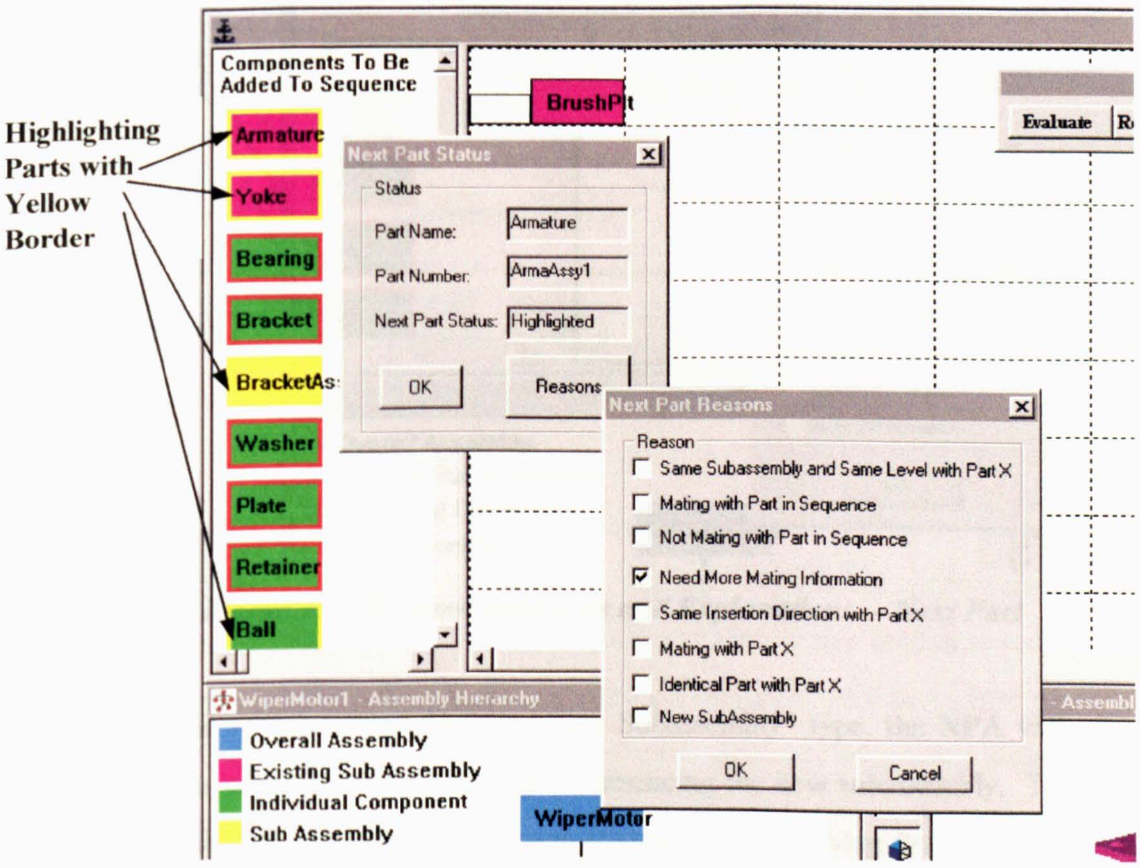
What Is The General Approach To Building The Assembly Sequence?

- ○ Top Down Strategy
- ○ Bottom Up Strategy
- ○ Outside In Strategy
- ○ Inside Out Strategy
- ● No Overall Strategy

Figure 7.8 Sequence Construction Strategies

When the base part, recommended or not, is dragged into the *Sequence Builder*, the

NPA starts to reason the best candidate of the next part based on the attributes of the part just added: the level and the subassembly that it belongs to, the mating information, insertion direction, and joining process information. In the wiper motor example, *BrushPlt*, not recommended, has been dragged into the *Sequence Builder* Figure 7.9. All the other parts at the same level that belong to the same subassembly with *BrushPlt* which are not yet added to the sequence are added to the NextPSet. If the NextPSet was empty, it would mean all the parts in the same level that belonged to the same subassembly with *BrushPlt* had already been added to the sequence. The **NPA** would check one level up. If the upper level was the *overall assembly*, it would mean that the sequence definition process was complete. But this is not the case in the example, so the **NPA** searches the best next part in the first level, where the *BrushPlt* is sited, based on mating and other information explained in the last section. It highlights *Amature*, *Yoke*, *BracketAssy* and *Ball* as the possible best next parts with *Yellow* borders Figure 7.9 as relevant mating information is not available, and asks for user input Figure 7.10. These highlighted parts are the parts in the same level, belonging to the same subassembly with *BrushPlt* that have not added to sequence.



Next Part Status

Status

Part Name: Armature

Part Number: ArmaAssy1

Next Part Status: Highlighted

OK | Reasons

Next Part Reasons

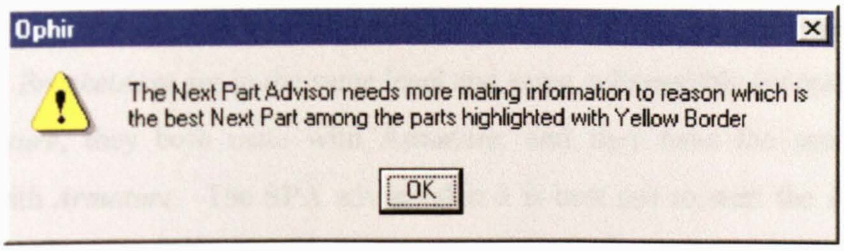
Reason

- [ ] Same Subassembly and Same Level with Part X
- [ ] Mating with Part in Sequence
- [ ] Not Mating with Part in Sequence
- [x] Need More Mating Information
- [ ] Same Insertion Direction with Part X
- [ ] Mating with Part X
- [ ] Identical Part with Part X
- [ ] New SubAssembly

OK | Cancel

Legend: 🟦 Overall Assembly; 🟥 Existing Sub Assembly; 🟩 Individual Component; 🟨 Sub Assembly

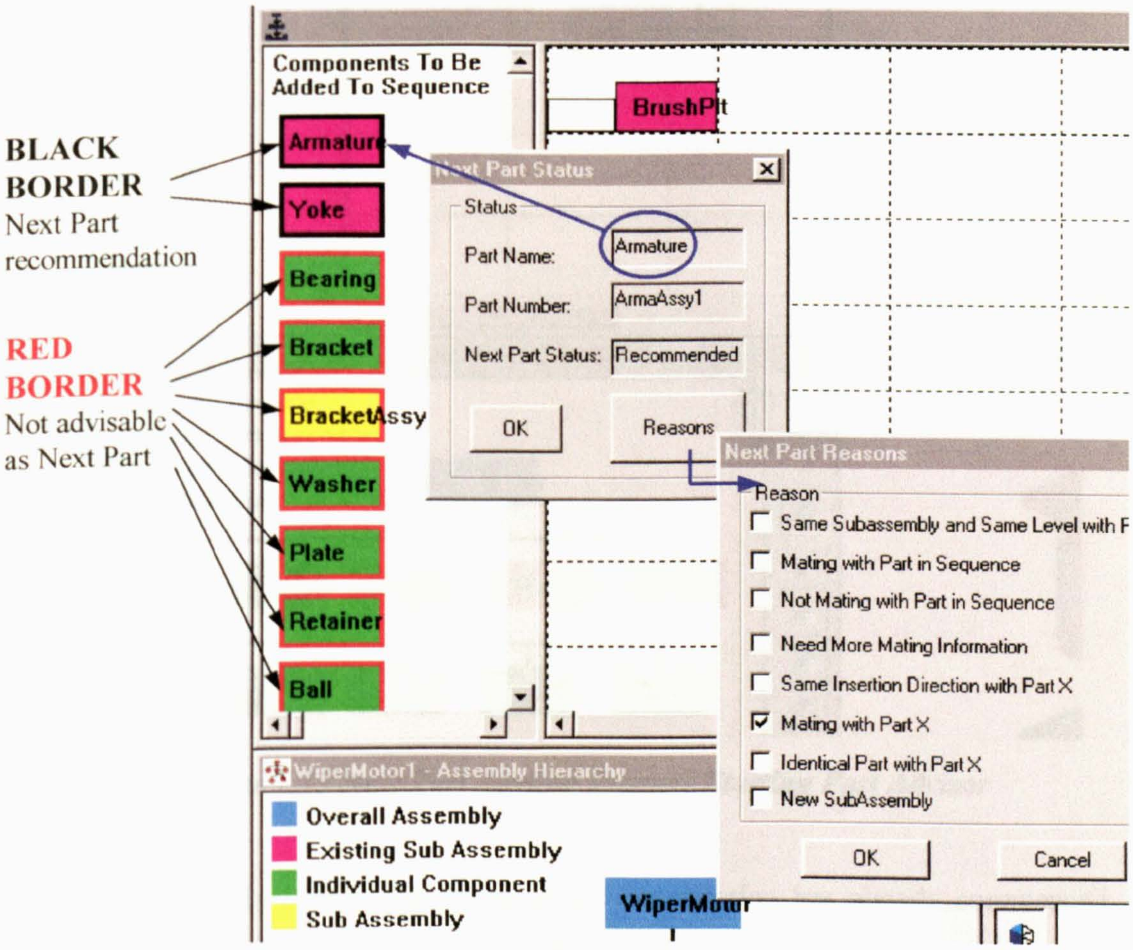
Figure 7.9 More Mating Information Is Required for the Highlighted Parts



The Next Part Advisor needs more mating information to reason which is the best Next Part among the parts highlighted with Yellow Border

Figure 7.10 Ask for User Input

From user's response, inputting the information that *Armature* and *Yoke* mates with *BrushPlt*, the **NPA** adds the relevant mating parts: *Armature* and *Yoke* to the NextPSet. At this point, it highlights *Armature* and *Yoke* as recommended next parts Figure 7.11.



Status
Part Name: Armature
Part Number: ArmaAssy1
Next Part Status: Recommended

Reason
- [ ] Same Subassembly and Same Level with F
- [ ] Mating with Part in Sequence
- [ ] Not Mating with Part in Sequence
- [ ] Need More Mating Information
- [ ] Same Insertion Direction with Part X
- [x] Mating with Part X
- [ ] Identical Part with Part X
- [ ] New SubAssembly

🟦 Overall Assembly
🟥 Existing Sub Assembly
🟩 Individual Component
🟨 Sub Assembly

Figure 7.11 Recommendations and Explanations -- Next Part

If any of the recommended parts is "New Subassembly" type, the **NPA** will trig the **SPA** to infer which is the best part for commencing the new subassembly. This is why after the *Yoke* and *Armature* are added to the sequence following *BrushPlt*, in the wiper motor example, *Bracket* is recommended Figure 7.12. It is the best starting point for

BracketAssy subassembly. The recommended shortlist of next part also includes *Ball* as *Ball* and *BracketAssy* are in the same level and same subassembly (overall assembly) with *Armature*, they both mate with Armature, and they have the same insertion direction with *Armature*. The **SPA** advises that it is best not to start the *BracketAssy* with *Washer*, *Plate* or *Retainer*.

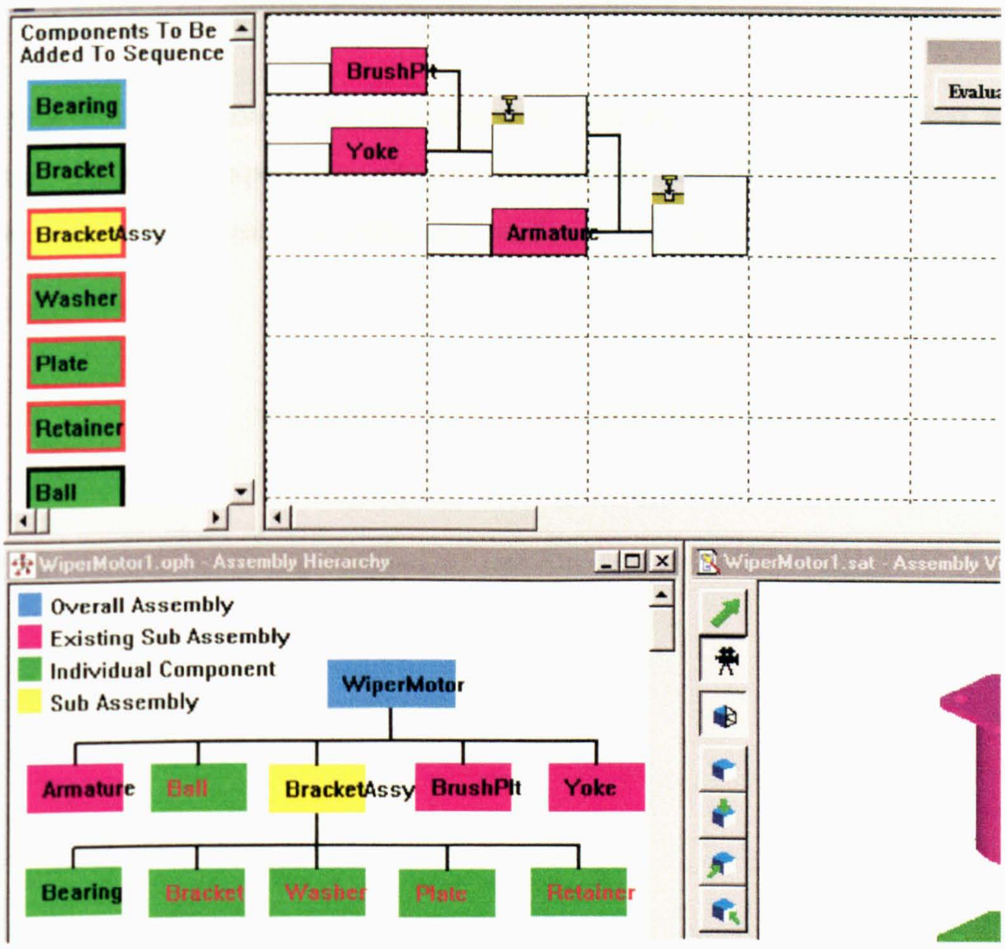
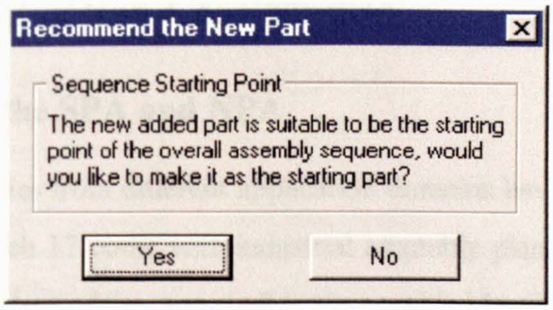


Figure 7.12 Next Part Advisor Triggers Starting Part Advisor

If any new part is added after the sequence construction has already commenced, a recommendation dialog Figure 7.13 will pop up, should the new added part be recommended by the **SPA** as the best starting point of the overall assembly or an unfinished subassembly. If the recommendation is accepted, the new added part should be automatically inserted to the right place in the assembly sequence. Otherwise, the new part is added to the *Holding Bay*, and in here, next part is recommended by the **NPA**.


Recommend the New Part

Sequence Starting Point

The new added part is suitable to be the starting point of the overall assembly sequence, would you like to make it as the starting part?

Figure 7.13 Starting Part Advisor Recommends the New Added Part

The **SPA** and **NPA** cooperate with each other and with the user as detailed in this chapter. These two expert elements and the other element **PCA** described in the last chapter will be tested using industrial case studies in the next chapter.

Chapter 8 Application Case Studies

8.1 Validation of the SPA and NPA

A number of case studies from different application domains have been used to test the **SPA** and **NPA** of which 17 come with industrial assembly plan information and these are listed in Table 8. Most of the case studies are provided by CSC Computer Sciences Ltd. In 82% of the listed case studies, the **SPA** identified the actual component used to start the assembly sequence i.e. in 14 out of the 17 cases, and on average 65% of the recommendation shortlists from the **NPA** contained the same next part as in industry assembly plan. Some recommendations are believed to be better than the industrial assembly plans in terms of assemblability. The Oil Pump can be used to illustrate the case study test process.

Table 8: Some of the Case Studies Used

Case Study Name	Parts Recommended by SPA	Part Used in Industrial Assembly Plan
Grating Arm Assembly	Grating Lever, Grating Arm, Grating Post	Grating Lever
Pintle Hook Mounting Assembly	Frame	Frame
Pintle Hook Mounting Assembly (Redesign)	Frame	Frame
Injector	Body	Body
Lens Assembly	Housing, body	Body
Actuator	Housing, body	Housing
EUI DL	Body	Body
EUI DL (Redesign)	Body	Body
Oil Pump	Body, Cover	Body
Oil Pump (Redesign)	Body	Body
Gear Chang Assembly	Housing	Housing
Wiper Motor	Yoke	Nut

Locker Handle	Main Body	Main Body
Locker Handle (Redesign)	Main Body	Main Body
Stapler Remover	Claw	Claw
Solenoid	Housing, Saddle	Bobbin
Solenoid (Redesign)	Body	Bobbin

▨ Grey shaded recommendations from the **SPA** matched the industrial assembly plans

○ Blue circle indicates a case for which the base part recommendation from the **SPA** did not match the industrial assembly plan

8.1.1 Oil Pump -- A Case Study Example

The Oil Pump Figure 8.1 is a fairly complicated product with several subassemblies, and it consists of 32 components. Its assembly structure is as Figure 8.2, which is defined in the Ophir environment as Figure 8.3.

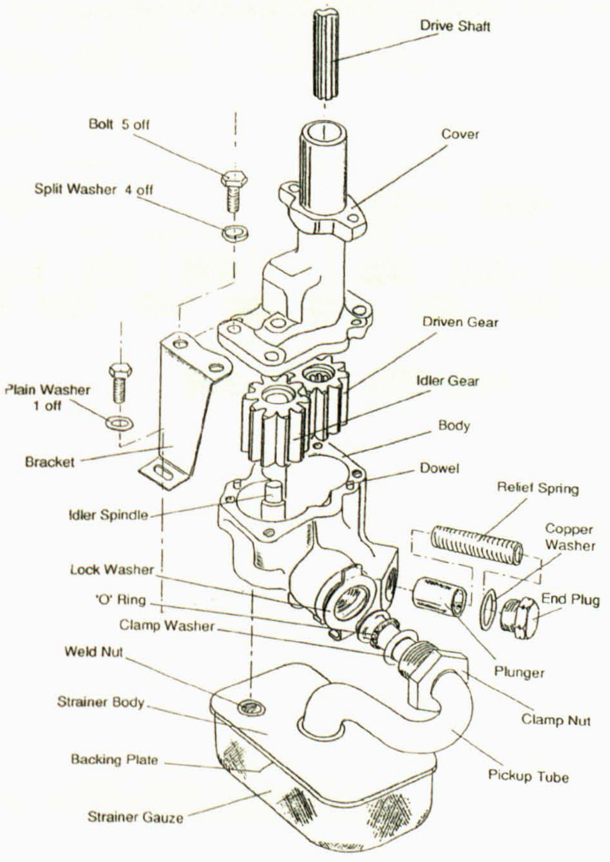


Figure 8.1 Oil Pump

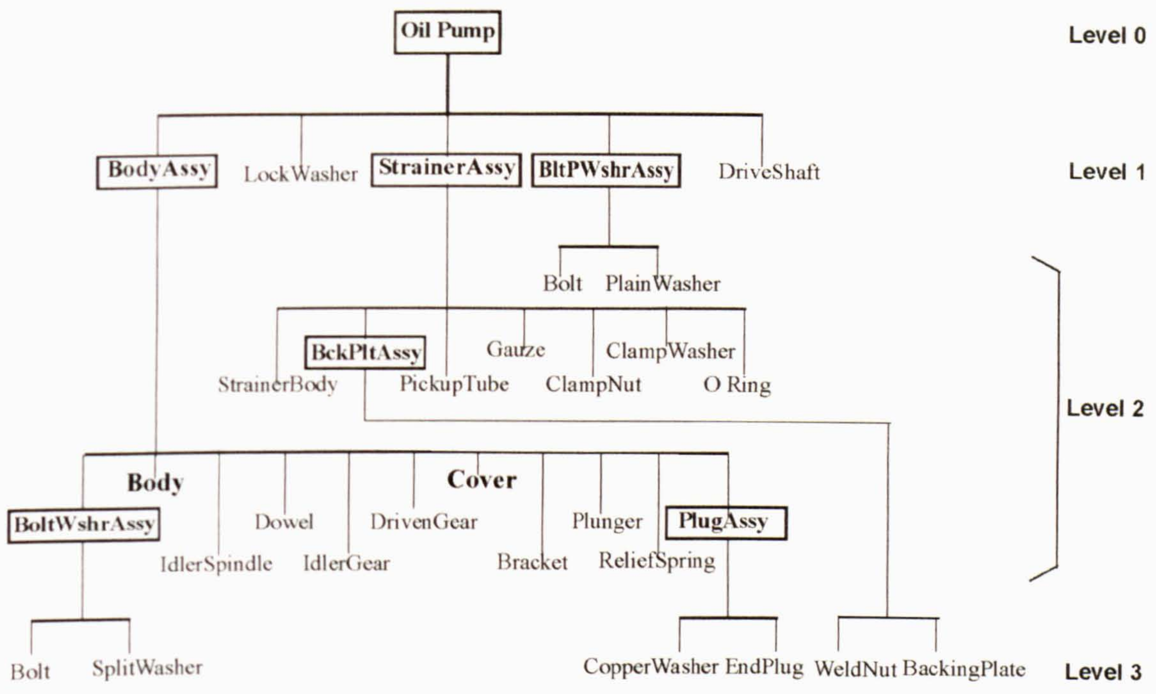


Figure 8.2 Oil Pump Assembly Structure

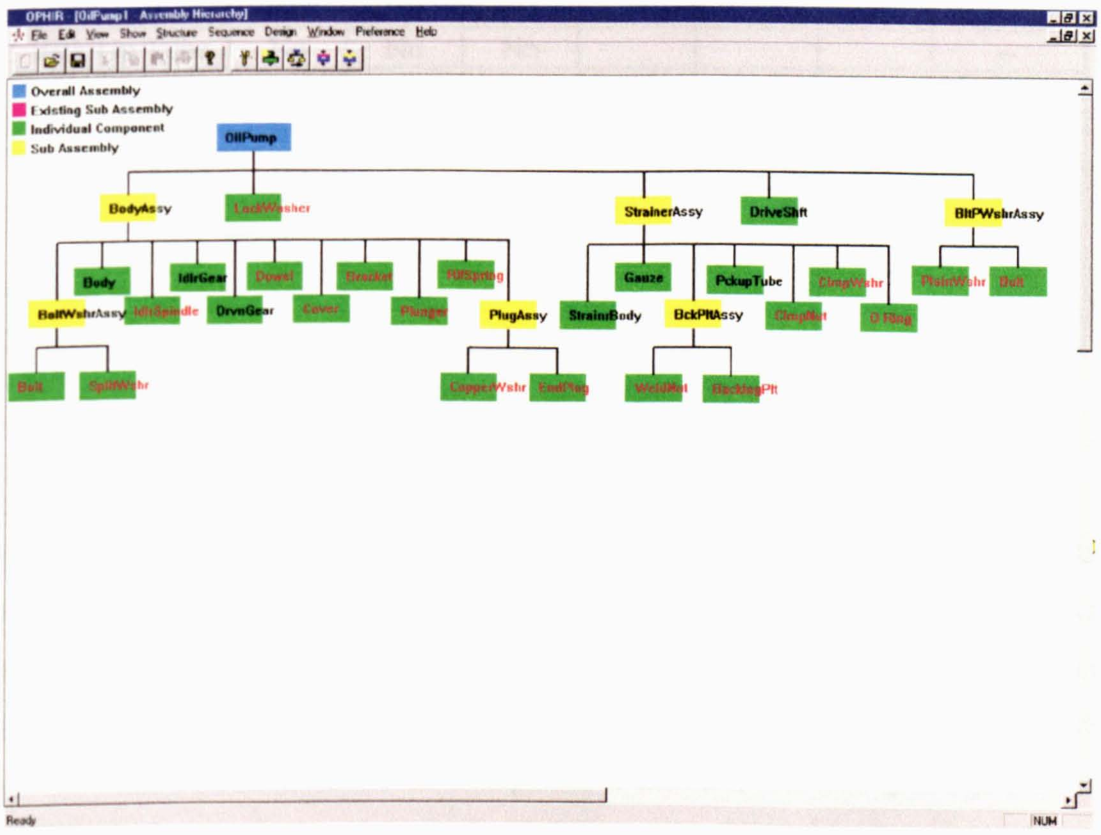


Figure 8.3 Oil Pump Assembly Structure Defined in the Ophir Environment

The summary of the recommendations from the **SPA** for the Oil Pump is presented in Table 9.

Table 9: Recommendations of Starting Part for Oil Pump

Part Name	Small	Light	Large Heavy	Free Moving	Flexible	Fragile	Start Part Status
Body	No	No	Yes	No	No	No	Yes
Idler Spindle	No	No	No				--
Dowel	Yes	Yes					No
Idler Gear	No	No	No				--
Driven Gear	No	No	No				--
Cover	No	No	Yes	No	No	No	Yes
Split Washer	Yes	Yes					No
Bolt	Yes	Yes					No
Bracket	No	No	No				--
Plunger	No	No	No				--
Relief Spring	No	No	No				--
Cu Washer	Yes	Yes					No
End Plug	No	No	No				--
Weld Nut	Yes	Yes					No
Backing Plate	No	No	No				--
Strainer Body	No	No	No				--
Pick Up Tube	No	No	No				--
Strainer Gauze	No	No	No				--
Clamp Nut	No	No	No				--
Clamp Washer	Yes	Yes					No
O Ring	Yes	Yes					No
Lock Washer	No	Yes					No
Plain Washer	Yes	Yes					No
Drive Shaft	No	No	No				--

▨ Grey shaded parts are the base parts recommended by the **SPA**

◯ Purple circled part is the base part used in the industrial assembly plan

The **SPA** recommended *Body* and *Cover* as potential starting parts of assembly sequence Figure 8.4. This is consistent with the industrial practice, as *Body* was actually used as the base part. Other nine components, mainly fasteners were eliminated and a definitive answer was not available for the remainder. Also the reasons of the recommendation are transparent to the user. For example, right clicking the part *body* icon in the *Holding Bay* brings up the *Recommendation* and *Reason Explanation Dialogs* Figure 8.5 which indicate that the reason to recommend the *body* is that its relatively large and heavy.

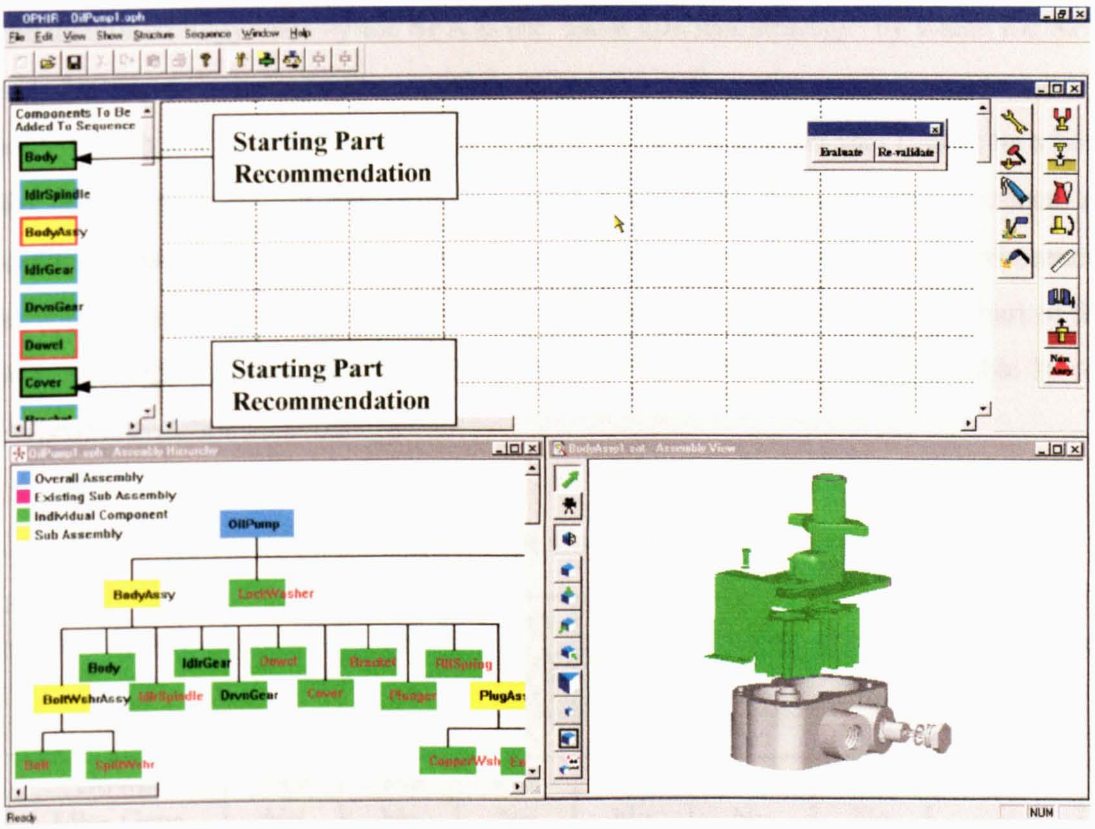
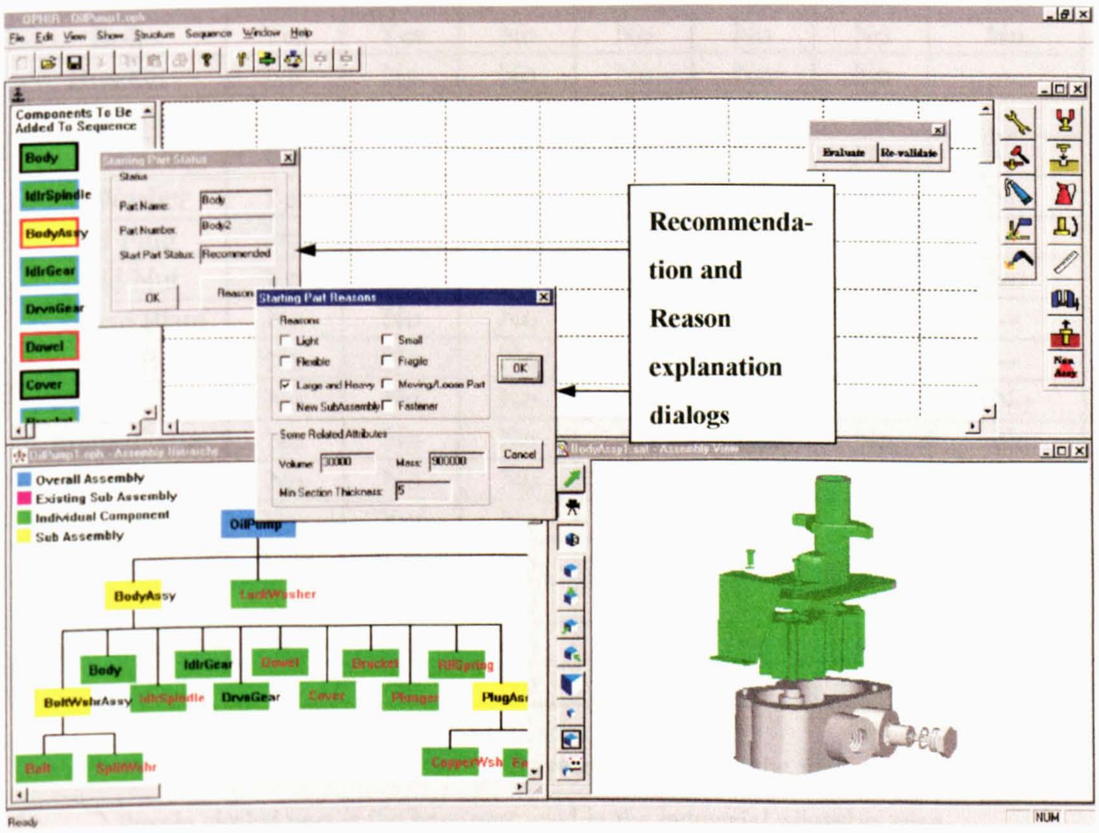


Figure 8.4 Starting Part Recommendations for Oil Pump



Starting Part Status

Status

Part Name: Body

Part Number: Body2

Start Part Status: Recommended

OK Reason

Starting Part Reasons

Reasons

☐ Light ☐ Small

☐ Flexible ☐ Fragile

☑ Large and Heavy ☐ Moving/Loose Part

☐ New SubAssembly ☐ Fastener

Some Related Attributes

Volume: 30000 Mass: 900000

Min Section Thickness: 5

OK Cancel

Figure 8.5 Starting Part Recommendation and Reason Explanation Dialogs

The default strategy used by the **SPA** is the "most efficient strategy" by which the **SPA** excludes parts for base parts as quickly as possible. For some user, more support is needed, such as they want to know more information about a part that cannot be a base part and the reasons. In that case, the "most transparent strategy" which collects all possible reasons for each conclusion may be preferred. The Ophir environment facilitates this. If the most transparent strategy is selected, the working flowchart of the **SPA** is slightly different, and the **SPA** provides all possible reasons as in Table 10 for its recommendations.

Table 10: Recommendations from SPA Using the "Most Transparent Strategy"

Part Name	Small	Light	Large Heavy	Free Moving	Flexible	Fragile	Start Part Status
Body	No	No	Yes	No	No	No	Yes
Idler Spindle	No	No	No	No	No	No	--
Dowel	Yes	Yes	No	No	No	No	No
Idler Gear	No	No	No	No	No	No	--
Driven Gear	No	No	No	No	No	No	--
Cover	No	No	Yes	No	No	No	Yes
Split Washer	Yes	Yes	No	No	No	No	No
Bolt	Yes	Yes	No	No	No	No	No
Bracket	No	No	No	No	No	No	--
Plunger	No	No	No	No	No	No	--
Relief Spring	No	No	No	No	Yes	No	No
Cu Washer	Yes	Yes	No	No	Yes	No	No
End Plug	No	No	No	No	No	No	--
Weld Nut	Yes	Yes	No	No	No	No	No
Backing Plate	No	No	No	No	No	No	--
Strainer Body	No	No	No	No	No	No	--
Pick Up Tube	No	No	No	No	Yes	No	No
Strainer Gauze	No	No	No	No	Yes	No	No
Clamp Nut	No	No	No	No	No	No	--
Clamp Washer	Yes	Yes	No	No	No	No	No
O Ring	Yes	Yes	No	No	Yes	No	No
Lock Washer	No	Yes	No	No	Yes	No	No
Plain Washer	Yes	Yes	No	No	No	No	No
Drive Shaft	No	No	No	No	No	No	--

▨ Grey shaded parts are the base parts recommended by the **SPA**

◯ Purple circled part is the base part used in the industrial assembly plan

After the sequence construction commenced with the selection of base part, *Body*, the **NPA** was invoked to find the best candidate for the next part. This considers the assembly structure of the Oil Pump (see Figure 8.2) and the attributes of *Body*. After the **NPA** finished reasoning, five candidates remained for the next part recommendation, *Idler Spindle*, *Dowel*, *Idler Gear*, *Driven Gear*, *Cover*, which were shown in Figure 8.6. All the five parts belong to the same subassembly *BodyAssy*, and on the same level (level 2) in the assembly structure as the initial part, *Body*. In addition, they all mate and have the same insertion direction with *Body*. The industrial assembly plan chose *Idler Spindle* as the next part. Thus the **NPA** has suggested a reasonable shortlist which includes the correct part.

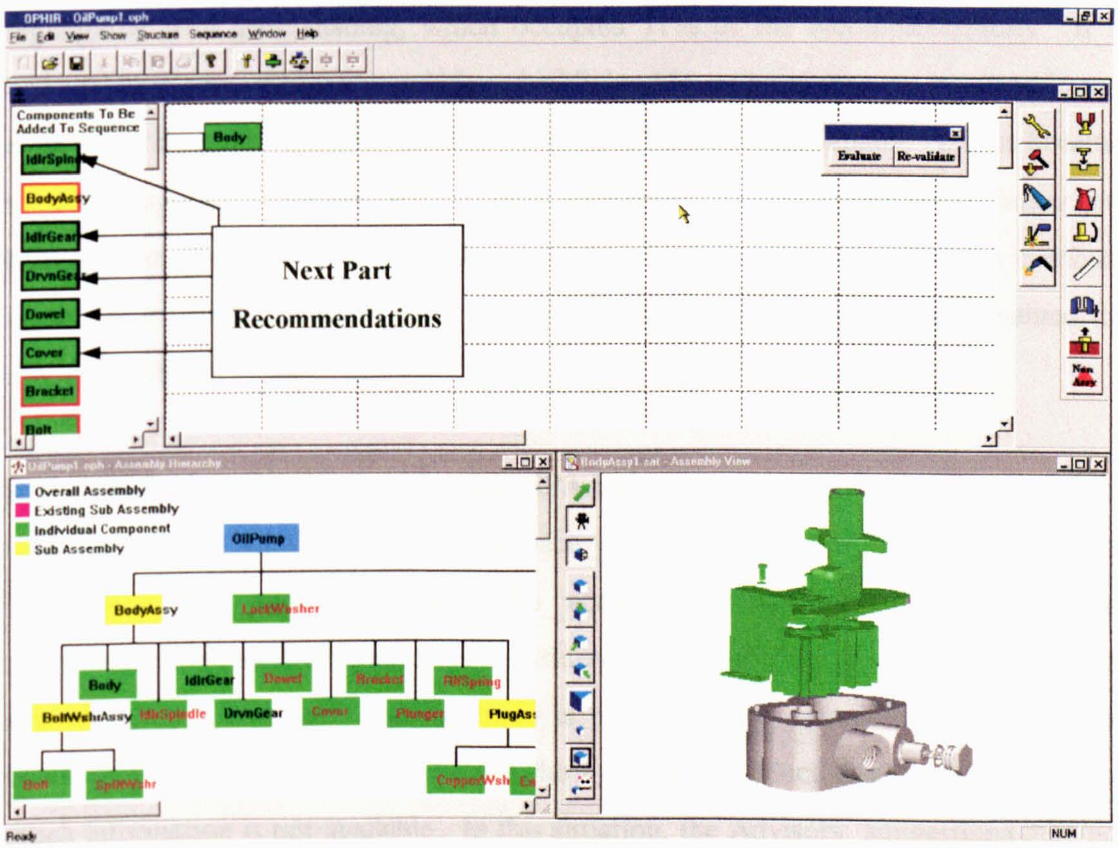


Figure 8.6 Next Part Recommendations for Oil Pump

When the five parts and *Bracket* have been added to the sequence, *Bolt* in *BltWshrAssy* was advised should be the next part. *BltWshrAssy* subassembly which includes the *Bolt* was identified as the best next part because it has the same insertion direction and mates with the last part selected. However, a subassembly must be built prior to insertion into the partial assembly. Thus **SPA** was triggered to identify the best base part of *BltWshrAssy*, which is *Bolt*. *Bolt* was recommended even though it is small and light and defined as a standard fastener. This occurs because all the parts in this subassembly are fasteners and relatively small and light. The rule precedence, described earlier in Chapter 7, was implemented and found *Bolt* is the larger and heavier, is not free moving, not flexible and not fragile. This differs from the industrial assembly plan which uses the *SplitWasher* as starting component of *BltWshrAssy*, however it is believed that this suggestion is reasonable.

The summary of the *Next Part* recommendation and the actually used next part in the industrial assembly plan are listed in Table 11. Consistent recommendations are highlighted with grey shading, which occupied 71% of the recommendations. It is believed that several recommendations highlighted in green circles are reasonable and even better than the industrial assembly plan in terms of assemblability. This is because the **SPA** suggested *Strainer Body* to start *StrainerAssy* subassembly as it is larger and heavier than the rest of parts. This suggestion is reasonable according to common sense. *Bolt* was suggested to start *BltPWshrAssy* with the same reason explained for the *Bolt* in *BltWshrAssy*.

In this Oil Pump example, the **NPA** and **SPA** have given reasonable suggestions for the base part of the whole assembly sequence and the set of best next parts. These recommendations mostly matched the actual industrial assembly plan. Where alternatives were offered, these are considered to benefit the sequence in terms of assemblability. However, this example is based on the complete product model. Generating such an assembly sequence during a new product design process, means much information is not available. In this situation, the Advisors' suggestions may not be as accurate. In some cases suggestions may not be made until user input relevant information, such as part mating information.

Table 11: Recommendations of Next Part for Oil Pump

Part Added to the Sequence by User	Next Part Recommended By the NPA	Part Used in Industrial Assembly Plan
	Body, Cover	Body
Body	Idler Spindle, Dowel, Dowel, Idler Gear, Driven Gear, Cover	Idler Spindle
Idler Spindle	Idler Gear, Dowel, Dowel, Driven Gear, Cover	Dowel
Idler Gear	Dowel, Dowel, Driven Gear, Cover	Dowel
Driven Gear	Dowel, Dowel, Cover	Idler Gear
Dowel	Dowel, Cover	Driven Gear
Dowel	Cover	Cover
Cover	Bolt, Bolt, Bolt, Bolt, Bracket	Split Washer
Bolt	Split Washer,…, Split Washer, Bolt, Bolt, Bolt, Bracket	Bolt
Split Washer	Split Washer, Split Washer, …, Bolt, Bolt, Bolt, Bracket	Split Washer
Bolt	Split Washer, Split Washer, …, Bolt, Bolt, Bracket	Bolt
Split Washer	Split Washer, Split Washer, Bolt, Bolt, Bracket	Bracket
Bracket	Split Washer, Split Washer, Bolt, Bolt	Split Washer
Bolt	Split Washer, Split Washer, Bolt	Bolt
Split Washer	Split Washer, Bolt	Split Washer
Bolt	Split Washer	Bolt
Split Washer	Plunger, Relief Spring, Copper Washer, End Plug	Plunger
Plunger	Relief Spring, Copper Washer, End Plug	Relief Spring
Relief Spring	Copper Washer, End Plug	Copper Washer
Copper Washer	End Plug	End Plug
End Plug	Strainer Body	Weld Nut
Strainer Body	Gauze, Backing Plate	Backing Plate
Gauze	Backing Plate	Strainer Body
Backing Plate	Weld Nut, Pickup Tube	Pickup Tube
Weld Nut	Pickup Tube	Gauze
Pickup Tube	Clamp Nut, Clamp Washer, O Ring	Clamp Nut
Clamp Nut	Lock Washer, Clamp Washer, O Ring	Clamp Washer
Clamp Washer	Lock Washer, O Ring	O Ring
O Ring	Lock Washer	Lock Washer
Lock Washer	Bolt, Drive Shaft	Plain Washer
Bolt	Plain Washer, Drive Shaft	Bolt
Plain Washer	Drive Shaft	Drive Shaft
Drive Shaft		

▨ Grey shadings highlight recommendations that match the industrial assembly plan

○ Green circles highlight recommendations that are believed to be better than the industrial assembly plan

Besides the 17 case studies, some other products have also been used to test the **SPA** and **NPA**. As there is not any industrial assembly sequence information provided with these products, the recommendations from the **SPA** and **NPA** have to be evaluated by experienced assembly engineers. The Vaporisor Valve Block is an example.

8.1.2 Vaporisor Valve Block – Another Case Study Example

The *Valve Block* is a large subassembly of a vaporiser; a real product from a health care product company – Pelon. A vaporiser mixes anaesthesia gases for operating theatre usage. Figure 8.7 shows the *Valve Block* in its final position in the overall product. Also identified in the figure is the *Inlet Valve Subassembly*. The *Valve Block* consists of 29 parts and is to be designed appropriately for manual assembly along a suitable assembly line. The *Inlet Valve Subassembly*, modelled in
Figure 8.8, is an important part of this product, thus the sequence generation support here was focused on the *Inlet Valve Subassembly* (short for *InletAssy* in assembly structure). Another subassembly, *Control Plate Assembly*, in the *Valve Block* is treated as an 'Existing Subassembly' type to simplify the analysis and it is short for *ContrlPltAssy* in assembly structure. The functional requirements of the *Inlet Valve Subassembly* have dictated the necessary parts required to regulate the gas flow. After parts and they based *Block* was added into the Ophir environment, the *Block* was recommended as the starting part of the *Valve Block*. This is obvious, as the *Block* is relatively large and heavy.

After the *Block* was added to the sequence, *ContrlPltAssy* and *Spool* were recommended Figure 8.9. This is because the *ContrlPltAssy* and *InletAssy* both mate with *Block*. As the *InletAssy* was treated as a 'New Subassembly' type with all its composing components and subassemblies, it has to be assembled before it goes into the sequence. So the **NPA** did not recommend the *InletAssy* itself. Instead, it triggered the **SPA** which inferred *Spool* as the best candidate to start the *InletAssy*.

Feedback from experienced assembly engineers confirmed that the recommendations are logic and reasonable.

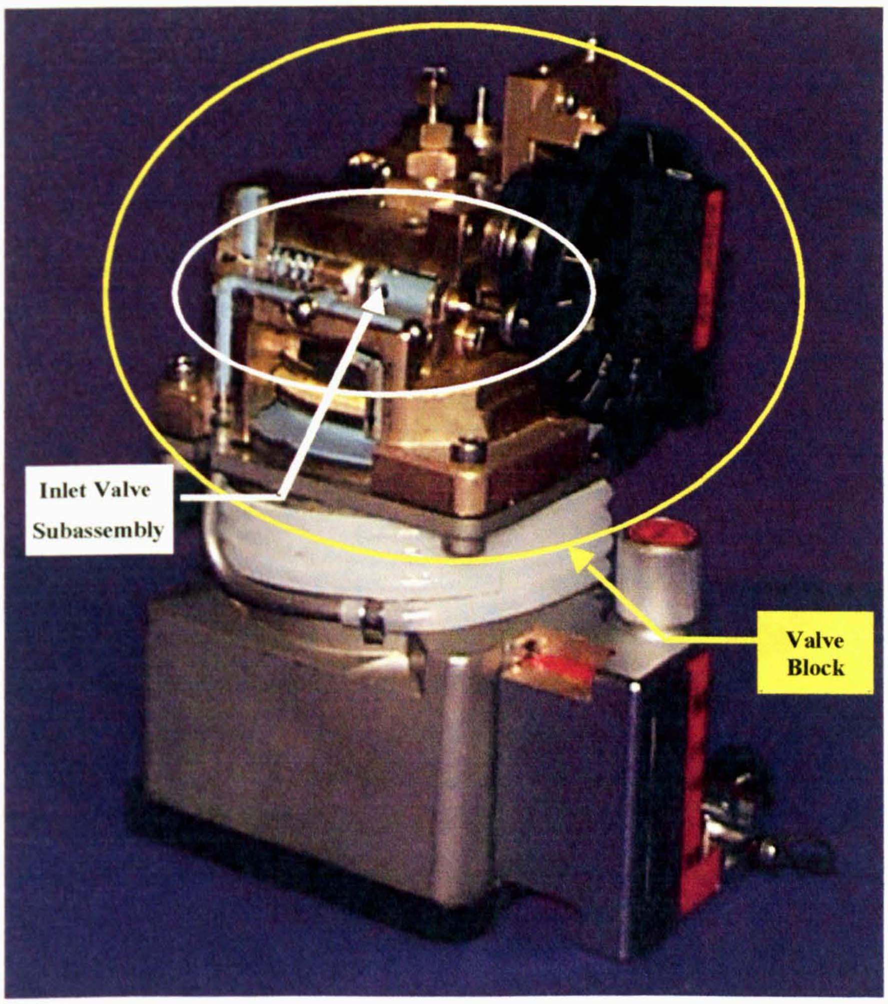


Figure 8.7 Inlet Valve Subassembly in Valve Block of the Vaporiser

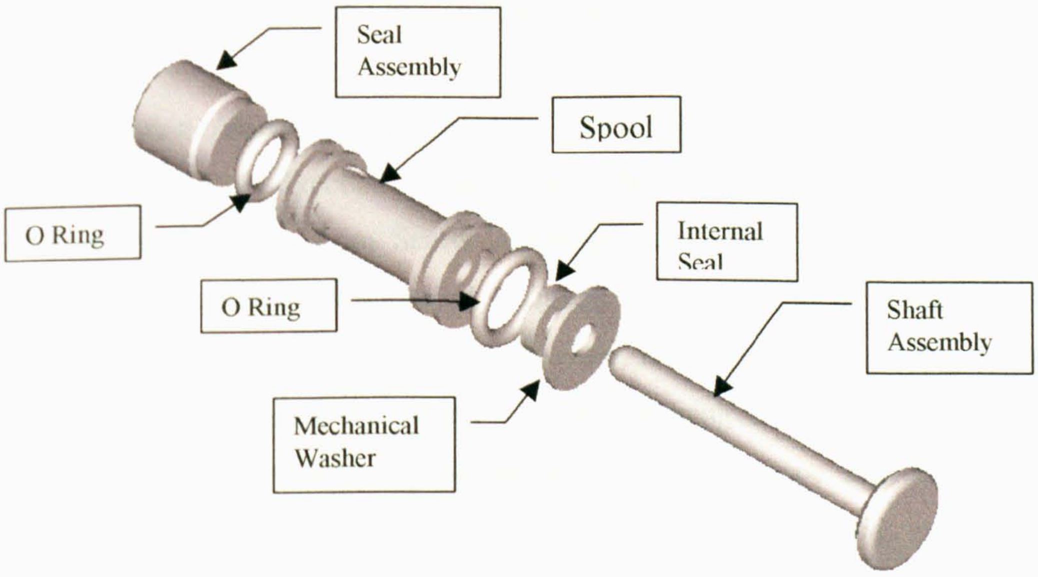
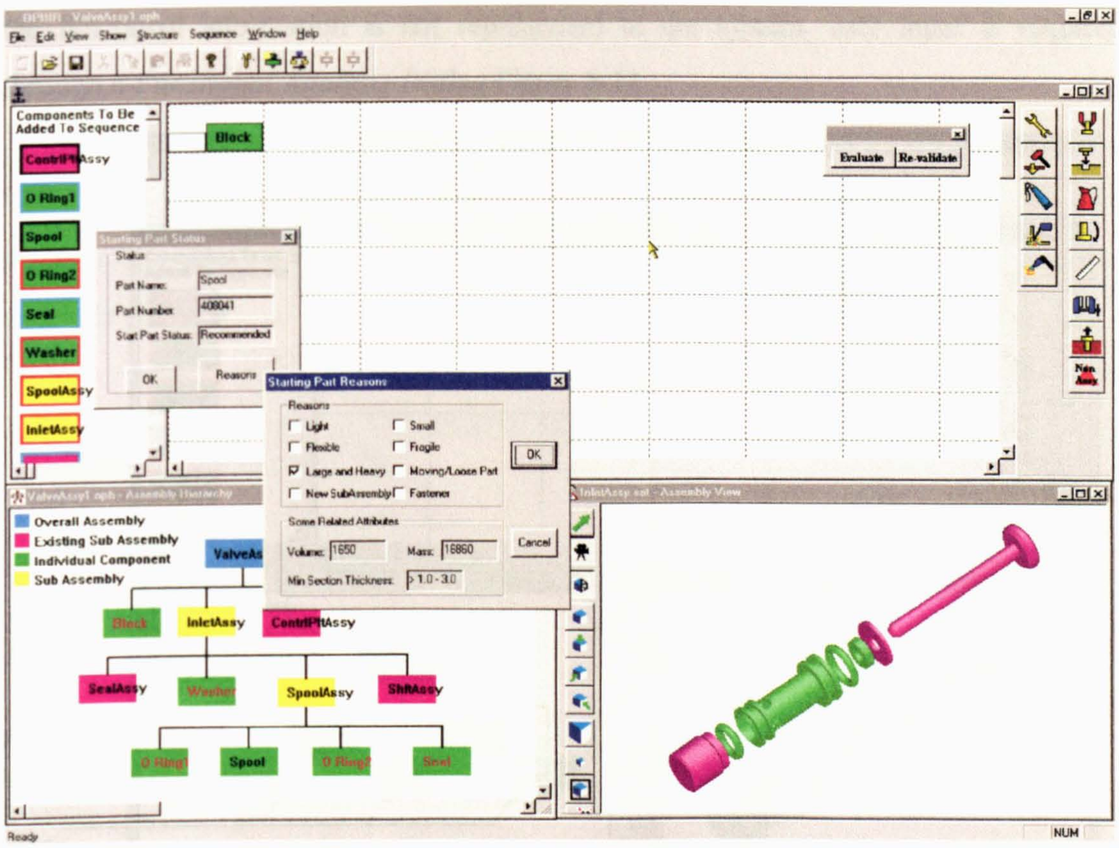


Figure 8.8 Exploded Model of Inlet Valve Subassembly



Starting Part Status

Status

Part Name: Spool

Part Number: 408041

Start Part Status: Recommended

OK Reasons

Starting Part Reasons

Reasons

☐ Light ☐ Small

☐ Flexible ☐ Fragile

☑ Large and Heavy ☐ Moving/Loose Part

☐ New SubAssembly ☐ Fastener

Some Related Attributes

Volume: 1650 Mass: 16860

Min Section Thickness: > 1.0 - 3.0

OK Cancel

Figure 8.9 Next Part Recommendations for Valve Block Assembly

8.2 Validation of the PCA

Several case studies have been used to test the **PCA**. The Trim Screw Assembly in the Lucas DFA Manual [30] is an simple example to demonstrate the validation process -- comparing the indications of the **PCA** with the manual DFA analysis.

The Trim Screw Assembly is a part of a car headlight assembly which aids the adjustment of headlight direction. It has 5 parts. When the Headlight Body Assembly and all the parts of the Trim Screw Assembly were interactively added into the assembly structure, the **PCA** identified *BodyAssy, Insert* and *Screw* as 'A' parts, and highlighted all the other parts, *Washer, Lockwasher, Knob* with names in red, indicating that they may be 'B' parts Figure 8.10. The reasoning of the **PCA** is based on functional requirements, material information, mating joint type information of the parts. The indications are consistent with the result of the manual DFA analysis in the Lucas DFA Manual. This also means that relevant functional information was brought into the Ophir environment with the parts and subassemblies. In situations that some of the functional information is not represented in the system, user input is required through a *Functional Analysis Dialog* Figure 8.11.

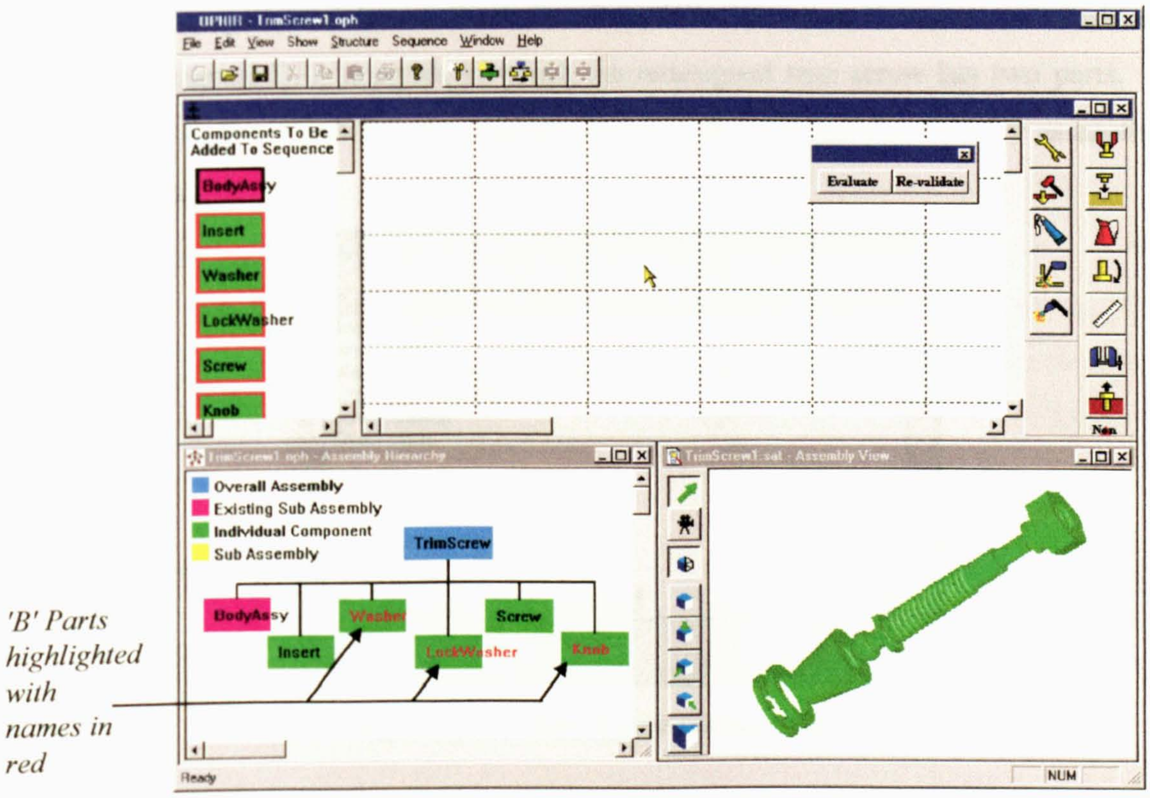
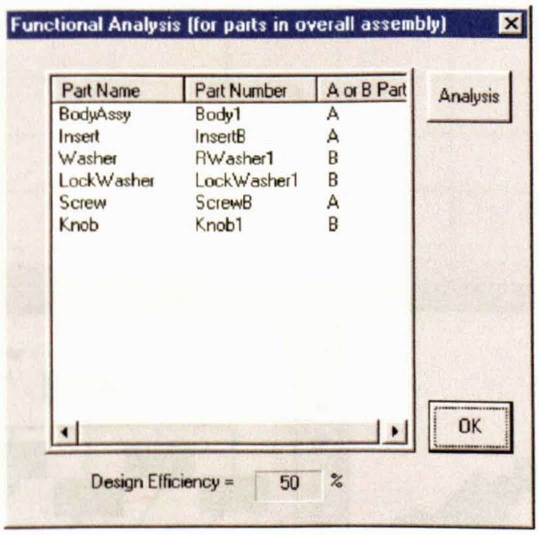


Figure 8.10 Starting Point Recommendation for Headlight Trim Screw

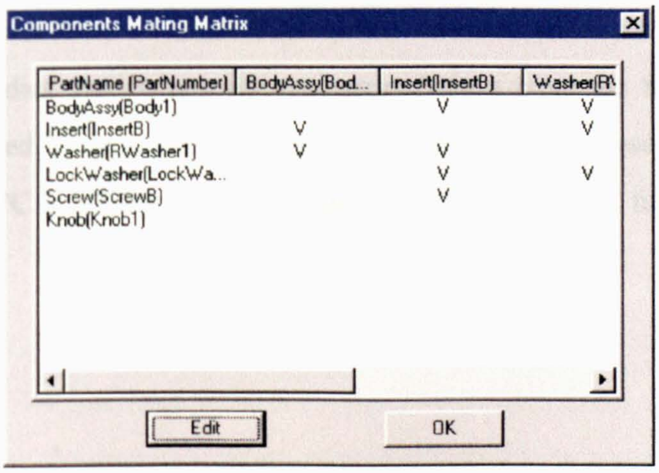


Part Name	Part Number	A or B Part
BodyAssy	Body1	A
Insert	InsertB	A
Washer	RWasher1	B
LockWasher	LockWasher1	B
Screw	ScrewB	A
Knob	Knob1	B

Design Efficiency = 50 %

Figure 8.11 Functional Analysis Dialog

When mating information was presented in the system, which can be viewed from the *Component Mating Matrix* Figure 8.12, the **PCA** highlighted one pair of parts: *Screw* and *Knob* with blue borders and another group of parts: *Insert*, *Washer* and *LockWashe* with black borders respectively Figure 8.13. The suggestions to combine *Screw* and *Knob* as one part, and *Insert*, *Washer* and *LockWasher* as another part are consistent with the manual DFA analysis, in which the redesigned trim screw has two parts. It combined *Screw*, *Knob* as one component, I*nsert*, *Washe*r and *LockWasher* as another component.



PartName (PartNumber)	BodyAssy(Bod...	Insert(InsertB)	Washer(R\
BodyAssy(Body1)		V	V
Insert(InsertB)	V		V
Washer(RWasher1)	V	V	
LockWasher(LockWa...		V	V
Screw(ScrewB)		V	
Knob(Knob1)			

Figure 8.12 Trim Screw Mating Matrix

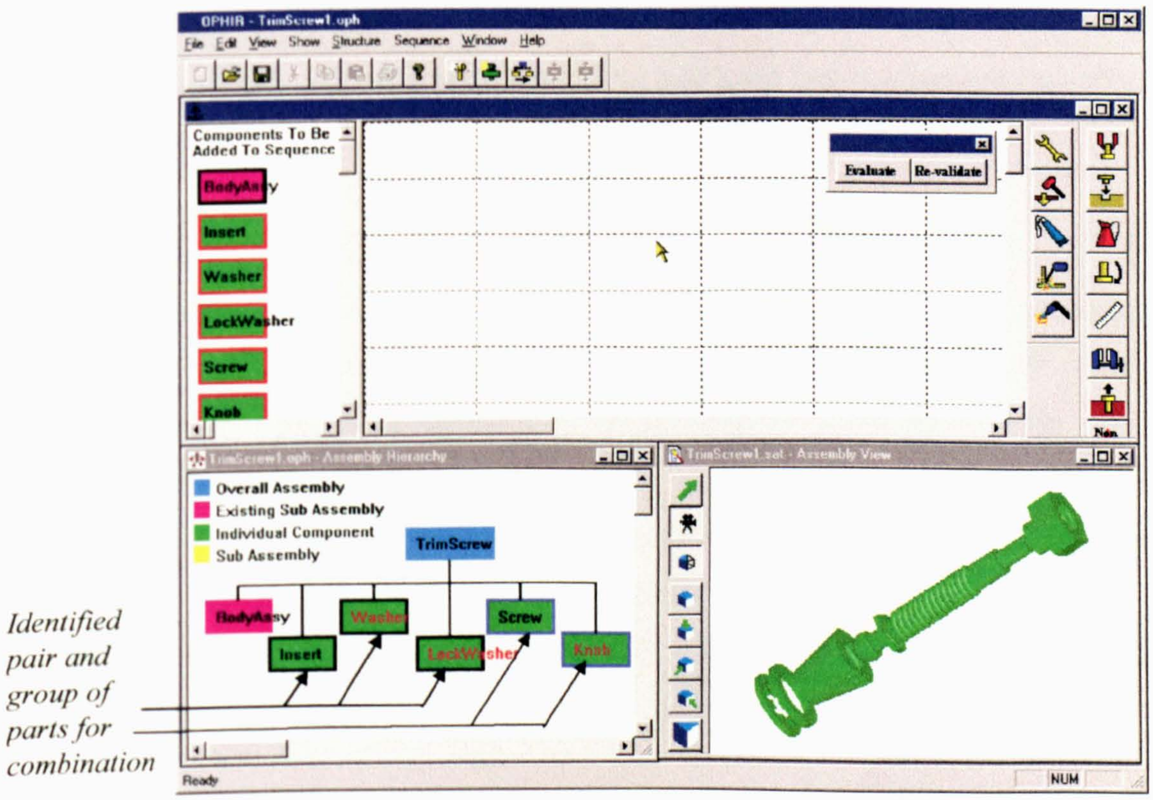


Figure 8.13 Highlighted Parts Group for Combinations

In summary, a number of case studies have been used to test the **SPA** and **NPA**. In 14 out of the 17 case studies that come with industrial assembly plan information, the **SPA** identified the actual component which was used to start the assembly sequence in the industrial assembly plan. The 17 case studies show that on average 65% of the recommendation shortlists from the **NPA** contain the part used in the industrial assembly plan. Some recommendations of the **SPA** and **NPA** are considered to benefit assembly sequence in terms of assemblability than the industrial assembly plans. There are other case studies used where the recommendations from the **SPA** and the **NPA** have been confirmed by experienced assembly engineers. A few case studies have been used to test the **PCA**. There are some satisfactory results, but further validation is needed.

Chapter 9 Discussion, Conclusions and Recommendations for Further Research

9.1 Discussion

To reduce product cost, lead time, and at the same time design quality into a product, assembly oriented design is needed. This requires consideration of the product as a whole as an assembly to optimise product structure and to explore how components can be fitted together, before the consideration of individual components in detail. This has many implications, including changing the way we design our CAD systems, to realise product design from top down rather than from bottom up.

Design for Assembly evaluation (DFA) can play an important role in realising assembly-oriented design. This has been proved by many successful DFA case studies. DFA attracts attention on the complete product (or subassembly) as a whole to promote the ideas of part count reduction, part standardisation and product modularization. It provides a systematic procedure for analysing proposed designs from the point of view of assembly and manufacture and highlights problems of assembly operation. DFA results in simpler and more reliable products which are less expensive to assembly and manufacture.

Literature review of DFA shows that many researchers believed that DFA should be incorporated into CAD systems to facilitate the automatic evaluation of assemblability from the early design stage even though that current CAD systems themselves need to be improved to give more support in design synthesis. This enables assembly problems to be tackled early and reduces product cost, as in the integrated system the product definition can be taken as the input for evaluation from the early design stage, and the improvement can be made directly on the product design. This integrated system also facilities automatic inference of DFA related component properties directly from CAD model to reduce subjective user input in DFA evaluation.

The importance of assembly sequence generation and feature recognition in automatic DFA evaluation has been evidenced, since not only product structure, component features, but also assembly process (sequence and operations) influence assemblability. So assembly planning (not necessarily fully detailed) should be brought into the design environment to support DFA evaluation from the early design stage. Enhanced product models to extend beyond current geometric-modelling packages to represent component interactions and assembly process information is needed to facilitate automatic DFA evaluation. Furthermore, functional information should be considered during the evaluation process.

Current research in DFA, which includes integrating DFA with CAD and assembly planning and providing intelligent support for DFA, shows that there is an essential need to improve DFA and associated design environment in the following aspects:

- Incorporating DFA into product design process as early as possible to highlight problems proactively and make changes for improvement at a minimum cost - Proactive DFA;
- Constructing an assembly plan, especially an appropriate assembly sequence, in parallel with design process to facilitate proactive DFA evaluation;
- Providing intelligent support for automatic inference of DFA related properties to reduce tedious and error-prone re-entry of data.

From the review of computer-aided assembly planning methods and investigations of industrial assembly planning practice, differences between the requirements of assembly engineers and the automated assembly planning systems have been found. To accommodate the difference, an interactive planning approach with decision support rather than a completely automated assembly planning system is needed. In such an interactive system, geometric hard constraints and heuristic based soft constraints should be integrally applied, and the user should be able to collaborate with artificial intelligence throughout the planing process with the system focusing on computable constraints, and the user focusing on the more incomputable constraints and making the final decisions. This is close to industrial practice, and enhances the efficiency of planning. It is such an assembly planning system that should be integrated with CAD environment to facilitate planning in parallel with product design to support the early assemblability evaluation in the design process.

One of the approaches adopted by the Ophir Project is to interrogate the product model by geometric reasoning to infer DFA and assembly sequence data. To provide advice at an early stage of the design process, there will be situations where the necessary geometric information is uncertain or not available. Expert support may be used to deploy heuristic knowledge about components to provide 'best guess' answers for product model interrogations. Heuristic methods can also replace geometric reasoning algorithms where an approximate answer provided quickly is more useful than a precise answer produced by lengthy computation. Most importantly, it is always useful to provide the designer with decision support based on relevant best practice, as design is a creative activity built on established knowledge and experience. So expert support is needed for the Ophir environment.

There are several issues regarding DFA and the assembly sequence definition requiring expert support. It has been found that each has very different requirements, and the knowledge involved is also different. In many cases, the support is required concurrently - the nature of the Ophir environment is concurrent. For some of these reasons, it is very difficult to adopt a single approach, such as a conventional dominant knowledge-based expert system to support the Ophir environment. So the strategy used for the expert support is to deploy an *Expert Assembler* which contains several separated modules: each tackling a problem area with a suitable problem solving strategy, knowledge representation and reasoning method. The rule-based approach can provide a convenient means to represent assembly expertise, and new knowledge can be appended easily to the knowledge base. The case-based approach is helpful in situations where a number of case studies can be employed to support the design decisions. Procedural and combined implementations may also be useful to deploy heuristics. The self-contained modularity has improved the maintainability of the *Expert Assembler*.

The role of the expert support for the Ophir environment is to guide the assembly sequence definition, automatically infer DFA related properties, highlight assembly problems instantly and provide improvement suggestions if possible. So the properties required of the experts are: detecting relevant information automatically, reasoning about the detected information, updating the environment data, feeding back the reasoning results in user friendly ways. These clearly include perceiving environment and goal-directed behaviour. As changes made in one area will influence another area and the concurrent nature of the Ophir environment requires the expert support provided concurrently, this implies that different expert modules should communicate and cooperate with each other. In an interactive environment, they should also cooperate with the user. As the properties required of the expert modules: perceiving environment, goal-directed behaviour and cooperation, are intelligent agent properties, so we call each expert module an expert agent. In each expert agent proposed, there are several expert elements to fulfil closely related tasks. Because of the autonomy nature of the expert agents, inference can be made continuously 'in the background' of the Ophir environment in relation to available information, thus a user is relieved from the extra burden of responding to many assembly-related questions concerning the design and its components. This also makes it possible to automatically highlight assembly problems instantly in the design environment.

For implementation of the experts, an expert system shell, CLIPS, has been selected for its flexibility, easy integration, reasonable performance, and low cost, and it is embedded into C++ environment to hold production rules. Heuristic search algorithms and C++ functions have been developed to complement the rule-based reasoning.

The thesis systematically presented ideas in support for proactive DFA evaluation from the early design stage, with focus on support for part count reduction. Part count reduction is usually based on dialogue with the user. There is little computational support in this aspect in any of the DFA methodologies and related literature. A method has been proposed to automatically combine candidate parts to reduce part count, but it is based on complete geometry, which means that the method may only be applied when product design is complete. This research fills up the gap: it provides computational support for part count reduction from the early design stage. The *Part Count Advisor* - an element of the *Expert Assembler*, automatically highlights candidate parts for elimination, candidate pairs or groups of parts for combination. Case-based reasoning has been explored for providing redesign support, such as support the generation of redesign solutions in part consolidation.

The current work has also made new progress in assembly sequence generation. An interactive sequence generation process which best suits industrial requirements and proactive DFA evaluation has been validated. Computer support for the selection of the base part and the most suitable next part in an interactive assembly sequence generation process has been realised by two expert elements: the *Starting Part Advisor* and the *Next Part Advisor*. The Advisors cooperate with each other and with the user. They provide suggestions dynamically and transparently.

The relationship between the expert support and geometric reasoning is very interesting. Due to limited time, how the expert system could support the product model interrogation has not been investigated deeply. However it is felt that this is both necessary and very important.

In developing the expert agents, it is revealed that different approaches can be used. They could be implemented entirely by C++, rather than using a rule-based expert system shell and C++ as adopted in the prototype system. The quantity of knowledge is currently not large: mainly some general knowledge, and some domain specific knowledge, but without any company specific data. For a more comprehensive assembly-oriented CAD system that can be used in industrial environment, a more sophisticated expert system shell may need to be integrated to hold the vast amount of knowledge involved, otherwise a lot of software coding for the *Expert Assembler* could be involved. However the performance of the system with the integrated CLIPS expert system shell demonstrates the feasibility of a larger scale implementation.

9.2 Conclusions

The research has focused on investigating the notion of assembly-oriented design, and has presented an innovative approach towards supporting an assembly-oriented CAD environment. An *Expert Assembler* approach has been adopted and deployed to support for proactive DFA and assembly planning in a CAD environment. Expert agents are created to fulfil the support requirements.

The strategy adopted to realise the *Expert Assembler* has been to create several separated modules: each is an expert agent devised to tackle a problem area. The strategy and the expert agent properties bring a number of advantages:

- Different knowledge representation and reasoning methods can be used
- Better maintainability
- Inference can be made continuously 'in the background' in relation to available data
- Automatically highlight assembly problems instantly.

Three expert elements: *Part Count Advisor*, *Starting Part Advisor*, and *Next Part Advisor* have been realised to support part count reduction and Assembly sequence generation.

The research brings computer support for part count reduction into the design process. The *Part Count Advisor* automatically highlights parts whose purpose is not critical to product function as candidates for elimination. It also highlights pairs or groups of parts that could be combined. Case-based reasoning has been explored for providing redesign solutions, such as how to combine candidate parts.

The *Starting Part Advisor* and the *Next Part Advisor* cooperate with each other and with the user to dynamically and transparently provide suggestions on the base part and the next part selections for assembly sequence definition. This brings new progress in assembly sequence generation.

Case study validation has shown that the three implemented *Advisors* can greatly assist the designer in sequence definition and part count reduction.

9.3 Recommendations for Further Research

It is desirable to extend the work to provide more depth and breath, such as to increase and refine the knowledge in use, so the recommendations from the *Advisors* can be more accurate.

It is also very useful to develop the assembly-oriented CAD environment into an integrated design and assembly planning system, so the assembly sequence generated for proactive DFA evaluation can be eventually developed into an assembly plan which can be used by industry.

Highlighting opportunities and assembly problems is an important support step. Providing redesign solutions is a more difficult task. Case-based reasoning has been explored for support of part count reduction based on a number of successful DFA case studies. From the experience gained, it suggests that much further efforts are required to provide redesign solution support.

Also any design changes, such as the change of assembly structure, should not adversely affect the product performance, so functional understanding of the assembly design is important. Jin-kang Gui et al [180], have described that functional understanding of assembly modelling is a key step towards a real computer-aided design environment that can support the early design stage. So further research is also directed to the mapping from functional modelling to the assembly modelling. This can bridge the assembly structure to functional structure and provides essential support for product structure optimisation.

References:

1 Li R.K. and Hwang, C.L. "A framework for automatic DFA system development", *Computers and Industrial Engineering*, 1992, Vol. 22, No. 4, pp. 403-413.

2 Owen, A.E. Assembly with Robots, London, Kogan Page, 1985.

3 Nevins, J.L. and Whitney, D.E. *Concurrent Design of Products and Processes*, New York, McGraw-Hill, 1989.

4 Hsu, W., Fuh, J.Y.H. and Zhang, Y.F. "Synthesis of design concepts from a design for assembly perspective", *Computer Integrated Manufacturing Systems*, 1998, Vol. 11, No. 1-2, pp. 1-13.

5 Andreasen, M.M., Kähler, S., Lund, T. with Swift, K. *Design for Assembly*, Second Edition, IFS Publications, UK, 1988.

6 Holbrook, A.E.K. and Sackett, P.J. "Positive design advice for high precision robotically-assembled product", *Assembly Automation: Proceedings of the 9th International Conference*, 1988, pp. 90-181.

7 Warnecke, H.J., Bäßler, R. "Design for assembly – Part of the design process", *Annals of the CIRP*, 1988, Vol. 37, No. 1, pp. 1-4.

8 Boothroyd, G. "Product design for manufacture and assembly", *Computer-Aided Design*, July 1994, Vol. 26, No. 7, pp. 505-520.

9 Boothroyd, G., Alting, L. "Design for assembly and disassembly", *Annals of the CIRP*, 1992, Vol. 41, No. 2, pp. 625-636.

10 Boothroyd, G., Dewhurst, P., Knight, W. *Product Design for Manufacture and Assembly*, Marcel Dekker, Inc., 1994.

11 Tatikonda M, "Design for assembly: a critical methodology for product reengineering and new product development", *Production and Inventory Management*, 1995, Vol. 35, No. 1, pp. 31-37.

12 Redford, A. "Design for assembly", *Engineering Designer*, pp. 12-14, Sep/Oct 1994.

13 Barnes, J.C., Dalgleigh, G.F., Jared, G.E.M., Swift, K.G., Tate, S.J. "Assembly sequence structures in design for assembly", In *Proceedings of IEEE International Symposium on Assembly and Task Planning (ISATP '97)*, 1997, pp164-169.

14 De Fazio, T., Rhee, S.J. and Whitney, D.E. "Design-specific approach to Design for Assembly (DFA) for complex mechanical assemblies", *IEEE Transactiojs on Robotics and Automation*, October 1999, Vol. 15, No. 5, pp. 869-881.

15 Hsu, W., Lee, C.S.G. and Su, S.F. "Feedback approach to design for assembly by evaluation of assembly plan", *Computer-Aided Design*, July 1993, Vol. 25, No. 7, pp. 395-409.

16 Ophir Project, *Knowledge Engineering Report*, 1998.

17 Jared, G., Limage, M., Sherrin, I., Swift, K. "Geometric reasoning and design for manufacture", *Computer-Aided Design*, 1994, Vol 26, No 7, pp. 528 - 536.

18 Simpson, T.V., Bauer, M.D., Allen J.K. and Mistree, F. "Implementation of DFA in conceptual and embodiment design using decision support problems", DE-Vol. 82, *Design Engineering Technical Conferences Volume 1 ASME 1995.*

19 Lee, S., Kim, G.J. and Bekey, G.A. "Combining assembly planning with redesign: an approach for more effective DFA", In Proceedings IEEE International Symposium on Assembly and Task Planning, 1993, pp. 319-325.

20 Kim, G.J., Lee, S. and Bekey, G.A. "Comparative Assembly Planning During Assembly Design", In *Proceedings IEEE International Symposium on Assembly and Task Planning*, 1995, pp. 319-326.

21 Angermüller, G., Moritzen, K. "A knowledge-based system supporting product design for mechanical assembly", In *Proceedings of the 1st Conference of Artificial Intelligence and Expert System in Manufacturing*, March 1990, pp. 181-192.

22 Boothroyd, G. and Dewhurst, P. Design for Assembly -- A Designer's Handbook, Department of Mechanical Engineering, University of Massachusetts at Amherst, 1983.

23 Salford University Industrial Center (UK), *Design for Assembly Handbook*, 1981.

24 Jakiela, M. and Papalambros, P. "Design and implementation of a prototype intelligent CAD system", *ASME Journal of Mechanisms, Transmission, and Automation in Design*, June 1989, Vol. 111, No. 2, pp. 252-258.

25 Myers, W. L., Dixon, J. R. and Simmons, M. K. "Computer analysis of mechanical assemblies from a CAD database: mechanical handling times", In *Proceedings of ASME Computers in Engineering Conference*, New York, USA 9-13 Aug 1987, pp. 167-172.

26 Sturges, R.H. and Kilani, M.I. "Towards an integrated design for an assembly evaluation and reasoning system", *Computer Aided Design*, 1992, Vol. 24, No. 2, pp. 67-79.

27 Miyakawa, S., Ohashi, T. "The Hitachi Assemblability Evaluation Method", In *Proceedings of International Conference on Product Design for Assembly*, Newport, RI, April 15-17, 1986.

28 Miyakawa, S., Ohashi, T. and Iwata, M. "The Hitachi new Assemblability Evaluation Method (AEM)", In *Transactions of the North American Manufacturing Research Institute*. SME, 1990, pp. 352-359.

29 Miles, B. L. and Swift, K. G. "Working together", *Manufacturing Breakthrough*, March/April 1992, pp. 69-73.

30 Lucas Engineering Systems Ltd, University Of Hull, *Design For Manufacture and Assembly Practitioners Manual*, Version 10, 1993.

31 C.S.C Manufacturing., University of Hull, *Design for Assembly / Manufacturing Analysis Practitioners Manual*, Version 10.5, 1994.

32 Kroll, E., Lenz, E. and Wolberg, J.R. "A knowledge-based solution to the design-for-assembly problem", *Manufacturing Review*, June 1988, Vol. 1, No. 2, pp. 104-108.

33 Hsu, W., Lim, A., Lee, C.S.G. "Conceptual level Design for Assembly analysis using state transitional approach", In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, April 1996, pp. 3355-3360.

34 Rosario, L.M. and Knight, W.A. "Design for assembly analysis: extraction of features from a CAD system database", *Annals of CIRP*, 1989, Vol. 38, No. 1, pp.13-16.

35 Eversheim, W. and Baumann, M. "Assembly-oriented design process", *Computers in Industry*, 1991, Vol. 17, pp. 287-300.

36 Molloy, E., Yang, H., Browne, J. "Design for assembly within concurrent engineering", *Annals of the CIRP*, 1991, Vol. 40, No. 1, pp. 107-110.

37 Scarr, A.J. "Product design for automated manufacture and assembly", *Annals of the CIRP*, 1986, Vol. 35, No. 1.

38 Stoll, H.W. "Design for manufacture: an overview", *Design for Manufacture*, Addison-Wesley Publishers Ltd. 1991, pp.107-129.

39 Miles, L.D. *Techniques of Value Analysis and Engineering*, New York: McGraw-Hill, 1961.

40 Zorowski, C.F. "Product design merit: relating design to the ease of assembly", In *Proceedings of the 8th Internation Conference on Assembly Automation*, 1987, pp161-169.

41 Boothroyd G. and Dewhurst P. *Product Design for Assembly*, Wakefield, RI, USA, 1989.

42 Boothroyd Dewhust Inc, "Design for Assembly product design analysis software", *http://www.design-iv.com.*

43 CSC Computer Sciences Ltd, *TeamSET™ User Guide*, Version 2, 1996.

44 Huang, G.Q. *Design for X: concurrent engineering imperatives*, Chapman & Hall, London, 1996.

45 TeamSet home page, *http//www.teamset.com/frame.html.*

46 Wood B.O., Cohen P.H., Medeiros D.J. and Goodrich J.L. "Design for robotic assembly", *Robotics and Manufacturing Automation*, PED, 1986, Vol. 15, ASME book No. Goo321.

47 Gairola, A. "Design for Assembly: a challenge for expert systems", *Robotics*, 1986, Vol. 2, No. 3, pp. 249-257.

48 Jakiela, M., Papalambros, P., Ulsoy, A.G. "Programming optimal suggestions in the design concept phase: application to the Boothroyd assembly charts", *Journal of Mechanisms, Transmissions, and Automation in Design*, June 1985, Vol. 107, pp. 285-291.

49 Molloy, O., Tilley S. and Warman, E. A. *Design for Manufacturing and Assembly: Concepts, Architectures and Implementation*, Chapman & Hall, 1998.

50 Chen, Y.T., Young, R.E. "PACIES: A part code identification expert system", *IIE Transactions*, Part 2, 1988, Vol. 20, pp. 132-136.

51 Swift, K.G. *Knowledge-Based Design for Manufacture*, Kogan Page Ltd, 1987.

52 Lucas Engineering & Systems, "Expert system aids design for assembly", *Assembly Automation*, 1989, Vol. 9, No. 3, pp. 132-136.

53 Daabub, A.M. and Abdalla, H.S. "A computer-based intelligent system for Design for Assembly", *Computers & Industrial Engineering*, 1999, Vol. 37, No. 1-2, pp. 111-115.

54 Young, R.E., Greef, A. and O'Grady, P. "An artificial intelligence-based constraint network system for concurrent engineering", *International Journal of Production Research*, 1992, Vol.30, No. 7, pp. 1715-1735.

55 Oh, J.S., O'Grady P. and Young, R.E. "Constraint network approach to Design for Assembly", *IIE Transactions*, Institute of Industrial Engineers, Feb. 1995, Vol. 27, No. 1, pp. 72-80.

56 O'Grady, P. and Oh, J.S. "A review of approaches to Design for Assembly", *Concurrent Engineering*, LISDEM Technical Report, 1991, pp. 5-11.

57 Sackett, P. J. and Holbrook, A.E.K. "DFA as a primary process decrease design deficencies", *Assembly Automation*, 1988, Vol. 8, No. 3, 137-140.

58 Sherrin, I., Jared, G., Limage, M. and Swift, K.G. "Automated manufacturability evaluation" In *Proceedings of International Forum DFMA*, Newport, RI, USA, June 1991.

59 Molloy, E., Yang, H. and Browne, J. "Feature-based modelling in design for assemby", *International Journal of Computer Integrated Manufacturing*, 1993, Vol. 6, No. 1&2, pp.119-125.

60 Li, R.K. and Huang, C.L. "Assembly code generation from a feature-based geometric model", *International Journal of Production Research*, 1992, Vol. 30, No. 3, pp.627-646.

61 Hoummady, A. and Ghosh, K. "Generation and evaluation of assembly sequences in computer-automated process planning", *International Journal of Computer Applications in Technology*, 1989, Vol. 2, No. 3, pp. 151-158.

62 Seidel, U.A., Swift, K.G. "Operation networks for coordinated DFA and sequence planning", In *Proceedings of 10th International Conference on Assembly Automation"*, 1989, pp. 539-546.

63 Kim, G.J., Lee, S. and Bekey, G.A. "Interleaving Assembly Planning and Design", *IEEE Transactions on Robotics and Automation*, April 1996, Vol. 12, No. 2, pp. 246-251.

64 Gu, P. and Yan, X. "CAD-directed automatic assembly sequence planning", *International Journal of Production Research*, 1994, pp. 3069-3100.

65 Lim, S.S., Lee, I.B.H., Lim, L.N. and Ngoi, B.K.A. "Computer-aided concurrent design of product and assembly processes: a literature review", *Journal of Design and Manufacturing*, 1995, Vol. 5, pp. 67-88.

66 Nilsson, Nils J. *Principles of Artificial Intelligence*, Springer-Verlag, 1980.

67 Rich, E. *Artificial Intelligence*, New York, McGraw-Hill, Inc.1983.

68 Hoffman, R.L. "Automated assembly in a CSG domain", *IEEE International Conference on Robotics and Automation*, May 1989, pp. 210-215.

69 Sanderson, A.C., Homem de Mello, L.S. and Zhang, H. "Assembly sequence planning", *AI Magazine*, Spring 1990, Vol. 11, No. 1, pp. 62-81.

70 Bourjault, A. *Contribution a une Approche Methodologique de L'assemblage Automatise: Elaboration Automatique des Sequences Operatoires*, PhD thesis, Faculte des Sciences et des Techniques de l'Universite de Franche-Comte, Besancon, France, Nov. 1984.

71 De Fazio, T.L., Whitney, D.E., "Simplified generation of all mechanical assembly sequence", *IEEE Journal of Robotics and Automation*, 1987, Vol. RA-3, No. 6, pp. 640-658.

72 Sanderson, A., Homem de Mello, L., "AND/OR graph representation of assembly plans", *IEEE Transactions on Robotics and Automation*, 1990, Vol. 6, No. 2, pp. 188-199.

73 Homem de Mello, L., Sanderson, A. "A correct and complete algorithm for the generation of mechanical assembly sequences", *IEEE Transactions on Robotics and Automation*, April 1991, Vol. 7, No. 2, pp. 228-240. (Also in Homem De Mello, L., Lee, S. editors: *Computer-Aided Mechanical Assembly Planning*, Kluwer Academic Pub, 1991, pp. 129-162.)

74 Homem de Mello, L., Sanderson, A. "A basic algorithm for the generation of mechanical assembly sequences," in *Computer-Aided Mechanical Assembly Planning*, Kluwer Academic Pub, 1991, pp. 163-190.

75 Zorc, S., Noe, D. and Konoenko, I. "Efficient derivation of the optimal assembly sequence from product description", *Cybernetics and Systems*, 1998, Vol. 29, No. 2, pp. 159-179.

76 Delchambre, A. *Computer-aided Assembly Planning*, Chapman & Hall, London, UK, 1992.

77 Lui, M.M., *Generation and Evaluation of Mechanical Assembly Sequences Using the Liaision Sequence Method*, S. M. Thesis, MIT, Cambridge, MA, May 1988.

78 Abell, L.E. *An Interactive Software Tool for Editing and Evaluating Mechanical Assembly Sequences Based on Fixturing and Orientation Requirements*, S. M. Thesis, MIT, Mechanical Engineering Department, Cambridge, MA, August 1989.

79 Baldwin, D.F., Abell, T.E., Lui, M.C., De Fazio, T.L. and Whitney, D.E. "An integrated computer aid for generating and evaluating assembly sequences for mechanical products", *IEEE Transactions on Robotics and Automation*, 1991, Vol. 7, No. 1, pp. 78-94.

80 Wolter, J. "On the automatic generation of assembly plans", *Computer-Aided Mechanical Assembly Planning*, edited by Homem de Mello, L.S. and Lee, S., Kluwer Academic Pub, Boston, 1991, pp. 263-288.

81 Wolter, J. "A constraint-based approach to planning with subassemblies", *IEEE ICSE*, Pittsburgh, PA, August 1990, pp. 412-415.

82 Wolter, J. *Representing Subassembly Tree by Deepest Dommon Ancestor Relations*, Tech Rept, 90-009, Computer Science Dept, Texas A&M University, May 1990.

83 Huang, Y.F. and Lee, C.S.G. "Precedence knowledge in feature mating operation assembly planning", In *Proceedings of the IEEE International Conference on Robotics and Automation*, Scottsdale, Arizona, May 14-19, 1989, pp. 216-221.

84 Huang, Y.F. and Lee, C.S.G. "An automatic assembly planning system", In *Proceedings of the IEEE International Conference on Robotics and Automation*, Cincinnati, Ohio, May 13-18, 1990, pp. 1594-1599.

85 Huang, Y.F. and Lee, C.S.G. "A framework of knowledge-based assembly planning", In *Proceedings of the IEEE Intentional Conference on Robotics and Automation*, Sacramento, California, April 9-11, 1991, pp. 599-604.

86 Lin, A.C. and Chang, T.C. "An integrated approach to automated assembly planning for three-dimensional mechanical products", *International Journal of Production Research*, 1993, Vol. 31, No. 5, pp. 1201-1227.

87 Lin, A.C. and Chang, T.C. "3D MAPS: Three-dimensional mechanical assembly planning system", *Journal of Manufacturing Systems*, 1993, Vol. 12, No. 6, pp. 437-456.

88 Mazouz, A.K., Souilah, A. and Talbi, M. "Design of an expert system for generating optimal assembly sequence", *Computer-Aided Engineering Journal*, December 1991, pp. 225-259.

89 Murayama, T. and Oba, F. "An efficient method for generating assembly sequences in product design stages", In *Proceedings IEEE International Symposium on Assembly and Task Planning*, 1993, pp. 564-569.

90 Heemskerk, C.J.M. and Van Luttervelt, C.A. "The use of heuristics in assembly sequence planning", *Annals of the CIRP*, 1989, Vol. 38, No. 1, pp. 37-40.

91 Lee, S. and Shin, Y.G. "Assembly planning based on geometric reasoning", *Computation and Graphics*, 1990, Vol. 14, No. 2, pp. 237-250.

92 Chen, C.L.P. and Wichman, C.A. "A systematic approach for design and planning of mechanical assembly", *AI EDAM*, 1993, Vol. 7, No. 1, pp. 19-36.

93 Hong, D.S. and Cho, H.S. "A neural-network-based computational scheme for generating optimized robotic assembly sequences", *Engineering Applications of Artificial Intelligence*, 1995, Vol. 8, No. 2, pp. 129-145.

94 Swaminathan, A. Shaikh, S.A. and Barber, K.S. "Design of an experience-based assembly sequence planner for mechanical assemblies", *Robotica*, 1998, Vol. 16, pp. 265-283.

95 Pu, P. "An assembly sequence generation algorithm using case-based search techniques", In *Proceedings of the 1992 IEEE International Conference on Robotics and Automation*, Nice, France, May 1992, pp. 2425-2430.

96 Chang, P.T., Ben-Arieh, D. and Lee, E.S. "Generation of mechanical assembly sequences with fuzzy weights", In *Proceedings of First Industrial Engineering Research Conference*, 1992, pp. 97-101.

97 Delchambre, A. and Wafflard, A. "A pragmatic approach to computer-aided assembly planning", In *Proceedings 1990 IEEE International Conference on Robotics and Automation*, May 1990, Cincinnati, Ohio, USA.

98 Khosda, P.K. and Mattikali, R. "Determining the assembly sequence from a 3-D model", *Journal of Mechanical Working Technology*, 1989, Vol. 20, pp153-162.

99 Deshmukh, A., Yung, J.P. and Wang, H.P. "Automated generation of assembly sequence based on geometric and functional reasoning", *Journal of Intelligent Manufacturing*, 1993, Vol. 4, pp. 269-284.

100 Gottipolu, R.B. "An approach for assembly sequence planning using solid models", *PhD Thesis*, Department of Industrial Engineering, Ecole Polytechnique, University of Montreal, Montreal, Canada, 1994.

101 Gottipolu, R.B. and Ghosh, K. "An integrated approach to the generation of assembly sequence", *International Journal of Computer Applications in Technology*, 1995, Vol. 8, No. 3, pp. 125-138.

102 Santochi, M. and Dini, G. "Computer-aided planning of assembly operations: the selection of assembly sequences", *Robotics & Computer-Integrated Manufacturing*, 1992, Vol. 9, No. 6, pp. 439-446.

103 Kaufman, S.G., Wilson, R.H., Jones, R.E., Calton, T.L. "The Archimedes 2 mechanical assembly planning system", In *Proceedings 1996 IEEE International Conference On Robotics and Automation*, 1996, pp. 3361-3368.

104 Jones, R.E. "User documentation for Archimedes 3.x", *http://www.sandia.gov/archimedes/userdoc/userdoc.html*, March 1997.

105 Calton, T.L. "Advancing design-for-assembly the next generation in assembly planning", In *Proceedings of the 1999 IEEE International symposium on Assembly and Task Planning*, July 1999, pp. 57-62.

106 Halperin, D. and Wilson, R.H. "Assembly partitioning along simple paths: the case of multiple translations", *IEEE International Conference on Robotics and Automation*, 1995, pp. 1585-1592.

107 Wilson, R.H. and Latombe, J.C. "Geometric reasoning about mechanical assembly", *Artificial Intelligence*, Dec. 1994, Vol. 71, No. 2, pp. 371-396.

108 Wilson, R.H. *On Geometric Assembly Planning*, PhD Dissertation, March 1992.

109 Wilson, R.H. and Rit, J.F. "Maintaining geometric dependencies in assembly planning", *Computer-Aided Mechanical Assembly Planning*, Edited by Homem de Mello, L., Lee, S., Kluwer Academic Publishers, 1991, pp. 217-241.

110 Ames, A.L., Calton, T.L., Jones, R.E., Kaufman, S.G., Laguna, C.A. and Wilson, R.H. "Lessons learned from a second generation assembly planning system", *IEEE*, 1995, pp. 41-47.

111 Jones, R.E., Wilson, R.H. and Calton, T.L. "Constraint-based Interactive Assembly Planning", In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, April 1997, pp. 913-921.

112 Jones, R.E. and Wilson, R.H. "A survey of constraints in automated assembly planning", In *Proceedings 1996 IEEE International Conference On Robotics and Automation*, 1996, pp. 1525-1532.

113 Calton, T.L. and Peters, R.R. "A framework for automating cost estimates in assembly process", In *Proceedings of the 1999 IEEE International Symposium on Assembly and Task Planning*, July 1999, pp. 100-105.

114 Grewal, S. "Integrated part and assembly planning", In *Proceedings of the 1997 IEEE International Symposium on Assembly and Task Planning*, 1997, pp. 36-41.

115 Grewal, S., Tran, P., Bhaskare, A., Submitted by Farmer, L. "Assembly planning software", *Annals of the CIRP*, 1995, Vol. 44, No. 1, pp. 1-6.

116 Editors: Homem de Mello, L., Lee, S. *Computer-Aided Mechanical Assembly Planning*, Kluwer Academic Publishers, 1991.

117 Ye, N. and Urzi, D.A. "Heuristic rules and strategies of assembly planning: experiment and implications in the design of assembly decision support system", *International Journal of Production Research*, 1996, Vol. 34, No. 8, pp. 2211-2228.

118 Almgren, R. and Martensson, N. "Rule-based assembly sequence exploring human planning experience", In *Proceedings 2nd Conference On Flexible Assembly Systems*, Chicago, 1990, Vol. 28, pp. 79-84.

119 Polfreman, D. *An Investigation into the Evaluation of Assembly Sequences*, MSc Thesis, Cranfield University, 1996.

120 Barnes, J.C. PhD Thesis, Cranfield University, U.K. 1999.

121 Bä□ler, R. and Schmaus, T. "Procedure for assembly-oriented product design", *2nd International Conference on Product Design for Manufacture & Assembly*, Rhode Island, 1987.

122 Zha, X.F., Lim, S.Y.E. and Fok, S.C. "Development of expert system for concurrent product design and planning for assembly", *The International Journal of Advanced Manufacturing Technology*, 1999, Vol. 15, pp. 153-162.

123 Zha, X.F., Lim, S.Y.E. and Fok, S.C. "Integrated knowledge-based approach and system for product design for assembly", *International Journal of Computer Integrated Manufacturing*, 1999, Vol. 12, No. 3, pp. 211-237.

124 Hird, G., Swift, K.G., Bässler, R., Seidel, U.A. and Richter, M. "Possibilities for integrated design and assembly planning", In *Proceedings of 9th International Conference on Assembly Automation*, 1988, pp. 155-166.

125 Wang, Y., Hsieh, L.H. and Seliger, G. "Knowledge-based integration of design and assembly process planning", In *Proceedings CIRP Seminars - Manufacturing System*, 1993, Vol. 22, No. 2, pp. 107-113.

126 Rampersad, H.K. "Integrated and assembly oriented product design", *Integrated Manufacturing System*, 1996, Vol. 7, No. 6, pp. 5-15.

127 Barnes, C.J., Dalgleish, G.F., Jared, G.E.M., Swift, K.G., Tate, S.J. "Computer support for proactive DFA," *Second International Symposium on Tools and Methods for Concurrent Engineering*, Manchester, UK, April 1998, pp. 313 - 320.

128 Barnes, C.J., Dalgleigh, G.F., Jared, G.E.M., Mei, H., Swift, K.G. "Assembly oriented design", In *Proceedings of the 1997 IEEE International Symposium on Assembly and Task Planning*, Porto, Portugal, July 1999, pp. 45-50.

129 Sodhi, R. and Turner, J.U. "Towards modeling of assemblies for product design", *Computer-Aided Design*, Feb. 1994, Vol. 26, No. 2, pp. 85-97.

130 Tate, S.J., Jared, G.E.M., Swift, K.G. "Detection of symmetry and primary axes in support of proactive Design for Assembly", In *Proceedings of ACM Symposium on Solid Modeling*, Ann Arbor, Michigan, USA, June 1999, pp. 151-158.

131 Swift K.G. and Booker, J.D. *Process Selection – from Design to Manufacture*, Arnold, a member of the Hodder Headline Group, London, 1997.

132 Mei, H., Robinson, P.A., Dalgleish, G.F., Jared, G.E.M. and Swift, K.G. "Strategies for expert system support in assembly oriented CAD", *Computer Aided Conceptual Design '98 --Proceedings of the 1998 Lancaster International Workshop on Engineering Design*, May 1998, pp. 231-242.

133 Wooldridge, M. "Intelligent agents", In G. Weiss, editor: *Multiagent Systems*, The MIT Press, April 1999, pp. 1-51.

134 Wooldridge, M. and Jennings, N.R. "Intelligent agents: theory and practice", *The Knowledge Engineering Review*, 1995, Vol. 10, No. 2, pp. 115-152.

135 Nwana, H.S. "Software agents: an overview", *The Knowledge Engineering Review*, 1996, Vol. 11, No. 3, pp.205-244.

136 Hintikka, J. *Knowledge and Belief*, Cornell University Press, 1962.

137 Rao, A.S. and Georgeff, M.P. "Modelling rational agents within a BDI-architecture" In: Fikes, R. and Sandewall, E. editor: *Proceedings of Knowledge Representation and Reasoning (KR&R-91)*, Morgan Kaufmann, 1991, pp. 473-484.

138 Chapman, D. "Planning for conjunctive goals", *Artificial Intelligence*, 1987, Vol. 32, pp. 333-378.

139 Agre, P.E. and Chapman, D. "Pengi: An implementation of a theory of activity", In *Proceedings of the 6th National Conference on Artificial Intelligence*, Morgan Kaufmann, 1987, pp. 268-272.

140 Brooks, R.A. "A robust layered control system for a mobile robot", *IEEE Journal of Robotics and Automation*, 1986, Vol. 2, No. 1, pp. 14-23.

141 Steels, L. "Cooperation between distributed agents through self organization", In Demazeau, Y. and Müller, J.P. editor: *Decentralized AI -- Proceedings of the First European Workshop on Modelling Autonomous Agents in Multi-Agent Worlds (MAAMAW-89)*, Elsevier, 1990, pp. 175-196.

142 Ferguson, I.A. *Touring Machines: An Architecture for Dynamic, Rational Mobile Agents*, PhD thesis, Clare Hall, University of Cambridge, UK. (Also available as Technical Report No. 273, University of Cambridge Computer Laboratory.)

143 Müller, J.P. and Pischel, M. "Modelling interacting agents in dynamic environments", In *Proceedings of the Eleventh European Conference on Artificial Intelligence (ECAI-94)*, Amsterdam, the Netherlands, 1994, pp. 709-713.

144 Gasser, L., Braganza, C. and Hermann, N. "MACE: A flexible testbed for distributed AI research", In Huhns, M. editor: *Distributed Artificial Intelligence*, Pitman, 1987, pp. 119-152.

145 Ben-Arieh, D. "A methodology for analysis of assembly operations' difficulty", *International Journal of Production Research*, 1994, Vol. 32, No. 8, pp1879-1895.

146 ROY, U. "Intelligent CAD system in concurrent engineering environment: A knowledge-based approach", *Cybernetics and System: An International Journal*, 1994, Vol. 25, pp. 611-628.

147 Myers, L., Pohl, J. and Chapman, A. "The ICADS expert design advisor: concepts and directions", *AI In Design*, 1991, pp. 897-920.

148 Tsang, C.H.K., Bloor, C. "An object-oriented automobile diagnosis expert system", In *Proceedings of Expert Systems*, 1994, pp. 111-124.

149 Kiritsis, D. "A peview of knowledge-based expert systems for process planning: methods and problems", *International Journal of Advanced Manufacturing Technology*, 1995, Vol. 10, pp. 240-262.

150 Price, G., Dunne, F.P.E., Teoh, K.S. and Walters, D.G. "Knowledge elicitation and development of a knowledge-based expert system for production scheduling", *Proceedings of the Institution of Mechanical Engineers*, Part B: Journal of Engineering Manufacture, 1995, Vol. 209, pp. 99-106.

151 Pham, D.T., Oztemel, E. "An integrated neural network and expert system tools for statistical process control", In *Proceedings of the Institution of Mechanical Engineers*, Part B, 1995, Vol. 209, pp. 91-97.

152 Hayes, C.C. "Plan-based manufacturability analysis and generation of shape-changing redesign suggestions", *Journal of Intelligent Manufacturing*, 1996, Vol. 7, pp. 121-132.

153 Valavanis, K.P., Kokkinaki, A.I. "Knowledge-based expert systems in engineering application: a survey", *Journal of Intelligent and Robotic Systems*, 1994, Vol. 10, pp. 113-145.

154 Feigenbaum, E.A. *Knowledge Engineering in the 1980's*, Dept. of Computer Science, Stanford University, Stanford CA, 1982.

155 Krause, F.L. and Schilingheider, J. "Development and design with knowledge-based software tools -- an overview", *Expert Systems With Applications*, 1995, Vol. 8, No. 2, pp. 233-248.

156 Ignizio, J.P. *Introduction to Expert Systems: the Development and Implementation of Rule-Based Expert Systems*, McGraw-Hill Book Co. Singapore, International edition, 1991.

157 Newell, A. and Simon, H.A. *Human Problem Solving*, Prentice-Hall, 1972.

158 Watson, I. and Marir, F. "Case-based reasoning: a review", *The Knowledge Engineering Review*, 1994, Vol.9, No. 4, pp. 327-354.

159 Althoff, K., Auriol, E., Barletta, R., & Manago, M. *A Review of Industrial Case-based Reasoning Tools*, An AI Perspectives Report, Series Editor: Goodall, A., 1995.

160 Bardasz, T. and Zeid, I. "DEJAVU: Case-based reasoning for mechanical design", *AI EDAM*, 1993, Vol. 7, No. 2, pp.111-124.

161 Sycara, K., Navin Chandra, D., Guttal, R. "CADET: A case-based synthesis tool for engineering design", *International Journal of Expert System*, 1992, Vol. 4, No. 2, pp.157-188.

162 Kim, G.J. "Case-based design for assembly", *Computer-Aided Design*, 1997, Vol. 29, No. 7, pp. 497-506.

163 Dechter, R. "Constraint satisfaction", In Shapiro, S.C. editor: *Encyclopedia of Artificial Intelligence*, John Wiley, New York, 1992, pp. 285-293.

164 Giarratano, J. and Riley, C. *Expert Systems: Principles and Programming*, International Thompson Publishing, 1993.

165 Software Technology Branch, Lyndon B. Johnson Space Center, *CLIPS Advanced Programming Guide*, Version 6.10, Aug. 1998.

166 *CLIPS Homepage*, http://www.ghgcorp.com/clips/CLIPS.html

167 *CLIPS Dynamic Link Library for Windows/95/NT*, http://ourworld.compuserve.com/homepages/marktoml/clipstuf.htm

168 IntelliCorp, Inc., *KAPPA-PC Advanced Topics*, June 1992.

169 *Kappa-PC Information*, http://www.intellicorp.com/kappa-pc/default.htm

170 *Eclipse Documentation*, http://www.haley.com/Eclipse.html

171 *Rete++ Data Sheet* and *Seamless Integration of the Rete Algorithm with C++*, http://www.haley.com/Rete++.html

172 Pahl, G. Beitz, W. *Engineering Design*, Design Council, Springer - Verlag, 1984.

173 Lin, A.C. and Chang, T.C. "A framework for automated mechanical assembly planning", *Journal of Mechanical Working Technology*, 1989, Vol. 20, pp. 237-248.

174 Holbrook, A., Sackett, P. "Design for Assembly through knowledge application", *Assembly Automation*, May 1990, Vol. 10, No 2, pp. 87-92.

175 Wilson, R. and Rit, J. "Maintaining geometric dependencies in assembly planning", *Computer-Aided Mechanical Assembly Planning*, Kluwer Academic Publishers, USA, 1991, pp. 217-239.

176 Golabi, S., Abhary, K. and Luong, L.H.S. "Contact recognition for assembly planning using solid modeling", In *Proceedings of the 12th International Conference on CAD/CAM Robotics and Factories of Future*, Middlesex

University Press, 1996, pp. 1033-1040.

177 Thomas, F. and Torras, C. "A group-theoretic approach to computation of symbolic part relations", *IEEE Journal of Robotics and Automation*, Dec. 1988, Vol. 4, No. 6, pp. 622-634.

178 Hurst, K. *Engineering Design Principles*, John Wiley & Sons Inc., 1999.

179 Atiyeh, P.G. "Design for Assembly: sometimes more is less", *Assembly Automation*, 1992, Vol. 12, No. 2, pp. 26-30.

180 Gui, J.K. and Mantyla, M. "Functional understanding of assembly modelling", *Computer-Aided Design*, June 1994, Vol. 26, No. 6, pp. 435-451.

APPENDIX

PAPERS ARISING FROM THE RESEARCH

Mei, H., Robinson, P.A., "Adding expert support to assembly-oriented CAD tools", *Short Communications in Manufacture and Design Section* in *Journal of Engineering Manufacture*, Part B, January 2000, Vol. 214, pp. 81-88.

Barnes, C.J., Jared, G.E.M., Mei, H., Swift, K.G., "Human/Computer Collaboration in Assembly Sequence Generation", submitted to the Journal *IEEE Transactions on Robotics and Automation.*

Barnes, C.J., Dalgleigh, G.F., Jared, G.E.M., Mei, H., Swift, K.G., "Assembly oriented design", In *Proceedings of the 1997 IEEE International Symposium on Assembly and Task Planning*, Porto, Portugal, July 1999, pp. 45-50.

Mei, H., Robinson, P.A., Dalgleish, G.F., Jared, G.E.M. and Swift, K.G. "Strategies for expert system support in assembly oriented CAD", *Computer Aided Conceptual Design '98 –Proceedings of the 1998 Lancaster International Workshop on Engineering Design*, May 1998, pp. 231-242.