A methodology for aggregate assembly modelling and planning

Michael Betteridge

A METHODOLOGY FOR AGGREGATE ASSEMBLY MODELLING AND PLANNING

A thesis submitted to the
University of Durham
for the degree of
Doctor of Philosophy

by

Michael Betteridge

School of Engineering
University of Durham

June 2000

Abstract

The introduction of Concurrent Engineering highlights the need for a link between the early stages of product design and assembly planning. This thesis presents aggregate assembly process planning as a novel methodology to provide this link. The theory behind the research is to bring all aspects of product development together to consider assembly planning at the conceptual stage of design. Decisions taken during the early design stage not only have the greatest influence on production times and costs, but also should ensure that a design is easy to manufacture and assemble.

An automated computer-based system has been developed to implement the methodology. The system generates aggregate assembly process plans which give details of feasible sequences, assembly process times and costs, resource requirements, and factory loadings. The Aggregate Assembly Modelling and Planning (AAMP) system employs object-oriented modelling techniques to represent designs, process planning knowledge, and assembly resources. The minimum information requirements have been identified, and a product model encompassing this data has been developed. An innovative factor of this thesis is to employ Assembly Feature Connections (AFCs) within the product model to represent assembly connectivity. Detailed generic assembly process models, functioning with limited design data, are used to calculate assembly criteria. The introduction of a detailed resource model to represent assembly facilities enables the system to calculate accurate assembly times, dependent on which resources are used within a factory, or even which factory is employed. A new algorithm uses the structure of the product model, process constraints and assembly rules to efficiently generate accurate assembly sequences. Another new algorithm loads the assembly operations onto workstations, ensuring that the capability and capacity are available.

The aggregate assembly process planning functionality has been tested using products from industry, and has yielded accurate results that prove to be both technically feasible and realistic. Industrial response has been extremely favourable. Specific comments on the usefulness and simplicity of such a comprehensive system gives encouragement to the concept that aggregate assembly process planning provides the required link between the early stages of product design and assembly planning.

Acknowledgements

Firstly, I would like to acknowledge the contribution of my supervisor Professor Paul Maropoulos, for all his guidance and encouragement over the period of this research. I would also like to thank Dr. Hugh Bradley for his valuable discussions and programming support. Also, I would like to thank all the members of the Design and Manufacturing Research Group, and all those people in the department and the rest of the University that have made my time in Durham so pleasurable, especially the Graduate Society AFC.

I am grateful to the Durham University Society for providing funding which has enabled me to undertake this project. I am also grateful to my friends in industry for providing important industrial knowledge and technical support.

I would like to thank my family for their motivation and interest, especially my parents. Finally, I offer my thanks to Rachel, who has given me all her support, encouragement and love throughout this endeavour.

Table of Contents

Chapter 1: Introduction

Chapter 2: Literature Review

Chapter 3: System Overview

Chapter 4: Product Model and Assembly Connections

Chapter 5: Process Models and Resource Models

Chapter 6: Aggregate Assembly Process Planning

Chapter 7: Testing and Results

Chapter 8: Discussion and Conclusion

List of Figures

List of Tables

Glossary

3D Maps - Three-Dimensional Mechanical Assembly Planning System

AAMP - Aggregate Assembly Modelling and Planning

ADAM - Assisted Design for Manufacturing and Assembly

AEM - Assembly Evaluation Method

AFC - Assembly Feature Connection

AGV - Automated Guided Vehicle

AMD - Aggregate Management and Detailed

B-Rep - Boundary Representation

BS - British Standards

CAAPP - Computer-Aided Assembly Process Planning

CAD - Computer-Aided Design

CAE - Computer-Aided Engineering

CAM - Computer-Aided Manufacturing

CAPP - Computer-Aided Process Planning

CIM - Computer-Integrated Manufacturing

CNC - Computer Numerical Control

CSG - Constructive Solid Geometry

DACON - Design for Assembly CONsultation

DFA - Design for Assembly

DFMA - Design for Manufacture and Assembly

DFX - Design For X

FADES - FAcility Design Expert System

FLAP - FLexible Assembly Planning

GAPP - Generative Assembly Process Planner

GUI - Graphical User Interface

HTML - HyperText Mark-up Language

ISO - International Standards Organisation

MOSES - Model Oriented Simultaneous Engineering System

MOST - Maynard Operations Sequence Technique

MTM - Methods Time Measurement

OOA - Object-Oriented Analysis

OOP - Object-Oriented Programming

PACT - Palo Alto Collaboration Testbed

PMTS - Predetermined Motion Time System

STEP - STandard for the Exchange of Product data

TMU - Time-Measurement Unit

WF - Work Factor

Declaration

This thesis is the result of my own work. No part of the thesis has been submitted for any other degree in this or any other University.

Michael Betteridge
Durham University
June 2000

Chapter One

Introduction

1.1 Background

In today's intensively competitive global market place, manufacturing companies are facing a wide range of issues, such as how to develop a product in less time, at lower cost, with higher quality, and 'right first time'. The benefits of bringing products to market quicker than competitors include extra sales revenue, earlier breakeven, premium pricing giving bonus profits from being first to market, and extended sales life. Other advantages include developing customer loyalty, increasing market share, improving innovation image, and an increased product range.

Companies are finding that traditional product development practices and tools can no longer keep pace with the reducing product life cycles and changing global market. Increasingly, companies are turning to strategic initiatives such as Concurrent Engineering and Design for Manufacture and Assembly (DFMA) to improve current practices of product development. The philosophy of Concurrent Engineering, also known as Simultaneous Engineering, is to consider all aspects of the product in parallel during the early stages of product development, in order to avoid costly and time-consuming activities downstream associated with traditional design and manufacturing processes (Dong *et al*, 1996).

Concurrent Engineering has been recognised as a concept since the late 1980s. However, successful companies have been using these ideas for many years before that.

Concurrent Engineering has its historical roots in the management approaches of Japanese manufacturers (examples include Xerox and Digital), many of whom have been using Concurrent Engineering principles, without setting out a specific terminology, since the 1970s. Hartley (1990) identifies several Japanese systems which foreshadow the philosophy of Concurrent Engineering, in which the unifying factor is the requirement for a consensus of agreement on decisions from all members of the organisation.

This approach leads to a full commitment to the project, and allows potential problems arising from a course of action to be identified from the beginning. Three important principles can be identified from the Concurrent Engineering philosophy. These are performing activities concurrently, not sequentially, to reduce the overall development time; involving representatives from all disciplines in every decision, since it is not always clear in advance where the influences will be observed; and finally, concentrating more effort and attention on early design, since it costs less to change the design at this point and the effects can be greater. These concepts attempt to address the issue of product development productivity by helping the designer to make early decisions that minimise costs over the life of the product.

A critical aspect of implementing these concepts is the integration of design and manufacturing issues in the early conceptual stages of design, to ensure that a design is easy to manufacture and assemble. Studies estimate that up to eighty per cent of a product's cost is already fixed by the end of the design stage, even though less than ten per cent of the total development costs have been expended. Of the total manufacturing cost, forty per cent is often accounted to assembly (Pawar *et al*, 1994).

Although leading manufacturing companies have begun to be aware of the immense benefits that can be derived from Concurrent Engineering, they are often forced to adopt a pragmatic approach in order to solve more immediate problems. The use of advanced information technology seems to play a minor role in the introduction of Concurrent Engineering. Organisational issues take priority, followed by the use of formal methodologies such as DFMA. Although this is the current situation, as companies become more experienced in Concurrent Engineering, they will start to look

for more sophisticated information technology tools to make the design process more effective.

At present, however, most computer tools for aiding design are directed at the creation of geometry, or pictures of geometry of single parts, together with some analytical capability for functional performance, such as finite element analysis. Little has been done to link geometric models to cost analysis, process planning for fabrication or assembly, or other aspects of product design and manufacture. Furthermore, there are few or no tools for these functions.

There is now a need for new product development tools that can assist developers performing traditionally downstream product development activities such as process planning. Existing tools are used too late in the development process, when they can have little impact. The new tools must work early in the development process, where the greatest benefits may be achieved. However, to accomplish this, the tools must be able to operate with the reduced amounts of information available in early design. The lack of suitable support tools is likely to prove a serious obstacle to the implementation of Concurrent Engineering practice in most companies. Therefore, this work has been undertaken to develop a scheme of support for Concurrent Engineering through the provision of appropriate information technology tools.

1.2 Aggregate Process Planning

Maropoulos (1995a) proposes a novel methodology for process planning to suit concurrent product and process development. This approach is based on the fragmentation of the process planning function into three levels, according to the detail of the task. This will result in the Aggregate, Management and Detailed (AMD) process planning architecture, as shown in Figure 1-1.

It is suggested that the process planning function will evolve into a three-tiered structure, with detailed process plans being delayed until near the time of production, whilst aggregate plans will be made as early as possible to facilitate strategic decision-making. The management process planning function will control the project planning of manufacture, ensure that the design and capacity constraints are satisfied, and manage

the manufacturing resources in the production facility. Aggregate Process Planning is the generation of manufacturing instructions for a given product based on a partially specified design. Aggregate assembly process planning is specific to assembly processes during manufacture. The aggregate production plan specifies a list of alternative manufacturing options for a product, which include the processes to be used, the resources required, and the factory routings. Full specification of process parameters is left to the detailed process planning stage, which will be carried out when the detailed design is finalised. Aggregate process and production plans provide quantitative feedback on the manufacture and assembly of a design, and a comparison between alternative production and processing options. Early identification of processing options allows the designs to be optimised for that process.

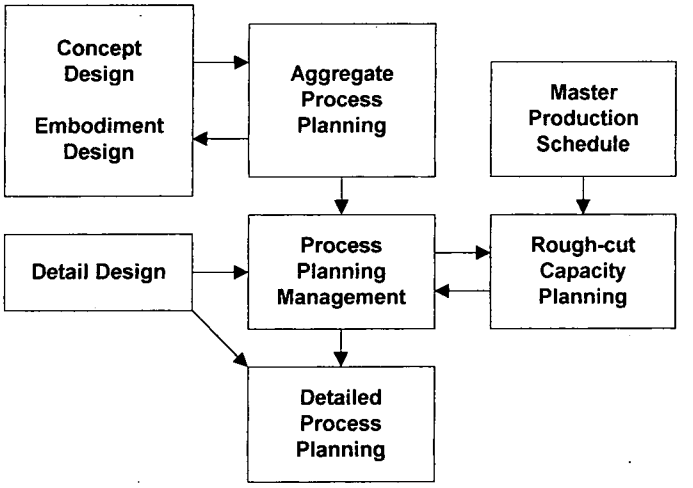


Figure 1-1: AMD Architecture for Process Planning

Aggregate process plans consist of a hierarchical set of instructions that can be mapped against a structured model of the product design. A key feature of aggregate process planning is that it identifies production alternatives and encourages the designer to explore the use of processes which might not otherwise be considered. In addition, the designer is able to receive an early breakdown of the relative costs of the product features and therefore identify the areas where additional work might result in the greatest cost savings.

1.3 Research Objectives

The aim of this thesis is to discuss the development of methods and a computer-based tool for concurrent design and manufacturing, focussing on assembly modelling and planning during the early stages of design. The objectives of this research are as follows:

- To assess the impact of Concurrent Engineering on Computer-Aided Engineering (CAE) tools for product development, in order to identify the requirements for new technology for assembly modelling and planning.

- To propose a new methodology for the computer support of product development which is tailored to the requirements of a Concurrent Engineering environment. In particular, to provide support for the assessment of assembly processes in the early stages of product design.

- To develop and implement a prototype computer-based system which provides the functionality identified. It is expected that a Concurrent Engineering tool for assembly planning should provide integration between design and production knowledge. In particular, the ability to identify and evaluate alternative options that will give a designer rapid feedback concerning manufacturability, cost or other important criteria, is critical to early product development.

- To evaluate this prototype system by thorough testing with industrial designs and data in order to allow comparisons with existing engineering methods. The system will be judged on criteria including accuracy of results and outputs, the ability to cover a wide range of design configurations and processing options, impact on design time, and ease of use.

The prototype computer system is to serve as a test-bed for the ideas which are proposed in this thesis; it is not intended for commercial use.

1.4 Outline of Thesis

Chapter two presents a review of research in the fields of product development, assembly and product modelling, and process planning. A general overview of the computer system which has been developed is given in chapter three, showing how the system works as a whole. The next two chapters deal with the individual models which have been developed for the system. In chapter four, the aggregate product model which forms the input to the system is detailed. Chapter five details the assembly process models which are used with the aggregate assembly planning assessment, and describes the time and cost calculation methods which are used. Chapter five also discusses the way in which resource information, such as factories, tools, machines, labour and transportation equipment, are modelled. Chapter six details the implementation of aggregate assembly process planning within the system. The operation sequencing, resource selection, and factory loading and balancing are described. Chapter seven presents the results of the testing of the system with example data, including examples from industry. A summary of the work, including suggestions for future work and conclusions, is presented in chapter eight.

Chapter Two

Literature Review

2.1 Introduction

Having established that the successful integration of a Concurrent Engineering methodology into the product development process will result in a great improvement in a manufacturing company's productivity and performance, the task is then to determine the requirements to achieve this purpose. This chapter consists of a review of published literature on the state of the product development process, focussing on assembly product modelling and process planning. In particular, this survey includes the methodologies and support systems which have been proposed to support Concurrent Engineering. Further, the requirements for improvements and alterations in the technology are identified and research efforts in this area are reviewed. It is suggested that modelling product designs and all aspects of the manufacturing process, together with the integration and management of data, are of particular importance in the pursuit of Concurrent Engineering manufacturing systems.

2.2 Product Development

The product development process, also referred to as total design, involves identifying the market or user needs, developing these, and ultimately, introducing new products into the market. It is the principle business of a large proportion of manufacturing companies, and in order to understand the impact of Concurrent Engineering on product development, it is necessary to review the task of product development and identify

each of its elements. There have been many attempts to draw up maps or models of the design process. Some of these models simply describe the sequence of activities that typically occur during design, whilst others attempt to prescribe a better or more appropriate pattern of activities.

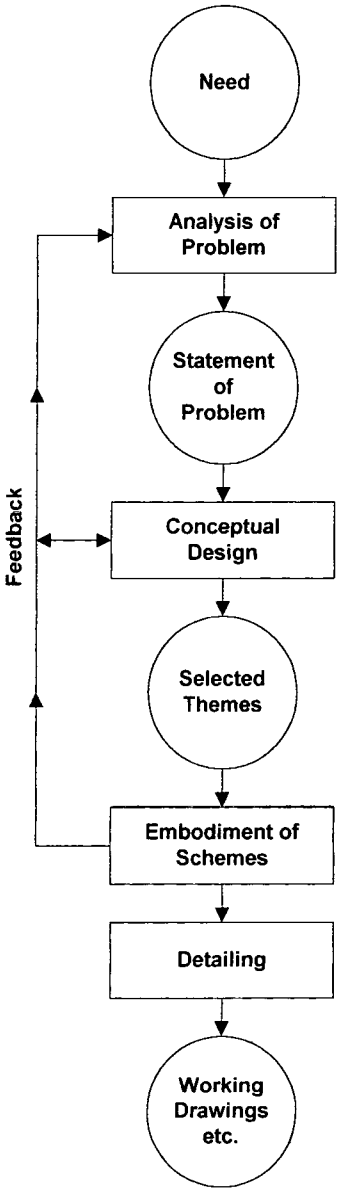


Figure 2-1: French's Model of the Design Process

Descriptive models of the design process usually emphasise the importance of generating a solution concept early in the process, thus reflecting the 'solution-focussed' nature of design thinking. This initial solution conjecture is then subjected to analysis, evaluation, refinement and development. Sometimes, of course, the analysis

and evaluation show up fundamental flaws in the initial solution and it has to be abandoned, a new concept generated, and the cycle repeated. This process is heuristic, using previous experience, general guidelines, and 'rules of thumb' that lead in what the designer hopes is the right direction. A simple four-stage model of the design process (Cross, 1994) includes: Exploration of the ill-defined problem space; generation of a concept; evaluation against the goals, constraints and criteria; and communication of a design, ready for manufacture. A feedback loop exists between the evaluation and generation stage for re-analysis. French (1985) has developed a more detailed model of the design process, as shown in Figure 2-1, based on the following activities: Analysis of the problem; conceptual design; embodiment design; and detailed design. In the diagram, the circles represent stages reached, or outputs, and the rectangles represent activities or work in progress.

As well as models that simply describe a more-or-less conventional, heuristic process of design, there have been several attempts at building prescriptive models of the design process. These latter models are concerned with trying to persuade or encourage designers to adopt improved ways of working. A comprehensive model that still retains some clarity is offered by Pahl and Beitz (1988). It is based on the following design stages:

- *Clarification of the design*: Collect information regarding the requirements to be embodied in the solution and also in the constraints.

- *Conception of the design*: Establish function structures; search for suitable solution principles; combine into concept variants.

- *Embodiment of the design*: From the concept, the design determines the layout and form, and develops a technical product or system in accordance with technical and economic considerations.

- *Detailing the design*: The arrangement, form, dimensions and surface properties of all the individual parts are finally laid down; materials specified; technical and economic feasibility re-checked; all drawings and other production documents produced.

At every step, a decision has to be made as to whether the next step can be taken, or whether previous steps have to first be repeated. Pahl and Beitz note that "continuing right to the end to discover that a serious mistake has been made at an earlier stage is something that must be avoided at all costs".

The BS7000 model, Standard for Engineering Management, commences with a feasibility study stage, and proceeds through conceptual design, embodiment design, detailed design, and design for manufacture stages. Outputs at each design stage include the form of design brief, conceptual drawings, layout drawings, detailed product definition and manufacturing instructions respectively. It can be observed that this model is derived from other models by Pahl and Beitz, and French. An ideal prescriptive model aims to consider the later stages of the life-cycle (e.g. production and disposal) before being committed to detailed design, whereas, the traditional approach leads to backtracking along the development path when unfeasible suggestions are only recognised after further stages. With such a Concurrent Engineering approach, designs should be 'right first time'.

In a review of management approaches, Andreason and Gudnason (1992) stress the importance of a planned product development and the integration of various activities. Kunz *et al* (1996) discuss the way in which Concurrent Engineering has been implemented in companies. They note that Concurrent Engineering implementation often only links product and process, instead of including the design of the manufacturing facility and organisation. A schema for the integration of organisational and facility design with product and process development is also presented. Sohlenius (1992) presents an overview of the early impact of Concurrent Engineering, stressing the need for education and for good team-working skills for its success, and also stating that Concurrent Engineering tools must ideally be able to "incorporate multiple perspectives, support multiple stages and work with multiple participants".

2.2.1 Concurrent Engineering

Concurrent Engineering is defined by Winner *et al* (1988) as a "systematic approach to the integrated, concurrent design of products and their related processes, including manufacture and support". This approach is intended to cause the developers, from the

outset, to consider all elements of the product life cycle from conception through to disposal, including quality, cost, schedule, and user requirements. Cleetus (1992) proposes another definition: "Concurrent Engineering is a systematic approach to integrated product development, that emphasises response to customer expectations and embodies team values of co-operation, trust and sharing in such a manner that decision-making proceeds with large intervals of parallel working by all life cycle perspectives, synchronised by comparatively brief exchanges to product consensus". Different aspects of Concurrent Engineering are addressed in both definitions including:

- New organisational structures, teamwork, leadership and customer understanding.
- Improvement, integration and concurrent product life cycle process activities.

A more pragmatic definition by Smith (1997) states that "Concurrent Engineering can be seen as a summary of best practice in product development, rather than an adoption of a radically new set of ideas". These best practice principles, according to Smith, are: Manufacturing and functional design constraints need to be considered simultaneously; combining people with different functional backgrounds into the design team is a useful way to combine the different knowledge bases; engineering designers must bear in mind customer preferences during the design process; and time to market is an important determinant of eventual success in the market.

Academic research institutions are commonly divided in their thinking on Concurrent Engineering, varying between providing organisational and technological support. Research into the former has concentrated on developing methodologies for the introduction of both new organisational structures and team working [(Evans, 1990) (Gillen and Fitzgerald, 1991)]. The technology-based initiatives have focussed largely on developing frameworks that allow the capture and sharing of cross-functional information, and on developing software applications to support the implementation of specific process improvements.

2.2.1.1 Computer-Aided Support for Concurrent Engineering

Molina *et al* (1995) believe that to achieve an integrated environment for the support of Concurrent Engineering, it is necessary to define and develop information models,

integrate and implement decision support applications, and provide an adequate information system architecture. The development of these key technological requirements should be based on frameworks which enable a computer system to be defined, configured and implemented according to the requirements dictated by the enterprise integration strategy. The integrated system should seek to provide a Concurrent Engineering system as a unifying module in a CAE environment. Typically, an integrated Concurrent Engineering system should link Computer-Aided Design (CAD) and Computer-Aided Process Planning (CAPP) to perform manufacturing assessment.

Several research groups at Stanford University jointly developed the Palo Alto Collaboration Testbed (PACT) (Cutkosky *et al,* 1993). This is a Concurrent Engineering infrastructure that encompasses multiple sites, sub-systems and disciplines. The PACT integrates existing multi-tool systems including: A distributed knowledge-based environment; a model formulation and simulation environment; a mechanical design and process planning system; and a digital electronics design, simulation, assembly and testing system. PACT demonstrated good preliminary results and the next version plans to use commercial sub-systems within its infrastructure.

Meerkamm (1993) describes the design system mfk, a prototype 'engineering workbench' which combines functional and geometric design (synthesis) with a multi-functional analysis system. The synthesis module of the system has four elements: Geometry; technology; function; and organisation. This is more than just a CAD modeller, since it allows the user to specify the product structure in conceptual terms, and to model the functions of components in terms of forces and so forth. The analysis module performs a checking function on the design, and incorporates a knowledge base for production, with links to external analysis engines such as finite element analysis tools for specific checks. This system comes close to the required goal of an integrated product development tool, although it does not have universal coverage of product development activities.

The sharing of common, consistent product and manufacturing data between a range of software applications and design teams is considered a key element to the effective support of Concurrent Engineering. Two data models incorporated into the Model-

Oriented Simultaneous Engineering System (MOSES) research concept, developed in collaboration by Leeds and Loughborough Universities, are the product model and the manufacturing model. The product model contains all data related to a product's life cycle, while the manufacturing model captures all data related to process capabilities.

Abdalla and Knight (1994) developed a knowledge-based system for automatically assessing component designs for manufacture. In this system, a rule-based feature recognition system interfaces with a solid modeller to develop a feature-based representation. The features on this model are then assessed individually using a process knowledge-base. The system can identify feasible processes and estimate process capabilities based on the feature tolerances, and relative cost values for each process are also produced. However, this system suffers from a number of drawbacks. In particular, because features are considered one at a time, a high number of processes will be suggested. Also, the system assumes a single process is used for each stage, and does not calculate actual costs to assess which process to use.

CAPP determines how a design is to be produced in a manufacturing system. CAPP is the important link between CAD and Computer-Aided Manufacturing (CAM), and to a great extent determines the success of Computer-Integrated Manufacturing (CIM). It is for this reason that CAPP is often referred to as a critical step in achieving CIM. Although CAD/CAM techniques have undergone a relatively long period of development (it was during the 1960s that CAPP began to evolve), its significance was not realised until the 1980s. In recent years a number of CAPP systems have been developed, although only a few appear in the manufacturing sector. So called 'integrated CAD/CAM systems' have become available during the last decade, but these rarely achieve the interface between CAD and CAM.

Many researchers and practitioners around the world have been focussing their efforts on developing new CAPP systems, as well as on the research of CAPP techniques. It is believed that with world-wide effort and co-operation, the development of CAPP will meet the needs of CIM implementation and the ever-increasing challenge in manufacturing industry. CAPP is not just computer work, but depends on the development and ingenious application of various logics, artificial intelligence and expert systems, computer graphics, database structure and management, computer

language and programming, and so forth. However, it is the principles and methodologies of process planning that provide the basis for developing efficient CAPP systems.

2.3 Assembly

Assembly is the aggregation of all processes by which various parts and sub-assemblies are "built together to form a complete, geometrically designed assembly or product" (Nof *et al*, 1997), such as an engine or an electronic circuit, either by individual, batch or a continuous assembly process. Another definition for assembly is the act of "putting together all the individual parts and sub-assemblies of a given product" (Delchambre, 1992). Assembly includes both reversible fastening processes such as screwing and bolting, and irreversible ones including riveting, soldering, glueing and so forth.

Assembly is a major part of the production system, and research has shown that it accounts for a large proportion of a product's production time, costs and labour. According to various studies, the assembly of manufactured goods accounts for over fifty per cent of total production time (Nevins and Whitney, 1978), for up to forty per cent of total unit production cost in Europe's consumer goods industry (Bullinger and Richter, 1991), and typically, one third of manufacturing companies are involved with assembly tasks (Martin-Vega *et al*, 1995). In the automotive industry, fifty per cent of the direct labour costs are in the area of assembly, and in precision instruments this value is between twenty and seventy per cent (Warnecke *et al*, 1992). These values point to the potential savings that can be generated by efforts to understand and improve assembly technology, systems and methodologies.

2.3.1 History of Assembly

The history of assembly can be divided roughly into three main periods: Pre-industrial; continuous; and flexible assembly. Before the industrial revolution, products were assembled manually. The two essential features influencing industrial assembly evolution were the interchangeability of parts, and the introduction of conveyors. The use of interchangeable parts at the turn of the eighteenth century enabled the manual assembly of products such as rifles and clocks. During the nineteenth century, the

introduction of conveyors and chutes to provide continuous parts and materials mobility was gradually adopted. In the twentieth century, the Ford Motor Company demonstrated the effectiveness of continuous assembly lines for mass assembly by combining the conveyor and the use of interchangeable parts. In parallel, automatic assembly machines, for example rotary and multi-spindle assembly machines, became common in industry, and provided more precise and faster assembly cycles. A new phase in seeking to understand assembly began in the late 1960s with the advent of robots and the possibility of robot assembly. However, robots have so little dexterity, sensing or brainpower, that assembly must be planned down to the last detail in order that robotisation is successful. Early attempts to achieve this revealed how little about assembly was understood. Progress was made in the 1970s on robot programming, machine vision, physics of parts and so forth. Also during this decade, market pressures for flexibility in design and production introduced the current era of flexible assembly. The two essential features were the availability of computers, and the use of robots for assembly. The late 1980s saw a great increase in the capabilities of computers as well as the software to support product design. From this period to date, a number of new techniques and methods have come together in the form of product DFMA, feature-based design for assembly modelling, assembly sequence analysis and assembly process planning.

2.3.2 Design for Assembly

Although there are many ways to increase manufacturing productivity (plant layout, automation, tools, processes and so forth), consideration of manufacturing and assembly during product design, according to Boothroyd (1992), holds the "greatest potential for significant reduction in production costs and increased productivity". Improving the design itself is not worth considering at a late stage, as usually too much time and money has been expended in justifying the design to consider major changes or even a completely new design. If a product is poorly designed for manufacturing and assembly, techniques can only be applied to reduce to a minimum the impact of the poor design (Boothroyd and Alting, 1992). Only when manufacturing and assembly techniques are incorporated early in the design process will productivity be significantly affected. The design of products, tools, and processes for ease of assembly is needed if

a reduction in assembly cost and an increase in the effectiveness of assembly operations are to be realised.

Redford and Chal (1994) stress the importance of Design for Assembly (DFA), stating that it should be "considered at all stages of the design process from the conceptual stage, where alternative solutions are considered, through to the detailed stage, where dimensions and tolerances are formulated". DFA analysis by Nof (1997) of product design alternatives includes: "Minimise the number of components and sub-assemblies; minimise the time and cost and maximise the reliability of assembly tasks; maximise assembly stability; maximise process yields; eliminate 'hidden' assembly faults and defects; and standardise by common components, processes and methods".

Although products have been designed for assembly as far back as the sixteenth century, it is only since the 1970s that these methods have been scientifically studied and systems developed to aid the designer. There are three general types of approach to DFA: Rule-based methods; procedural methods; and artificial intelligence-based approaches.

2.3.2.1 DFA Guidelines

Rule-based approaches follow a list of guidelines developed and established as 'best practice'. DFA guidelines have evolved for manual, automated and flexible robotic operations. They comprise a multi-disciplinary combination of experimental, analytical and theory-based recommendations, serving as checklists and advice. Recent guidelines have been developed by Boothroyd *et al* (1994) and Edan and Nof (1995), and include advice on product, component, operation, operator, tool, cell considerations and so forth.

2.3.2.2 Procedural Systems for Assemblability Evaluation

Procedural assemblability evaluation is applied by designers for quantitatively estimating the degree of difficulty and associated cost of assembly. While DFA guidelines are general, quantitative ranking enables designers to compare and analyse trade-offs. Hitachi (Miyakawa and Ohashi, 1986) developed an Assembly Evaluation Method (AEM) in 1975 as an effective procedural tool to improve design quality for

better assemblability. The AEM uses two indices at the earliest possible stage of design, namely, the Assembly Evaluation Score *E*, which is used to assess design quality or the difficulty of assembly, and the Assembly Cost Ratio *K*, used to project assembly costs relative to current assembly costs. In the AEM, approximately twenty symbols are used to represent assembly operations. Each symbol has an index which can be used to assess the assemblability of the part under consideration. Corporations such as Sony, Toshiba and NEC have followed Hitachi in developing their own methods.

Another developed system is the Lucas DFA method which arose out of collaboration work between the Lucas organisation and the University of Hull. This method is based around an 'assembly sequence flowchart'. The research group has developed a knowledge-based evaluation technique, the Lucas DFA Evaluation Method, that systematically follows a procedure in which the important aspects of assemblability and component manufacture are considered and rated. As product design commences, it is important to decide whether the product is unique, or whether there are similarities, and therefore opportunities, for standardisation of components and/or assembly procedures, and the establishment of a product family theme. The system is meant to be implemented into a CAD system and because of this, it should be possible to obtain most of the information required for the analysis with the minimum of time and effort. This is a major advantage over most systems that effectively operate in stand-alone mode.

Boothroyd and Dewhurst (1987) have probably developed the most famous DFMA tools and systems. Their DFA method addresses the problems of determining the appropriate assembly method, reducing the number of parts that must be assembled, and ensuring that the remaining parts are easy to assemble. The first step in their procedures is to select the appropriate assembly method for the product. Once this has been established, then an analysis of the design, identifying the assembly difficulties and estimating assembly times, is made for the chosen assembly method. Although Boothroyd and Dewhurst's handling and insertion assembly times are very thorough and useful, some of their assembly process times were found to be over-simplified. The most powerful tool of this or any DFA system, is the reduction in the number of parts required for the product to be functionally acceptable. Boothroyd and Dewhurst's

market leader DFA analysis can be carried out using either their handbook or, more recently, a software package for the personal computer.

Huang and Mak (1999) undertook an experiment to exploit the Internet providing DFMA techniques on the World Wide Web. A well-known DFA technique was converted into a Web-based version functionally equivalent to its version on a standalone workstation. A number of important insights were gained from the experiment: The Web-based client and server architecture are found to be attractive for collaborative DFMA; generic frameworks can be developed for applying different DFX techniques in an integrated way; and integration with other decision-support systems, such as CAD and CAPP, can be readily exploited.

2.3.2.3 Artificial Intelligence Approaches to DFA

Recent computer-assisted approaches to DFA include artificial intelligence techniques which add reasoning and decision-support capabilities. Such systems advise the users on how to improve their work to enable better quality designs with less errors, resulting in lower assembly cost. Knowledge-based systems provide new information-processing capabilities such as rule-based, inference, knowledge-based management, search mechanisms and so forth, combined with conventional computer capabilities. Several systems have been developed with the relatively mature technology of rule-based knowledge systems. Design for Assembly Consultation (DACON) (Swift, 1987) provides a CAD interface for drawing assembly components after they are designed with expert analysis. Hernani and Scarr (1987) developed an expert system interfaced with CAD to recommend assembly design rules. Facility Design Expert System (FADES) (Fisher and Nof, 1989), provides economic analysis and selection of assembly technology. Assisted Design for Assembly and Manufacture (ADAM) (Sackett and Holbrook, 1988), generates advice on reducing the number of components and rationalising the assembly. Numerous rule-based systems for electronic circuit design for assembly have also been developed. Zha *et al* (1999) propose a knowledge-based approach and an expert system for integrated product design for assembly modelling and process planning, and assemblability analysis and evaluation. Key issues in the development of the Design for Assembly Expert System (DFAES) include the system structure, knowledge representation and acquisition, problem solving tools, and

knowledge management system. The intelligent system is aimed to provide the user with suggestions in improving a design, and also offer design ideas.

In a constraint network approach for DFA (Oh *et al*, 1995), design knowledge is represented not as a collection of rules, but as a collection of inter-connected assembly constraint objects. An efficient search can be performed over these networks to evaluate the propagation of these design changes. Other approaches in the field of artificial intelligence to DFA include search techniques for assembly planning and feature-based assembly design. These are both discussed later in this chapter.

De Fazio *et al* (1993) developed a prototype software system that implements a form of feature-based DFA. It is not an automated design system, but instead, a decision and design aid for designers interested in concurrent design. DFA modules in the system include analysing part shapes using DFA rules, part count, assembly process planning, assembly sequence generation, and assembly process costs. It is programmed to act like a manufacturing expert looking over the designer's shoulder, providing suggestions, comments and information about fabrication and assembly. Li and Hwang (1992) also developed a DFA evaluation procedure to achieve a concurrent design environment. Inputs to this system include product engineering drawings, exploded 3D views and assembly sequences, while the outputs include assembly codes, costs and times, and valuable information for re-design suggestions.

2.3.3 Assembly Sequence Generation and Planning

The assembly planning activities in manufacturing companies are still very much pencil-and-paper-based, and are driven by the experience of product planners. However, the task is becoming increasingly difficult due to the changing nature of the manufacturing environment, where the push is towards producing a variant of models every day. The amount of information to be processed is therefore increasing rapidly, and the time required to process it is becoming shorter.

An assembly plan describes how to assemble the product, i.e. specifies a sequence of assembly operations that has to be carried out in order to make the final product from constituent parts and resources. The ability to generate an optimal plan can result in

significant savings, both in terms of time and money. Because of this, many researchers have looked into the problem of the automatic generation of feasible assembly sequences.

There has been a considerable growth of interest in recent years in developing Computer-Aided Assembly Process Planning (CAAPP) for mechanical and electromechanical products. This is not only because CAAPP provides a means of systematically discovering an optimal assembly sequence which may be overlooked by a human designer due to the inherent complexity involved in planning, but also because CAAPP provides the capability of analysing products in terms of ease of assemblability and maintainability, tolerancing, fixturing and overall assembly cost. This can also be linked back to the design level for the modification of product design as well as assembly floor layouts, and to workcell level for programming instruction. CAAPP can thus play an important role for CIM and Concurrent Engineering.

CAAPP is mainly concerned with automatic and interactive generation of feasible, yet cost-effective, assembly sequences. This requires identifying the precedence relationships in part mating, based on reasoning of geometric and physical inferences between parts and sub-assemblies, which affect assembly orders. It also requires selecting preferred assembly sequences out of a large number of feasible assembly sequences, based on analysing assembly costs associated with handling and mating of parts and sub-assemblies.

There are many possible assembly plans for each product. However, assembly operations cannot be implemented in a random order because some operations may prevent the execution of others owing to geometric or precedence constraints. A precedence graph is a type of directed graph which illustrates the precedence constraints between parts. The precedence graph is not a product structure graph, but a constraint graph which describes the order of the operation sequence. It can also be used as an implicit representation of a possible assembly sequence of the product. Much effort has been expended to make assembly planning more autonomous, more efficient, and closer to reality. Early work on interactive assembly planning was concerned with formulating a necessary and sufficient set of questions to be answered by a human designer that

leads to the complete set of precedence relationships, with the minimum number of question-and-answer operations.

Bourjault (1984) presents an approach to generate all possible and valid assembly sequences for a set of parts that form an assembly. The algorithm is based on the rules which come from users' answers to a series of questions about the mating of pairs and multiples of parts. Each user query focusses on a connection or liaison between a pair of parts in the assembly. The answers are expressed as precedence relationships between logical combinations of liaisons, from which assembly plans can be generated in a straightforward manner.

De Fazio and Whitney (1987) found that the question-and-answer approach proposed by Bourjault can lead to serious problems when the number of connections increase. The number of queries directly relates to the number of connections. Bourjault's method requires $2l^2$ questions, plus the possibility of a large number of subsequent questions (here l is the number of connections between parts). For products with more than ten relationships, the questions number several hundred. De Fazio and Whitney propose a technique where, instead of numerous simple yes-no answers, the users' answers directly evoke relationships. For each liaison, the system asks which other liaisons must be established prior and which must be established after. Valid linear and partial order connection sequences can be obtained algorithmically directly from these relationships. As with Bourjault's network, the nodes are the parts and the liaisons are the relationships between the parts. The algorithm can significantly improve the tedious process of dealing with questions and answers and reduces the number of queries to $2l$.

Wilson (1995) takes a dual approach to minimising user queries. Firstly, most assembly operations are validated automatically from the CAD models of the assembly's parts using simple, fast techniques. Secondly, with each query the user is allowed to identify a set of parts that constrain a sub-assembly. The system uses this information to answer future queries automatically. This powerful approach dramatically reduces the number of queries without sacrificing accuracy, and can be used for real products, unlike the above techniques.

Baldwin *et al* (1991) built an integrated set of user-interactive computer programs that generates all feasible assembly sequences for a product, and then aids the user in judging their value based on various criteria. The programs use a disassembly analysis for generating sequences, and provide on-line visual aids during generation and evaluation. During evaluation, matters such as avoiding difficult assembly states or moves, stability, fixturing, orientation, re-fixturing and re-orientation are important, and inclusion of states are considered to highlight desirable or undesirable sequences. The designer edits the set of sequences according to these criteria, leading to an informed sequence choice or the need for design refinement.

Homem de Mello and Sanderson (1990) also use interactive methods to tackle the problem. It was also assumed that the assembly sequence is the reverse of the disassembly sequence and therefore, the problem of generating an assembly sequence becomes the generation of a disassembly sequence. Although there are cases where assembly sequences may not be the reverse of the corresponding disassembly sequences, disassembly planning is widely used in the research community due to its advantage in planning efficiency. The disassemblability of a part or a sub-assembly directly implies the satisfaction of precedence relationships, whereas in the forward search, the satisfaction of precedence relationship between a pair of mating parts may not be known immediately until an exhaustive search is completed. The algorithm used generates all cut-sets of the assembly's graph of connections, and checks which cut-set corresponds to feasible decompositions by generating questions which are to be answered by the user. AND/OR graph representations of assembly sequences are then generated and the sequence is created. Kunica and Vranjes (1999) also employ disassembly planning in their CAD-based prototype system for the automatic generation of plans for automated assembly.

Arai and Iwata (1993) developed a kinematic simulation system for disassembly sequence planning that simulates physical phenomena, including the effect of gravity. The part to be removed is decided by comparing the evaluation standard values of candidate parts when several parts can be removed at the same time. With the use of a product model, presented by 3-D solid geometry, possible movements of each part in the product are calculated, and the removal possibilities from the product can be

searched. When there are several ways to remove the part from the product, the best disassembly operation for the part is selected on the basis of shortest distance of part movement. When given an assembly of rigid parts, the partitioning problem of identifying a proper sub-assembly that can be removed as a rigid object without disturbing the rest of the assembly, is addressed by Wilson *et al* (1995). Ben-Arieh and Kramer (1994) also focus on sub-assembly combinations, using them to constrain the number of feasible assembly sequences generated.

Lee and Shin (1993) use liaison graphs to determine the assembly order through the extraction of preferred sub-assemblies. An assembly planning system, called the Assembly Coplanner, which automatically constructs an assembly partial order and generates a set of assembly instructions from a liaison graph representation, was developed. The planning is carried out with respect to the geometry, physical nature and resources, to find a cost-effective assembly plan in a flexible assembly system.

Laperrière and ElMaraghy (1992) initially presented an integrated approach to assembly planning where the evaluation of assembly sequences is performed as they are generated. Geometric feasibility, stability and accessibility constraints were introduced to reduce the size of the directed assembly graph to be searched. This approach led to the development of a Generative Assembly Process Planner (GAPP) taking as input a solid model, and outputting feasible assembly sequences (Laperrière and ElMaraghy, 1996). The relative quality of different assembly sequences can be determined using pertinent criteria such as the number of re-orientations, concurrency and grouping of operations.

Mazouz *et al* (1991) consider artificial intelligence techniques, and use the knowledge-based systems concept to generate optimal assembly sequences without the need for user involvement. They define two kinds of parts, internal and external. The function of the liaison between parts can be 'maintained', 'putting on' or 'putting on-maintaining'. A series of rules was developed based on these definitions. However, these definitions sometimes seem ambiguous. Nevertheless, the suggested idea aims for a higher level abstraction in representing assembly problems, which offers an opportunity for more effective use of computer planning systems. Tönshoff *et al* (1992) use a knowledge-based approach to generate assembly sequences. The system selects the optimum

assembly sequence on the basis of data on the possibilities of connecting jointing positions, and on assembly technology, e.g. jointing method, technical function and jointing direction. The rules and priorities that make up the kernel of the static knowledge-base were derived from numerous analyses of assembly documents. Senin *et al* (2000) investigate the application of genetic algorithm-based search techniques to concurrent assembly planning, where product design and assembly process planning are performed in parallel, and the evaluation of a design configuration is influenced by the performance of its related assembly process. Genetic algorithms are optimisation methods which adopt search strategies that imitate mechanisms of natural selection. The main problem with such an approach is finding an optimal reliable solution in a feasible time-scale, especially when the number of parts to be assembled increases.

The increasing demand for product variety forces manufacturers to design mixed-model assembly lines on which different product models can be switched back and forth and mixed together with little change over costs. This leads to a requirement for better co-ordination of components supplied to the assembly lines, otherwise, lines may have to be stopped due to part starvation. Zhang *et al* (2000) developed an optimisation-based scheduling algorithm, using the Lagrangian relaxation technique, to deliver products 'just in time', whilst avoiding possible component shortage. Based on the results presented, high quality schedules were generated whilst satisfying all the constraints of this problem.

2.4 Assembly Product Modelling

Human beings have a long history in the use of graphical methods to express their ideas and thoughts. Engineering drawings are graphical representations of real parts and products, and can be considered as being the graphical language for industry. Engineering drawings are the most useful and universal means of describing artefacts which have not been made. Progressively, however, they will be replaced by advanced information technologies that integrate and co-ordinate various life-cycle considerations during product development. A central issue among these information technologies is product modelling, which generates an information reservoir of complete data to

support various activities at different product development phases. Product modelling is at the centre of current research into integrated product development.

A product model is a means of storing and representing data about the product. Historically, product models have been designed specifically for a particular CAE element to perform a particular task. Examples of early product models include two-dimensional CAD and finite element mesh representations. Although each model performs adequately for its specified task, it is nearly impossible to translate one model automatically into another because it represents and stores only a subset of the total product data according to the task in hand. In addition, the different nature of the engineering disciplines involved lead to a fundamentally different approach to modelling the product (Salomons *et al*, 1993).

The importance of a properly structured and powerful product model can be seen from the requirements to improve the integration between CAE elements. It has been suggested (Spur *et al*, 1986) that data flow in product development can be classified as either geometry-oriented or administration-oriented data. In order to manage the integration of separate software systems, it is important that these data flows be incorporated into the same model. A comprehensive attempt to define a specification for product modelling is presented by Krause *et al* (1993). They define a product model as "the logical accumulation of all relevant information concerning a given product during the product life-cycle". A methodology for the design of product models for specific manufacturing systems is set out using the concept of process chains, which represent the set of technical and management functions required to develop products from beginning to end. The requirements of a product model can be summarised as: Create a consistent product description for all stages in design and manufacturing; present the actual model data; capture and record the design intent; facilitate product documentation; offer decision alternatives; and ensure manufacturability whilst designing [(Krause *et al*, 1993) (van der Net *et al*, 1996)]. The model can help prevent unnecessary iterations in the design process in various ways. By maintaining alternative decisions, it provides protection from downstream uncertainties, and manufacturability checks can immediately identify some impossible or undesirable designs.

Arai and Iwata (1992) discuss the specific requirements for product models in the conceptual design stage. In particular, the need to integrate functional modelling with geometric modelling is stressed. This approach is supported by other researchers [(Salomons *et al,* 1993) (van der Net *et al,* 1996) (Meerkamm, 1993)]. The authors state that a conceptual product model should support representation of the functional requirement, the design specification and the rough structure of design solution. In order to link functional and geometric modelling, the representation of the designer's intent is critical. A structured 'design process description language' is proposed as a means of standardising the design intent of a particular action. This is an attempt to devise a language which may be processed automatically, or by users, to pass on the design intent.

Another approach to the capturing of design intent is presented by van der Net *et al* (1996), using the concept of manufacturable design transformations. In this modelling system, the designer is restricted to a pre-determined set of manufacturable geometric transformations. These are characterised by an operator and an associated design object, which is represented in the resulting model as a reference element, linking features together according to either topology, tolerances or assembly relations. The advantage of this approach is that design manufacturability is ensured, and downstream users of the model can see the relationships intended between features. However, this scheme does not capture functional design intent at this stage. The requirements for product models go beyond merely representing the product from the point of view of one engineering discipline. The product model should provide an integrated data set which maintains all product data, from initial concept through to disposal. This means that the product model must be capable of changing with the evolution of the product, and supplying data in formats suitable for all engineering disciplines. The product models available from commercial vendors do not currently meet these requirements. In general, product models are geometrical models based on CAD systems.

A number of software tools are available which claim to offer an integrated CAE environment based on a core CAD system (Pro/Engineer from Parametric Technology Corporation, Euclid from Matra Datavision, CATIA from Dassault/IBM). Each of these uses a primarily geometrical model to represent the product, although additional data

may be stored in some cases. In order to provide integrated finite element analysis, the most advanced systems allow the automatic generation of meshes from the product geometric model. Similarly, there is software available which can provide dynamic analysis of CAD solid models. However, none of these models provides a suitable solution to the representation of design intent, and most are inadequate for the demands of analysis such as automated assembly process planning tasks. Current research into product models has concentrated on enhancing geometric product models, either to produce an integrated product model suitable for all product development domains, or to tailor the model for use in a particular domain.

2.4.1 Geometric Models

Solid modelling is the most advanced modelling technique used in geometric modelling software. It can provide mathematically unambiguous information and complete models for real world objects. There are several representation schemes developed and used in solid modelling software, such as Constructive Solid Geometry (CSG) and Boundary representation (B-rep). In addition, many systems now combine these approaches into a hybrid B-rep/CSG scheme. The most popular representations for CAD solid modelling packages are CSG and B-rep. The reason for the popularity of CSG is its robustness and its simplicity for validity and integrity checks. It uses solid primitives and regular Boolean operations for constructing models of products. A variety of solid primitives can be modelled. Common solid primitives such as cylinder, cone, block, sphere and wedge are often supported in modelling systems, and some systems also allow users to model their own primitives.

B-rep technique represents a solid through its boundary surfaces. The basic idea of B-rep is to represent a solid by decomposing its surfaces into a collection of faces which have mathematical representations. The disadvantages of B-rep systems are that models are difficult to construct and they are poor at capturing the design intent. The main drawback to CSG representations is the lack of explicit representation of the lower level entities of the part, such as lines, points and surfaces. Hybrid solid models (Werling and Wild, 1994) seek to combine the advantages of both solid modelling approaches. CSG representation is used for the macroscopic representation of geometry, whilst lower level entities are represented through the modelling of each CSG primitive in B-rep

format. It is found that both B-rep and CSG representations cannot provide the technical and functional information essential for subsequent manufacturing applications such as process planning. The following section therefore deals with feature representations which aim at bridging the gap between design and manufacture.

2.4.2 Feature Technology

The use of features can be seen by many researchers as the key to genuine integration of many aspects of design, and the planning of manufacture and assembly [(Denzel and Vosniakos, 1993) (Molloy *et al*, 1993) (Case *et al*, 1994)]. On the design side, this could relate to the fulfilment of functional requirements, the building of geometric models, or as preparation for design analysis such as finite element analysis. On the planning side, activities such as process planning, assembly planning, inspection planning, part programming, and so forth, could potentially be based upon a feature representation of the product. A large body of research has been generated on feature-based product models [(Case and Gao, 1993) (Salomons *et al*, 1993)]. Many researchers have tried to define the term 'feature', but there is much disagreement over the use of the term. What is commonly known, is that the term 'feature' is defined differently according to the points of view of research. Definitions range from the broad definitions given by Pratt and Wilson (1985): "A feature is a region of interest on the surface of the part" and "A generic shape that carries some engineering meaning"; to those more specifically related to a particular domain, such as Henderson (1986): "Features are defined as geometrical and topological patterns of interest in a part model and which represent higher level entities useful in analysis". Van't Erve (1988) defines features for process planning as "a distinctive characteristic part of a workpiece defining a geometric shape". Fu *et al* (1993) define features thus: "A feature is an abstraction of a set of geometric constraints and can be associated with a meaningful context". Example contexts are either manufacturing or functional. Lenau and Mu (1993) suggest two complementary definitions of features: "Information sets that refer to aspects of form or other attributes of a part", and "a group of geometric entities that together have some higher-level meaning". The first definition is more general, whilst the second limits the term to geometric entities.

Many different types of features are identifiable, with the three main types being 'design or functional features', 'geometrical features', and 'manufacturing features'. Pratt (1993) is more specific in his definition of form features and also includes assembly, tolerance, inspection, fixturing, robotics and analysis features. For true integration of product life-cycle activities, it is preferable to have a single unified feature representation, or failing this, a number of representations which can be readily mapped between each other. Feature-related research can be divided into two main fields: The representation and data structures of features; and the means of obtaining the feature data to create the model. The former may be viewed as the development of feature taxonomies, whilst in the latter case, two approaches dominate, design by features and feature recognition.

2.4.3 Feature Taxonomies

In practice, features are usually divided into different classes to help the designer to access the feature data and assist the manufacturing engineer to generate process plans for a group of features that have some common geometrical, topological or other properties. Such classes can be further divided into sub-classes, so that classes and sub-classes form a hierarchy. This classification structure is known as a feature taxonomy. A feature taxonomy is central to the development of a feature-based product model, and many researchers have developed such taxonomies. The failure of a standard feature taxonomy to emerge can be explained by the assertion that "the way of classifying features is highly dependent on feature representation methodologies and strategies for the eventual use of the feature data" (Case and Gao, 1993).

Butterfield *et al* (1985) classify form features into three main categories: Sheet features; rotational features; and non-rotational features. Each of these classes is further divided: Sheet features as either flat or formed; rotational features as either concentric or non-concentric; and non-rotational features as either depressions, protrusions or surfaces. Because this scheme was intended to be the standard for all the application programs carried out in the Computer-Aided Manufacturing International project, it is broad and general. Pratt and Wilson (1985) divide feature representations into two types, explicit and implicit. In an explicit feature the geometry is fully defined, whilst for an implicit feature, the feature is represented parametrically by attribute values, and the full

geometry must be calculated as required. Tönshoff *et al* (1996) use explicit and implicit features in a modified manner, using a dual representation for each feature in order to integrate regular and free-form features into a unified classification.

Gindy *et al* (1993) treat form features as volumes enveloped by entry/exit and depth boundaries. This taxonomy is particularly suited to manufacturing representations. A feature is defined by imaginary and real faces. The number of imaginary faces determines the 'external access directions' which can be used for process planning. The result of grouping features according to these characteristics is a list of form feature classes or primary features, such as holes, steps, pockets, bosses, and real and imaginary surfaces. The scheme is closely linked to the process planning requirements, and is sufficient to classify the features used in this domain.

Gandhi and Myklebust (1989) use a parametric approach to the definition of features. The taxonomy is based on the topology of feature primitives, i.e. features having the same topology are grouped together so that they can be defined by using the same number of parameters. An example would be the group of features which can be described by the parameters of a length and a radius, which includes cylinder, disk and cylindrical plate. An additional level of classification can be applied according to form, such as angularity, curvature, rotundity, straightness and circularity. This taxonomy is perhaps less logical because of the need for a combination of two separate classification schemes.

Gao and Huang (1996) classify features into three levels: Atomic features; primitive features; and compound features. Atomic features include points, lines, arcs, planes, surfaces and so forth. These features represent the constituent elements of primitive and compound features, such as faces, edges, axes and so forth, and are essential for dimensioning and tolerancing. Primitive features include the classes surface, boss, pocket, hole, slot and so forth. Each feature class is further characterised by a number of profile shapes. The topology of a primitive feature is determined by its class and profile shape. Each primitive feature can be decomposed into atomic features and therefore can be referred to independently. Compound features are a collection of primitive features and/or atomic features which may together perform a single function, or may be manufactured by similar operations.

Latif *et al* (1993) describe an object-oriented feature taxonomy based on the definition of a base or stock feature, such as a box, which is then modified by the addition of features, such as holes or pockets. Maropoulos *et al* (1998) also use this scheme to represent products. Components are described using both positive and negative form features. A positive feature is a geometric shape that encloses a material volume, such as a cylinder, a prism or a sheet. A negative form feature is a geometric shape where material has been removed from a part, such as a hole, a slot or a recess. A minimum of one positive feature is required for each component, because it is only the positive features which hold material information. With an object-oriented model of the product, it is possible to use inheritance to infer properties of the components from its features and *vice versa.*

Taxonomy schemes must be measured against two requirements. Firstly, a rigorous taxonomy is a prerequisite for the production of predictable analytical algorithms for engineering systems; and secondly, the feature taxonomies and representations must support the generation of the geometry during design. Gindy's scheme is aimed at providing a structure which simplifies the generation of process plans, and meets the first criterion very well for a particular analysis requirement. The schemes of Gao and Huang, Latif, and Maropoulos *et al* similarly use the vocabulary of process planning, and are most suited to this domain. Butterfield's taxonomy is less specialised, and suitable for an integrated product model used by many analysis systems, whilst Gandhi and Myklebust's scheme is strongly aligned to the second criterion.

2.4.4 Feature Data Models

Manufacturing planning systems need to extract feature-based component information from CAD systems both accurately and efficiently. Currently, there are two main approaches to obtaining feature information automatically from CAD systems. These are feature recognition and feature-based design. The basic idea in feature recognition is to analyse a traditional CAD model and identify or recognise form features in it. Feature recognition can be divided into a number of categories or approaches dependent on the type of geometrical model to be analysed. Lenau and Mu (1993) list five categories of feature recognition methods: Syntactic pattern recognition; state transition diagrams; decomposition approach; CSG (set theoretic approach); and graph-based approach.

Singh and Dengzhou (1992), and Subrahmanyam and Wozny (1995) discuss these methods in more detail. The feature recognition approach is inherently unsatisfactory, because the CAD data does not originate in that form, but rather from the designer. Hence, any feature model that is generated in this way is inevitably a translation of a translation, with a resulting loss of accuracy of information content.

One of the chief criticisms of the feature recognition approach is that it promotes a "wanton abandonment of design intent" (Case and Gao, 1993). Any design intent captured in the geometric model is not passed on to the features. Other criticisms of feature recognition techniques are that technological information or some features are not recognised, and have to be entered afterwards, and that this technique can become very complex and computer intensive. The errors caused by feature recognition can be avoided by using feature-based design.

In the 'design by features' approach, the designer is provided with a features library, similar to the primitives of a CSG system, which can be used with a set of operators such as add, delete and modify, to create a feature representation. The feature representation maintains additional information such as feature names, taxonomy codes and attributes that are not kept in a conventional solid modeller, and this eliminates the need for feature recognition. The functional requirements of a feature-based design system are summarised by Pratt and Wilson (1985), Shah and Rogers (1988), and Case and Gao (1993), as follows:

- The data supported must be sufficient for all applications that will use the database.
- The mechanism for feature definitions must be flexible (generic) to allow designers to define their own needs.
- The product definition system must provide an attractive environment for creating, manipulating and deleting feature entities. Feature relationships should also be defined.

- The design system should be able to integrate with different application software and the interface mechanism should be flexible or generic so that the effort to integrate with different software can be minimised.

Case (1994) describes a 'design by features' implementation using Gindy's feature taxonomy to create feature-based product models. In this approach, the feature representation is maintained in a parallel data structure, although the author suggests that it would be preferable to redesign the solid modeller's structure to add the feature data if possible. An iconic user interface is used to select feature types to add to the model and to define relations between features. It is claimed that the iconic feature-based interface proves a more efficient and robust means of specifying geometry than the underlying solid modeller. Further work from this project is reported by Rahman *et al* (1995), where the feature taxonomy is extended using an object-oriented approach to add functionality to the geometric reasoning process.

Latif and Hannam (1996) discuss the practicalities of amalgamating both approaches to produce an object-oriented, feature-based design system. They conclude that this approach can be successful due to a number of reasons, namely: The modularity of the approach allows for the system to evolve efficiently; corresponds to every operation required in a CAD system, including tolerancing and links to CAM; and is a natural way of organising data, allowing the user to interact easily. The main criticism of design by features is that the limitation of a defined feature library will over-constrain the designer. However, this is equally a problem with feature recognition, which will also fail if a design has features outside the existing taxonomy. Furthermore, it can be considered a benefit of the system that the designer is required to use standard solutions to problems by restricting the allowed geometry.

A technology which is related to both feature recognition and feature-based design research is that of feature mapping. In this process, a product modelled using one feature representation is converted into an alternative representation which is to be used for a particular activity. Fu *et al* (1993) present a feature representation which is tailored to the requirements of feature mapping or feature transformation. They specify the need to "support automatically the different ways specialists view the same object" as the main drive of their research.

2.4.5 Representation of Assembly Data

Although there has been much research in modelling geometry aspects of a product, there has been little work undertaken representing assembly data and linking geometry models to process planning for fabrication and assembly, cost analysis, quality analysis, tolerance analysis, or other aspects of product DFMA. Thus, there is a need to develop a complete product model which incorporates information on how all components are stored in an assembly, in addition to the geometrical and topological data on each component (Case and Harun, 1998). An ideal system allows the link to be established between the geometric and assembly model so that the designers need only to modify individual parts for design modification by using the geometric modeller, and the assembly model is updated automatically (Zeid, 1991). The information used to describe an assembly includes the data of each individual part, and the relationships between the parts. The relationships between the parts describes how these parts should be assembled, including orientation, location and mating conditions.

Gui and Mäntylä (1994) state that a functional understanding of assembly modelling is a key step towards a real CAD environment that can support early design. Delchambre (1992) suggests a structured model of assembly containing almost all the required information. In this model, the geometrical information specifies the shape and dimensions of the parts, as well as their relative positions within the final assembly. Component information includes the features of the components and their roles in the assembly. There is also the topological information that indicates the type of contacts between the parts in the assembly. Lin and Chang (1993a) include both geometric and non-geometric information in their assembly product model. The non-geometric information includes standardised machine elements, mechanical fasteners and assembly design intents. This assembly product model is used in the Three-Dimensional Mechanical Assembly Planning System (3D Maps) (Lin and Chang, 1993b). Li and Hwang (1992) include material type, handling and feeding conditions as non-geometric features (e.g. abrasive, fragile, tangle and weight). They also use operational assembly features such as insertion path, direction and difficulty, holding conditions, insertion resistance and so forth in their assembly product model.

Lee and Gossard (1985) provide a hierarchical two-part data structure for representing components and sub-assemblies in a database. The first part is the data structure used to store topological and geometrical information on each component in an assembly. The second part is the data structure used to store information on how all the components in an assembly are connected. A tree structure, using the concept of the 'virtual link', is created to represent the relationships between the components in the assembly. Lee and Andrews (1985) create a data structure to represent an assembly based on the spatial representations between its components. In this structure, the relationships are defined by 'fits' and 'against'. The 'fits' condition applies to the relationship between a solid cylinder and a hole, and the 'against' condition applies to the relationship between two planar of two components. Hsu *et al* (1993) have developed such a taxonomic approach to representing assembly mating relationships. Their scheme includes general conditions such as 'fit', 'place', 'stack' and 'insert', as well as more specific assembly operational terms, including 'screw'. Henrioud and Bourjault (1992) also include non-assembly data such as labelling and checking into their product model representation.

The definition of components, and the mating relations between the components, yields to a connected graph often referred to as a relational model. In typical relational models, the components are represented by the nodes of the graph, and the links define the relations between the components. Each link contains all the relations between the related components (Ben-Arieh and Kramer, 1994). However, some researchers use a separate link for each contact relation. For example, the relational graph of Homem de Mello and Sanderson (1991) has a separate link for each contact relation. In their model they use three types of entities, parts, contacts and attachments. The relational model can be directly used for assembly reasoning and analysis, or it can be the starting point for the generation of other types of assembly model.

The creation of hierarchical models organises the relations in the assembly, thereby reducing the computational complexity of algorithms for assembly analysis. A hierarchical model is also a more realistic representation of the functional intent. Most assemblies are designed sequentially, with groups of functionally-related parts forming sub-assemblies. Santochi and Dini (1992) identify 'sub-groups' that can be assembled separately before the final assembly of the whole product. They also use a 'table of

contact' formalism in their FLexible Assembly Planning (FLAP) system. Chakrabarty and Wolter (1997) use structured hierarchy product models. These are arranged naturally, with the large, high-level structures containing smaller sub-structures. They use libraries of structures for efficient product definition.

Mo *et al* (1999) put forward a DFA-oriented assembly relation model composing of a function relation model, geometry relation model, and connecting relation model. The function model describes the functions of the product and component, the geometry model details the contacting and positioning relations among parts, and the connecting model relates to the connecting methods employed. The connecting model is divided into two categories, direct and indirect connecting. Typical direct methods are interference fit and bending, whereas standard indirect methods are thread and pin connecting. Based on the relation models, a DFA expert system module can evaluate the joining process and suggest improvements to the design if required.

You and Chiu (1996) use feature-based libraries for standard parts such as bolts, nuts, bearings and so forth, that are frequently used for assembly purposes. Their definition of the main difference between standard and common parts is that standard parts "have some specific meaning in the design". For example, bolts must be used to fasten several parts, and bearings must mate with shafts or holes. It is noted that the hierarchy representation and inheritance offered by object-oriented programming is suited to standard part libraries and databases.

2.5 Assembly Process Modelling

There is growing awareness that the product designer's decisions are responsible for a major part of the total product cost. In order to ensure that the best decisions are taken, it is vital that the designer or the design team has easy access to all relevant information and data. Information about manufacturing and assembly processes can be obtained by including a production expert in the design team. Another possibility is to capture production expertise of process engineers, and with determining process models, supply knowledge to the whole production development team. The aim is to store artificial expertise which can be accessed by whichever engineer has a requirement for it, and additionally, can be built into automated analysis systems, whether they are Design for

X (DFX), CAPP or Concurrent Engineering systems. Process models can be used both to analyse planned production, and for comparison between measured performance and theoretical targets. They should contribute to the understanding, management and improvement of production.

Lenau and Alting (1992) identify a number of process modelling technologies: The morphological process model; group technology; books on manufacturing and assembly processes; CAPP; and constraint modelling. They point out that most sources of information on processes do not adopt a uniform method of description, making it difficult to compare processes and codify process knowledge. They propose a design-oriented process model based on the following: Basic transformation (including characteristic motions, energy, material, features, fixtures and reliability); equipment, machines and availability; pre- and post-consequences; company policies; cost; and the environment. Although numerous researchers have developed process models for machining [(SECO, 1997) (Sandvik, 1997)] there has been little such work in the field of assembly process modelling. This is mainly due to the complex nature of assembly process models. Work undertaken in this field includes work measurement, assembly time data sheets and process-based cost modelling.

2.5.1 Motion Time Studies.

The field of work measurement evolved to estimate the time needed by suitably qualified and adequately motivated workers to perform a specified task at a specified level of performance. Work measurement techniques encompass: Time study (direct observation with performance rating); work sampling; standard data; and Predetermined Motion Time Systems (PMTS). A variety of predetermined time standards are currently used to establish assembly times in industry. The most common systems are the Methods Time Measurement (MTM), Maynard Operations Sequence Technique (MOST) and Work Factor (WF). These are similar in the way that human motions are classified, but each employs a different coding system (Dossett, 1992).

WF is based on the pioneering work of Frank Gilbreth, who points out that all human labour is composed of the same elemental motions, which in WF are called standard elements. When setting up the system, various factors were identified which influenced

the time to complete these standard elements. These WFs are expressed in numerical values, e.g. 1 WF, 2 WFs and so forth. They supply information about the difficulty involved in a motion. Decreasing the difficulty of an operation means decreasing the time needed and a reduction in the physical and/or mental load.

The MTM system is the most popular PMTS. It is a detailed system which divides any operation into single motions (Maynard *et al*, 1948), including obtain, locate, rotate, force and so forth. It also contains four combination motions: Consecutive; combined; simultaneous; and compound. Each MTM motion corresponds to a number of Time-Measurement Units (TMUs) and these equate to an actual assembly time in seconds. The MOST, developed by Zandin (1980), consists of three versions: Basic, Mini and Maxi. Basic MOST is comprised of three basic sequence models: General move sequence; controlled move sequence; and tool use sequence. In addition to the three basic sequences, an equipment-handling sequence is available to analyse the movement of heavy objects which require a manually-operated crane. Motions included in MOST are get, move, actuate, return and so forth. The time for each sequence is obtained by adding together the index numbers.

Computerised versions for MTM and MOST have been developed (Genaidy *et al*, 1990). They require the user to gather workplace information and key the information into the computer database. Commonly, the computer-based systems are two to five times faster than the manual application. Although motion time studies generate accurate assembly times, they are generally used in the latter stages of the product development process as a stand-alone tool, or during the product re-design process. This is contrary to the Concurrent Engineering philosophy. Many companies, including Nissan, Phillips and Flymo, have their own PMTS designed specifically for their product base.

2.5.2 Assembly Data Sheets

Assembly time standards were developed by Boothroyd and Dewhurst (1987) as a result of extensive experimental studies to measure the effect of part characteristics on assembly times. The results of this work were collated into three sets of assembly data sheets (manual, automatic and robotic) and are a major function of their DFA analysis

tool. To extract correct assembly times, their classification system needs to be understood. The classification system for manual handling is a systematic arrangement of part features in order of increasing handling-difficulty levels. The classification system for manual insertion and fastening processes is concentrated with the interaction between mating parts as they contact and go together. Two-digit codes range from 00 to 99 and have an assembly time specific for each code.

Experience has shown that, under normal circumstances, the time error results in some cases over-estimating, and in other cases under-estimating. Hence, these tend to cancel each other out. However, if an assembly contains a large number of identical parts and operations, care must be taken to check whether the part characteristics fall close to the limits of the classification. A weak point of these data sheets is that they only contain one time for operations such as screw tightening, riveting, welding and so forth. In practice, these times would vary considerably depending on the tool used, or process parameters such as the length of weld and so forth.

2.5.3 Cost Modelling

Bloch and Ranganathan (1992) developed a process-based cost modelling suite as a suitable decision support tool for evaluating different technology choices. This method models the material flow to and from each process step, and calculates the cost of processing at each step. The overall cost is the sum of materials, manufacturing and assembly process costs, and latent costs. Some of the applications of the tool include: Selection of material, technologies, processes and equipment; vendor evaluation and make or buy decisions; and competitive bench-marking. Machine utilisation costs, operator or direct labour costs, indirect labour costs, and overhead costs are elements used to make up the overall assembly cost.

2.6 Conclusions

From the review of state-of-the-art assembly modelling and planning techniques, there appeared to be many proposed methods, but no developed system that could handle a realistic product assembly. A bias showed that the proposed systems either required vast input by the user, or could only handle assemblies with few components. It was

also found that there was limited support for assembly modelling and planning at the conceptual stage of the product development cycle. DFA approaches give a qualitative solution to gain an optimum design, and not a quantitative value for the actual assembly times, costs and resources.

In spite of the great success of both DFA and assembly process planning methodologies and systems, it is felt that there would be more benefits and savings if the two approaches could somehow be integrated. This is because, when the two approaches are examined closely, it is realised that something is missing in each approach. The DFA approach provides guidelines on how best to design a part or a component based on some previously performed empirical studies. However, it gives little or no consideration to the actual assembly plan by which the product is to be assembled. For example, a certain design change may require a change in assembly direction or it may require an extra tool change. Such cases cannot be detected by the existing DFA approach because of its ignorance about the assembly plan. On the other hand, during the planning process, a great deal of information is discovered which is helpful to the designer. Unfortunately, there is no systematic way of gathering this information that is useful to the designer, and even if such information could be gathered, no means are available of analysing the information in a form that can be understood by the designer. As a result, many of these pieces of information are buried, and one is content simply to find the best assembly plan for a less optimal design. Therefore, it can be seen that if the link between the design process and the planning process could be supplied and both methods integrated, many benefits could potentially be reaped.

Firstly, such integration would shorten the product life-cycle time considerably, and hence increase competitiveness. Secondly, this integration would enable good use of the information that is available from the planning phase to improve the design for ease of assembly. Thirdly, substantial costs could be saved by changing the design in the early stages of design and planning, instead of changing it later in the development cycle, which is costly. Fourthly, it would provide a fast and efficient way of evaluating different design options for a given product. Finally, with such integration, the designer would be free from having to worry about DFA issues. Instead, the designer could focus his/her efforts on satisfying the functional requirements of the product.

Chapter Three

System Overview

3.1 Introduction and Theory

As described in the preceding chapter, there is limited support available for assembly modelling and planning during the early stages of product development. A new contribution to this field of investigation is the integrated computer support system operating at an aggregate level, which has been developed as part of this research in order to address these limitations. The system is called the Aggregate Assembly Modelling and Planning (AAMP) system. The methodology developed throughout this thesis is defined through the specification and functionality of the prototype support system. It is important to note that the development of the tool was not the purpose of this work; rather, the system was developed to test the theories which have been applied in its development. This chapter presents an overall description of the system which has been developed.

The AAMP system is a CAE tool which is targeted at filling the perceived gap in support for product development at the early design stages. The primary requirement for designers during this stage of product development is the analysis of the ability of a given design to perform its required function. For Concurrent Engineering, however, it is important to ensure that the designers are also able to consider the manufacture, assembly and subsequent life-cycle issues of the product. The assembly constraints should be considered by the designer, along with the product performance constraints.

Whilst the design of a product is the principal influence on the assembly process, the designer is not necessarily an expert on assembly processes. Therefore, in order to properly consider the consequences of a design, the following questions should be addressed: (i) How easily can this design be assembled? (ii) What is the assembly time and cost for this design? Whilst it is usually possible to assemble any given design, the assembly duration and cost of doing so may be unacceptably high. These questions lead to the definition of the 'assemblability' of a design, which is an indication of the suitability for production. Assemblability can be measured in many ways, most notably the assembly time and cost. Factors contributing to the cost include: Labour; investment in assembly resources; transportation cost; resource depreciation; material; energy; cost due to ensuring quality; storage (space provision); and investment cost. Additional measures of assemblability which are useful include: The ease of assembly; the assembly lead time for the product; the critical assembly path; the critical assembly time; and the effect on factory loading of different designs.

In order to assess the assemblability of a design, it is necessary to identify possible ways of assembly, then to check the implications of the use of each of these alternative methods, thereby arriving at times and costs for each alternative. The AAMP system successfully achieves this through the generation of aggregate assembly process plans. In order to operate during the early design stages, a Concurrent Engineering support system for assembly must fulfil a number of criteria:

- Provide a link between the early stages of product design and assembly processes and methods.
- Have the ability to represent alternative design concepts during conceptual, embodiment and detailed design stages.
- Efficiently acquire accurate assembly decisions from limited product information.
- Derive accurate estimated assembly times and costs, assembly sequences and required resources.

- Aim to create and compare alternative product designs and configurations, and also different assembly methods, processes and manufacturing assembly resources.

- Have the ability to assess alternative aggregate assembly process plans in terms of assemblability criteria.

These criteria lead to more detailed requirements about the structure and elements required in the system. In particular, in order to perform aggregate assembly process planning and assessment of the production routes generated, it is necessary to provide production process expertise and knowledge of the factory's manufacturing resources within the system. To perform automated assembly process planning, the computer system must capture assembly process knowledge, and rules for selection of processes and calculation of input and output criteria. Aggregate assembly process planning should be integrated with the production capabilities of an individual organisation. This implies that the system must have access to appropriate data on the factory resources, including the assembly machines and tools available, the layout of the factory and cost rates for resources and labour time.

The AAMP system is believed to be the first developed system that fulfills all of the above critera within an integrated CAE tool, thus making an important and original input to this area of research. The fundamental theory behind this work is a unique methodology that will bring all aspects of product development together to consider assembly planning at the conceptual stage of design.

A prototype Concurrent Engineering support system, known as CAPABLE, (Maropoulos *et al*, 1998) which aims to provide multi-disciplinary support in the product development process, has been developed at the University of Durham by Dr. Hugh Bradley and Dr. Zhihui Yao. The AAMP system is designed to sit on top of this existing system and use some of its basic functions, such as loading, saving and so forth. Most importantly, the AAMP system uses CAPABLE's feature-based product model. This is further developed to include representations of assembly connections and additionally required assembly data necessary for aggregate assembly process planning.

3.2 Overall Layout

The AAMP system is organised as an event-driven system which allows the user to construct and modify a number of structured models. Specifically, these models represent the current design idea which has been generated. In addition, the system uses an object-oriented model of production processes which can only be modified by the administrator. In normal use, the product developer would use the design editor function to enter information available about the proposed product design, make modifications to the factory resource model with the factory editor function, and then analyse the assemblability by running the aggregate assembly process planning function. The architecture of the system is shown in Figure 3-1.

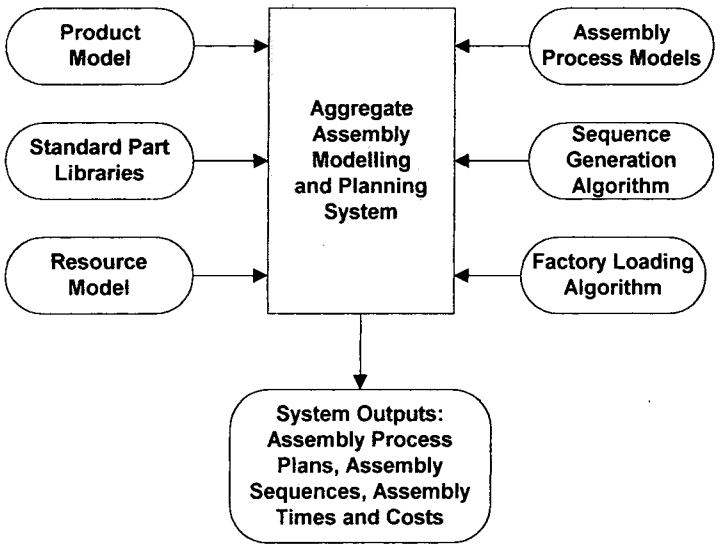


Figure 3-1: System Architecture of the AAMP System

The functions of the AAMP system can be broken down into a number of separate modules which are linked together to provide the overall novel assemblability assessment function. In particular, it is important that the system allows the designer to browse and modify the current design model in order to compare alternative design configurations.

3.2.1 Product Model

The user of the AAMP system is able to enter a description of the product design and create a model of the product within the system. The model can be modified through

editing existing information, deleting information and adding new structures. It is recognised that a fully functional Concurrent Engineering support system would provide an integrated link to a three-dimensional CAD system, extracting data from the more detailed solid model into an aggregate product model. For an initial conceptual design, however, the use of a solid modelling system is not always appropriate. The AAMP product model allows the representation of design information on components which cannot yet be drawn because they are not fully defined. Along with a means of editing designs within the system, this also provides the necessary function to load and save current design ideas to files on disk.

The product model is an object-oriented representation of the product structure, and uses a feature-based solid modelling approach which is compatible with the latest CAD systems and is highly suited to assembly planning. The product model is made up of information about both the geometry and the product structure. The model utilises a schema based upon a bill of materials, a familiar technique used to visualise the product because it is constructed from its sub-assemblies and components. A product browser is the primary means of design specification at present. This allows the user to alter the design product model in any way, including the loading and saving of product definitions and the modification, addition and subtraction of product features.

To aid the process of product modelling, standard part libraries have been specifically developed for the AAMP system. These libraries reduce the time for product specification, and hold information on part features, geometry and assembly process data. The product model can represent a high level of detail when required, including dimensional information, whilst retaining the ability to simplify data required in the early stages of design.

3.2.2 Assembly Feature Connections

An innovative factor of this research is the introduction and use of Assembly Feature Connections (AFCs) within a conceptual product model. The function of AFCs is to indicate which component features are linked together, as shown in Figure 3-2. Also represented is the generic form that such a connection will take, such as a placement, snap fit, plug and target, threaded and so forth. The assembly connection object is

attached to the joining features on the product model, and data entered during the modelling stage, or derived during later processing, is stored attached to this AFC object. The type of AFC is determined automatically, or semi-automatically, depending on the range of possible options and the confidence of decision-making. It will be automatic when a number of features can be assembled in only one way. If however, several types are possible, then the user will be presented with the range of possible AFCs, sorted in order of choice preference, and the user will suggest one.

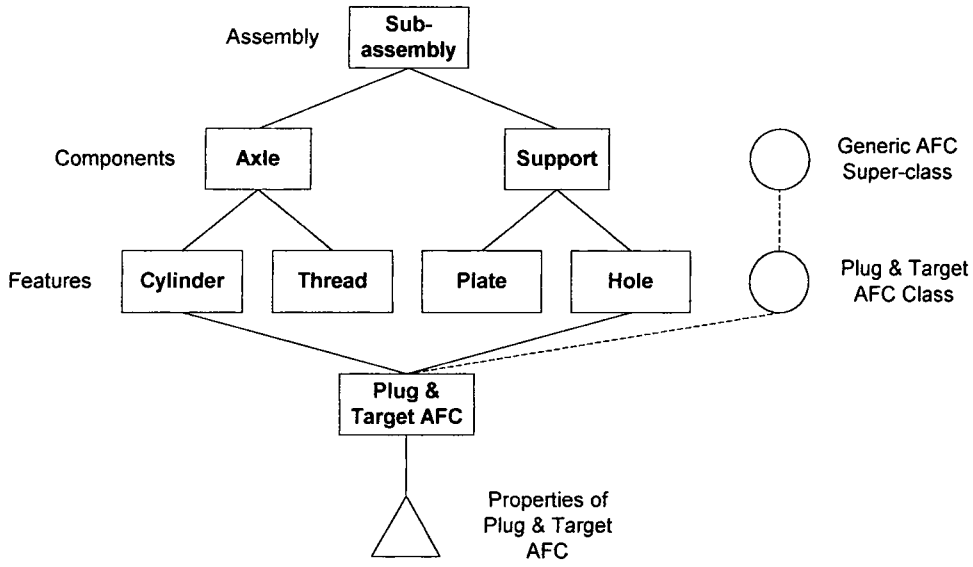


Figure 3-2: Example AFC

A major advantage of using a bill of materials product model structure, and including the AFCs as part of this product model, is that it increases the efficiency of the sequence generation algorithm during aggregate assembly process planning. All aspects of the product model and AFCs are discussed in detail in chapter four.

3.2.3 Resource Model

A substantial element of this research is the introduction of a resource model for aggregate assembly process planning. This gives the ability to model all aspects of a factory, thus allowing the AAMP system to calculate accurate assembly times dependent on where the product is assembled within a factory, or even which factory is employed. The resource model is an object-oriented model of the factory resources available to the process planner, and contains information on the factory, cells,

workstations and assembly resources, including machines, tools, transfer machines and personnel. A unique assembly resource classification is used to enable the assembly processes to be mapped to resources available. Assembly machine and tool attributes include power, operation rates, torque and speed constraints.

A factory resource browser allows the user to investigate the details of the factory database. This may be used in order to tailor the system's analysis through the selection of a particular workstation, cell, or ultimately, factory, which should be considered for the assembly of a product. The factory resource browser allows the user to edit the resource model to ensure it is up-to-date. The resource model is discussed in detail in chapter five.

3.2.4 Aggregate Assembly Process Planning

Aggregate assembly process planning, together with the development of the AAMP system, is the main work of this thesis. The generation of aggregate assembly process plans allows the assemblability of a given design to be assessed, which includes estimated assemblability criteria. All aspects of product development are considered concurrently at a very early stage of design, making aggregate assembly process planning a new concept in this field of research.

The aggregate assembly process planning function is divided into a number of stages. The main requirements for aggregate assembly process plans are the selection of assembly process, selection of assembly machines and tools to perform the process, and the sequencing of the assembly process steps. The aggregate assembly process planning function of the AAMP system is a generative automated process planning system, operating at an aggregate level. Although the plans are at the aggregate level, they are detailed enough to include factory loading down to machine and tool level, accurate estimation of costings and timings, and are based on realistic sequencing. Aggregate process data stored in object-oriented databases is combined with individual process equations, features, and resource parameters to calculate the assemblability indicators.

3.2.5 Sequence Generation and Factory Loading

A novel development of this research has been the generation of two completely new assembly sequencing and factory loading algorithms. One of the functions of the AAMP system is to derive an assembly sequence for a product. The first new algorithm is used to generate a feasible assembly sequence using the structure of the product model, process constraints, and methods to calculate the base and moving parts. This sequence is then used for subsequent system functions, including assigning assembly operations to factory resources, and calculating times and costs. The second new algorithm is a factory loading and balancing algorithm, and its fundamental objective is to load all the assembly operations onto workstations, whilst ensuring the workstations have the capacity and capability. The algorithm is divided into two routes: Loading and balancing an existing factory; and creating and loading a new factory resource. Aggregate assembly process planning, the sequence generation algorithm, and the loading and balancing algorithm are discussed further in chapter six.

3.2.6 System Outputs

Once the system has generated a set of aggregate assembly process plans, the results are outputted to a number of HyperText Mark-up Language (HTML) files. Each AAMP output HTML page shows a specific section of the results, including: A summary of results; cell loadings; workstation loadings; full details of assembly operations; and the assembly resources. Hyperlinks allow the user to jump between pages to associated data. For example, it is possible to select a workstation on the cell loading page and jump to the workstation loading page to view the workstation in more detail, and see the assembly operations and resources associated with this workstation.

3.3 Development Tools

The AAMP system was developed using Nexpert Object (Neuron Data, 1995), an object-oriented, knowledge-based system environment, designed for the rapid prototyping of artificial intelligence-based computer systems. This system was chosen because it provides the required object-oriented modelling ability, along with a powerful implementation of the knowledge-based system. In addition, the environment

provides an integrated library of routines for the development of Graphical User Interfaces (GUIs). Result outputs are saved in HTML files. This is the industry standard for describing the contents and layout of World Wide Web internet pages.

3.3.1 Object-Oriented Analysis and Programming

Object-orientation is a technique for system modelling and understanding complex systems. Coad and Yourdon (1991) describe Object-Oriented Analysis (OOA) as the "challenge of understanding the problem domain, and then the system's responsibilities in that light". The key to understanding complex systems is to decompose the system into manageable pieces which can be more easily understood. Traditionally, systems have been decomposed on the basis of algorithmic decomposition, which breaks the processes down into individual steps. In object-oriented decomposition, the system is decomposed according to the key abstractions in the problem domain. Thus, instead of a set of process steps, the system is represented as a set of objects which are described in terms of their properties and behaviour.

The advantages of OOA lie in the benefits of abstraction, encapsulation, inheritance and organisation methods. Abstraction allows the analyst to ignore those aspects of the system which are irrelevant, and concentrate on the important factors. Encapsulation relates to the practice of hiding the complexity of an object from view when looking at the wider picture, thus reducing the complexity which must be handled at any one time. Inheritance allows the analyst to express commonality amongst objects, by defining attributes and behaviour to classes to which several objects belong. The objects inherit the attributes and services of the parent classes, thus sparing the definition of each separately. Coad and Yourdon identify three pervading methods of organisation which are inherent to OOA: Objects and attributes; wholes and parts; and classes and members. Each of these enhances the understanding of the system and leads to a more complete description.

Within this thesis, two different schemas have been used to represent object-oriented models. Whilst in general a single representation schema might be thought to be more consistent, there are advantages to using a mixture of two styles. The first schema is that adopted by Coad and Yourdon, as shown in Figure 3-3. This representation

highlights the encapsulation of data within objects, and emphasises the relationships of wholes and parts. It is particularly good for representing the details of a class structure, and defining objects which are sub-objects of others, as shown in the figure. However, this system has weaknesses. In particular, it is difficult to represent multiple objects belonging to the same class and to represent objects which are instances of more than one class, the concept of 'multiple inheritance'. In these cases, the object/class model cannot be represented without showing the same class or object more than once on the diagram, which is confusing. In such cases, a second representation has been used which is more flexible.

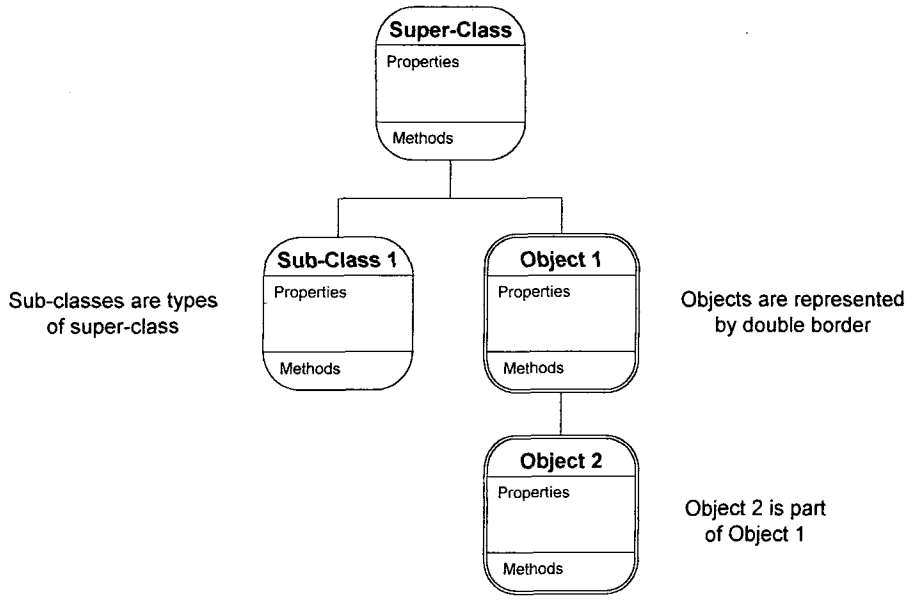


Figure 3-3: Coad and Yourdon Schema for Object Model Representation

The second representation schema is a more basic method adopted from the manuals of Nexpert Object (Neuron Data, 1995), the software development system used. In this representation, different symbols are used to represent classes, objects, properties and methods, as shown in Figure 3-4. Thus, it is easy to represent two objects which belong to the same class, or a single object which is an instance of two distinct classes. The main drawback of this approach is that it results in larger diagrams, making it difficult to represent complex situations.

Object-Oriented Programming (OOP) is the development of computer systems based upon models generated through OOA. This is a particularly powerful programming

approach which has become the most common type of computer language today. Examples of OOP languages include C++, Object Pascal and Java. OOP is particularly suited to manufacturing applications, because the data models relate closely to real world objects. Furthermore, object orientation supports the maintenance of models at multiple levels of detail. This is particularly useful in the modelling of a product throughout its development, since the initial model will be far less detailed than the final one.

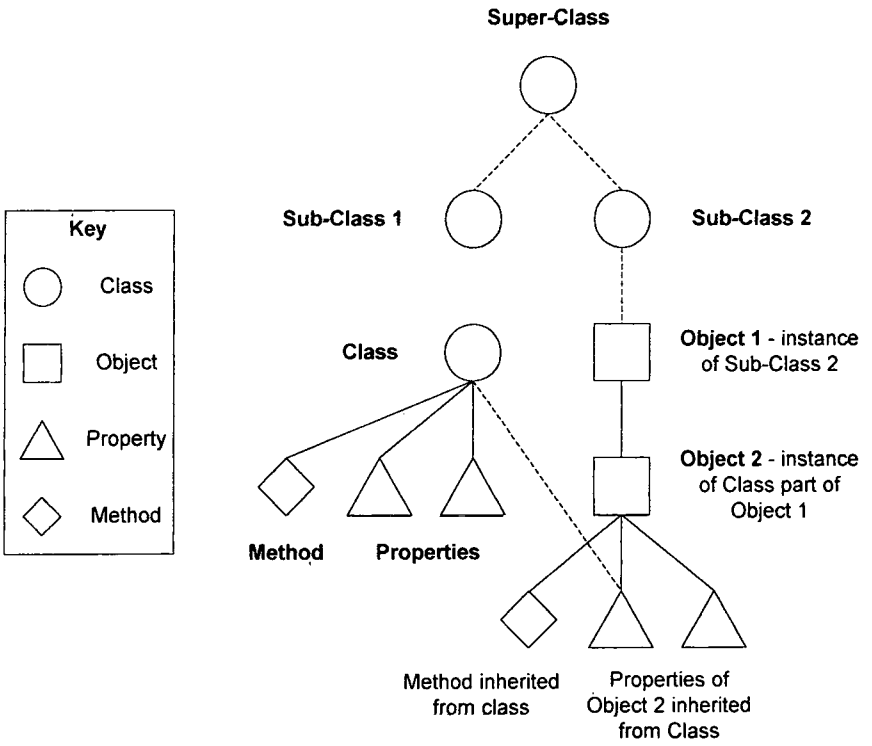


Figure 3-4: Neuron Data Schema for Object Model Representation

In an objected-oriented program the data is stored as objects which are members of one or more type of class. The types of class to which the object belongs determines the functionality of the program and is stored as 'methods' attached to classes. These methods are sets of instructions which are executed by the sending of 'messages' to the object or class to which the method belongs. An object-oriented program operates by sending messages from one object to another, causing methods to be executed which may in turn generate further messages.

An important concept of OOP is 'inheritance', which is the mechanism by which the functionality of the program that is stored in the classes is propagated to the objects created during the program execution. Objects and classes may inherit methods and properties from their parents. Thus, all the functionality which is required to be associated with an object may be assigned through the membership of particular classes. Objects can be members of several classes and have multiple parent objects. Thus, an object-oriented model stores information not simply in the properties of the objects, but in the linkages between the objects and the relationships which are created. The use of multiple classes for single objects gives the programmer a finer degree of control over the system behaviour. This programming method is suited to the generation of product models in particular, because the class of the objects within the model can be changed during the development process so that more detailed methods can be applied to the increasingly detailed product design.

3.3.2 Knowledge-Based Systems

As stated previously, the Nexpert Object language used for the development of the AAMP system is a knowledge-based system engine. A knowledge-based system is a computer program which systematically encodes human expertise in a particular field into a data retrieval mechanism, allowing automated interrogation of the data to solve given problems. Use of a knowledge-based system structure is an approach well-suited to the design of a decision support system, because the process of decision-making can be made transparent to the user so that the reasoning behind each system suggestion can be traced. This enhances the reliability of the computer system because any errors which are made can be picked up.

3.3.3 Hybrid Systems

A hybrid system is one which combines the elements of two or more alternative programming systems. Nexpert is an example of a hybrid system. The chief advantage of this hybrid system is that it allows the flexibility of modelling, and has the ability to generate generic data structures, a characteristic of OOP, with knowledge-based system functionality such as inferencing. This is highly suited to the encapsulation of engineering knowledge such as assembly process planning expertise.

3.4 User Interface

As part of this thesis, the AAMP system is implemented on a UNIX platform using the X-Windows environment to provide a GUI. The user interface is based around a main development manager window which allows access to each of the functions of the system. The functions of the system call up additional windows to provide specific information such as the product model browser and the factory layout browser. These windows are programmed to be modeless, i.e. the program focus can shift to any of several open windows, allowing the system to be used in a non-linear fashion. The window controls are implemented with functions which read the data from the knowledge-base and use it to populate the elements of the windows. The user interacts with the window data and this is then passed back to the knowledge-base which processes the data.

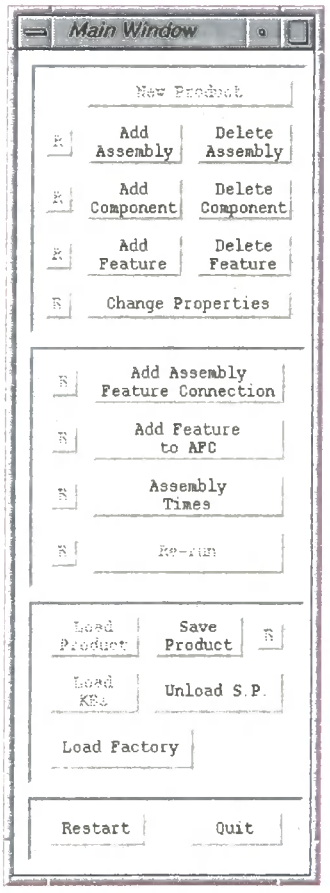


Figure 3-5: AAMP System Main Window

The interface is based around a 'main window', as shown in Figure 3-5, which can be used at management level for: Loading and saving products; loading and editing factory

resources; the addition and modification of products, assemblies, components, features and AFCs; and the remaining system functions, including deriving assembly times, sequences and aggregate assembly process plans. The 'product and resource browser window', as shown in Figure 3-6, is the primary means of analysing the state of the product and factory resource. The browser window consists of a node diagram of the product and resource, with an additional overview window to allow rapid navigation of this area.

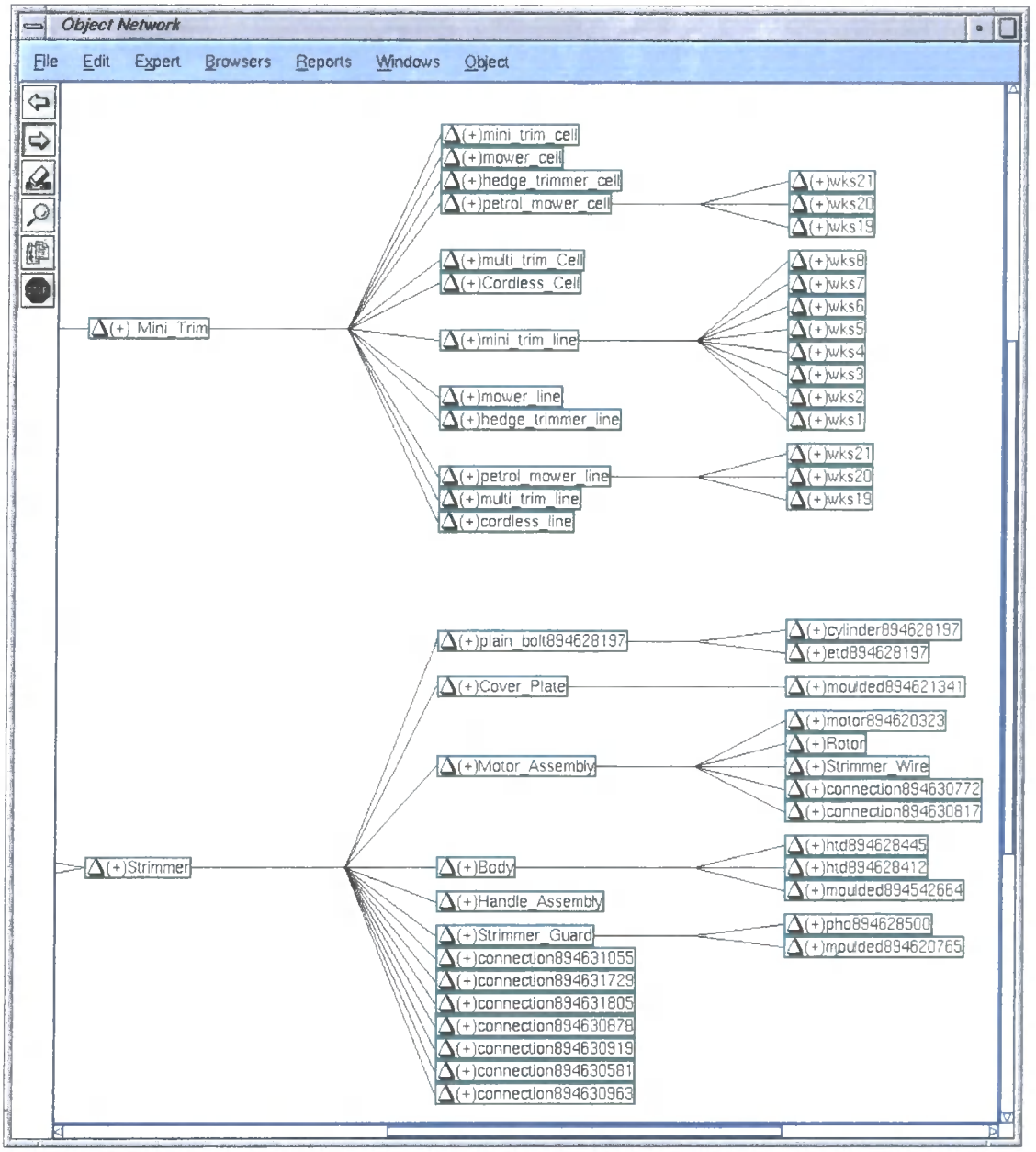


Figure 3-6: Product and Resource Browser Window

Whilst the user interface design is only peripheral to the research objectives of this work, it is, unfortunately, a necessary requirement if the system developed is to be tested properly. A significant amount of work has been done on developing the interface. However, this work will not benefit this project alone, because it is to be used in the testing of other research tools currently under development.

3.5 Summary

This chapter has discussed the overall structure of ideas and the developed AAMP computer system. The system is divided into a number of models which store the data and the functionality of the system. The next three chapters detail each of the models and the functionality of the system in turn, after which, an example of the system as it would be used is given in the testing and results chapter.

Chapter Four

Product Model and Assembly Connections

4.1 Introduction

This chapter describes the aggregate product model, standard part libraries and AFCs which are used by the AAMP system. The aggregate product model has been developed in order to satisfy the requirements of the proposed aggregate assembly process planning function and design methodology. The chapter is split into a number of sections, including the requirements and specifications of an aggregate product model, standard part libraries and AFCs. Secondly, the implementation of the above functions is discussed. Examples will demonstrate the aggregate product model in use with real product assemblies. The chapter also includes a discussion of the implementations of using the aggregate product model and some conclusions.

4.2 Product Model

A product model is a representation of the intended physical product. This model may represent any level of detail that is required at a particular stage in the design process, such as functional or geometrical information. The requirements of a product model will differ from the conceptual through to the detailed stage of the design process in the quantity and quality of information. At the conceptual stage, a product structure is established that can meet the product specification. Detailed geometrical data is not desirable at this early stage. The designer should make some decisions regarding the relative ease of manufacture and assembly of the alternative options.

During the embodiment design stage, the components of the product are designed in more detail, identifying key dimensions. The functional requirements of specific components should be mapped onto the product model. At this stage, a schematic representation of the product can be produced with some geometrical data. Studies into the manufacturability and assembly processes required can be undertaken, together with the use of DFX techniques to establish indeterminate product structure and geometry. However, much of the geometrical data will be assigned later at the detailed design stage. At the detailed final design stage, full part geometry is verified and specified, and final detailed manufacturing and assembly process planning is performed. A solid model is preferred at this stage so that visualisation of details and access checks can be undertaken, as well as the generation of numerical machine code.

4.2.1 Aggregate Product Model Specifications

With an understanding of the information required through the product development cycle, the specifications for an aggregate product model for the earlier design stages can be identified. The requirements of an aggregate product model for assembly include:

- Maintain a structured generic product component model through the design stages. The product model will only store critical information, reducing processing requirements.
- Support for functional representations of designs. An important requirement is that it should support the designer in mapping the function into appropriate design concepts.
- Allow the addition or modification of information to the model at any stage. This implies that the environment should provide modelling tools that are consistent with recognised design vocabulary.
- Support for abstract and implicit representation of incomplete data at the early design stages. Since the modelling of functionality does not require a complete description of geometry, it should be possible to allow for the creation of abstract, or incompletely specified designs. The verification of product data such as dimensioning can be undertaken at the later design stages.
- Support integration with aggregate manufacturing and assembly process planning.

- The model should be assembly-based rather than component-based. This representation provides a functional skeleton for the design, assembly connections and assembly constraints. Detailed component geometry, dimensions and tolerancing can be added later.
- Support for evaluating the design at any stage. Evaluations such as DFMA should suggest changes in component geometries. Such changes should only be allowed if the functionality is not violated.
- Support the integration of standard part libraries to aid the creation of an aggregate product model.

The above requirements lead to the selection of an object-oriented product model which uses feature-based solid modelling techniques to define the product structure. It is felt that this approach will provide the flexibility and ease of manipulation to meet the requirements of an aggregate product model. The product model uses a bill of materials structure, a familar technique used to visualise a product as it is constructed from assemblies and components. Modelling using a feature-based bill of materials structure is similar to the process of solid modelling and hence, the aggregate product model is in line with current proprietary systems.

4.2.2 Structure of the Aggregate Product Model

A product model can be considered as a set of components connected together. Simple products consist of few components, whereas complex products contain numerous components at many levels of sub-assemblies. An important function of the aggregate product model is the representation of the logical grouping of components into assemblies and sub-assemblies resembling the product's bill of material. When seeking to represent the design, a flexible product model is required to allow for change through the process of design. The challenge is to provide a design product model which can represent the design, including the undetermined values, and can perform analysis on this representation, despite only a limited amount of data being available, using the same object constructs.

This leads to a model based on simple geometry, and the most suitable modelling system for this approach is feature-based representation. Features are a natural

representation of a product, and can be categorised into three main types: Functional features (e.g. cylinders); manufacturing and assembly features (e.g. threads or fillets); and aesthetic features (e.g. chamfers). In the conceptual stage, features can represent just the basic requirements of a design, whilst in the later detailed stages, they can represent aesthetics and production features. The three stages of product development are shown in Table 4-1.

Table 4-1: Staged Introduction of Features During Design

Design stage	Features added to product model
Conceptual	Major positive and negative functional features
Embodiment	Minor functional and major production features
Detailed	Minor production and aesthetic features

These three stages of product development are discussed for an example component, one half of the body of a strimmer, as shown in Figure 4-1. At the conceptual stage, the designer is interested in the principle purpose of the case. Thus it can be represented by its key functional features. These are: A moulded body to carry a motor sub-assembly; a slot to locate a switch; a hole to allow access for wiring; and threaded cavities to allow another casing to be joined to it. The aggregate product model consists of these features. At the embodiment stage, the remainder of the functional features are considered, along with the major production features. Some features may fall into both categories. For example, the profile of the moulded body is required both for manufacturing purposes (enhances moulding process) and for functional reasons (reduces weight and increases structural strength). At the detailed stage, the component is fully specified with all dimensions and tolerances. The key elements of the detailed product model comprise of the aesthetic profile and surface finish of the casing body. In addition, the tolerance boundaries on each parameter value have been specified.

The dimensional and tolerance information which is available varies through the product development. There is a gradual introduction of product detail at each stage of the design process. When a feature is first identified, the actual dimension values might not have been determined. By the detailed stage, the tolerances of individual dimensions will have been specified. The product model must represent incomplete

variable data in a coherent manner by specifying boundary limits or using standard tolerance intervals.

Figure 4-1: Strimmer Body

4.3 Standard Parts

Defining a product model is frequently a laborious process undertaken by a designer. Any assistance during this process is advantageous in primarily reducing the time to complete this operation. The use of standard parts in engineering design has been greatly advocated both in design textbooks and by experienced design engineers in industry, Elola *et al* (1996). The concept of standard parts is to utilise the commonalities found in several parts during assembly. Within an assembly, such as an engine, numerous standard parts can be found, including nuts, bolts, gaskets and bearings. The main difference between standard parts and non-standard parts is that standard parts have some pre-specified function in the design. For example, bolts must be used to fasten several parts together and bearings must mate with shafts (or axles) and holes.

The advantages of using standard part libraries are numerous. Their main function is to aid the process of product specification and the derivation of accurate aggregate assembly process plans, whilst limiting the required user input. It is also realised that undertaking numerous tedious operations, such as defining a nut, results in inaccurate data being entered into the system. Assistance with the modelling of standard parts will

leave the designer more time to concentrate on the non-standard parts. The main advantages of creating such databases include:

- Contain all product data such as product structure, part features, critical dimensions, assembly process data, and resources in a required standard format.
- The utilisation of standard parts will decrease the data required from the designer. This ultimately leads to a reduction in the product modelling lead time and cost.
- Standard parts will also minimise computational time and human resource requirements during process planning.
- Libraries will increase the quality of data in the product model, leading to more accurate aggregate assembly process plans.
- Provide data in standard formats such as British Standards (BS) or International Standards Organisation (ISO).

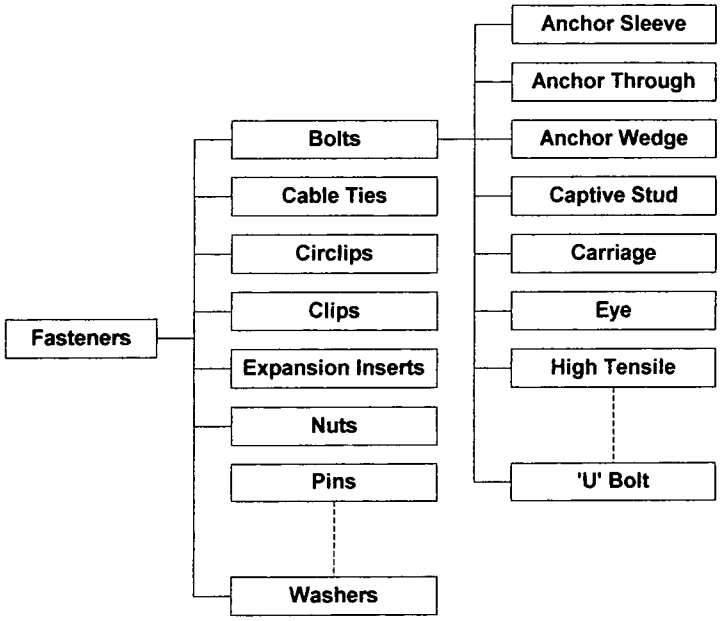


Figure 4-2: The Successive Hierarchy of the Bolt Family

There is another advantage in adopting such a representation of standard parts. Because OOP has been adopted to construct the product model, the advantage of 'succession' can be realised in the management of the various standard parts. Figure 4-2 illustrates the successive hierarchy of the bolt family. The carriage bolt possesses both the

common data of bolts and its specific data. As a result, standard parts can be updated easily and efficiently.

Understanding that standard parts are beneficial to the design process, it is important that the standard part structure and data conform to that of the aggregate product model.

4.4 Assembly Feature Connections

Current CAD systems provide exhaustive capabilities for specifying detailed component geometry in support of the product design process. However, before detailed geometric designs are finally produced, the designer must first map the product requirements into functional specifications. An aggregate product model should aim to support the design process better by consistently representing and maintaining the functional intent of the designer. To provide such support, the product model must address the assembly level, because only at this level is it possible to model the functionality of the design in terms of the significant geometry and assembly connections.

Many current proprietary systems provide a detailed component design capability as support for the design process. The designer first details the individual component geometry. Finally, assembly mating connections are specified to complete the product definition. This is known as bottom-up (component-to-assembly) CAD support. In top-down (assembly-to-component) design environments, the designer should first generate a functional representation of the design, including mating connections. Once this is completed, the designer refines the design by the addition of component geometry. For an aggregate product model, a top-down process is preferred, allowing design analysis and evaluation to be performed, such as DFMA, at a much earlier stage in the design process.

Feature relations provide the aggregate product model with an integrated method for dimension, tolerance and connectivity definition. Feature relations can be added as child objects to features in order to specify additional detail about the product geometry. AFCs provide the system with the ability to model connectivity, and allow the aggregate assessment of assembly process planning and DFA. Assembly connections

are modelled early in the design process, often prior to dimensioning, so it is fitting that the aggregate product model can represent the assembly configuration.

An assembly connection node defines features on two or more distinct components which are linked together by a joint relationship, such as a threaded joint (e.g. nut and bolt), or a placement joint (e.g. block and sheet). The AFC nodes represent the joint that is created rather than the process of creating the joint. Thus, a weld joint could be produced by a number of alternative thermal joining processes. Figure 4-3 shows an example of a threaded AFC node. Both geometric and non-geometric assembly data can be stored attached to the AFC node. Due to the object-oriented nature of the product model, the AFC nodes can also be attached to a class from the classification of assembly connection types. 'Succession' again allows assembly and resource data, and process planning methods to be implied from its parent super-class.

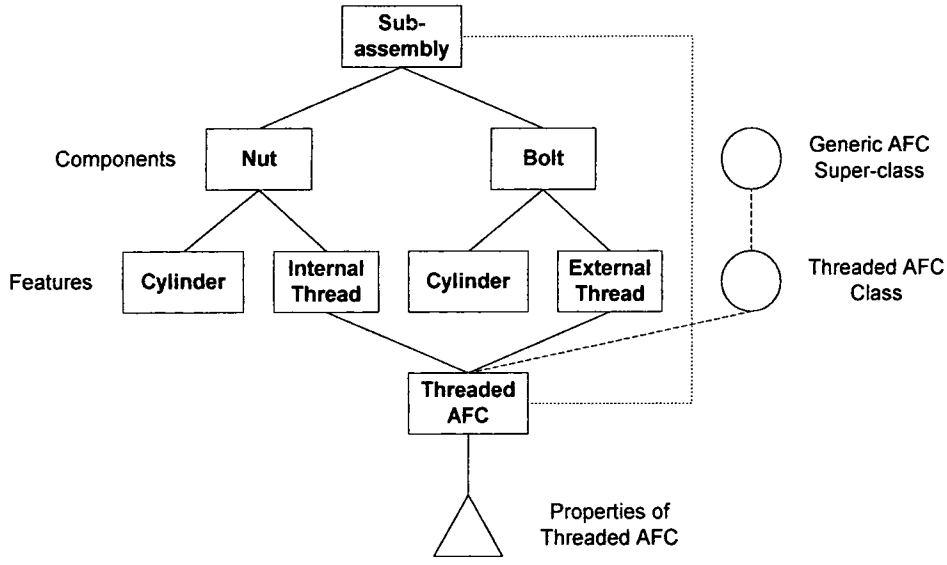


Figure 4-3: An Example Threaded AFC Node

A major advantage of implementing a bill of material top-down aggregate product model which uses AFC nodes, is that it increases the efficiency of the sequence generation algorithm during aggregate assembly process planning. The bill of material feature-based product model can be used to limit the number of feasible sequences generated due to a significant amount of assembly order being derived from the structure of the model. For example, it can be assumed that all assembly joins for a sub-assembly should be undertaken prior to the assembly joins at the parent sub-assembly.

Attaching the AFC node to both the features that make up the join, and at the parent assembly level which both features belong to, as shown in Figure 4-3, also limits the number of generated sequences, because a hierarchy is created when the assembly join should be undertaken. The sequence generation algorithm implemented in the AAMP system is discussed in greater detail later in chapter six of this thesis.

4.5 Implementation of the Aggregate Product Model

As discussed in the previous chapter, the generic aggregate product model and class structure that is implemented in the AAMP system was initially developed by Bradley for a Concurrent Engineering support system known as CAPABLE (Maropoulos *et al*, 1998). The extra functions required for AAMP were subsequently developed by the author of this thesis.

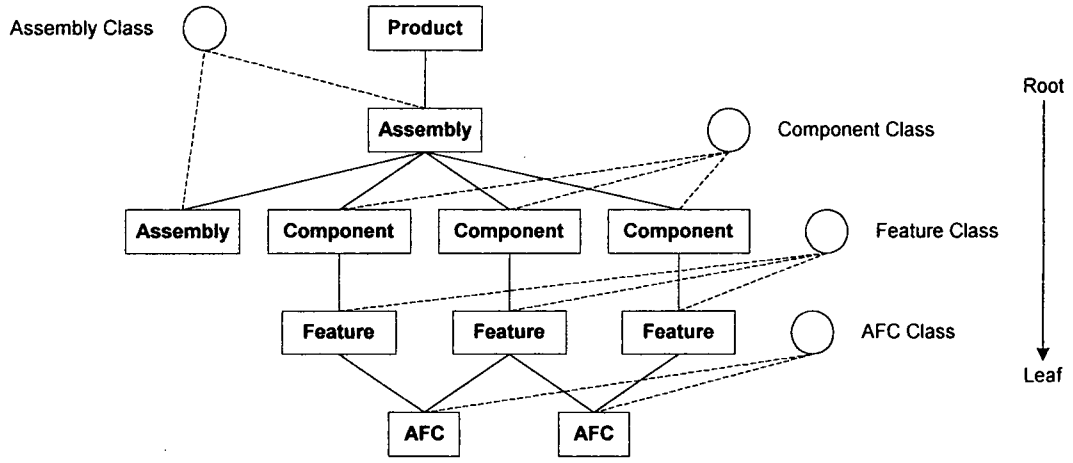


Figure 4-4: Object Product Model

In this section the implementation of the aggregate product model will be discussed with reference to simple examples. Instances of each class of the model are presented, and the attributes and functionality which are associated with the object classes will be outlined. Each product modelled with the system is made up of a hierarchy of objects which are instances of a variety of different classes. Each different class represents an increasing level of information as the tree is traversed from root to leaf nodes. At each level of the tree, the siblings of an object will be of the same generic class, although there may be instances of different specific classes. For example, components are made up of many feature objects which are all instances of specific feature classes within the

generic feature super-class. Figure 4-4 displays a simplified example of the object product model. The following sections describe each class of the aggregate product model.

4.5.1 Assemblies Class

Although a product is basically a collection of components connected together, assemblies and sub-assemblies are used to define a product into a logical product structure. This bill of material representation of the product aids the modelling process, presents a clearer view of the product structure, and assists subsequent operations such as assembly process planning.

The assembly objects form root nodes in the product tree and can have either sub-assemblies, components or features as child objects. Sub-assemblies and components are the most typical children of the object. Properties of the assembly class include geometric values such as size and weight. Its other properties relate to the assembly functions of the AAMP system, including handling and orientating assembly data. Values for these properties are either entered at the time of creation, or calculated at a later time during process planning activities. Information entered at the time of creation comprises the name of the assembly, number of such assemblies and parent object; and data calculated at a later time includes size, weight and number of child components. In addition, an instance of the assembly class can be the 'product' which is currently being modelled. The product is defined here as the completed part which is sold to the customer. The product class allows the definition of additional functionality to the product assembly object.

4.5.1.1 Assembly Creation

A GUI window was coded to assist a user when creating assemblies. This window, as shown in Figure 4-5, allows the user to enter the name and quantity of the new assembly. A pull down list box displays all the existing assemblies to which it could be attached. The assembly object is created as a child to the selected assembly parent object, and also attached to the generic assembly class.

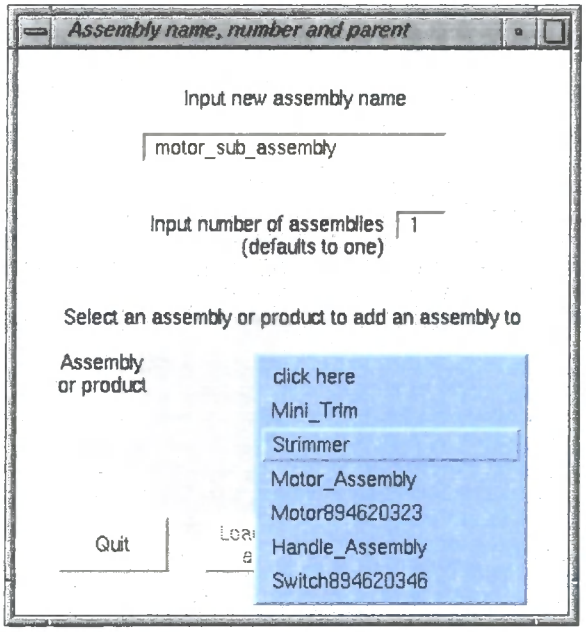


Figure 4-5: Assembly Creation GUI

4.5.2 Components Class

Component objects are the building blocks used to define a product, and usually belong to assembly objects. These discrete parts are usually created from a single piece of material. The component class objects represent the basic information of the part, and stores the sum of the properties of its child features. The difference between individual component types is represented at the feature level. Properties of components include some basic geometric information such as weight and volume. Other attributes include material, quantity, and handling and orientating assembly data. The detailed geometry of the component is stored at feature level.

A component class can only have features as children. These are divided into two types, positive and negative features (Bradley, 1997). Each component has one positive feature, which defines the overall geometry of the part. Examples of positive features include prism, cylinder and sheet. The geometry of these positive features is refined through the addition of negative features. Negative features define the material to be removed from the positive feature in order to generate the component shape. Positive and negative features are defined by using separate class structures, as discussed in the following sections.

4.5.2.1 Positive Feature Class

As previously stated, the positive feature defines the overall shape of the component, such as a cylinder, prism or moulded part. Positive feature objects can only exist in the model as child objects of components, and each component has one, and only one, positive feature. The only child objects a positive feature can possess is a feature relation which will be discussed later in this chapter. Positive features can be of six basic types: Prism; cylinder; sheet; solid; moulded part; and wire. Attributes of the positive feature depend on which is selected. They all share a basic array of properties, including volume and weight, but specific geometric properties such as diameter and length, are inherited from their positive feature parent class.

4.5.2.2 Negative Feature Class

The geometry of the individual components is constructed by the addition of negative features to the positive feature which describes the basic shape of the component. In the aggregate product model, a negative feature is defined as "individual geometric characteristics of a solid part, the sum of which makes up the full geometry of the part" (Bradley, 1997). Examples of negative features include holes, threads, chamfers, slots, profiles and so forth. This approach to defining the feature geometry is the major difference between this model and most feature-based models. The feature classes are not defined with a fixed and limited set of geometry information which must be specified in order to store the feature within the model. Instead, the system seeks to allow the user as much flexibility as possible in the definition of the geometry. This leads to the adoption of a two-layer data model for the representation of features. An additional class of objects is defined, called feature relations or connections, which allows the geometry and connectivity to be stored and modified.

The feature relations have been proposed as a means of solving the problem of tolerancing and assembly modelling in solid models. Feature relations are an integrated schema for the representation of feature characteristics, including linear and geometrical dimensions and tolerances, and also component connections within assemblies. A classification of feature relations has been developed for dimensioning and tolerancing in accordance with the industry standards (British Standard 308:3).

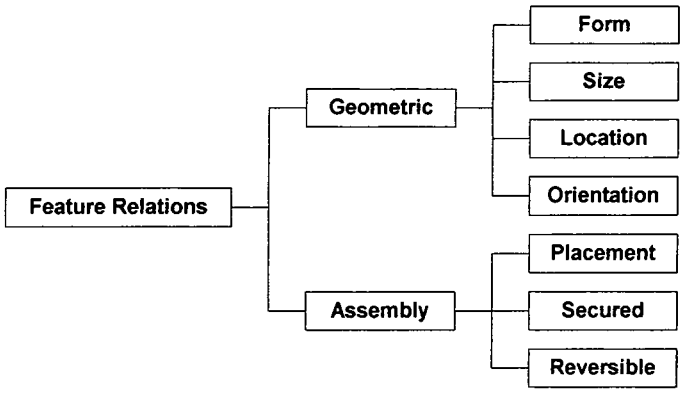


Figure 4-6: Feature Relation Classification

4.5.2.3 Feature Relation Class

Feature relations are an integrated method for dimension, tolerance and connectivity definition. Feature relation objects can be added as child objects to feature objects to connect two such objects together and specify additional geometry detail. Feature relation objects belong to one of the leaf nodes in the feature relation hierarchy. The top levels of the feature relation classification are shown in Figure 4-6. In the product model, geometry feature relation objects can be used to represent simple geometry such as tolerancing data, as well as more complex specifications such as concentricity and flatness. The classification of AFCs is used to define the way in which individual components are connected together to create assemblies and fabrications. The implementation of these feature connections is discussed later in this chapter.

4.5.2.4 Feature classification.

The class of features can be divided into many sub-classes based on the characteristics of the individual features, (Figure 4-7). The full set of positive and negative features used in the aggregate product model can be found in Appendix A. Figure 4-8 displays an example of a Jacquard lift arm component, displaying both positive and negative features. The component's positive feature is a sheet, and the negative features include through holes, blind holes and internal threads. Dimensioning and tolerancing feature relations would be used to specify the location of the negative features on the product model.

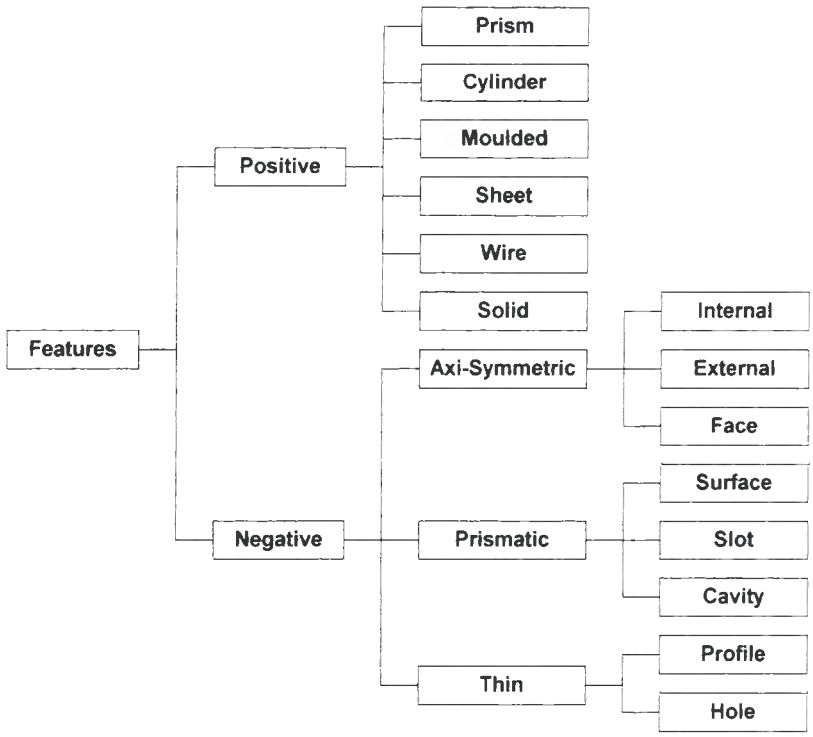


Figure 4-7: Feature Taxonomy

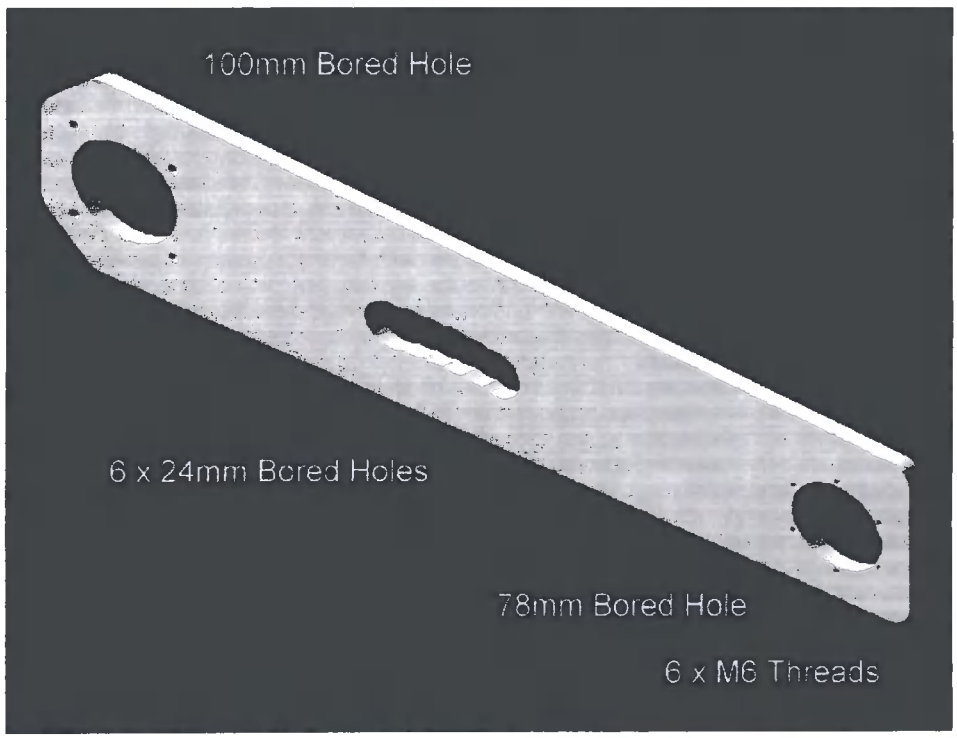


Figure 4-8: Jacquard Lift Arm Component

4.5.2.5 Component and Feature Creation

GUI windows were coded to assist a user creating components and defining features. The component creation GUIs allow the user to enter the quantity and name of the new component, and a pull down list box displays all the assemblies to which the component can be attached, allowing the user to select one. Another pull down list box allows the user to derive the main positive feature type. Then the user is asked to enter the overall dimensions of the positive feature, dependent on which positive feature is selected. A final GUI, as shown in Figure 4-9, allows the user to specify component features that will present handling difficulties. The component object is created as a child to the selected assembly and also attached to the generic component class.

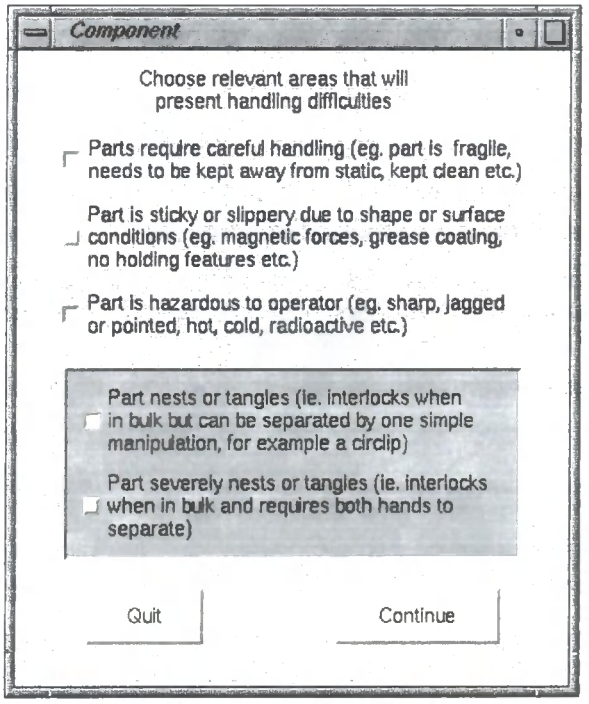


Figure 4-9: Component Creation GUI

Negative features are defined in a similar way to components, the user selecting the component to which the feature is to be attached, and the feature type. Dimensions are entered with the option of inputting tolerances. Positive and negative features are attached as a child object to the component, and are also attached to their own feature type super-class.

4.6 Implementation of Standard Part Libraries

In this section, the implementation of standard part libraries will be discussed with reference to simple examples. The classification, structure and information included in the libraries are presented. As discussed previously, standard part libraries will contain product model and assembly data to assist the design process. As only the assembly planning and scheduling processes are being considered, only data relevant to this is required for the AAMP system. However, manufacturing data would probably also be entered into the libraries in a commercial package to aid decisions such as make or buy.

A requirement of the part libraries is that the data conforms to that already employed in the aggregate product model. This includes the product structure, including all connections to parent objects and classes. Secondly, the type of data and information held for each object is included. Finally, it includes the definition of relevant AFCs. This will ensure that a standard part can easily be loaded into the aggregate product model and be instantly ready to use. A further requirement of the libraries is that they only hold required data, thus limiting the amount of information. For example, if an electric motor is considered as a standard brought-in part, as shown in Figure 4-10, data is required on the shape, size and assembly for the body, axle, securing points and power connectors. However, data is not required on the internal structure, components and workings of the motor. This notion reduces the amount of data stored in standard part libraries. The main benefits of reducing the amount of library data includes: Reducing the time to develop standard libraries; reducing the size of the standard part libraries, and aggregate product models and files; and reducing the amount of processing required during aggregate assembly process planning.

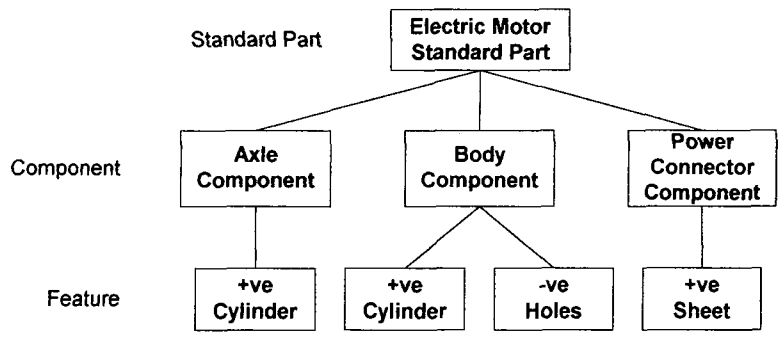


Figure 4-10: Electric Motor Standard Part

Investigating the electric motor standard part further, it can be seen that the complete motor makes up a sub-assembly, with the body, axle and power connector components attached as child objects. The positive feature of the body and axle are cylinders, and the positive feature of the power connector tabs are sheets. The only negative features in the model are the through holes on the body, which function as securing points. Only required standard part data is attached to each object of the motor in the same manner as a normal product model. The main data includes: The dimensions and weight of the complete sub-assembly; the dimensions of each component and feature; and assembly processing data. The assembly processing data is comprised of handling and orientating information, and intelligent suggestions for the AFC type of key joining features. It would be assumed that for the motor, a pulley or gear would be attached to the axle, and that a power source would be attached to the electrical connectors. The part library therefore contains data suggesting that an AFC to the axle would be a plug and target AFC, and to the power connectors a wire tab AFC. These suggestions are additionally confirmed by the user during the definition of AFCs.

Prior to designing standard part database libraries, a process of information gathering has to be undertaken to define the classifications and required data. The majority of data available is found in dedicated supplier databases and product catalogues, e.g. the R.S. catalogue. Standard parts have been classified within four broad types, as shown in Figure 4-11: Mechanical; electro-mechanical; electrical; and general. Each type can subsequently be broken down to sub-classifications, including fasteners, fixings, capacitors and switches. For the prototype system, a selection of parts libraries were created to demonstrate their aid to the designer. Within mechanical assemblies, the main joining operations performed are comprised of fastening by screw or nut and bolt, riveting, pressing and welding. Indeed, threaded fastening and riveting alone constitute over sixty per cent of all assembly operations (Martin-Vega *et al*, 1995). With these facts in mind, and considering that their function, structure and shape are similar, threaded fasteners are an ideal product for which to develop standard part libraries.

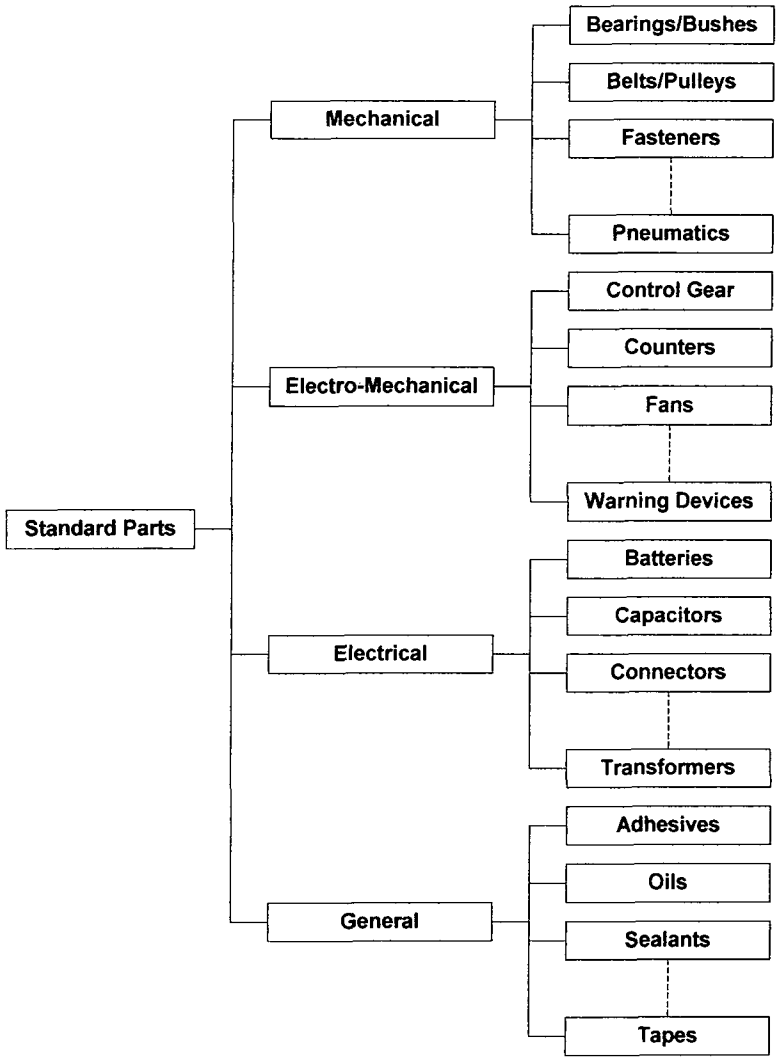


Figure 4-11: Standard Part Classification

Figure 4-12 shows a selection of standard bolts, all designed to perform a different task, but all with the same shape and product features. Libraries were initially created for a selection of nuts, bolts, washers and rivets. Subsequently, libraries were created for the example components used during testing, such as electric motors, switches and capacitors.

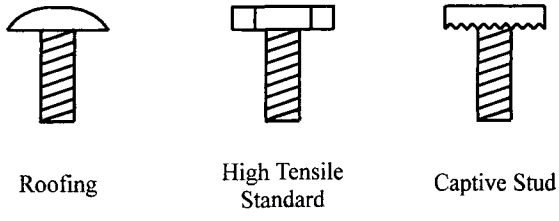


Figure 4-12: Example Standard Bolts

A GUI was coded to assist the user loading standard parts. This window allows the user to select the class, type and model, as well as the size of the standard part, from a selection of pull down list boxes, as shown in Figure 4-13. It also allows the user to select the assembly to which the standard part should be attached on the aggregate product model.

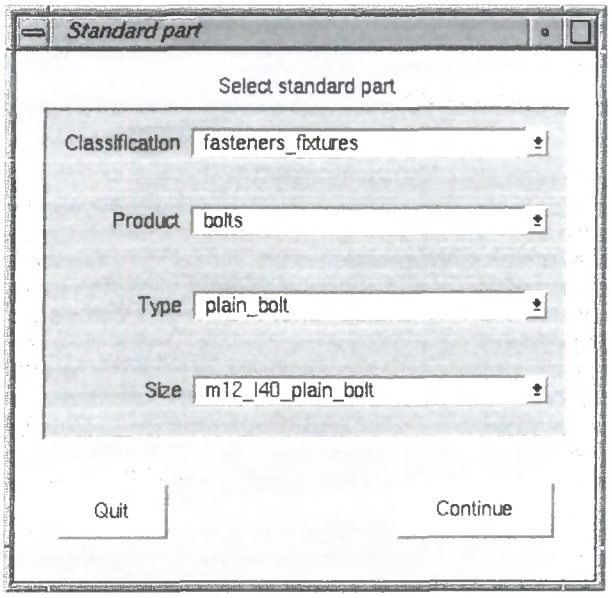


Figure 4-13: Standard Part Loading GUI

4.7 Implementation of Assembly Connections

As discussed earlier in this chapter, AFCs are used within the product model to define assembly connections and allow the aggregate assessment of assembly production. In this section the implementation of assembly connections will be discussed with reference to simple examples. The classification of the connections is presented and the attributes and functionality which is associated with the connection class will be outlined.

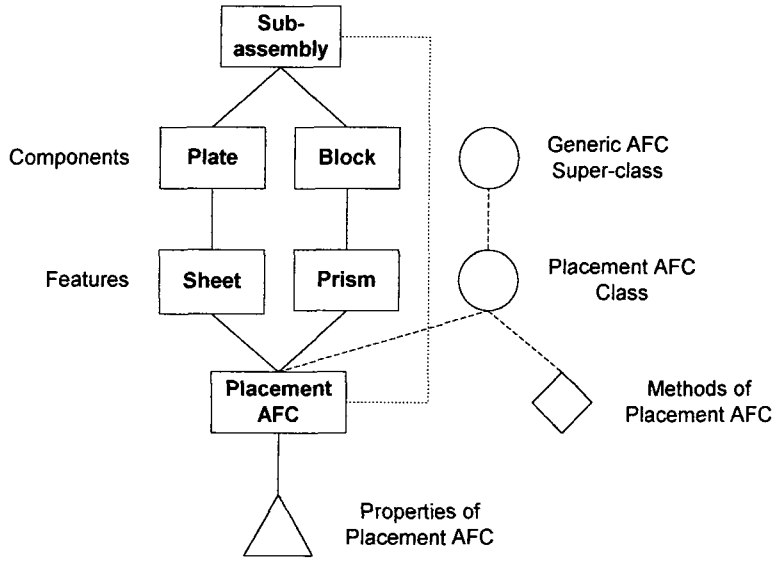


Figure 4-14: Example Placement AFC

4.7.1 Assembly Connection Classification

An assembly connection node defines features on two or more distinct components which are linked together by a joint relationship. These connections are classified according to the type of connection, and represent the physical link between the features. Figure 4-14 displays an example placement AFC node and the objects and classes to which it is joined. The example connection is attached to the two features as well as their parent assembly, the connection type class and the AFC super-class. The object properties and methods of the connection object vary depending on which connection type is selected. All connections share a basic array of assembly process properties, handling and orientating data, resource data, assembly times and parent objects. They also inherit specific process and geometry data, such as pitch and length of threads, and assembly process time calculation methods, from the parent connection class.

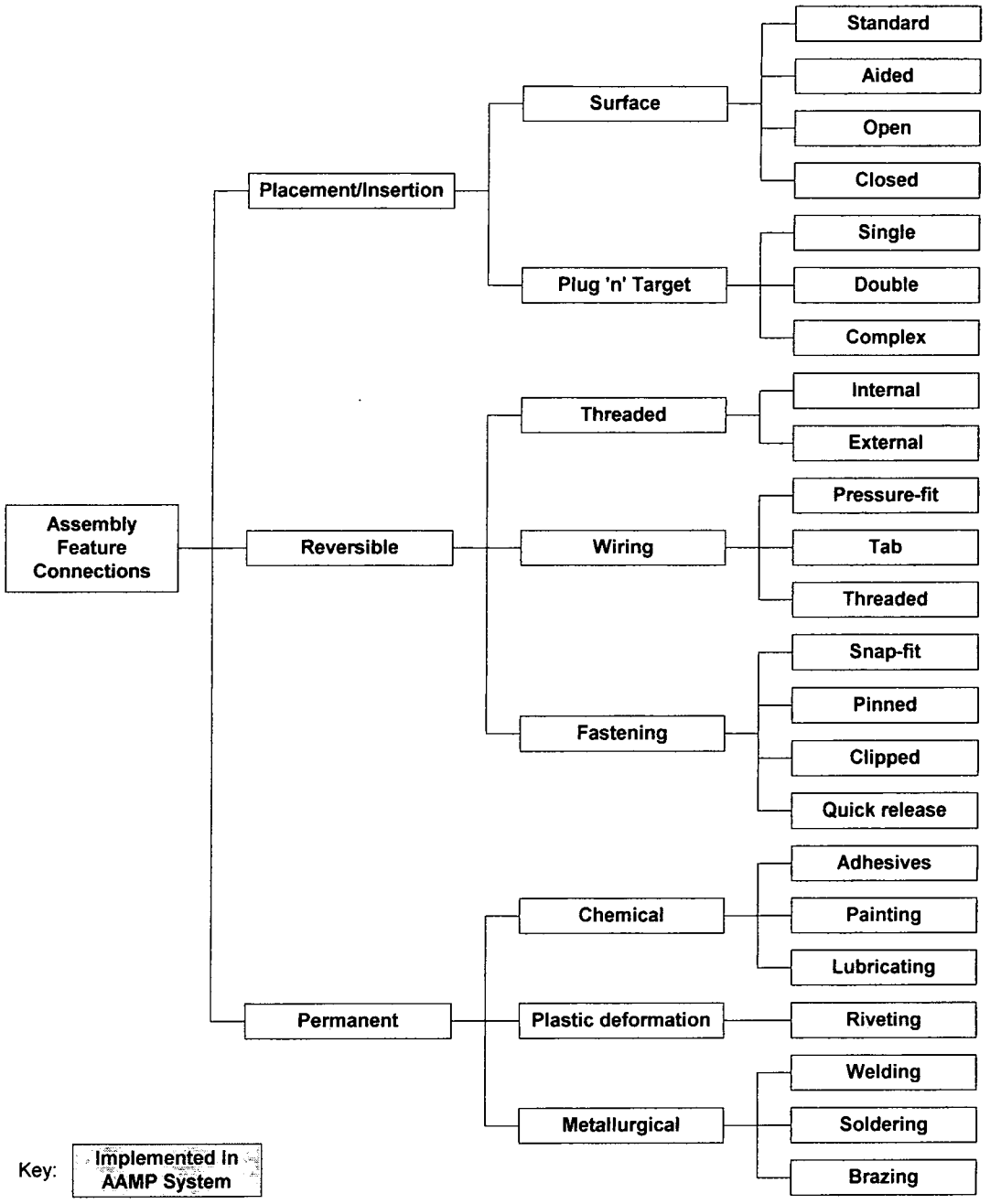


Figure 4-15: AFC Classification

Prior to developing the functionality associated with AFCs, a broad classification was developed. AFCs were initially divided into three main types: Placement and insertion (e.g. surface placement); reversible (e.g. threaded); and permanent (e.g. chemical). The classification was further broken down into sub-classes as shown in Figure 4-15. Sub-classes of the placement and insertion class include a range of placement and plug and target AFCs. Placement AFCs are general pick and place processes, whereas plug and

target AFCs involve placing a component in some form of hole or recess. Sub-classes of reversible connections include threaded, wiring and an array of fastening AFCs (e.g. clipped, self-fasteners and snap-fit). Threaded connections are further broken down into internal and external sub-classes. Sub-classes of permanent connections include: Metallurgical (e.g. thermal); chemical (e.g. adhesives and solvents); and plastic deformation (e.g. riveting).

During this research, connection classes were created for a range of join types to test the different AFC's functions and methods. These classes included: Placement; plug and target; threaded; riveted; wiring; and snap-fit fasteners. The frequency of these types of connection in a mechanical assembly, or an electronic assembly operation, is greater than seventy per cent and fifty per cent respectively (Martin-Vega *et al*, 1995).

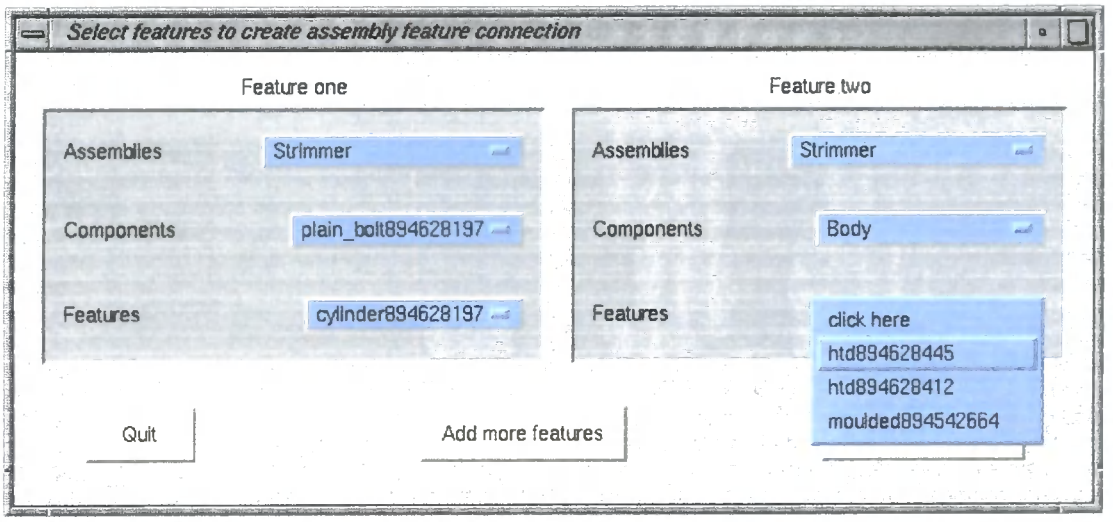


Figure 4-16: Feature Selection GUI

4.7.2 Assembly Connection Creation

The process of creating and assigning information to AFCs is divided into three stages: Selecting features to join; deriving an assembly connection type; and inputting relevant assembly process data. To aid the creation of assembly connections, a selection of GUIs were coded for the above functions. Initially, the user selects the features to be joined together from pull down list boxes, as shown in Figure 4-16. A minimum of two distinct component features are required to create a connection. A basic rule is included in the AAMP system, ensuring that the user does not select features from the same

component; once a feature is selected, the user is unable to select another feature from the same component.

The AAMP system then attempts to automatically derive the connection type. Each feature has a list of probable connection types stored as a property, and the system attempts to derive if a link exists between these lists. If a match does exist, this is presented to the user, who has the option to accept the connection type. If more than one match exists, the user is presented with all possible selections from which to choose. If no match exists, or an unsuitable match is presented, then the user selects a connection type manually. An example scenario is the assembly of a nut and a bolt. The part features to be connected are an internal and external thread. Obviously, a screw tightening process is a probable connection type for threaded features, so the system derives that this is the most suitable connection type, which the user accepts.

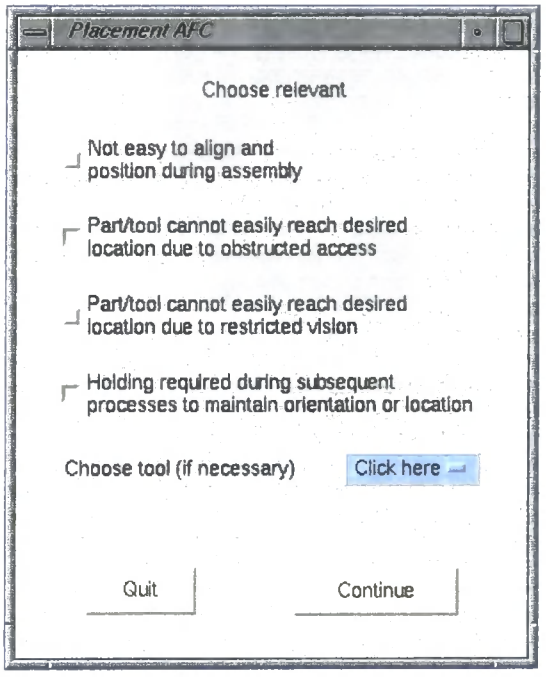


Figure 4-17: Data Inputting GUI

The final step in the operation of creating an assembly connection object is inputting assembly process information into the system. The data is required to compute accurate process times and costs and, ultimately, valid assembly sequences and aggregate assembly process plans. The data requested by the system varies on the selected assembly connection, and includes both assembly process and resource data. For each

connection type, a GUI is presented to the user to aid the input of information. An example GUI for a placement AFC is shown in Figure 4-17.

4.8 Manipulating the Product Model

This section will discuss the various functions of the system relating to the manipulation of the product model by the user. In addition, the requirements for transfer of data between other systems and the AAMP system will be discussed. In order for the AAMP system to be integrated into a real engineering design environment, it is essential that information be retrieved automatically from proprietary design software systems which are currently employed. It is not the aim for the AAMP system to be a dedicated CAD tool in the traditional sense. This task is best performed by the innumerable CAD systems which are currently available. In particular, the functionality of CAD systems, such as Pro/Engineer from Parametric Technology Corporation, Euclid from Matra Datavision, and CATIA from Dassault/IBM, makes them far more suitable for formulating the product design. What is required is a means of taking the product data from such as these and extracting the information required to create the AAMP aggregate product model.

Since there are many different CAD systems available on the market, each using a different format for modelling the product both within the CAD system and in stored data files, it is not feasible for a system such as the AAMP system to be developed to read and manipulate the data models of all other available systems. For this reason the Standard for the Exchange of Product Data (STEP) is being developed for ISO to facilitate the transfer of product models from one system to another. This, theoretically, enables software developers to retain a proprietary modelling system, whilst providing cross-system compatibility through a standard file format, thus avoiding the need to write a translator for each system with which they might wish to share data.

However, because of the vast number of different uses to which the product data is put, and the requirement for the standard to be generic, the development of a complete, workable standard is not progressing sufficiently to allow widespread use. In most fields of engineering, the standard is still in its infancy, with little take up in industry.

The exceptions appear to be in the aeronautical and automotive industries, which were the main drivers in the STEP project.

4.9 Conclusions

This chapter describes a flexible aggregate product model which can represent data over the early stages of product development. All the information which is required for aggregate assembly process planning and design assessment can be stored in the model. This includes: Product structure; assembly and sub-assembly groupings; component geometry via specific features; and feature relations for dimensional, tolerance and assembly connectivity. Standard part libraries have been designed to aid the user to create the product model, and AFCs have been included in the model to allow the representation of assembly joins. The product model has been designed to be compatible with the emerging STEP standard to enable rapid data transfer from solid modelling CAD environments.

With a suitable aggregate product model, it is possible to analyse the production and assembly options which are available, and to make suggestions as to the best production route. Alternatively, the processing information can be used as a design feedback to alter the design in order to produce a product which is cheaper or quicker to make, or can be made to a higher quality. In order to achieve this aim, it is necessary to develop a model of the production and assembly processes, and how they are applied. A detailed description of the assembly process model developed for this purpose is given in the next chapter.

Chapter Five

Assembly Process and Resource Models

5.1 Introduction

This chapter details the work relating to the process and resource models. The process model embodies expert assembly knowledge which allows the system to perform automated aggregate assembly process planning and product evaluation. The resource model represents the tools, equipment and facilities available to the company to assembly the product. Process and resource models are a prerequisite for aggregate assembly process planning, which considers both the processes and the equipment which can be used for assembly.

The overall aim of the process model is to rapidly generate assembly times, sequences, plans and costs, using limited information available during the early product design stages. This is vitally important in order to apply process planning considerations within Concurrent Engineering. The aggregate planning objectives are the early identification of assembly constraints and bottlenecks, and the definition of best product configuration and assembly methods. The most suitable product and process plans will be retained, and will form the basis of full detailed and optimal plans which will be created by using detailed process planning systems.

The process model consists of a hierarchical taxonomy of individual aggregate process models, and together with the attendant architecture and functionality, allows them to be used for aggregate assembly process planning. The philosophy of aggregate process

models is that they should provide the means of making adequate predictions about assembly operations with either uncertain, or incomplete, knowledge. The aggregate process modelling of assembly processes involves the translation of product design data into initial assembly planning information. Ideally, these activities should be performed as early as possible in the product development cycle, because then there is a wide range of options both in terms of product configuration and process selection.

Aggregate process models are obtained by the controlled simplification of detailed process models so that they can function using only the limited product information available during the conceptual and embodiment design stages (Bradley, 1997). Aggregate process models are simplified descriptions of capabilities, requirements and parameters of assembly processes which allow aggregate process planning to be executed. Any such simplification will almost inevitably result in a loss of accuracy in the associated assembly planning predictions. This drawback is outweighed by the ability to rapidly evaluate alternative product configurations and processing options at an early design stage, so that best design options can be developed later. The assembly process model contains a comprehensive classification of aggregate assembly processes. Due to the limitations of time, however, aggregate assembly process models have been developed for only a selection of assembly processes, specifically, the high frequency assembly operations such as pick and placement, snap-fit fastening, screw tightening, wiring and riveting.

Resource information which is required for aggregate process planning includes both equipment and organisational data. The resource model is hierarchical in structure, based on the concept of factories; a factory is a production unit which consists of a number of manufacturing cells. Within the factory, information on cells, workstations, process equipment, transportation, labour resources and storage are modelled. The resource model supports the use of multiple factories, which can represent either alternative locations for assembling a product (useful for make or buy decisions), or alternative configurations of the same location (useful for facility design). The resource model allows users to customise the system to suit their own requirements. A generic modelling scheme has been developed which can apply to any factory system. The

model structures are populated with data about the resources present in the particular factory.

5.2 Principles of Aggregate Assembly Process Models

In the development of the aggregate assembly process models, a number of principles have been applied. These principles, initially identified by Maropoulos (1995b), provide a specification for the process models in the system, and are discussed in this section.

- *Controlled simplification of detailed process models:* Complicated process models utilised by detailed process planning systems are unsuitable for aggregate process planning. These models require too much data and too much computation to produce results. Therefore, it was necessary to simplify the process models so that the core function is retained, whilst the unnecessary processing is eliminated. It is important, however, that this process is controlled, so that the information that is retained still provides an accurate assessment of process performance and results.

- *Limited data input requirements:* Of key importance to the aggregate process models is that they require a limited set of data inputs. This is necessary if the models are to be used in the conceptual and embodiment design stages, when full data is not yet determined. The aggregate process models should incorporate only the basic geometric information of the component parts and features, without requiring the specification of more detailed aspects which might not be determined until the detailed design stage.

- *Model assembly operations:* The process models should allow the automated aggregate process planner to model the assembly operations as they would be carried out on the shop floor, so that production routes may be passed to the process planning engineers for further consideration. Additionally, assembly process plans at this level allow a contribution to be made to factory layout design and production management.

- *Measure assembly performance:* For each process, it is necessary to measure a key set of assembly performance indicators. These should include the cost and delivery

implications of the process. Provision for both the handling and processing assembly times must be included in the calculation of cost and delivery.

- *Perform core capability checks:* The aggregate process models must retain the ability to check the most important assembly process constraints, so that processes are not suggested which are clearly impossible from the information available. However, detailed capability checks that require long computational detail, and those constraints that rarely are broken, can be omitted. An example of a check which could be made is to ensure that the dimensions of a threaded operation correspond, whilst full geometric checking for robot path interference is best left to detailed systems such as Computer Numerical Control (CNC) code generators.

- *Utilise company-specific knowledge:* The assembly process models must take into consideration the individual characteristics of the particular company's product and process knowledge. Also, the process models must incorporate inputs from the factory resource model so that the processes are evaluated according to the facilities of the company.

- *Function-driven operation:* The process models should be oriented towards providing the necessary functionality for aggregate assembly process planning. An object-oriented approach, with encapsulated methods for interacting with the process models, is the preferred structure.

- *Conformance with team-based engineering:* The process models must support the conformance of the overall system to the team-based approach of Concurrent Engineering. This means that the models should be accessible to, and usable by, developers from all disciplines within the company, and not restricted to use by process planners through over-complexity or the requirement for process planning expertise.

5.3 Assembly Process Model Overview

The assembly process model consists of a generic classification of assembly process types, which cover all forms of joining and fastening. Assembly processes are used to join components and sub-assemblies together to ultimately create a final assembled

product. Assembly operations involve the collection of components and placing them in their required location. Generally, a subsequent securing process is necessary. The securing operation can include both reversible processes (e.g. threading and snap-fits) and irreversible ones (e.g. riveting and soldering). Assembly processes can be classified into these two groups: Processes that only involve the collection and placement of parts together; and processes requiring a subsequent operation. The higher levels of the assembly process taxonomy are shown in Figure 5-1.

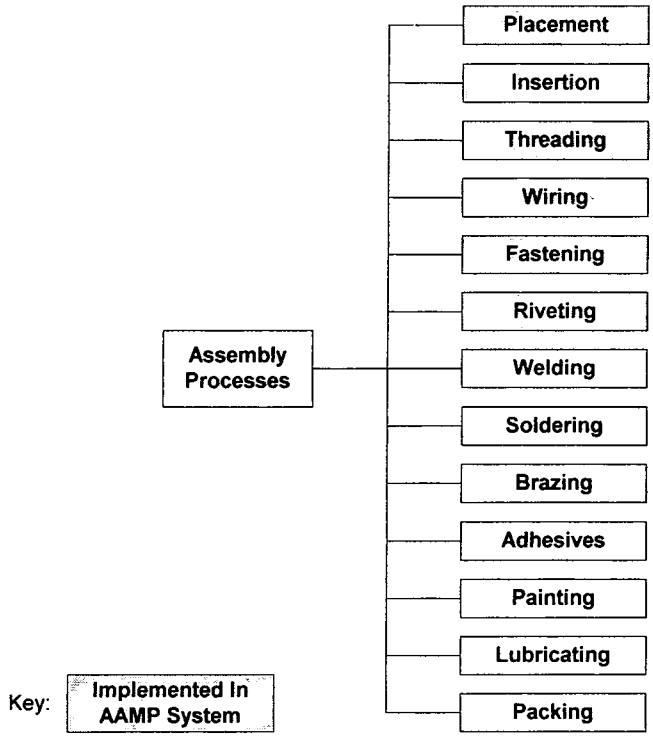


Figure 5-1: Assembly Process Taxonomy

In order to build useful assembly process models, decisions must be made about the level of detail to which the process will be represented. It is quite possible, and indeed common, for research into specific process optimisation to model processes in great detail. However, this is clearly inappropriate for a system which aims at aggregate planning information that covers a wide process range. On the other hand, some process models tend to over-simplify the representation of the process, so that important capability checks and parameters are not considered.

Each process class within the model must contain a set of parameters common to all classes that are used in the generic process planning functions. In addition, the process

classes have a set of methods which define the specific process planning knowledge. The principle method required is the processing time method. The aggregate assembly process planning function is based on a criterion which is derived from the processing time and transportation time. The system must be able to calculate a processing time for each of the process options, and this is done by using processing time methods. This factor has been considered in the design of the system's product and process models so that a generic method could be devised where possible for each assembly process.

For each assembly process type, a method is used to calculate the processing time for all operational elements, which is defined as the processing involved in undertaking an AFC. In the AAMP system, assembly process models have been developed for the implemented AFCs, documented in the preceding chapter. Therefore, discussion will centre on placement and insertion, threading, snap-fit fastening, riveting and wiring assembly process models from the overall operation classification.

The placement and insertion assembly process model is further divided into surface, and plug and target process models. Surface placement processes encompass basic placement operations, guided placement operations, and placement operations into open and closed slots or grooves. Plug and target processes are defined by placing a part into a recess, and include single cylindrical, single non-cylindrical, double and multiple plug and target operations. The threaded process model incorporates both internal (e.g. a screw into a block) and external (e.g. a nut onto a bolt) threaded operations. Wiring assembly operations are further broken down to include tab, threaded and pressure-fit wire connectors.

The primary benefit of classifying processes into a hierarchy, instead of using a simple flat structure, is the ability to use the concept of 'inheritance' to share functionality amongst similar process models. This reduces the amount of programming required and the size and complexity of the programs developed. Thus, in the model chosen, all the threading processes may be given the same basic set of attributes by defining the super-class threading, and linking each detailed threading class as child objects.

5.4 Processing Time Algorithms

This section describes the derivations of the generic process time models for the assembly processes implemented in the AAMP system. The process time algorithms are based on aggregate process models, which operate when the full planning details are not available and yet a process time estimation must be calculated. Naturally, this estimate should be as accurate as possible. To achieve this, the models use calculations based on product model geometry, assembly resource, and process data where possible.

The Boothroyd and Dewhurst assembly time standards were derived as a result of extensive experimental studies performed to measure the effect of product geometry, size, weight, ease of manipulation, access and vision on manual handling and insertion times. The results were published in the form of a Product Design for Assembly Handbook (Boothroyd and Dewhurst, 1987), and subsequently as a software package in 1992. These time standards are part of a structured design analysis method which guides the team to a functional, simplified product structure which is straightforward to assemble. The method also aims to reduce the part count and identify difficulties that may hinder the assembly process or affect the quality of the product. Indeed, there have been numerous published examples of successes obtained with the Boothroyd and Dewhurst's DFA method.

A drawback with the Boothroyd and Dewhurst's software package is that it was designed as a question-and-answer standalone system. With no link to a product model, the system requires a large input from the user to gain any form of design analysis. Another disadvantage of the approach is that the assembly process times (e.g. for riveting, screw-tightening and welding) do not take into consideration the product geometry or resources being utilised. An example of this problem can be seen when deriving an assembly process time for welding from the handbook. The limited method only gives one time, and does not consider the materials to be joined, the length of weld, or the welding process employed. Such generic process times remove the possibility of analysis using different tools and machines, or even state-of-the-art resources.

Three methods of calculating assembly process times have been implemented in the AAMP system to give the desired accuracy of results. Times are obtained from standard assembly time databases, calculated using assembly process equations, and procured from standard assembly process rates. The manual pick and place (handling, manipulation and insertion) elements of Boothroyd and Dewhurst's assembly time standards are implemented in the AAMP system. However, to increase the validity of the results when a frequent subsequent assembly process element of the assembly operation is necessary, the author of this thesis developed methods to calculate the process time using data from the product and resource model. The next sections discuss the time algorithms for the implemented process models.

5.4.1 Standard Assembly Operation Time Databases

The placement and insertion assembly process class includes all handling, placement and insertion elements of assembly operations. This process covers many variations of the basic pick and place operation and also numerous pick and insert operations, for example into a recess or groove. The pick and place time algorithm element is also used as part of the total assembly time calculations for process operations. For example, a screw-tightening assembly operation requires a part to be initially picked, manipulated and placed at a desired location prior to the screw-tightening process. To successfully extract assembly times from the Boothroyd and Dewhurst assembly time standards, the classification system employed needs to be fully understood. The system is divided into two main elements, manual handling and manual placement/insertion.

The classification system for manual handling is a systematic arrangement of part features in order of increasing handling difficulties. Manual handling is divided into four scenarios: Parts which are easy to handle with one hand; parts that require a grasping tool; parts that severely nest or tangle and require two hands for manipulation; and parts that require two hands or assistance due to their size or weight. Each group is further broken down based on dimensions, weight, part symmetry and additional handling difficulties, such as slippery, fragile, jagged, hot and so forth. Each of the different handling scenarios, and the required data to derive a handling time, is discussed in the following sections.

- Handling with one hand - parts that can be grasped and manipulated easily by one hand. This is the general case for handling and manipulating a part, and the majority of parts fall into this category. Additional part dimensions (e.g. thickness and size), part symmetry, and data on handling difficulties are required to derive a handling time. The thickness is defined as the maximum height of the part with its smallest dimension extending from a flat surface. A number of examples are shown in Figure 5-2. The size, also called the major dimension, is defined as the largest non-diagonal dimension of the part's outline when projected on a flat surface. Both the size and thickness values are available from the product model.

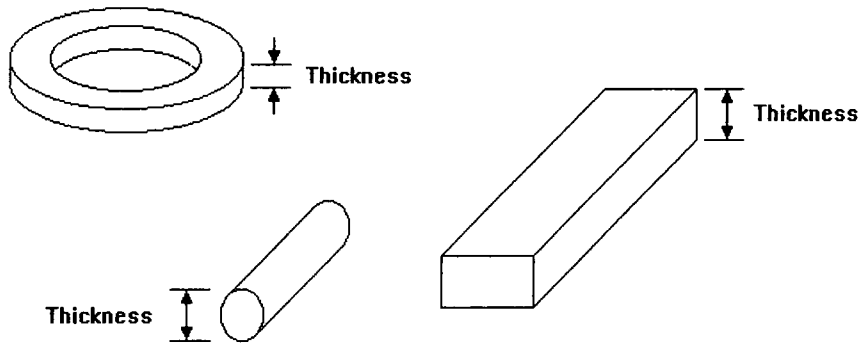
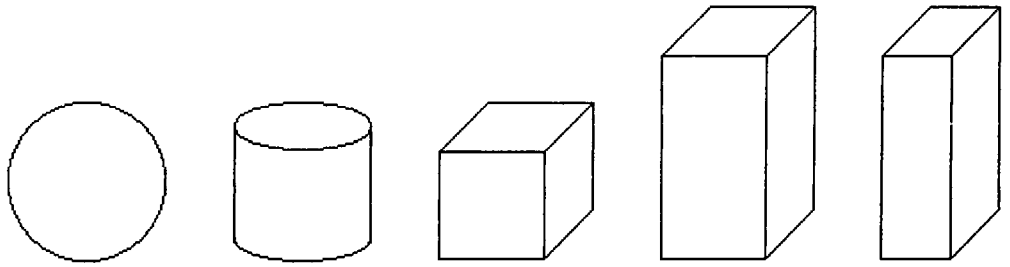
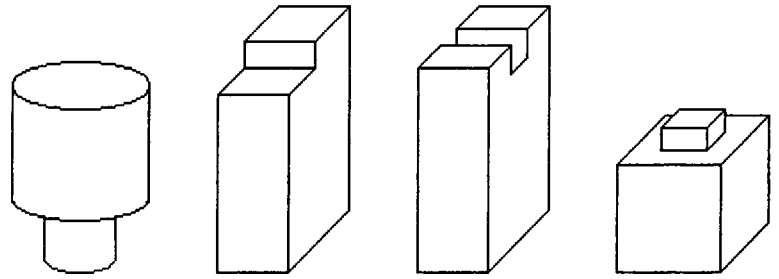


Figure 5-2: Examples of Part Thickness

One of the principle geometric design features that affect the handling time required to orientate a part is its symmetry. Orientation involves the proper alignment of the part prior to its placement or insertion, and can be divided into two distinct operations: Rotational alignment of the part about an axis perpendicular to the axis of insertion; and rotational alignment of the part about the axis of insertion. As a result, Boothroyd and Dewhurst define two types of rotational symmetry for a part: Alpha symmetry - maximum rotation required about the axis perpendicular to the axis of insertion; and beta symmetry - maximum rotation required about the axis of insertion prior to placement or insertion. Summing the alpha and beta symmetry values gives a single parameter for the classification system. A number of examples of alpha and beta symmetry values are displayed in Figure 5-3. Alpha and beta symmetry values can also be found from the product model.



Alpha	0	180	90	180	180
Beta	0	0	90	90	180



Alpha	360	360	360	360
Beta	0	360	180	90

Figure 5-3: Rotation Symmetries for Example Parts

Parts can present handling difficulties if they nest, tangle or stick together, are slippery, or require careful handling. Parts that nest or tangle are those that interlock when in bulk, but can be separated by one simple manipulation of a single part, for example taper cups, closed end helical springs, circlips, etc. Magnetic forces and grease coatings, are examples of circumstances when parts can stick together. Parts that are slippery are those which easily slip from fingers because of their shape and/or surface conditions. Parts that require careful handling are those that are fragile or delicate, have sharp corners or edges, or present other hazards to the operator. The product model contains data about part handling difficulties entered either by the user, or taken from standard part libraries during the product modelling process.

- Handling with grasping aid - parts that can be grasped and manipulated by one hand, but only with the use of grasping aids. Grasping tools can take the form of tweezers, standard tools such as circlips or pliers, or special tools to perform a specific task. Additional part dimensions (e.g. thickness and size), part symmetry, type of grasping

aid, and data on handling difficulties are required to derive a handling time. The majority of this data has been discussed in detail in the previous section. An additional time penalty is summed if optical magnification for manipulation is necessary, because the size and thickness of the part is exceptionally small. The type of grasping aid is entered by the user during the creation of the AFC.

- Two hands for manipulation - parts that severely nest or tangle or are flexible and require two hands for manipulation. Some parts naturally tangle and nestle together, so they need to be sorted before they are handled. However, once they are sorted, they only require one hand for handling. Parts may also need two hands for manipulation if they are flexible. Additional part dimensions (e.g. size), part symmetry and data on handling difficulties are required to derive a handling time.

- Two hands or assistance required for large size or weight - parts that require two hands, two persons, or mechanical assistance for grasping and transporting parts. Parts may need two hands for manipulation if the part is heavy, large, awkward to handle, does not possess holding features, or requires careful handling. Parts may also need two hands for manipulation if the component is unduly flexible. Additional part dimensions (e.g. size), part weight, part symmetry and data on handling difficulties are required to derive a handling time. This data is taken from the product model, or is entered by the user.

To derive a handling time, the AAMP system computes which scenario, from the above four, best fits the assembly operation. This is undertaken by analysing the size, weight, volume and shape of the moving part or sub-assembly in an assembly connection. The system then extracts a handling time from standard time databases, considering all the additional product data and handling information. A standard handling time database was designed for each of the above scenarios.

The design features that affect the classification system for manual placement/insertion include the accessibility and visibility of the assembly location, the ease of alignment, and if holding is required for subsequent operations. Accessibility and visibility can be impaired by either obstructed access or restricted vision. Obstructed access is defined because limited space is available for the assembly operation, causing a significant

increase in the assembly time. Restricted vision is defined because the operator has to rely mainly on tactile sensing during the assembly process. A part is easy to align and position if the position of the part is established by locating features on the part or on its mating part, and insertion is facilitated by well-designed chamfers or similar features. If holding is required for a subsequent operation, it is because the part is unstable after placement or insertion, and will require gripping or holding down before it is finally secured.

The data required to compute placement/insertion times is entered during the process of creating AFCs. This information is stored in properties attached to the feature connection. In a similar way that handling times are derived, standard manual placement/insertion times are also extracted from databases. Finally, the total assembly time can be found by summing the manual handling time and the placement/insertion time.

The practice of designing standard time databases for assembly operations is also employed in the calculation of process times for both joining with fasteners and wiring connectors. There are numerous different types of fastening methods available, but due to limitations of time, only snap fit fasteners were modelled. Three wiring connection methods were developed, namely, tab, threaded and pressure-fit: A threaded wiring connection involves placing a wire onto a joining component, and a subsequent threading operation to connect the two components together; a pressure-fit wiring connection involves pushing a wire into another component; and a tab wiring connection involves pushing a spade connector over a plain tab, or *vice versa*. For both the snap fit and wiring connectors, the final assembly time is made up of a handling and placement assembly time, and a subsequent process specific assembly time. The handling and placement time is taken from Boothroyd and Dewhurst's standard times, discussed earlier in this section. Standard assembly process times for both the snap fit and wiring operations were derived by measuring actual industrial assembly times. These values were then processed and entered into a number of databases to be used in the AAMP system.

5.4.2 Assembly Process Equations

The threaded connection is one of the most frequently used connection types in manual assembly. To calculate an assembly time for a threaded operation two elements are required: The manual handling and placement assembly time; and the screw-tightening process time. The handling and placement time is taken from Boothroyd and Dewhurst's standard time classification. This element has already been discussed in detail.

The second element required to compute an accurate threaded operation time is the screw-tightening process time. This process time is expressed in terms of the length and pitch of the thread, and the speed of the screw-tightening tool in revolutions per second. The speed of the tool is taken from the resource model. The process time is given by length of the thread divided by the product of the thread pitch and tool speed, as shown in the equation below.

$$t_p = \frac{l}{ps}$$

Where:

t_p = Processing time (seconds)
l = Length of thread (millimetres)
p = Pitch of thread (millimetres)
s = Speed of tool (revolutions/second)

The total threading operation assembly time can be found by summing the manual handling and placement assembly times, and the screw-tightening process time. Finally, assembly time penalties are summed to the process time for either obstructed access and/or restricted vision. This type of process equation is also used for the calculation of assembly times for welding, soldering, brazing and lubricating.

5.4.3 Standard Assembly Process Times

Without doubt, riveting is one of the 'classic' connection techniques and, together with threaded connections, is the most commonly applied. In aerospace engineering, riveting

is employed most frequently; over one hundred thousand rivets are used, for example, in the manufacture of large passenger aircraft. To calculate an assembly time for a riveted operation two elements are required: The manual handling and placement assembly time; and the riveting process time. The handling and placement time is taken from Boothroyd and Dewhurst's standard time classification. This element has been discussed in detail earlier in this chapter.

The second element required to compute an accurate riveted operation time is the riveting process time. The process time is controlled by the tool employed to perform the operation, and any additional assembly time penalties. Each individual riveting tool in the resource model has an operation rate property. This property is the time duration required to perform one riveting operation. Using actual resource data in this way allows a variety of process plans to be created using different factory resources, and also allows the introduction of state-of-the-art resource data. Assembly time penalties are summed to the process time for either obstructed access and/or restricted vision. The total riveting operation assembly time can be found by summing the manual handling and placement assembly times, the riveting operation rate, and any additional time penalties, as shown in the equation below. This method of computing assembly times from resource process rates is also employed in the calculation of times for insertion operations requiring a press.

$$t = t_h + t_i + t_o + t_a$$

Where:

t = Total assembly time (seconds)

t_h = Handling time (seconds)

t_i = Placement time (seconds)

t_o = Riveting operation time (seconds)

t_a = Additional time penalties (seconds)

5.5 Resource Model Elements

A number of elements are necessary to constitute a production and assembly facility. Many of these must be considered in order to develop aggregate process plans which

use that facility. The following list discusses each of these resource elements and the possible methods for modelling within a computer system.

- A Factory is a facility for the production and assembly of products. Each factory model contains models of all the manufacturing and assembly resources, which are grouped into cells, or may belong directly to the factory. For the purpose of this prototype system we assume that all operations are carried out at a single factory.

- A Cell is an administrative grouping of production and assembly resources within a factory. The cell contains workstations, storage and transportation.

- An assembly workstation is a grouping of assembly machines and tools, and labour force to undertake a specific set of assembly operations.

- An assembly machine or assembly tool is a device for performing a particular assembly process. Assembly resources can be of many different types, ranging from hand tools dedicated for a particular process (e.g. a spanner), to larger machines (e.g. an automatic press). Assembly resource models must represent the capabilities of each tool, in terms of processes which can be performed, duration and cost. Since assembly tools vary considerably depending on the processes for which they are designed, the assembly resource model must be specific to the class of assembly tool. Another important factor in assembly resourcing is the production capacity, which must not be exceeded when producing an assembly process plan.

- The transportation method employed within a factory can include conveyor belts, forklift trucks, hand trucks, Automated Guided Vehicles (AGVs) and manual handling. The movement of work around the factory contributes to the manufacturing lead time through the requirement of labour and equipment. Each transportation method has different properties of speed and cost per distance travelled. The algorithms for the calculation of both lead time and product cost should include a consideration of the transportation cost, a product of the method of transport and the distance travelled. For the purposes of aggregate process planning, each cell and workstation has a transportation method property which indicates the transportation options available within that area.

- Labour is required to carry out most assembly operations within a factory, and therefore it is important to know the cost of the labour. This will vary depending on the factory, because labour costs are related to the location of the factory and the hours worked. Each assembly operation will incur labour costs which must be calculated.

5.6 Resource Model Structure

The previous section has identified the elements which must be combined into the resource model. In Figure 5-4, the hierarchical structure of a single factory is highlighted. The factory is comprised of a number of cells, each of which contains a set of workstations, each of which consists of a collection of assembly resources.

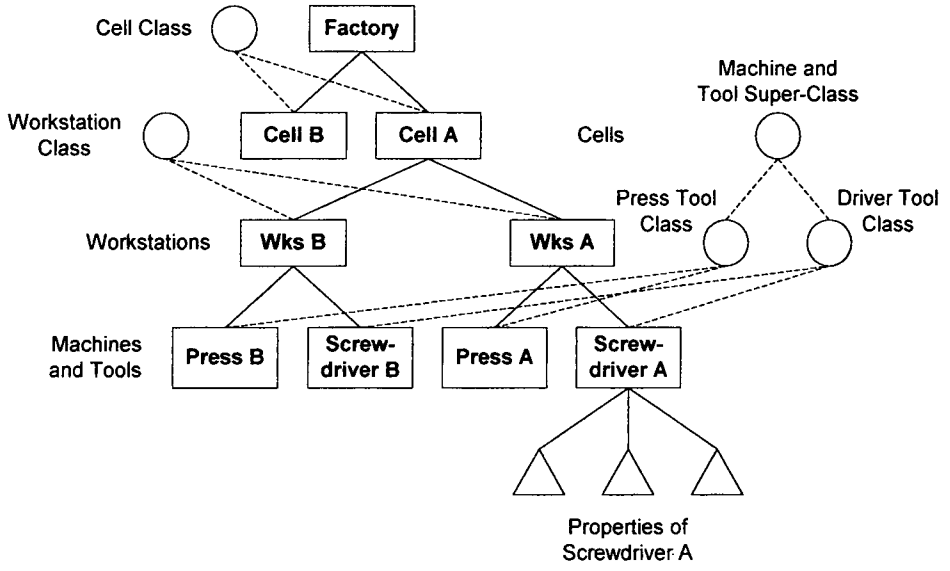


Figure 5-4: Factory Structure

Whilst the factory, cell and workstation models will remain similar for most examples, the assembly tool model will vary depending on the tool or machine type. A riveter and press have different properties and hence, different assembly resource models are required. Using an object-oriented model, however, specific models can be developed for each assembly tool type, so that only the appropriate properties and methods are supplied for each assembly tool. Each assembly tool is therefore an instance of a sub-class within the hierarchy of the assembly tool's class, also shown in Figure 5-4.

5.7 Resource Model Implementation

As described above, the resource model structure is implemented within the system by using a set of classes. These are briefly explained in this section, with examples of the class properties.

5.7.1 Factory Model

The factory concept is implemented within the system by using a single factory class. All factories are members of this class. Factories are loaded at separate occasions to allow the user to compare aggregate assembly process plans for different resources and set-ups. Examples of properties from the factory class include:

- Children: A list of the child objects of the factory, including any cells and tools.
- Name: A text identifier.
- Transport: A list of material handling methods between cells, including conveyors, AGVs and forklifts.
- X_ext, Y_ext: The width and length of the factory floor (in metres).

5.7.2 Cell Model

As with the factory concept, the cell concept is implemented within the system as a single cells class. It is at the cell level that the resource of labour is introduced to the model. Examples of the properties of the cell class include:

- Available: A flag to denote whether the cell is available for use in the aggregate assembly process plan.
- Children: A list of the child objects of the cell, including any workstations.
- Factory: The name of the factory to which the cell belongs.
- Name: A text identifier.
- X_coord, Y_coord: The position of the cell relative to the factory floor (in metres).
- X_ext, Y_ext: The width and length of the cell (in metres).

5.7.3 Workstation Model

The workstation concept is also implemented within the system as a single workstation class. A workstation is an area designated within a cell to carry out assembly operations. Workstation resources include assembly machines, tools and human operators. Examples of the properties of the workstation class include:

- Available: A flag to denote whether the workstation is available for use in the aggregate assembly process plan.
- Cell: The name of the cell to which the workstation belongs.
- Children: A list of the child objects of the workstation, including any assembly machines, tools and human operators.
- Name: A text identifier.
- Type: Denotes the type of workstation.
- X_coord, Y_coord: The position of the workstation relative to the factory floor (in metres).
- X_ext, Y_ext: The width and length of the workstation (in metres).

5.7.4 Assembly Tool Model

Unlike the factory, cell and workstation classes, each tool type requires a different class to represent its specific properties, capabilites and parameters, because its function and data differ so much. A classification of assembly machines and tools has been compiled containing a detailed model of each assembly tool type in its place. By classifying the different machines and tools into a hierarchy, it is possible to write generic models which apply to groupings of the classification, such as all presses or screwdrivers, and then to modify the details of these models to better represent individual variations.

For each process model within the classification, a number of key parameters are defined which hold all the information which is required by the rest of the system. Process data, such as rate and power parameters, and limits to the size and weight of components which can be considered, is common to all press types. The majority of the parameters are specific to the tool type because they would not be relevant to other tools. The assembly tool model maintains a model of each machine and tool in the

factory of the company. The machine and tool super-class models the generic attributes which are common to all assembly tools and machines and include:

- Available: Denotes whether the tool is available for use in the aggregate assembly process plan. Allows the user to remove tools for maintenance and so forth.
- Cell: The name of the cell to which the tool belongs.
- Cost or rate: The hourly rate or cost of the tool (pounds/hour).
- Name: A text identifier.
- Type: The generic type of tool.
- Workstation: The name of the workstation to which the tool belongs.

These are the properties which are used by the aggregate assembly process planning function, irrespective of the tool selected.

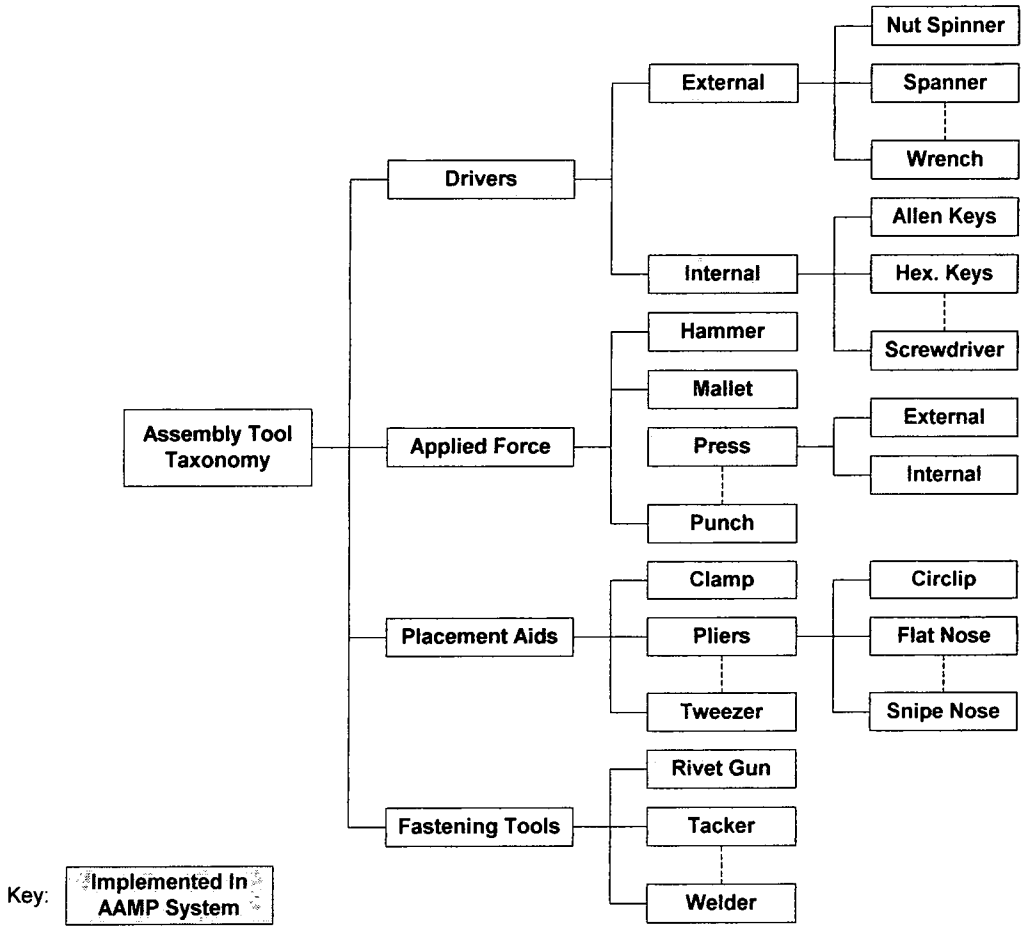


Figure 5-5: Assembly Tool Classification

5.7.4.1 Assembly Tool Class Structure

In building a taxonomy of assembly tool types, the aim was to develop a hierarchy which allows the definition of process capabilities for tools through the identification of a particular assembly tool class for each process. In other words, the system should define all the assembly tool types which can be used for a particular process as sub-classes of a single super-class, and no tool types which cannot perform the process should belong to this class. This requirement naturally results in a tool taxonomy which is similar to the process taxonomy. Figure 5-5 shows the top levels of the assembly tool taxonomy.

Each of these classes is divided into further sub-classes to model individual process capabilities. The three sets of tools that were fully implemented into the system to demonstrate a variety of tool models, were drivers, press machines and riveters. These were chosen to compliment the implemented aggregate assembly process models and are described in the following sections.

- Driver tools: This category of tool is used for screw-tightening assembly processes, and can be divided into two categories, external and internal drivers. The main distinction between external and internal drivers is that an external driver locates over the part that it is tightening, whereas, an internal driver sits within the part that it turns. There are numerous different types of driver available. Internal drivers include screwdrivers, Allen and hexagon keys; and examples of external drivers are socket drivers, spanners, nut spinners, Stillsons and monkey wrenches. Both categories can be powered manually, by air or by electricity. The main property of the driver class is the process rate (revolutions per second) that the driver can rotate. This value is used to calculate realistic and accurate screw-tightening process times, as discussed earlier in this chapter. Other properties of the driver tool are the range of torque which can be applied, and the size of components on which the driver can be used.

- Press machines: Presses are used to force a part into a recess when an interference fit occurs. An example of such an interference fit is a cylindrical cam shaft located in an engine bearing housing. Numerous different types of presses are available, from

standard bench presses that can perform many different jobs, to complicated one-off presses designed for a single task. Presses can be powered either manually or hydraulically, and can thus give a wide range of applied force. The main properties of the press class is the process rate (length of time per cycle), and the force that can be applied. This process rate is used to calculate realistic process times, as discussed earlier in this chapter.

- Riveting tools: This category of tool is used for riveting assembly processes. There is a number of different types of riveting tools, and they can be powered manually, by air or electricity. The main property of this class is the process rate (length of time per cycle). This process rate is used to calculate realistic process times, as discussed earlier in this chapter.

5.7.5 Transfer Model

The transfer model is implemented within the system by using a taxonomy of transfer types. In building a taxonomy of transfer types, as shown in Figure 5-6, the aim was to develop a hierarchy which allows definition of a transfer method through the identification of a particular transfer type class. The transportation method employed within a factory can include conveyor belts, forklift trucks, hand trucks, AGVs and manual handling. The movement of work around the factory contributes to the manufacturing lead time through the requirement of labour and equipment. Each transportation method has different properties of speed and cost per distance travelled. The algorithms for the calculation of both lead time and product cost should include a consideration of the transportation cost, a product of the method of transport and the distance travelled. For the purposes of aggregate process planning, each cell and workstation has a transportation method property which indicates the transportation options within that area. This includes data on the fixed and flexible transportation methods currently available. Although flexible transfer mechanisms can move products without major restrictions around a factory, fixed mechanisms are limited by their locations. Hence, fixed transfer types hold information on workstations and cells to which they linked, to restrict impossible assembly sequences being derived.

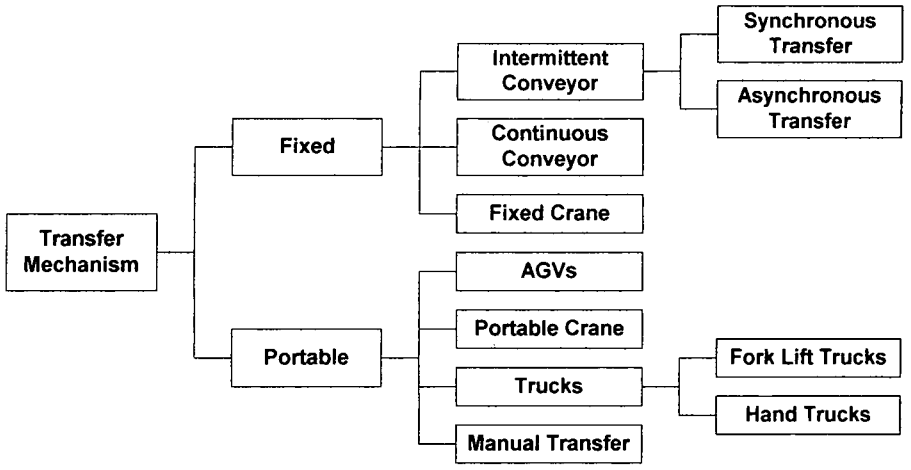


Figure 5-6: Transportation Taxonomy

5.8 State-of-the-Art Resource Model

During the development of new products, the process planner will often wish to consider the purchase of new tools. The best way in which to assess the impact of a new tool, and to determine if it would be a sound economic investment, is to calculate the effect on the production of new and existing products. If a model of a new tool is available within the process planning system, then the tool may be considered alongside the existing equipment and properly assessed. It is proposed that this would be a useful application of the system, because it is relatively simple to build models of state-of-the-art equipment. If parameter values such as speed, operation time and power limits are available for a new tool, then it can be included in the factory resource model, and the new process plans generated can be compared with previous sets.

5.9 Suppliers and Subcontracting

In today's business environment, it is common for companies to buy in services or components from other manufacturers. This is usually the case when the company does not have the capacity to perform a particular process, it is uneconomical to produce in-house, or a supplier can provide the component with higher quality and lower cost than is possible in-house. The decision to make or buy a particular component will depend on the availability of processes within the local factory, and the capacity of the resources which can perform these processes. In the future, it is anticipated that a

company using the system would use the tool to assist in the make or buy decision by comparing the cost of the component built in-house with that built by a known supplier, using a model of the supplier company's resources.

5.10 Summary

A set of assembly process models has been developed which allows the system to automatically assess the assembly of a given design. The process models provide information on the processing time required to undertake assembly operations. Models have also been developed to represent the resources available to the company in assembling the product. In combination with the process models, this allows the assessment of the manufacturing and assembly options for a given design. The system can predict the effects of changing the sequence of assembly, resource selection, factory layouts and staffing levels. The use of a detailed resource model within the system would allow the addition of extra functionality, such as the use of the system for performing benchmarking of the factory against state-of-the-art, and against other sourcing options, thus allowing the company to determine which parts should be made in-house and which should be brought in from suppliers.

Chapter Six

Aggregate Assembly Process Planning

6.1 Introduction

This chapter describes the concept of aggregate assembly process planning and defines the implementation of an aggregate assembly process planning methodology within the AAMP system. The aggregate assembly process planning functionality of the system analyses the product model, and produces an aggregate assembly process plan and routing using a model of the factory and the aggregate assembly process models.

An aggregate assembly process plan consists of a set of instructions which can be mapped against a structured aggregate model of the product design. The aggregate assembly process plan gives a general description of the assembly method for each of the assembly connections. Suitable combinations of assembly processes and resources are identified, and an appropriate sequence of operations is set out. An aggregate assembly process plan is intended as a guide to indicate assembly operations for a product, and an indication of cost, lead time, and resource requirements. It is not a complete set of assembly instructions because it does not include such things as 'numerical code' required by robots.

6.2 Aggregate Assembly Process Planning Requirements

There are a number of requirements which an aggregate assembly process planning algorithm must meet. The specifications have been used in order to develop the functionality described. A key consideration is that aggregate assembly process

planning is a generic technology which is intended to be applied across the full range of assembly processes to allow the comparison of all assembly options for any product. The other requirements for aggregate assembly process planning are listed below:

- *Early design:* The algorithm must be able to operate on early design data, i.e. at the conceptual and embodiment stages, when much of the detail required by traditional CAPP systems is not available.

- *Variable detail:* Aggregate designs will vary in detail, so the system must account for this variation and use extra detail where available.

- *Sequencing:* The aggregate assembly process plan will involve a number of assembly operations which must be organised into a logical sequence. Early knowledge of the assembly sequence enables the planning of facility layout and schedules to be brought forward in time.

- *Alternative routes:* Aggregate assembly process planning must allow a range of alternative routes, with comparative evaluations, to be modelled.

- *Process identification:* The algorithm must identify the assembly processes which could be used for assembly of the design. A key feature of aggregate assembly process planning is the consideration of a wide range of assembly processes.

- *Resource selection:* The system must involve the specification of resources, including tools, transportation and human resource. The plan should consider the capacity of each resource during processing.

- *Factory loading:* The algorithm must allow for loading the product onto the factory model. A key feature is the assembly time and cost calculation using actual resource data from the factory model.

- *Resource balancing:* The system must involve balancing the assembly resources to ensure a smooth flow through the factory, minimising assembly bottlenecks.

- *Multi-criteria analysis:* The optimisation within the program must reflect the multiple criteria which must be satisified to arrive at a good aggregate assembly process plan. These criteria include the assembly cost and time.

- *Transparency:* The algorithm should provide clear feedback to the user when decisions are made, to ensure the reasoning is understood.

- *Customisation:* Provision must be made to allow the user to influence the decisions made by the algorithm to reflect outside influences such as company business strategy.

- *Realism:* Clearly, the aggregate assembly process plans produced by the algorithm must be in line with those which would be adopted within the company so that they provide a reasonable guide to expected final production costs.

6.2.1 Task Sequencing

A requirement for the generation of a working aggregate assembly process plan is that the planner must specify the order in which the assembly tasks are to be carried out. There are many influences on the order in which the assembly operations should be carried out, some of which may be set aside, whilst others cannot be altered. Amongst these influences are: The product structure; assembly process type; geometrical constraints; and ergonomic constraints. The effect of the sequence is to apply constraints on the selection of assembly processes and resources, and the loading of the factory. In order to minimise costs it is important to keep to a minimum the number of assembly set-ups and the amount of transportation involved in the assembly route.

It is imperative that a feasible assembly sequence is derived because later system processing relies on this information, and the system will produce an incorrect aggregate assembly process plan, with spurious results, if an incorrect sequence is considered. Although the assembly sequence is derived automatically, it is important to allow the user to view the sequence and easily modify it if necessary.

6.2.2 Resource Selection

Resource selection refers to the specification of those assembly tools, equipment and human resources which will be used to carry out the assembly operations. Each tool can perform a particular set of assembly processes, typically from within the same overall classification. The criteria for the selection of assembly resources include: The capacity of the resource (including power, speed and geometry); the utilisation of the resource; the cost rate of the resource; and the location of the resource within the factory. In many cases the planner will wish to select assembly resources that ensure the product is made fully within one cell or area of the factory.

With resource selection, it is particularly important to provide comparative information between the different resources in the output from the system. It is often necessary within lower volume products to move production planned for one area to another. This can occur for maintenance or breakdown reasons, because of a lack of capacity or resources due to scheduling difficulties, or because of changes in the required output. The aggregate assembly process planning function can be valuable in this situation, because alternative production plans can be considered during the development stage, and the cost and time implications of alternative resources can be clearly determined.

6.2.3 Factory Loading and Balancing

Another requirement for the generation of a working aggregate assembly process plan is that assembly tasks must be loaded and balanced onto the factory resource model. Factory loading involves analysing the assembly of the product, identifying the operations and resources required, and calculating the time required by each task. In addition, technological constraints of the resources are considered to ensure that the operations can be carried out with the resources available. As well as the tool and human resources, transportation between workstations is to be considered in order to derive accurate assembly lead times and overall costs. Factory loading ensures that the actual assembly resource data is used for assembly time and cost calculations.

Assembly line balancing ensures that the production rate can be met by loading the resources in a manner which results in a smooth assembly line, with no major bottlenecks. Given the production rate (units per time period) that the line must

produce, a cycle time (time period per unit) can be calculated. The cycle time is the maximum time available at each workstation to achieve the production rate. It is possible that an actual assembly operation time is greater than the assembly cycle time. In this case, the line balancing function should load a number of mirrored workstations in parallel to ensure that the production rate is still maintained.

Allowing the user to create aggregate assembly process plans for any number of factory configurations is a valuable function within aggregate assembly process planning. This permits the user to consider his/her own existing factory configuration as well as new layouts, introduce improved resources into the factory, and even load supplier factories at an early stage in the design process. Automatically creating aggregate assembly process plans for both existing factory layouts and new layouts, using all existing resources within the factory, is a major function of aggregate assembly process planning.

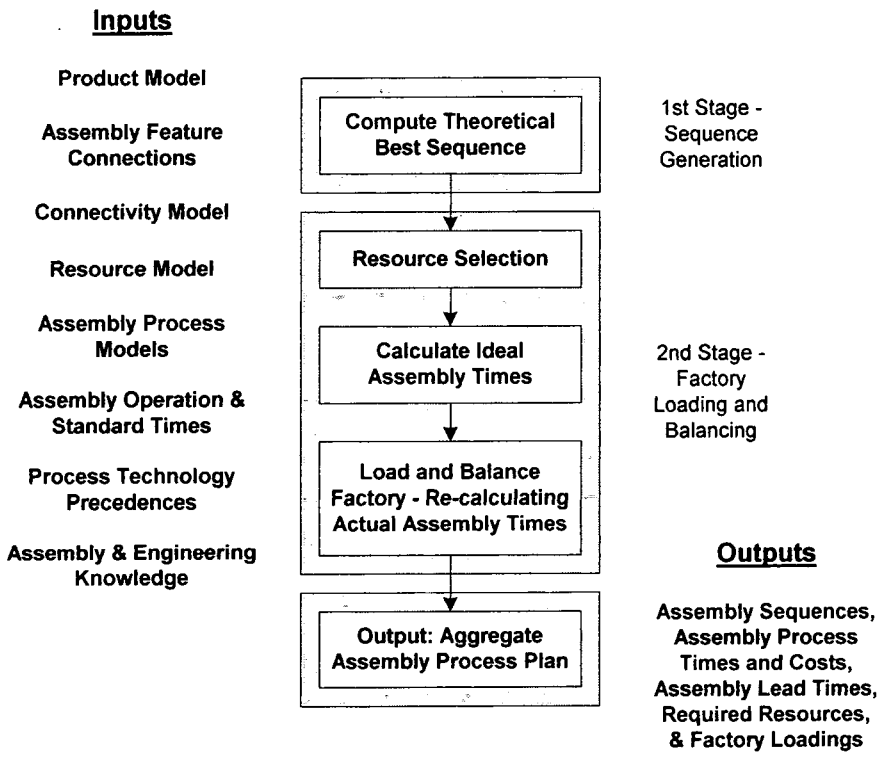


Figure 6-1: Aggregate Assembly Process Planning Structure

6.3 Aggregate Assembly Process Planning Functionality

The previous section sets out the requirements for aggregate assembly process planning and the AAMP system. In this section, the structure of the proposed aggregate assembly process planning algorithm will be detailed. The architecture which has been adopted is in two main processing stages. Initially, the theoretical best sequence is derived and assembly operation times are calculated using best in-house resources. Secondly, the factory is loaded and balanced and an aggregate assembly process plan is generated using specific resources, and actual assembly times and costs are re-calculated, as shown in Figure 6-1. The decision to perform the two task groups of sequencing and initial operation calculations, and secondly resource loading, balancing and re-calculation sequentially, instead of concurrently, was taken in order to reduce the computational load and complexity of the system.

It is important for an aggregate assembly process planning system to operate rapidly in order to provide immediate feedback to product developers, particularly when used by designers. This allows the evaluation of many alternative product ideas and resource specifications, and is the key to successful conceptual design. Whilst it is generally a straightforward procedure to generate the lists of possible sequences, finding feasible sequences, and the best combination of sequence and resources, is a more difficult task. The size of the search increases exponentially with the number of assembly connections and available resources. Indeed, for practical purposes, the search space becomes too large to be effectively searched with even advanced methods. For this algorithm, it has been decided that the effective way to reduce the search, whilst retaining the greatest chance of reaching the optimum solution, is to initially derive the theoretically best sequence before loading and balancing the assembly line. By determining the sequence in advance of resource selection, it is possible to greatly reduce the number of possible assembly routings and process plans, hence, reducing the processing time.

The sequence is the factor which is most dependent on engineering knowledge and expertise, and is therefore the least suitable for automation within a computer system. Hence, it is appropriate to use a knowledge-based computer system for this section of the algorithm. This technique allows the embodiment of human engineering knowledge and the easy integration of extra requirements and constraints from the user. The

sequence is therefore determined with an algorithm that is based on geometric product information, product structure, resource data, and accepted engineering and assembly practice. The user may also alter the generated sequence before the resource loading and balancing commences.

To derive an accurate and useful aggregate assembly process plan, it is vital to consider actual factory assembly resources in the algorithm. Only by using actual resources will a real life aggregate assembly process plan be computed giving genuine assembly indicators. The resource loading and balancing algorithm allows the assembly sequence to be realistically loaded onto the factory considering different resource capacities, capabilities and routings. This allows the user to load the product onto the complete factory, or turn individual cells, workstations, machines and tools on and off in order to compare alternative aggregate assembly process plans at different locations within the factory, and with different resources. An important function of the system is to load an existing factory and also design and load new factory layouts, allowing multiple plans to be derived efficiently. This permits factories to be re-designed with existing resources, or incorporate improved or world-class resources. The aggregate assembly process plan is outputted in a set of HTML World Wide Web files. Each dynamic file is linked, allowing fast and easy viewing of results, and can be saved for future reference.

The aggregate assembly process planning functions are implemented in an algorithm which divides the planning tasks into a sequence of discrete stages at which the user is consulted and is able to monitor the system's progress. This approach permits the user to develop an awareness of the tasks involved in aggregate assembly process planning, and understand the effect of each element on the design of the aggregate assembly process plan. It also gives the user the opportunity to override the computer-generated suggestions when special circumstances arise.

6.4 Sequencing

The initial stage in the algorithm is performed using an algorithm based on a knowledge-based system which embodies assembly planning expertise. The system aims to satisy the numerous constraints to determine an advantageous order to carry out

the elements of the assembly plan. The constraints which must be satisfied are detailed below, after which the algorithm is described in detail.

6.4.1 Factors Affecting Sequencing

The factors which affect sequencing can be classified into two groups, hard constraints and soft constraints. Hard constraints are those rules which must be followed because it is physically not possible to break them, and these are typically related to the product structure, geometry and user-defined constraints. Soft constraints are general engineering and assembly rules which should be followed in order to simplify production and create a logical assembly plan. These rules can be broken if they conflict with the hard constraints or cause very high assembly times and costs.

Hard Constraints

- *Precedent relationships from the product structure:* The aggregate product model utilises a bill of materials structure, a familiar technique used to visualise a product constructed from assemblies and components. This method gives a hierarchy of the product structure. The location of the connections on the product model greatly constrains the sequence, because it can be assumed that connections at a lower level in the hierarchy will be undertaken prior to those at a higher level. For example, a sub-assembly will be assembled before it is connected to its parent assembly.

- *Geometric:* Constraints can be imposed by component geometry, in particular the size, volume and weight of joining parts.

- *User-defined constraints:* The user may wish to intervene in the sequence selection process to specify additional constraints where the algorithm will not account for special cases.

Soft Constraints

- *Base fixturing parts:* At each stage of the sequence, a component, or collection of components, will take on the role as base part. It is usual that this base part remains stationary, with sub-assemblies and components being added to it. Frequently, the base part will sit in an assembly fixture as it moves along the assembly line. The

base part is defined by data such as size, volume, number of components, and number of connections.

- *Moving parts:* For each assembly operation one component will predominantly move in relation to its other components. This can be derived by considering the size, weight, volume and number of parts of each component or sub-assembly that is to be connected together.

- *Process technology:* Certain processes create constraints. In particular, when an operation requiring heat is used, it is advisable to locate temperature sensitive parts after this process.

- *Process precedence:* Processes can be ordered into a general sequence which should be followed. In particular, processes that require force or heat should be undertaken early in the sequence and placement processes should be undertaken before fastening processes such as threading or riveting.

- *Process clustering:* Clustering the operation elements according to the process used for assembly allows the resource selection algorithm to benefit from specifying the same tool or machine for multiple adjacent tasks.

It can be seen that certain of these constraints will cause conflict in the generation of a suitable sequence. A process plan will always require that compromises are made in order to arrive at a working solution.

6.4.2 Sequencing Algorithm

The sequencing algorithm, shown in Figure 6-2, operates in two main stages. Initially, an outline process plan is derived from the product structure hard constraints, i.e. the constraints imposed by location of the assembly connections on the bill of material product model. This gives a hierarchical structure of the assembly connections, and also groups together the connections at each level. The second stage of the algorithm orders the groups of connections at each level into a sequence. The second stage can be further broken down into applying the soft constraints, assigning weightings and ordering according to these weightings.

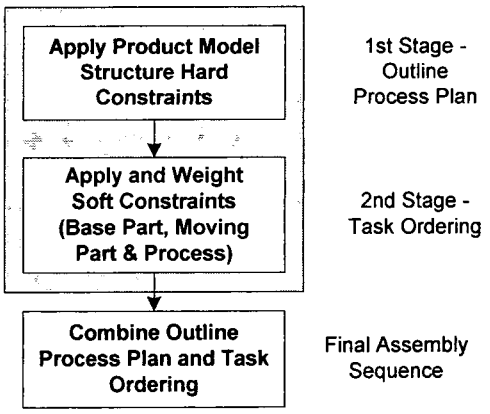


Figure 6-2: Sequence Generation Algorithm

6.4.2.1 Outline Process Plan

During the first stage of the sequencing algorithm, an outline assembly plan is found from the hard constraints imposed by the structure of the product model. The bill of materials model allows the product to be defined in a hierarchical structure. At the top of the model is the final product, and at each level down, the product is broken into sub-assemblies, components and features, as discussed previously in chapter four. Assembly connections are created by linking features together, and each connection is additionally attached to a sub-assembly node at a higher level on the product model where the feature network lines meet.

Figure 6-3 displays a section of a product connectivity model with a number of assembly connections. It can be seen that the connections are attached to the features that are being assembled and also a parent sub-assembly node. From the connectivity model we can see that connections A, B and C should be undertaken before connection D. Applying these constraints minimises the number of problematic generated sequences that occur because of access, location, and obstructed vision. This method reduces the number of possible sequences significantly, and so reduces the required computational time. The results of the first part of the sequence algorithm are an outline of the aggregate assembly process plan with groups of assembly operations at various stages. These groups are then ordered in the second part of the sequence algorithm.

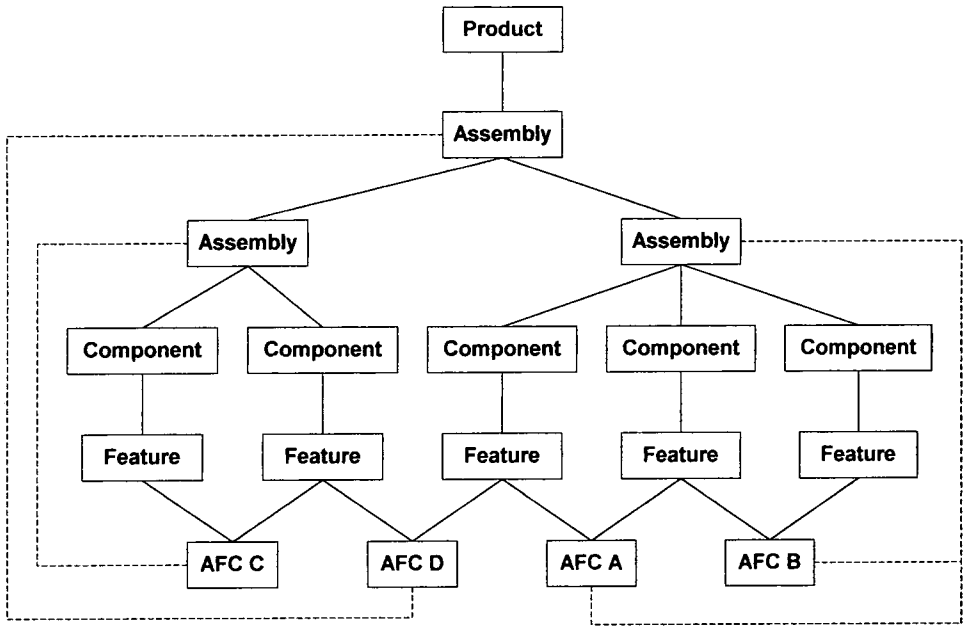


Figure 6-3: Example Product Connectivity Model with Assembly Connections

6.4.2.2 Second Stage Sorting

The second stage of the sequence algorithm involves a number of tasks. Initially, the moving part of each connection is derived before the base part for each set of connection groupings is found. The connections are next assigned process weightings, and ordered according to these.

- Moving part of each connection: An important task within the sequence algorithm is deriving the moving part for each AFC. This aims to remove the possibility of spurious sequences being computed. For example, if an assembly operator is placing a wheel onto a car, then it is sensible to move the wheel to the car, rather than *vice versa*. Another example is a micro-chip onto a circuit board. The algorithm carries out a number of checks to find the most obvious moving part or parts for each connection. These include analysing the dimensions, volume, weight, number of assembly connections and number of parts. If a sub-assembly is being considered as a possible moving part, the algorithm will calculate its weight, volume and so forth, by determining the sum of all its children components.

- Base part: The algorithm to derive the base part at each level of assembly is similar to the code to find the moving part as described above. The base part is defined as the part or sub-assembly to which all other components are connected. The base part is usually stationary at this point, and regularly located in a fixture to aid assembly. An example of a base part is a printed circuit board. This would generally sit in a fixture, and a number of electrical components would be inserted onto the board as it moves along an assembly line. The algorithm carries out a number of checks to derive the base part. These include analysing the size, weight, number of components and number of connections for all parts and sub-assemblies at each level.

- Process weightings: The next stage in the algorithm is to apply a process priority index to each of the unsequenced assembly operations. This index is based on process weightings which are determined by rules corresponding to assembly theory. Example rules include that operations requiring heat or force should be carried out early in the assembly sequence, fastening processes should be carried out after placement operations, and so forth. General precedences for each process have been determined to generate the process indices, as shown in Table 6-1. Low index processes should be carried out prior to high index processes. Note that processes in italics have not been modelled in the prototype AAMP system.

Table 6-1: Process Sequence Index

Metallurgical	*100*
Plug and target	200
Placement and Insertion	300
Wiring	400
Snap fit	500
Threaded	600
Chemical	*700*
Riveted	800

Once a process weighting has been assigned to each connection, the algorithm will order the assembly elements according to this value. This method of sorting allows the

grouping of similar processes. For example, if a number of operations require pressing, then it is logical that they should be carried out at the same time.

6.4.2.3 Final Assembly Sequence

The final stage of the sequencing algorithm is combining the outline process plan and the second stage sequences to give a final assembly sequence for the complete product model. Although the algorithm is described in two stages, the object-oriented architecture of the prototype system allows both functions to be carried out simultaneously. The final feasible assembly sequence is then presented to the user via a GUI for validation. An example is shown in Figure 6-4. The user can either confirm that the sequence is correct, or make desirable adjustments if necessary by cutting and pasting in assembly operations. This gives the user ultimate control over the sequence and allows the resolution of any conflicts between constraints.

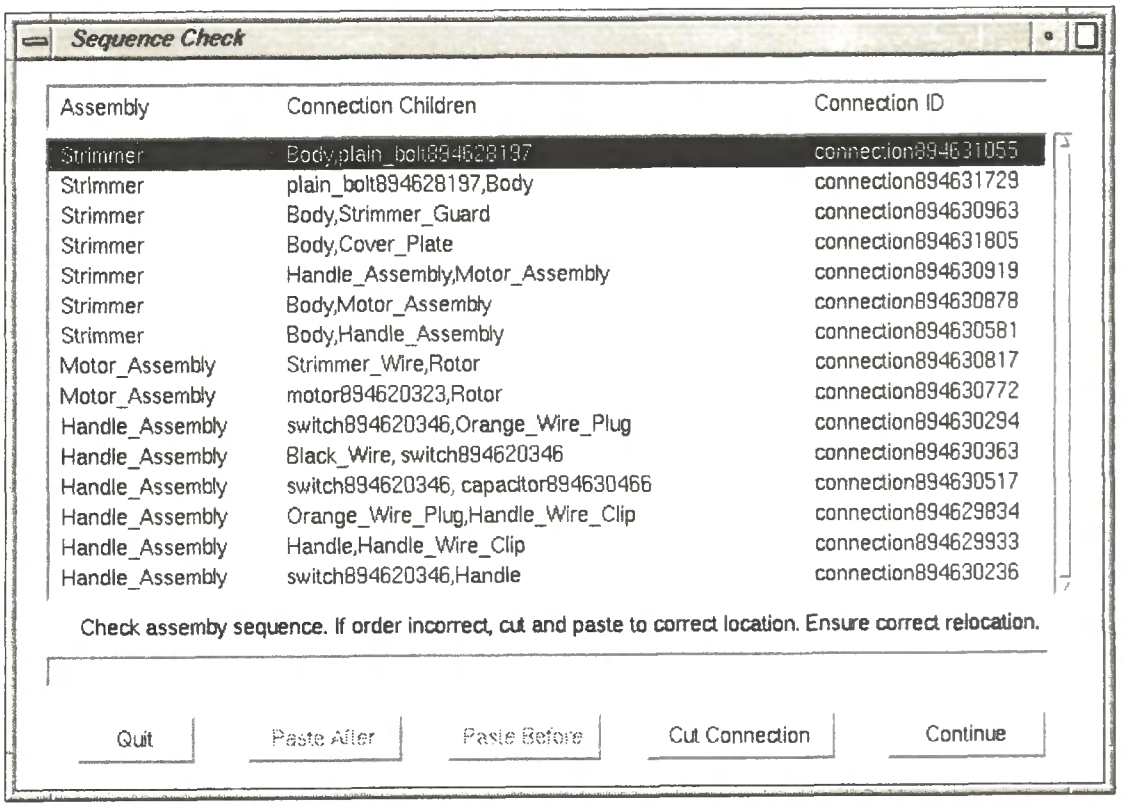


Figure 6-4: Sequence Output

6.5 Assembly Operation Indicators

One of the main functions of aggregate assembly process planning is the calculation of assembly times and costs for each of the assembly tasks. This enables the user to select the most suitable solution. This may be the solution with the shortest lead time or the lowest cost, or a combination of relatively short lead time and fairly low cost. Assembly times are calculated at two stages during the algorithm, whereas assembly costs are only calculated after the product model has been loaded onto a factory. The system makes use of the assembly time calculation methods which are defined for each process in the assembly process model, as detailed in chapter five. Each operation inherits a particular method from its parent process class. The method takes inputs from the properties of the components to be connected, information from the product model, and factory resource data.

6.5.1 Assembly Times

The algorithm calculates assembly task times in two parts, a general handling time and a more specific placement/insertion or process time. A process time is usually where a tool is required, whereas a placement/insertion time is generally the location of a part or collection of parts without tools. Process models for the process time calculations use data from the factory model to ensure that they are realistic. This can include tool speeds, machine rates, capability and so forth.

Assembly process times are calculated twice during the aggregate assembly process planning algorithm, once at an early stage in the algorithm, and later these times are re-calculated when the operations are assigned to actual workstation resources. The first set of assembly time calculations use the best in-house resources to give ideal assembly times. This is undertaken for two reasons. Firstly, ideal assembly times reduce the amount and complexity of computation required during the factory loading algorithm. This is because ideal times can only be increased when actual resources are used. Therefore, ideal times can be used to verify if a workstation has the available loading capacity. Secondly, ideal times can be used to calculate efficiencies of each assembly operation. During the loading and balancing of workstations, real assembly times are re-calculated using the actual resources available at each workstation.

Transfer times between workstations are also computed using resource information such as the location of workstations and transfer equipment being utilised. The transfer process models use data from the factory model to derive the distances between each workstation, and also the speeds and rates of the selected transfer equipment. Once the assembly process and transfer times have been calculated, the total assembly time, lead assembly time, and critical assembly time can be derived.

The total assembly time is composed of the sum of all the assembly operations' times and transfer times. The lead time is the actual time required to complete all operations and transfers in an industrial scenario. The most loaded workstation controls the cycle of the assembly line, and is referred to as the bottleneck workstation. This means that workstations not utilised at the same rate as the bottleneck workstation, spend time waiting for the bottleneck workstation to complete its tasks. The assembly lead time is calculated by multiplying the cycle time of the bottleneck workstation by the number of workstations loaded, and adding to this the total transfer time. It is nearly always the case that the lead time is greater than the total assembly time. The only situation where the total and lead assembly times are identical is if all workstations are loaded at the same rate, i.e. one hundred per cent. It can be seen from the lead time calculation method the importance of evenly loading and balancing the assembly line.

Determining the critical path and critical path assembly time involves finding each possible path from the start of the assembly process to its finish, then calculating the length of each path, and finally, determining the longest assembly path. The critical assembly time is generally lower than the total assembly time, unless there is only one assembly path through the product assembly model. The product model structure implemented in the AAMP system is ideal for calculating the critical path. The critical path, and hence the critical assembly time, is found by searching through the product assembly model to find the network path with the longest assembly time. An example product model and critical path is shown in Figure 6-5. Starting at the top of the model, a search method is used to find the child sub-assembly with the largest total assembly time. This is repeated until the critical path is found. The critical assembly time is finally computed by summing all the assembly operations on this path.

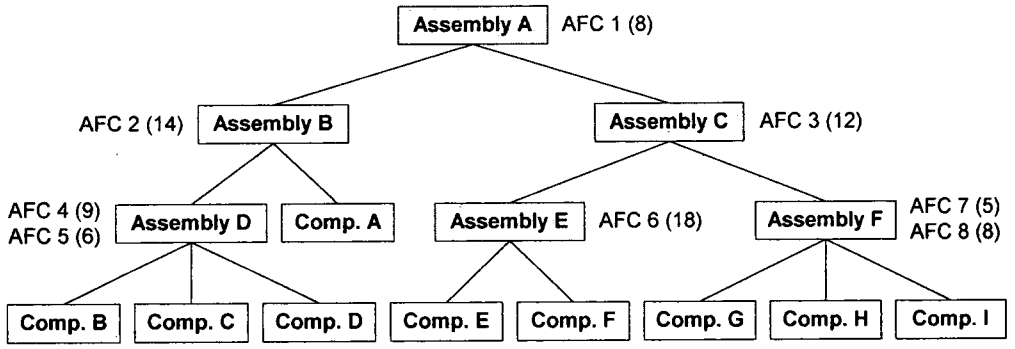


Figure 6-5: Critical Assembly Path

6.5.2 Assembly Costs

The total assembly cost of a product is a function of the product design and the assembly system used for its production. The lowest assembly cost can be achieved by designing the product so that it can be economically assembled by the most appropriate system. It is necessary to determine the costs accurately because companies generally make decisions solely on these results.

Assembly costs for each product are derived from a multitude of data. There are several contributing factors to the overall cost. These can be broken down to include:

- *Labour cost:* The cost of labour is regularly the highest factor in the total assembly cost. Labour costs for a process vary in importance with the process type. For highly automated assembly the cost is negligible, whereas for labour intensive processes such as manual assembly, the labour cost is probably the most important cost. The cost of labour is a function of time and the rate of pay.

- *Assembly tool cost:* The second major cost factor is the cost of using a specific assembly tool. This cost is proportional to the time which the component spends on each machine. Assembly machine costs can be divided into value-adding time when the processing is taking place, and non-value-adding time when the machine or tool is being set-up or the parts are being loaded or unloaded. For the algorithm, one time is used for this calculation, which is made up of a combination of the non-value-adding and value-adding times. The times are converted into costs using a cost rate

which is calculated for each tool and machine, based on its purchase cost, depreciation and running costs.

- *Overhead cost:* The overheads of a production company must be paid for through the profits on products, and therefore, to assign overhead costs individually to each product reflects the true cost of manufacture to the company.

- *Investment cost:* The investment cost represents the time value of the money which is tied up in the production of the process. Therefore, this is an indication of the cost of the factory inventory, and thus the cost consequences of a given production lead time. The shorter the assembly lead time, the sooner the customer receives the product, and therefore, the sooner the company recoups the investment through payment. In many companies this cost is overlooked, but where lead times are large, then this cost can be significant. It is through the consideration of this cost that the trade-off between shorter lead times and higher processing costs can be investigated.

The objective of the algorithm is to sum all these costs together for the product, and return a single cost per product which allows different products, sequencing and factory options to be compared. The system calculates an assembly cost per product by summing all the individual processing costs, including human resource and assembly tools and machines. This figure is then added to the workstation transfer costs before a combined value is mutiplied by individual overhead costs.

6.6 Factory Loading and Balancing

This section deals with the key aspects of designing and planning assembly systems. Assembly line loading and balancing is a key task in achieving effective material flow, controlling in-process inventory, and promoting assembly line performance. The fundamental aims of the loading and balancing algorithm are to assign all assembly operations to resources within a factory (whilst ensuring the resources have the capability and capacity), select the best resources, utilise the workstations efficiently, and ensure that the sequence is maintained. An important function of the AAMP system is the ability to create new factories, as well as load existing factories. The system can design and load a new factory, identify required resources, and also plan the factory

layout. This allows new factories to be completely re-designed using existing resources, or incorporating improved or world-class resources. A number of factors are considered to effectively load and balance the assembly line. These are detailed below, after which the implemented algorithms are described in detail.

6.6.1 Factors Affecting Loading and Balancing

One of the main factors in the algorithm is the factory to be loaded. The factory resource is made up of cells, workstations, machines, tools, human resources and transportation equipment. Cells are a collection of workstations, and workstations are a collection of assembly machines, tools and human resources. Transportation equipment can include assembly line conveyors, AGVs and fork-lift trucks. The AAMP system allows a number of options to be investigated. The user can load the product onto the complete factory, or turn individual cells, workstations or machines on and off to compare loadings at different locations in the factory, and with differing resources.

To ensure that the assembly line is operating efficiently, it is important to load and balance the resources in a manner which results in a smooth assembly line, with no major workstation bottlenecks. A bottleneck workstation resource is one that, because it is working at capacity, acts as a restriction on the total output. These are the resources where an increase in capacity would also increase the total output. A non-bottleneck workstation resource, on the other hand, is one that is not working at full capacity, and an increase in capacity of the non-bottleneck resource will have no effect on the output. All assembly lines possess a bottleneck workstation, but it is important that the bottleneck should be as minor as possible. Hence, it is important to attempt to utilise all the workstations on the assembly line at the same rate. There will be situations where a smooth assembly line cannot be designed with the available resources and sequence. It is important that the system outputs the results, highlighting the problem areas, and allows the user to re-design the factory or change the assembly sequence, and re-create the aggregate assembly process plan.

Prior to the running of the loading and balancing algorithm, production data has to be entered into the system by the user. The system then calculates the assembly line cycle time, i.e. the maximum time available at each workstation to achieve the production

rate. During loading and balancing each workstation can be loaded up to this value. Ideally, each workstation would be loaded and balanced to the same capacity, but unfortunately, in practice, this rarely occurs. Occasionally, there will be situations when the assembly cycle time will be less than the assembly time for a particular operation. Here, the system identifies the situation and loads mirrored workstations in parallel to ensure that the production rate can still be maintained.

6.6.2 Loading and Balancing Algorithm

The loading and balancing algorithm operates in a number of differing ways, but the fundamental objective of the algorithm is to load all the assembly operations onto workstations, whilst ensuring the workstations have the capacity and capability. There are also special scenarios that have to be included in the algorithm, such as designing and loading new workstations when the factory is fully loaded, creating new factories and layouts from all available resources, and loading operations on parallel mirrored workstations to ensure the production rate is maintained.

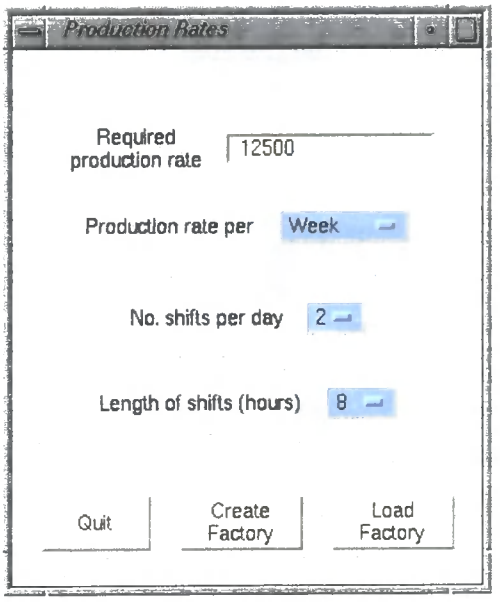


Figure 6-6: Production Rate GUI

The algorithm is divided into two routes, the loading of an existing factory, and the creating and loading of a new factory. In this section all scenarios will be discussed in detail. Prior to this split in the algorithm, the user inputs the required production data into the system and selects to load either an existing or a new factory via a GUI, as

shown in Figure 6-6. The inputted data includes the required production volume, the time duration to complete this volume (i.e. the number of days, weeks, months), and the number and length of shifts. From these values, the system can derive the production rate (or throughput rate) R, from the equation given below.

$$R = \frac{v}{tnl}$$

Where:

R = Production or throughput rate (units per time period)
v = Production volume (units)
t = Number of days to complete the production rate
n = Number of shifts per day
l = Time duration of shifts (time period)

Given the production rate R (units per time period) that the line has to achieve, the required assembly cycle time c (time period per unit) is derived by the equation below.

$$c = \frac{1}{R}$$

The assembly cycle time c is the maximum time available at each workstation to achieve the required production rate. The algorithm uses this value to ensure that workstations are not overloaded and are evenly balanced.

6.6.2.1 Loading Existing Factories

This part of the algorithm deals with the loading of assembly operations onto an existing factory. An existing factory has restrictions on the location of tools and machines at each workstation, and on the order of the workstations along the assembly line. For a re-designed product, it is likely that it will be loaded on an existing assembly line. Figure 6-7 shows the main functions and general sequence that the algorithm takes. Initially, the system selects the first assembly operation from the earlier derived sequence, and an available workstation to be loaded. Workstations also possess a sequence in which they should be loaded, which is stored in the factory model. The

algorithm then carries out a number of checks to ensure that an operation can be carried out at a workstation.

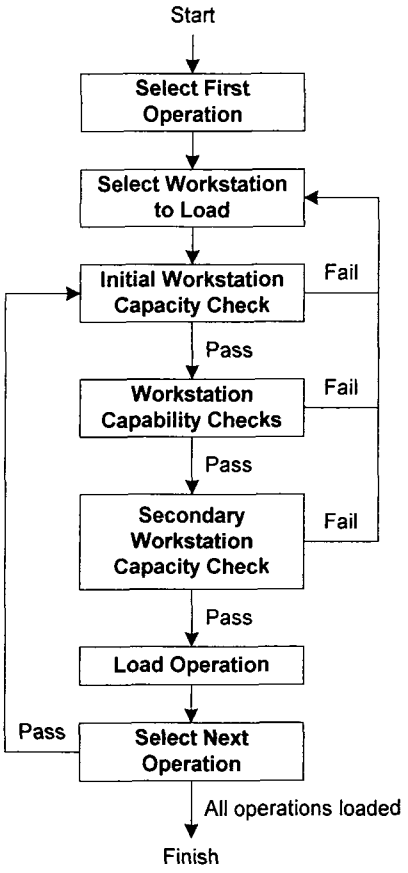


Figure 6-7: Factory Loading Algorithm

1. Initial Capacity Check

The first check is to ensure that the workstation has enough remaining capacity (time) to load the assembly operation. The ideal assembly times calculated earlier in the algorithm are used for this check. As the ideal times are derived using best in-house resources, the actual times, to be calculated later, cannot be less than these times. Hence, if the check fails, there is no possibility that the operation might have fitted on a workstation when the actual times are calculated. The check, shown below, sums the assembly operation time and the amount that the workstation is already loaded, and ensures that it is lower than the assembly cycle time.

$$c \geq t_{op} + w_l$$

Where:

c = Assembly cycle time (seconds)

t_{op} = Assembly operation time (seconds)

w_l = Current workstation loading (seconds)

If a workstation does not have the sufficient capacity for the operation, the algorithm aborts from these checks and selects the next available workstation. In some cases, the assembly time to complete the operation will be greater than the assembly cycle time, and the algorithm will fail at the initial capacity check for all the remaining workstations. Due to this possibility, prior to selecting another workstation, the amount that the current workstation is loaded is checked to ensure that it is not zero. If the workstation loading is equal to zero, parallel mirrored workstations are required to split this operation onto more than one workstation to maintain the production rate. This topic is discussed later in this chapter.

2. Capability Check and Tool Selection

The algorithm next checks to see if the required resources are available at the workstation, and determines the best available resource. Each tool or machine can perform a particular set of assembly operations, typically from within the same overall classification. The algorithm identifies all the factory resources suitable to undertake the operation, and matches this set to the resources available at the current workstation. This gives a sub-set of resources available for selection. The criteria for selection of assembly tools and machines include the operation time (speed or rate) and the capability (power, torque, size). The fundamental reason for the selection of resources is the operation time. If a resource is not available at a workstation, or does not possess the capability required, the system aborts from these checks and selects the next available workstation.

3. Secondary Capacity Check

If a workstation has the capacity and required resources, then the algorithm re-calculates the assembly time using the actual resource. This gives the real assembly operation time. The final check that the algorithm then performs is identical to the initial capacity check discussed earlier, but uses the 'actual' instead of 'ideal' assembly times.

If a workstation does not have the sufficient capacity for the operation, the algorithm selects the next available workstation. However, if the workstation does have sufficient capacity, the assembly operation is loaded onto the workstation. The system then selects the next operation from the assembly sequence and attempts to load it onto the same workstation, and the loading and balancing algorithm re-starts.

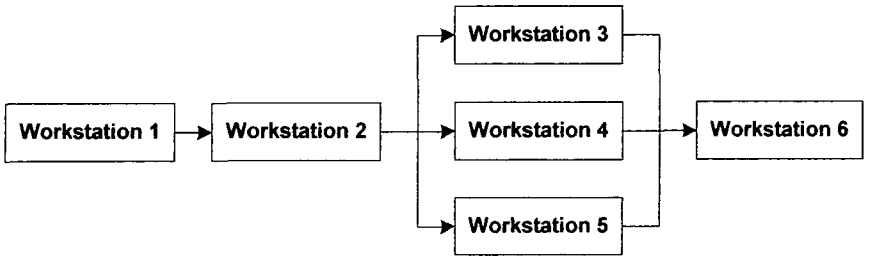


Figure 6-8: Parallel Workstations

Parallel Workstations

As mentioned earlier in this section, in some cases the assembly operation time will be greater than the assembly cycle time. The solution to this problem is to load the assembly operation on a number of mirrored parallel workstations to maintain the required production rate, as shown in Figure 6-8. The number of mirrored workstations required is calculated by dividing the assembly operation time by the assembly cycle time, as shown in the equation below, and rounding it up to the nearest integer.

$$n = \frac{t_{op}}{c}$$

Where:

n = Number of mirrored parallel workstations required

t_{op} = Assembly operation time (seconds)

c = Assembly cycle time (seconds)

The algorithm loads the parallel workstations in a similar approach to a single workstation. The next workstation on the assembly line is selected, and capability and secondary capacity checks, as discussed earlier in this section, are performed on the workstation. This routine is repeated until the number of loaded workstations is equal to the number of required mirrored workstations.

It must be observed that as there is now more than one workstation loaded with a particular assembly operation, the amount that each of the mirrored workstations is loaded has to be divided by the number of mirrored workstations. For example, if the cycle time is thirty seconds and the assembly time is forty seconds, two parallel workstations are required. However, the time that each workstation should be loaded is forty seconds (actual assembly time) divided by two (number of required parallel workstations), which equates to twenty seconds. This is because we assume that only half of the production goes through each mirrored workstation.

In a similar manner that normal workstations are loaded to their full capacity, the algorithm also attempts to load all of the parallel workstations with as many assembly operations as is feasible. The algorithm selects the next assembly operation in the sequence, and checks that all of the parallel workstations have the required capability and capacity. This process is repeated until a workstation fails to have either the capability or capacity. The algorithm then returns to the single workstation loading algorithm.

New Workstations

Whilst loading an existing factory, there will be occasions when a high production requirement and/or a low number of available workstations will result in the factory resources not having sufficient capacity. If this situation occurs, it is desirable for the system to fully load the existing factory, and then design and load new workstations for the remaining assembly operations. The algorithm for designing, assigning resources, and loading new workstations is discussed in detail in the next section.

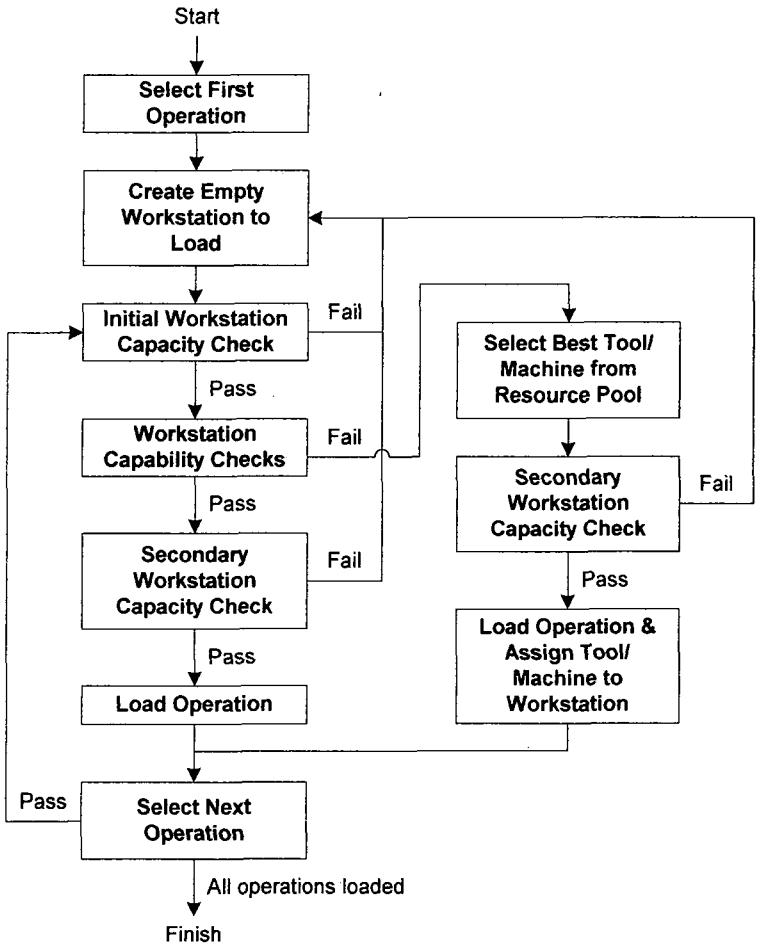


Figure 6-9: New Factory Loading Algorithm

6.6.2.2 Loading New Factories

During the design of a new product, it is usual that effort will be spent re-designing or creating new assembly lines. Whereas an existing factory has restrictions on the location of assembly tools and machines at each workstation and on the order of workstations along the assembly line, a new factory can be designed for a specific

product to gain the minimum lead time and the maximum throughput. This part of the algorithm deals with the designing of a new factory and subsequent loading and balancing of assembly operations. Figure 6-9 shows the main functions and sequence that the algorithm takes. The functions are very similar to those used for loading an existing factory, the main difference being that the factory is designed and resourced as each assembly operation is considered.

Initially, the system creates an empty workstation prior to selecting the first assembly operation from the earlier derived sequence. An initial capacity check is performed to ensure that the workstation has the capacity (time) for the assembly operation. The ideal assembly times calculated earlier in the algorithm are used for this check. As the ideal times are derived using best available resources, the actual times, to be calculated later, cannot be less than these times. Hence, if the check fails, there is no possibility that the operation might have fitted on a workstation when the actual times are calculated. If a workstation does not have the sufficient capacity for the operation, the algorithm creates another new workstation.

The system next identifies the resources required to undertake the assembly operation. The algorithm checks if the workstation has the required resources with the desired capability. If the resources are available they are used. However, if they are unavailable, the algorithm creates a set of tools that can undertake the operation from the resource pool. Resources are collectively stored in a main resource pool until they are assigned to a workstation, and can take the form of existing, new, or world-class assembly tools and machines. The best tool or machine from the pool is selected from the set, based on the fastest rate or quickest speed. The algorithm then re-calculates the assembly times using the actual resource data. This gives actual assembly operation times.

A final capacity check, identical to the initial capacity check, is then performed using 'actual' instead of 'ideal' assembly times. If a workstation does not have the sufficient capacity for the operation, the algorithm creates another new workstation. However, if the workstation does have sufficient capacity, the assembly operation is loaded onto the workstation. If a resource from the pool is used, the tool is also assigned to the workstation and removed from the main pool.

The system then selects the next operation from the assembly sequence and attempts to load it onto the same workstation, and the loading and balancing algorithm re-starts. Parallel workstations can be created in a similar manner, as explained in the Loading Existing Factories section.

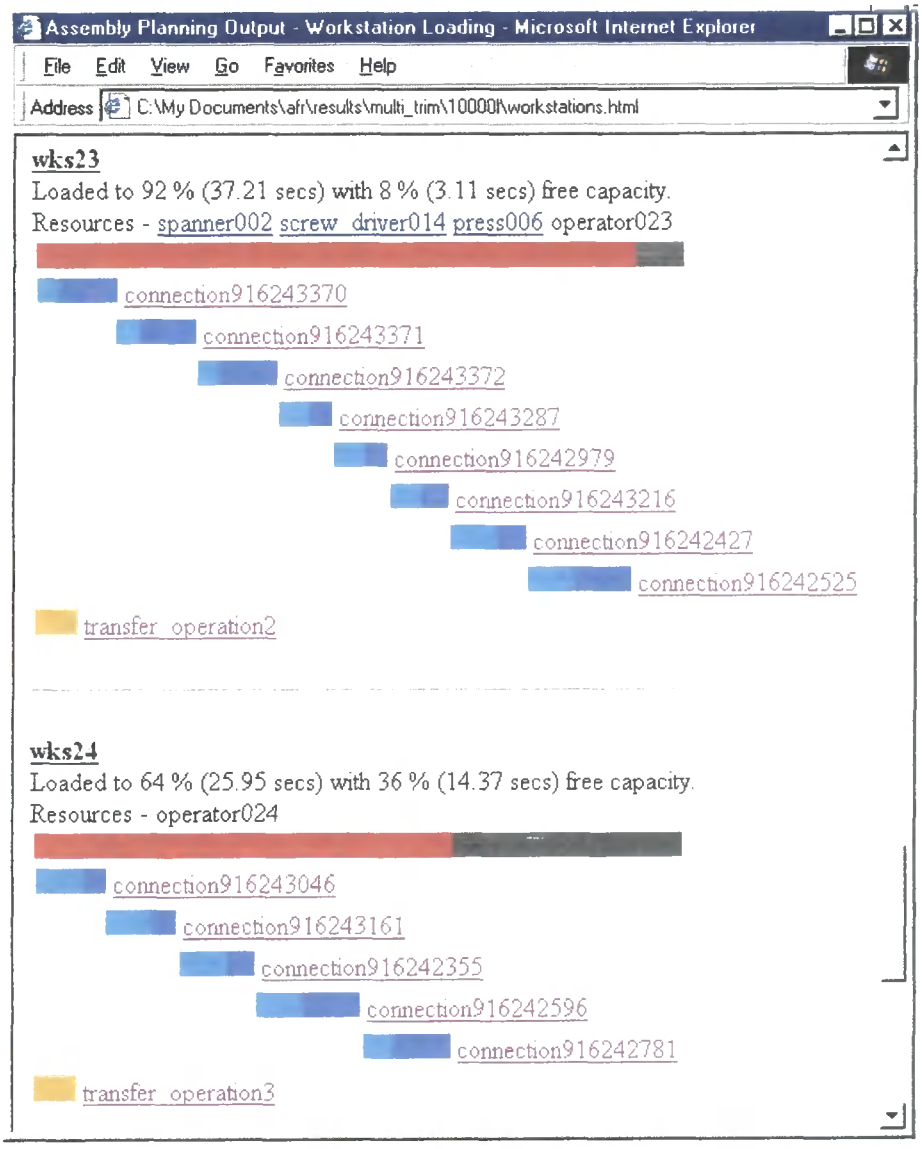


Figure 6-10: An Example HTML Results Page

6.7 Aggregate Assembly Process Planning Outputs.

Once the system has generated a set of aggregate assembly process plans, the results are outputted to five HTML files. HTML is a language used to write World Wide Web pages and using this format, allows the results to be viewed from any personal computer or workstation with an HTML browser. World Wide Web pages allow text,

graphics, colour and so forth to be viewed and are ideal for displaying graphical results. Each AAMP output HTML page shows a specific section of the results including: A summary of results; cell loadings; workstation loadings; assembly operations; and the assembly resources. An example of a results page is shown in Figure 6-10. Hyperlinks allow the user to jump between pages to associated data. For example, it is possible to select a workstation on the cell loading page and jump to the workstation loading page to view the workstation in more detail, and see the assembly operations and resources associated with this workstation.

6.8 Conclusions

An automated aggregate assembly process planning system has been developed which analyses the product model and generates an aggregate assembly process plan, using a model of the factory, assembly process models, and assembly data and process knowledge. The aggregate assembly process plan gives a general description of the assembly method for each of the assembly connections, suitable combinations of assembly processes and resources are identified, and a feasible sequence of operations is set out. The process plan is intended as a guide to indicate assembly operations, costs, times and resource requirements, which are outputted to a set of HTML World Wide Web files for viewing.

The automatic routing module generates a suitable sequence from the product structure and geometrical constraints, assembly knowledge, and process rules. The user is allowed to provide an input into the system to influence the sequence generation where preferences occur. On the other hand, the system is capable of producing a feasible sequence without reference to the user. A key strength of the aggregate assembly process planning algorithm is that alternative factory scenarios can be loaded and balanced, and the system outputs can be analysed by the user. This allows the user to consider his/her own existing factory configuration as well as new layouts, introduce improved resources into the factory, and even load supplier factories at an early stage in the design process. Automatically creating aggregate assembly process plans for both existing factory layouts and new layouts, using all existing resources within the factory, is a major function of aggregate assembly process planning.

Chapter Seven

Testing and Results

7.1 Introduction

This chapter details the work relating to the testing of the AAMP system. With a research project such as this, the design of an effective method of testing and evaluating the concepts and theories proposed gives rise to difficulties not found with more experimental work. In particular, it should be recognised that the most informative method of testing the work is not feasible. A production company would be taking unacceptable risks if it adopted an untried CAE tool as a major part of its product development strategy. It is therefore necessary to look for alternative ways of evaluating the methodology and strategies.

The principle evaluation strategy which was adopted in this project was to analyse the performance of the system when executing assembly product development tasks on example industrial products. The system outputs were then compared with the information available from traditional methods and industrial data. Data was gathered from an industrial partner, Company X, for a variety of product ranges. Although it was not always possible to breakdown industrial data for each individual assembly operation, analysis at a higher level was always possible. Two main strategies were used for testing the system and its functionality. The first approach was to model real products within the system, and the second was to generate aggregate assembly process plans for these products using the developed AAMP system.

Typical component designs can be extracted from a product and modelled in the system's aggregate product model. This enables the demonstration of the aggregate product model concept, structure and flexibility, by modelling a range of product geometries, standard parts and AFCs. Two approaches may be taken: Modelling of specific components to emphasise particular modelling functions; and modelling randomly selected products and components to test the generic nature of the model. For this project, the model was developed using complete products and components from Company X.

The aggregate assembly process planning function of the system can be demonstrated and tested through the use of example aggregate product models. It is necessary to test the following features and functions of aggregate assembly process planning:

- It can be applied to a variety of product model configurations.
- Suitable assembly processes and resources are selected and evaluated.
- The proposed sequences and routings are realistic.
- The aggregate assembly process plans produced are both technically feasible and realistic (i.e. no process or resource constraints are broken).
- It can produce alternative assembly options for the same design, using alternative sequences, processes and resources.
- Estimated assembly times and costs calculated are sufficiently accurate.
- Aggregate assembly process plans are produced in a sufficiently satisfactory way in an acceptable time scale. (The system is intended for use as a rapid evaluation tool, so that it may run as the design continually evolves).

Many of these criteria relate to the overall functionality of the system. There are others that relate more specifically to individual modules with the AAMP system. The aggregate assembly process planning evaluation can be divided into stages according to the main planning steps: Process identification and evaluation; sequencing generation; assembly resource selection; factory loading and balancing; and presentation of results.

7.2 Factory Model Used in Testing

In order to successfully run the system for testing purposes, a factory resource model is required. The assembly machines and tools were modelled according to the methods described in chapter five and used for this purpose. Figure 7-1 shows the overall layout and location of cells, workstations and assembly lines for a part of the Company X factory in County Durham. The area modelled is divided into six cells, as shown, and each cell is further broken down into a number of workstations. The workstations contain assembly equipment including presses and general manual assembly tools such as screwdrivers, socket-drivers and riveters, as shown in Table 7-1. Human resource is also assigned to each workstation. Resource data, such as process rates and speeds, are stored as properties attached to each machine or tool object on the resource model.

Separate palletised asynchronous assembly conveyors connect the workstations in each cell together. Four conveyors are present, linking workstations together in the Mini-Trimmer, Trimmer, Hedge-Trimmer and Cordless Trimmer cells in the modelled factory. Manual transfer between workstations is required in the petrol and standard Mower cells. Although each cell has been designed to assemble a specific product, the system can attempt to load a product on any cell. All resources can be removed from consideration by selecting them as unavailable. This is useful when restricting the loading to a specific area. In the examples discussed in this chapter, the aggregate assembly process planner was left free to select resources from the whole factory and also from specific areas.

Table 7-1: Factory Assembly Machines and Tools

Machines/Tools	Workstations
Presses	wks2, wks5, wks13, wks17, wks22, wks23, wks25, wks31, wks31
Screwdrivers	wks1, wks2, wks3, wks4, wks5, wks7, wks8, wks11, wks12, wks15, wks17, wks18, wks22, wks23, wks25, wks28
Socket-drivers	wks5, wks18
Spanners	wks5, wks23
Riveters	wks20, wks21

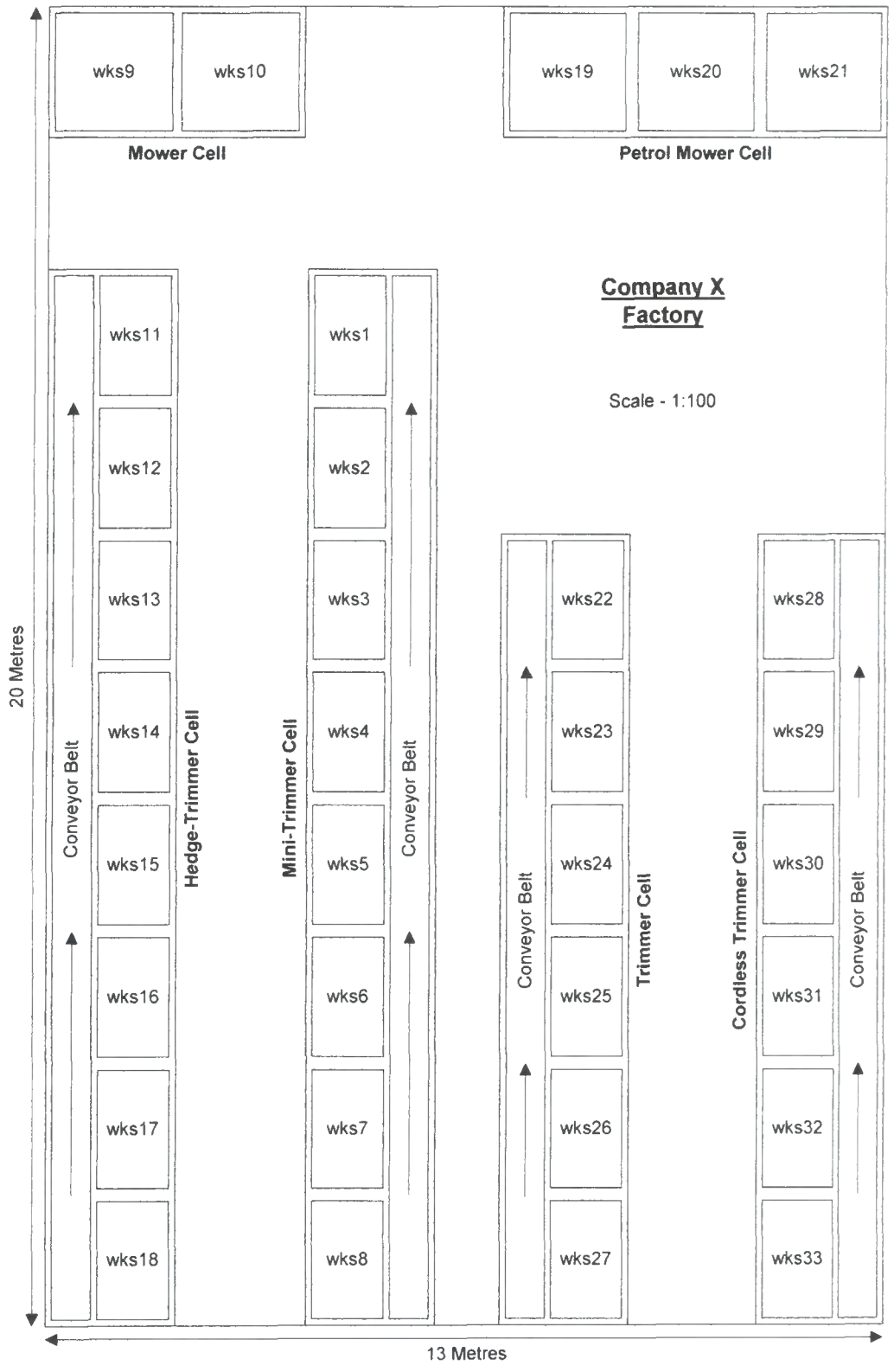


Figure 7-1: Company X Factory Layout

7.3 Product Model Tests

The product model was initially developed and tested by representing individual components which form part of a variety of products designed and manufactured by Company X. Further proving of the product model was then achieved by using the system to model a group of components to create sub-assemblies, and symbolise larger product structures. AFCs were next introduced into the product model to represent actual assembly joins. Finally, a range of Company X products were modelled incorporating all aspects of the system's modelling techniques, including the use of standard part libraries. These different products were chosen for the presence of a wide range of product model components and AFC types.

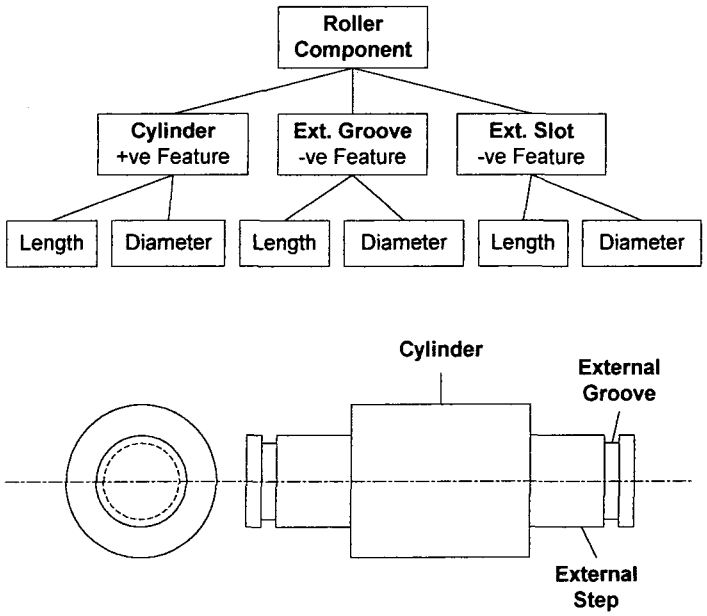


Figure 7-2: Roller Component

Figure 7-2 displays the product model for a roller component from a Company X Trimmer product. The overall dimensions of the component are a length of 100mm and a diameter of 40mm. The component is modelled using a single positive cylinder feature and four negative features. To create an aggregate product model, the correct selection of features is essential. It is possible that there is more than one positive feature that could represent the basic component shape. In these circumstances, the selection of the best feature should be made on the basis of preserving design intent. In this case, there is only one possible positive feature, the cylinder, that clearly represents

the roller component. Four negative features have been used to represent the steps and grooves at each end of the component. Two 'esp' features and two 'egv' features (see Appendix A for full descriptions of feature types) have been used to represent the external steps and external grooves on the cylindrical part. Dimensional detail for the positive and negative features are stored at a level below the features. Generic properties for the components and features are attached to their respective objects.

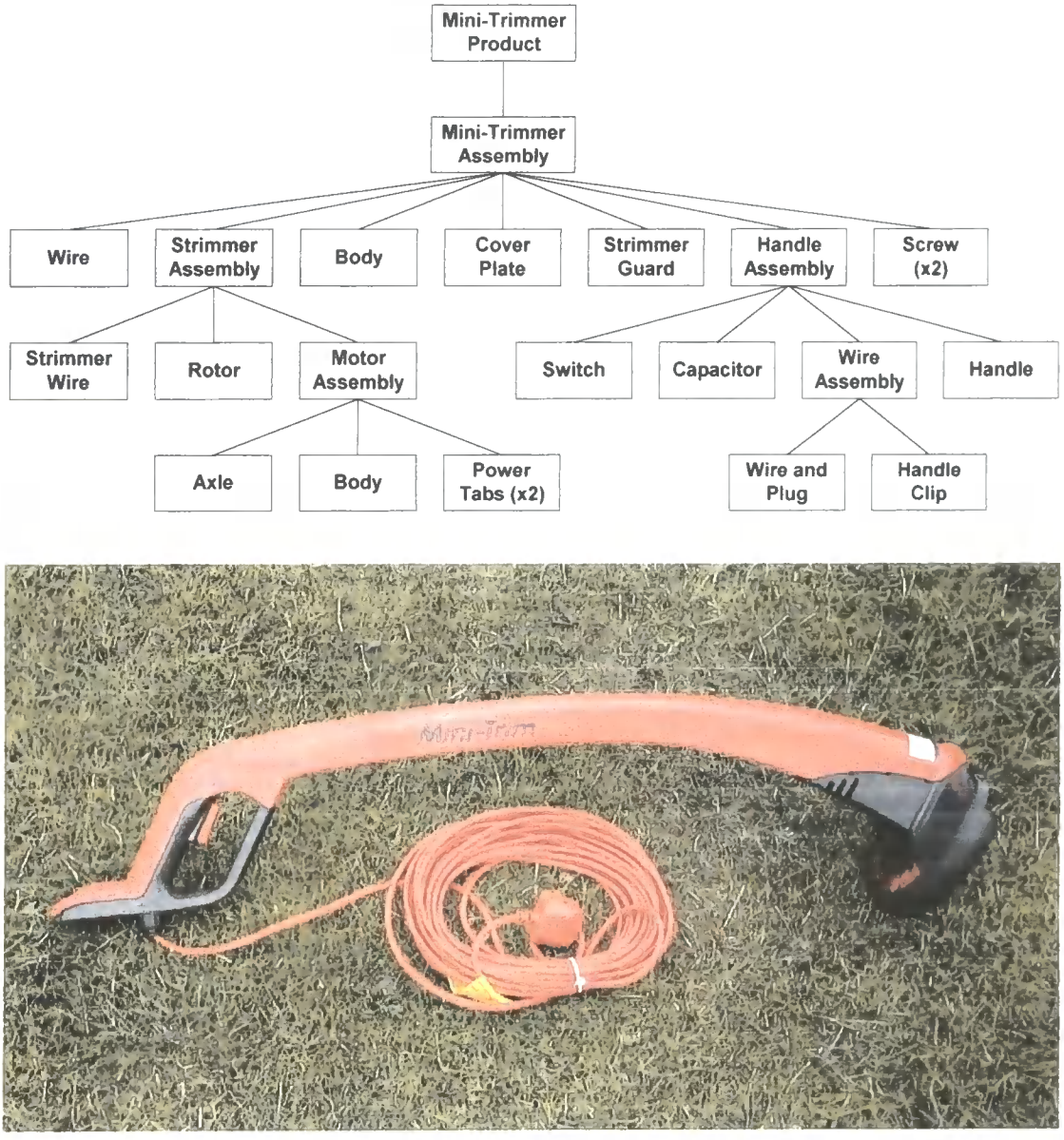


Figure 7-3: Mini-Trimmer Product and Structure

Figure 7-3 displays the assembly and component level of a Company X Mini-Trimmer product. The dimensions of the complete product are approximately a length of 800mm, a height of 200mm, and a width of 250mm. The Mini-Trimmer is composed of

seventeen components, one being the body part. The body is modelled using a moulded positive feature and a number of negative features. The body negative features are used to define additional features on the body to which other components are to be located, including 'htd' features, used to represent threads on a non-axial hole. The strimmer guard, handle, cover plate and handle clip are also represented using the moulded positive feature. The wire components are modelled using the wire positive feature, and the cylindrical positive feature is used to represent the motor body, axle and rotor components. The switch and capacitor are modelled using the prism positive feature. Finally, the sheet positive feature is used to represent the motor power connector tabs. Negative features are mainly used to model connection features on the Mini-Trimmer product model. Internal and external threads, holes and slots are examples of modelled negative features employed in this example. Dimensional details for all positive and negative features are also stored on the aggregate product model.

To compute accurate assembly sequences, it is essential that the bill of material hierarchical structure has been modelled correctly. This structure imposes important constraints on the possible assembly sequence. The Mini-Trimmer product hierarchy consists of four levels. At the top of the product model is the final assembled state of the Mini-Trimmer product. At each level beneath this, the product is broken down using sub-assemblies to group components together. The motor, rotor and strimmer wire components make up the strimmer sub-assembly. The motor is a standard part loaded from a product database. A requirement of standard parts is that they possess just the required product features, and hence, the motor product model consists of only a body, axle, and power connector tags. For the Mini-Trimmer product, the switch, capacitor and screws are also examples of standard parts loaded onto the model from databases. The handle sub-assembly is composed of the handle, switch and capacitor components, and the wire sub-assembly. The wire sub-assembly is made up of the wire and plug, and handle clip components.

AFC nodes are used on the Mini-Trimmer product model to define the features to be joined together to create assembly connections. For this example product, fifteen AFCs represent all the operations required to assemble the Mini-Trimmer. These AFC nodes are attached to the features they are joining, and also to the parent assembly level to

which both features are related. A summary of the AFC nodes is shown in Table 7-2. It highlights the connection components, the connection type, and the parent assembly level to which the AFC is attached.

Table 7-2: Mini-Trimmer Assembly Feature Connections

Connection	Components	Connection Type	Assembly Level
894629834	Switch, Handle	Plug and Target	Handle
894629933	Handle Clip, Wire and Plug	Plug and Target	Wire
894630236	Handle Clip, Handle	Snap Fit	Handle
894630294	Wire and Plug, Switch	Wiring - Screw	Handle
894630363	Black Wire, Switch	Wiring - Screw	Handle
894630517	Capacitor, Switch	Wiring - Tag	Handle
894630581	Handle, Body	Placement	Mini-Trimmer
894630772	Rotor, Motor	Plug and Target - Press	Strimmer
894630817	Strimmer Wire, Rotor	Placement	Strimmer
894630878	Motor, Body	Placement	Mini-Trimmer
894630919	Black Wire, Motor	Wiring - Screw	Mini-Trimmer
894630963	Cover Plate, Body	Placement	Mini-Trimmer
894631055	Screw, Body	Threaded	Mini-Trimmer
894631729	Screw, Body	Threaded	Mini-Trimmer
894631805	Strimmer Guard, Body	Snap Fit	Mini-Trimmer

As can be seen from the table, a range of connection types are modelled. Placement connections are used to join the cover plate, motor and handle to the main body, as well as locating the strimmer wire on the rotor. Snap fit connections are employed to model the fitting of the handle clip to the handle, and also the strimmer guard to the body. Locating and tightening the screws to the main body is modelled using the threaded connection. A plug and target connection is used to represent placing the motor axle into a hole on the rotor. A sub-class plug and target press type is automatically selected due to the tight tolerances of this connection. A plug and target connection type is also selected to model the wire and plug placement into the handle clip. Two different sub-class types of wiring connections are selected for joining the motor, switch, capacitor and wires together, these being the tag-wiring and screw-wiring AFC types. Table 7-2 also displays the parent assembly level to which each connection is attached. This information is used in the sequence generation algorithm as discussed later in this

chapter. Generic assembly information and specific connection data is stored as properties attached to each AFC node on the aggregate product model.

7.4 Testing the Aggregate Assembly Process Planning Functionality

The testing of the aggregate assembly process planning functionality will be described in two sections. Firstly, in this section, a case study will be used to demonstrate the system's functionality, focussing on the sequence generation algorithm, the resource loading and balancing algorithm, and assembly process calculations. In addition, a number of different scenarios will be used to demonstrate the effect of altering the sequence, available resources, capacity and required production rate on the outputted aggregate assembly process plan. In the second section, aggregate assembly process plans for three increasingly complex industrial products will be presented.

The aggregate assembly process plan outputs presented in this thesis have been formatted into tables and figures from the HTML files created by the AAMP system. The first of these tables displays the automatically generated assembly sequence. At this stage, a number of steps have been performed to compute the sequence. Initially, the outline sequence is generated from the hard constraints prior to the second sorting stage, where base parts, moving parts and process weighting are considered. Finally, the outline sequence and second stage sorting results are combined to give a feasible sequence of assembly operations. A part of an example sequence is presented in Table 7-3. Each row represents a single assembly operation step. The 'assembly level' column identifies the sub-assembly level on the product model to which the AFC is attached. The table also shows the connection reference number, the components to be joined, and the moving part for the connection. This sequence is presented to the user for confirmation, and the user is allowed to alter the sequence where preferences occur.

Table 7-3: Example of Sequence Generation Output

Connection	Components	Moving Part	Assembly Level
894630517	Capacitor, Switch	Capacitor	Handle
894630581	Handle, Body	Handle	Mini-Trimmer
894630772	Rotor, Motor	Rotor	Strimmer
894630817	Strimmer Wire, Rotor	Strimmer Wire	Strimmer

To generate the final aggregate assembly process plan, the system has undertaken a number of additional steps. Initially, ideal assembly times are computed using best in-house resources. These assembly times are used during the subsequent loading and balancing process stage. Here the system loads and balances the assembly operations onto the factory, using the previously computed sequence, ensuring that each loaded workstation has the required resources, capability and capacity. Actual assembly times are calculated during this stage using factory data from the loaded resource. Figure 7-4 displays graphically an example of a loaded assembly workstation. The figure shows the percentage and rate that the workstation is loaded, and the remaining free capacity. The available loading time at each workstation equates to the assembly cycle time. The assembly cycle time is the maximum time available at each workstation to achieve the production rate, which is calculated by dividing the time to complete production by the production volume. For the example in Figure 7-4, the assembly cycle time (twenty-six seconds) is the sum of the loaded rate (twenty-four seconds) and the free capacity (two seconds) of the workstation. The assembly and transfer operations are also shown in the figure in the form of a Gantt chart. Assembly operations are broken down into their handling and insertion/process time elements and displayed in the sequence that they are loaded onto the factory. The green portion of the operation time represents the handling assembly time, and the blue portion represents the insertion/process assembly time.

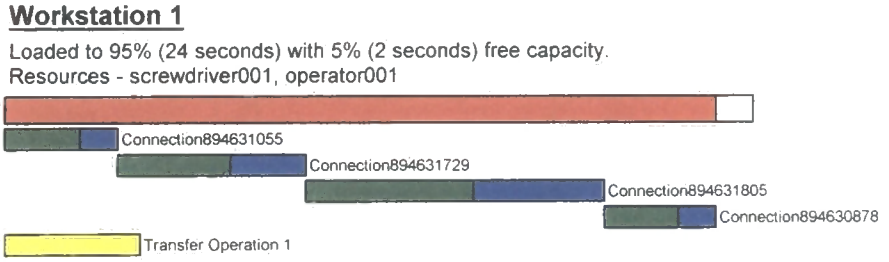


Figure 7-4: Example Workstation Loading

Another table displays a breakdown of the data for each assembly operation, with each row representing a single assembly connection. The table shows the connection reference number, the connection type and the resources employed. It also shows the handling, insertion/process and total assembly time, and the assembly cost for each connection. A summary of each generated aggregate assembly process plan is also

presented, including: The total assembly operation, transfer and lead times; the assembly critical path and critical assembly time; the total assembly cost; the average workstation efficiency; and production data.

7.4.1 Sequence Generation Algorithm

The case study product used to demonstrate the aggregate assembly process planning functionality is the Company X Mini-Trimmer, also used during the earlier product model testing. The Mini-Trimmer product model consists of seventeen components and fifteen AFCs. Figure 7-3, shown earlier, displays the product structure, and the assembly and component levels of the Mini-Trimmer. The sequence generation algorithm functions in two sections. Initially, an outline assembly plan is generated from the hard constraints. The main hard constraint is the location on the product model at which the AFCs are attached. AFCs are created by linking features together, and each connection is additionally attached to the parent assembly level to which both features are related.

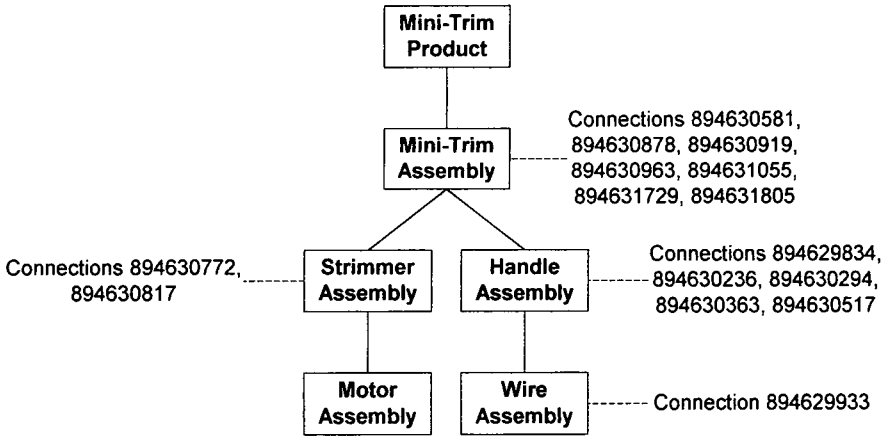


Figure 7-5: Mini-Trimmer Product Connectivity Model

Figure 7-5 displays the product connectivity model for the Mini-Trimmer product. From this figure we can see that the assembly connection 894629933, attached to the wire assembly, should be carried out prior to the connections attached to the handle assembly. In addition, these connections should be undertaken before the connections attached to the Mini-Trimmer assembly. It can also be seen that connections 894630772 and 894630817, attached to the strimmer assembly, should be undertaken prior to the

connections attached to the Mini-Trimmer assembly. The result of the first part of the sequence algorithm is an outline of the aggregate assembly process plan, with groupings of assembly connections at various stages. These groups are then ordered in the second part of the algorithm.

The second stage involves a number of tasks. Initially, the moving part for each connection is derived before the base part for each set of connection groupings is found. The connections are next assigned process weightings, and ordered accordingly. Examining the group of connections attached to the Mini-Trimmer assembly, it can be seen in Table 7-4 that six of the seven connections involve attaching a component or sub-assembly to the main body component. The body is one of the main parts of the Mini-Trimmer due to its size, volume, and the number of connections attached to it. Hence, it is obvious for these six connections that the moving part is the other component or sub-assembly in the join, namely the cover plate, strimmer guard, motor, handle and screws. From these results, we can also derive that the base part at this level on the product model is the body. The system computed the base part at the strimmer assembly and handle assembly level as the motor and handle respectively, which appeared to be logical. Table 7-4 displays the moving part for each connection obtained by the algorithm.

The next stage in the algorithm is applying a process priority index to each of the connections. This index is based on process weightings which are determined by rules corresponding to assembly theory. One such rule is that fastening operations should be carried out after placement operations. An example of this rule is found at the Mini-Trimmer level on the model, where a threading operation, connection 894631055, is carried out after a number of placement operations. Another example of a process weighting is that an operation requiring force, e.g. connection 894630772, should be carried out prior to placement operations, e.g. connection 894630817.

The final stage of the sequencing algorithm is combining the outline process plan and the second stage sequences to give a final assembly sequence for the complete Mini-Trimmer product. Table 7-4 displays the generated sequence obtained by the AAMP system in reverse order. Studying the outputted assembly sequence, the results appear to be in a logical and feasible order. Comparing it to the actual sequence in which

Company X assembles the product, there is only one difference. Company X locates the middle cover plate at the end of the sequence after the screws have been inserted and tightened, whereas the sequence generated by the AAMP system reverses these two operations. Both sequences are feasible, and depend on the preference of the assembly process planner. After examining the generated sequence, the user can either confirm that the sequence is functional, or make desirable adjustments if necessary.

Table 7-4: Mini-Trimmer Sequence

Connection	Components	Moving Part	Assembly Level
894631055	Screw, Body	Screw	Mini-Trimmer
894631729	Screw, Body	Screw	Mini-Trimmer
894630963	Cover Plate, Body	Cover Plate	Mini-Trimmer
894631805	Strimmer Guard, Body	Strimmer Guard	Mini-Trimmer
894630919	Black Wire, Motor	Black Wire	Mini-Trimmer
894630878	Motor, Body	Motor	Mini-Trimmer
894630581	Handle, Body	Handle	Mini-Trimmer
894630817	Strimmer Wire, Rotor	Strimmer Wire	Strimmer
894630772	Rotor, Motor	Rotor	Strimmer
894630294	Wire and Plug, Switch	Wire and Plug	Handle
894630363	Black Wire, Switch	Black Wire	Handle
894630517	Capacitor, Switch	Capacitor	Handle
894629834	Switch, Handle	Switch	Handle
894630236	Handle Clip, Handle	Handle Clip	Handle
894629933	Handle Clip, Wire and Plug	Handle Clip	Wire

7.4.2 Loading and Balancing Algorithm

The aims of the loading and balancing algorithm are to assign all assembly operations to resources within a factory (whilst ensuring the resources have the capability and capacity), select the best resources, utilise the workstations efficiently, and ensure that the sequence is maintained. In this section, six loading and balancing scenarios are discussed in detail including: An efficient loading; an inefficient loading; workstation overloading; loading parallel workstations; loading new workstations; and finally combining the loading of existing and new workstations. The factory resource to be loaded is the Mini-Trimmer cell from the Company X factory. The cell consists of eight workstations, each with a variety of assembly machines, tools and human resource, as

shown earlier in Figure 7-1. A palletised asynchronous assembly conveyor connects the workstations together in the cell. Prior to the loading and balancing algorithm commencing, ideal assembly times are calculated for each of the operations, using best in-house resources. These ideal times can only be increased when actual resources are used. Therefore, ideal times can be used for an initial check to verify if a workstation has the available loading capacity.

7.4.2.1 Efficient Workstation Loading

The production data entered into the system by the user for this aggregate assembly process plan is a volume of fourteen thousand and four hundred Mini-Trimmers, to be assembled in one week, using two eight-hour length shifts. From these values, an assembly line cycle time, the maximum time available at each workstation to achieve the production rate, is calculated to be twenty-eight seconds. The assembly operations are now loaded onto the factory resource using the earlier derived sequence.

The first operation to be loaded on the factory is connection 894631055, a threaded connection. The system attempts to load this operation on workstation one in the Mini-Trimmer cell. The algorithm initially confirms that the workstation has the available capacity using the ideal assembly time. Next, the system identifies all the factory resources suitable to undertake the operation, and matches this set to the resources available at the current workstation. At workstation one, there is an internal driver tool, screwdriver 001, suitable for selection. The system then checks that the driver tool has the capability to tighten the screw to the required torque. The 'actual' assembly time is next calculated using the process rate of screwdriver 001. The final check that the algorithm performs is a final capacity check, using the 'actual' assembly operation time. As all of these checks have been successful, the assembly connection 894631055 is loaded onto workstation one. The system now selects the next connection in the sequence and attempts to load it onto the same workstation.

Connections 894631729, 894630963, 894631805 and 894630919 are also successfully loaded onto workstation one. The system then attempts to load connection 894630878 onto workstation one. This is unsuccessful because the remaining available capacity of workstation one is less than the assembly operation time for this connection. The

system then selects the next workstation in the Mini-Trimmer cell, workstation two, and attempts to load the assembly operation onto this workstation. Connections 894630878, 894630581, 894630817, 894630772 and 894630294 are successfully loaded onto workstation two before it runs out of available capacity. Lastly, connections 894630363, 894630517, 894629834, 894630236 and 894629933 are loaded onto workstation three.

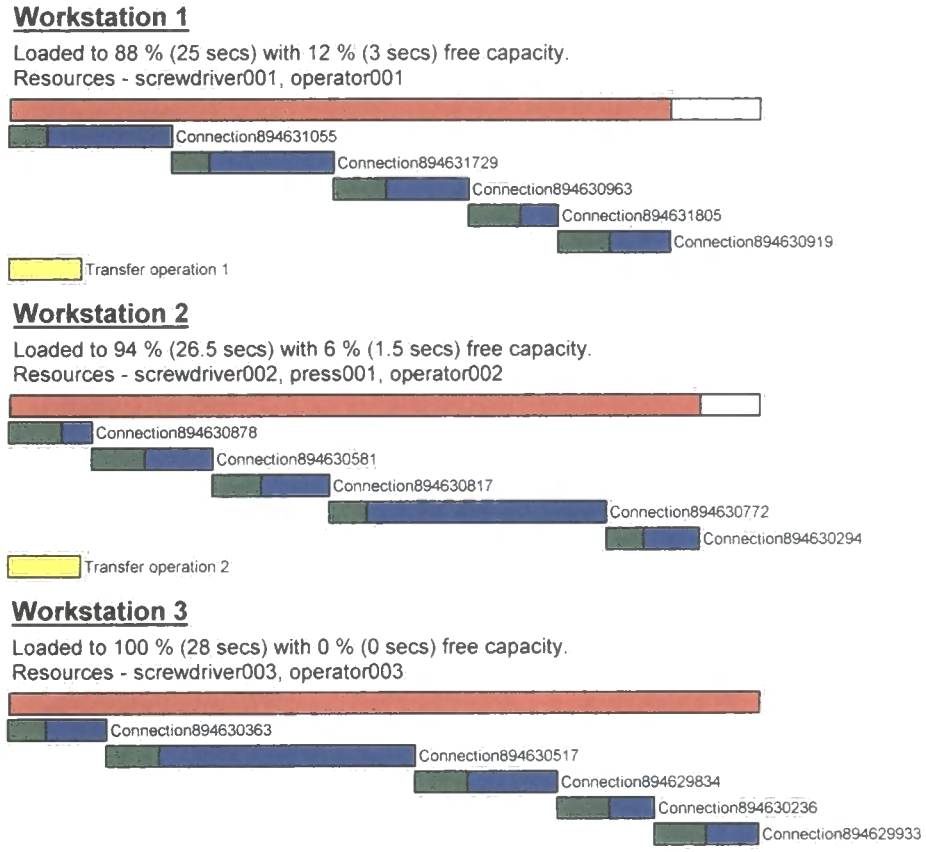


Figure 7-6: Workstation Loading

Figure 7-6 displays the loadings of workstations one, two and three. From this Gantt chart we can see that workstation one is loaded to eighty-eight per cent of its available capacity, workstation two to ninety-four per cent, and workstation three is one hundred per cent fully loaded. This gives an average efficient workstation loading of ninety-four per cent. Analysing Figure 7-6, we can see that without changing the assembly sequence or resource, workstation three is the bottleneck workstation for this scenario. Hence, if we increase the production volume, or reduce the number or length of the shifts, then we would require more than three workstations to be loaded.

Table 7-5 displays a breakdown of the assembly times and costs for each operation. The table shows the handling, insertion/process and total assembly times, and assembly cost for each connection. Summing the assembly times gives a total assembly time for the Mini-Trimmer operations of seventy-nine and a half seconds. Adding to this the transfer operation times gives a total operation time of eighty-five seconds. The lead time is the actual time to complete all operations and transfers in an industrial scenario. The assembly lead time is calculated by multiplying the cycle time of the bottleneck workstation by the number of workstations loaded, and adding to this the total transfer time. This gives an assembly lead time of eighty-nine and a half seconds. Summing the assembly costs for each connection results in a total assembly cost of seventy-six pence.

Table 7-5: Mini-Trimmer Assembly Times and Costs

Connection	Type	Tool	t_h	t_i/t_p	t_t	Cost
894631055	Threaded	Screwdriver 001	1.5	4.6	6.1	£0.06
894631729	Threaded	Screwdriver 001	1.5	4.6	6.1	£0.06
894630963	Placement	-	2.0	3.1	5.1	£0.05
894631805	Snap Fit	-	2.0	1.5	3.5	£0.03
894630919	Wiring - Screw	Screwdriver 001	1.9	2.3	4.2	£0.04
894630878	Placement	-	1.9	1.5	3.4	£0.03
894630581	Placement	-	2.0	2.5	4.5	£0.04
894630817	Placement	-	1.8	2.5	4.3	£0.04
894630772	Plug and Target - Press	Press 001	1.5	9.0	10.5	£0.12
894630294	Wiring - Screw	Screwdriver 002	1.5	2.3	3.8	£0.04
894630363	Wiring - Screw	Screwdriver 002	1.5	2.3	3.8	£0.04
894630517	Wiring - Tag	-	1.9	9.6	11.5	£0.10
894629834	Plug and Target	-	2.0	3.5	5.5	£0.05
894630236	Snap Fit	-	1.9	1.5	3.4	£0.03
894629933	Plug and Target	-	2.0	1.8	3.8	£0.03

The critical assembly path and time is found by searching through the product assembly model to find the network path with the longest assembly time. The critical assembly path for the Mini-Trimmer is displayed in Figure 7-7, with the grey assembly nodes representing the critical path. Summing the assembly times for all the connections on the critical path results in a critical assembly path time of sixty-four and a half seconds.

To verify the accuracy of the outputted assembly times from the AAMP system, it was important to compare these times with actual factory assembly process times. Although

it was not possible to accurately measure and compare each individual operation time, analysis at a higher level was possible. A number of measurements were taken of the complete Mini-Trimmer assembly process with different personnel. The average factory assembly process time for the complete Mini-Trimmer product was found to be ninety-one seconds. This is comparable to the AAMP result of eighty-five seconds, a difference in times of just seven per cent.

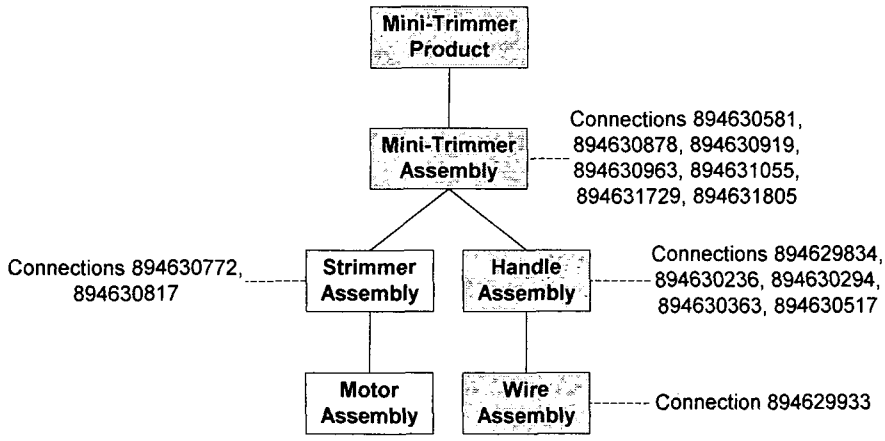


Figure 7-7: Mini-Trimmer Critical Assembly Path

7.4.2.2 Inefficient Workstation Loading

The above scenario demonstrates an efficient assembly line loading and balancing example, with an average workstation loading of ninety-four per cent. Using identical production data and assembly sequence as the first example, the effect of relocating resources is demonstrated in this second scenario. The only modification to the resource model is moving press 001 from workstation two to workstation three. The assembly line cycle time, the maximum time available at each workstation to achieve the production rate, is again calculated to be twenty-eight seconds. Connections 894631055, 894631729, 894630963, 894631805 and 894630919 are loaded onto workstation one before it runs out of available capacity. Connections 894630878, 894630581 and 894630817 are next loaded onto workstation two. Although workstation two has the available capacity for connection 894630772, it does not have the required press resource. Connection 894630772 is loaded onto workstation three as this has a press machine. Connections 894630294 and 894630363 are also loaded onto

workstation three before it runs out of available capacity. Finally, connections 894630517, 894629834, 894630236 and 894629933 are loaded onto workstation four.

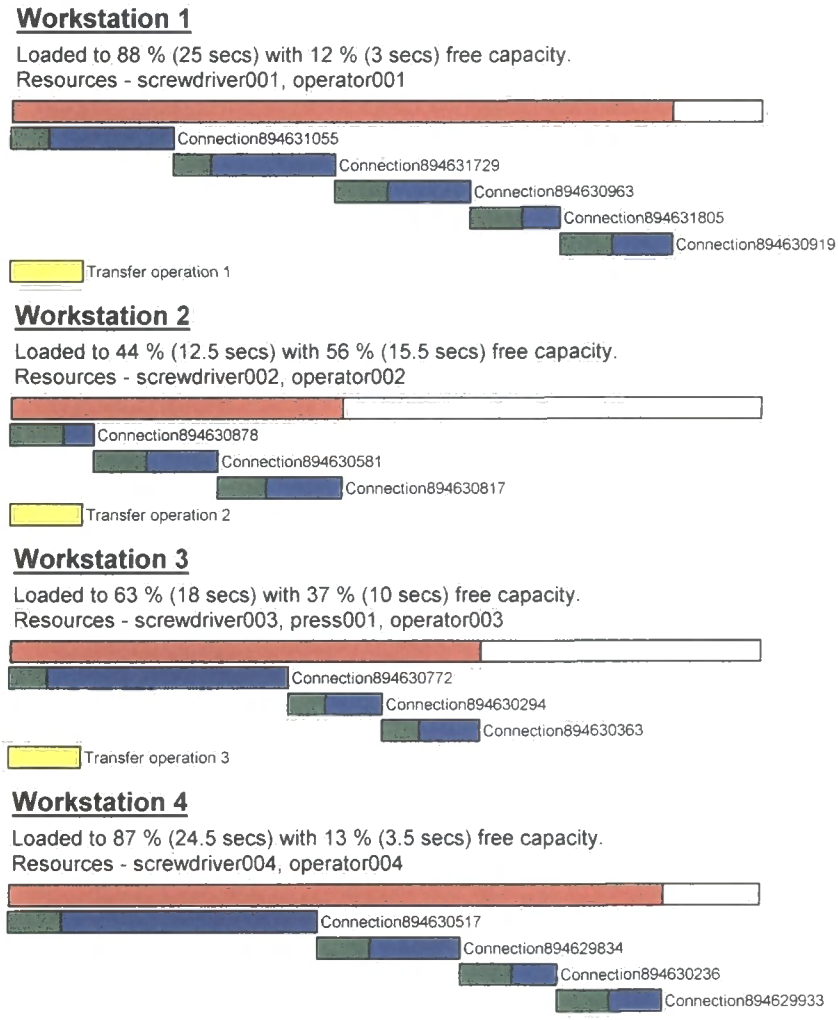


Figure 7-8: Inefficient Workstation Loading

Figure 7-8 displays the loading of workstations one to four. From this it can be seen that workstation one is loaded to eighty-eight per cent of its available capacity, workstation two to forty-four per cent, workstation three to sixty-three per cent and workstation four to eighty-seven per cent. This results in an average workstation loading of seventy and a half per cent. The total assembly time for this example is eighty seconds. Adding to this the transfer operation times results in a total operation time of eight-eight seconds. The lead time for this example is one hundred and eight seconds. Table 7-6 shows a comparison between the results of the efficient loading and inefficient loading scenarios. It can be seen that the total assembly times for both examples are very similar. The minor difference is due to different tools being utilised. The total transfer time for the inefficient loading is higher because of the additional transfer operation

required between workstations three and four. A significant difference can be seen in the assembly lead times and workstation efficiencies for each scenario. The assembly lead time is approximately twenty per cent greater, and the workstation loading is sixteen per cent lower for the inefficient example compared to the efficient scenario. These differences are because the inefficient example loads an additional workstation to achieve the production volume. This example shows the importance of correctly designing assembly lines and populating workstations with the required assembly resources.

Table 7-6: Efficient and Inefficient Loading Results

	Efficient Loading	Inefficient Loading
Total Assembly Time	79.5 seconds	80 seconds
Total Transfer Time	5.5 seconds	8 seconds
Total Operation Time	85 seconds	88 seconds
Assembly Lead Time	89.5 seconds	108 seconds
Average Workstation Loading	94 %	70.5 %

7.4.2.3 Workstation Overloading

This scenario demonstrates the effect of greatly increasing the required production volume whilst maintaining the same time period for assembling the Mini-Trimmers. The production data entered into the system by the user for this aggregate assembly process plan is a volume of twenty-five thousand Mini-Trimmers, to be assembled in one week, using two eight-hour length shifts. From these values an assembly line cycle time, the maximum time available at each workstation to achieve the production rate, is calculated to be sixteen seconds. To add extra complexity to this scenario, workstation six is made unavailable for selection. The assembly operations are now loaded onto the factory resource using the earlier derived sequence.

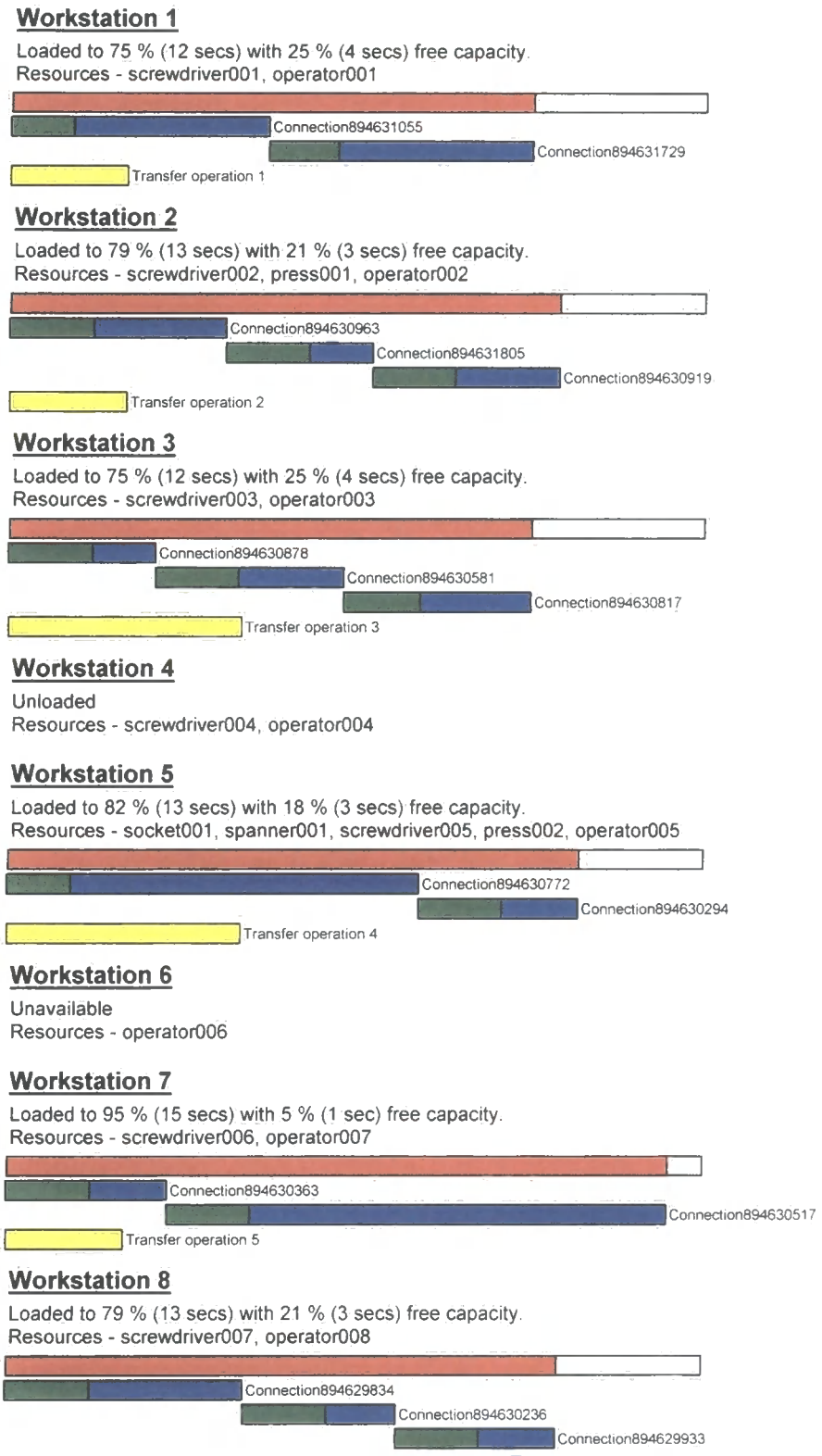


Figure 7-9: Heavy Workstation Loading

From Figure 7-9, we can see that connections 894631055 and 894631729 are loaded onto workstation one, connections 894630963, 894631805 and 894630919 are loaded onto workstation two, and connections 894630878, 894630581 and 894630817 are

loaded onto workstation three, before each workstation runs out of available capacity. Although workstation four has the available capacity for connection 894630772, it does not have the required press resource. Connection 894630772 is loaded onto workstation five as this has a press machine. Connection 894630294 is also loaded onto workstation five. No operations are loaded onto workstation six as this resource is unavailable. Finally, connections 894630363 and 894630517, are loaded onto workstation seven and connections 894629834, 894630236 and 894629933 are loaded onto workstation eight.

Table 7-7: Efficient and Overloading Results

	Efficient Loading	Overloading
Total Assembly Time	79.5 seconds	78 seconds
Total Transfer Time	5.5 seconds	19 seconds
Total Operation Time	85 seconds	97 seconds
Assembly Lead Time	89.5 seconds	109 seconds
Average Workstation Loading	94 %	81 %

Table 7-7 shows the contrast between the assembly, transfer, total operation and lead times, and average workstation loadings for the efficient and overloaded scenarios. It can be seen that that the total assembly times for both examples are comparable. The total transfer time for the overloaded scenario is higher because of the additional transfer operations required. A significant difference can be seen in the assembly lead times and workstation efficiencies for each scenario. The assembly lead time is almost twenty seconds greater, and the workstation loading is thirteen per cent lower for the overloaded example compared to the efficient scenario. These differences are because it is more difficult to efficiently balance an assembly line with a lower cycle time. It would be advisable in such a scenario to split the assembly line in two and assemble the Mini-Trimmer product in parallel. Doubling the assembly cycle time would result in a smoother, more efficient assembly line, with a lower assembly lead time.

7.4.2.4 Parallel Workstation Loading

In some cases an individual assembly operation time will be greater than the assembly cycle time. The solution to this problem is to load the assembly operation on a number of mirrored parallel workstations to maintain the required production rate. This scenario

demonstrates such an occurrence. The production data entered into the system by the user for this aggregate assembly process plan is a volume of ten thousand Mini-Trimmers, to be assembled in one week, using two eight-hour length shifts. From these values an assembly line cycle time is calculated to be forty seconds. To aid demonstrating parallel loading a modification is made to the resource model, increasing the process time of press 001 from nine seconds to fifty seconds.

Connections 894631055, 894631729, 894630963, 894631805, 894630919, 894630878, 894630581 and 894630817 are loaded onto workstation one. Connection 894630772 cannot be loaded onto workstation one as there is not the required press resource. Although workstation two has the required resource and is unloaded, loading fails as there is not the required capacity. This is because the assembly operation time is greater than the assembly cycle time, and hence, parallel workstation loading is required to maintain the production rate. The number of parallel workstations required is calculated by dividing the assembly operation time (fifty-two seconds) by the assembly cycle time (forty seconds) and rounding it up to the nearest integer. For this example, two parallel workstations are required. Connection 894630772 is now loaded onto workstation two. Connection 894630772 is unsuccessfully loaded onto workstations three and four, because these workstations do not have the required press resource. However, connection 894630772 is also loaded onto workstation five.

It must be observed that as there is now more than one workstation loaded with a particular assembly operation, the amount that each of the parallel workstations is loaded is calculated by dividing the assembly operation time by the number of loaded parallel workstations. It can be seen from Figure 7-10 that connection 894630772 loads workstation two by twenty six seconds, and also loads workstation five by eight and a half seconds. The large difference in loading rates is caused by the different process rates of presses 001 and 002. In a similar manner that normal workstations are loaded to their full capacity, the algorithm also attempts to load all of the parallel workstations with as many connections as is feasible. Connections 894630294, 894630363, 894630517, 894629834 and 894630236 are also loaded onto both workstations two and five. Connection 894629933 is not loaded onto both workstations because workstation

two does not have the required capacity. However, connection 894629933 is loaded onto workstation five back in the single workstation loading mode.

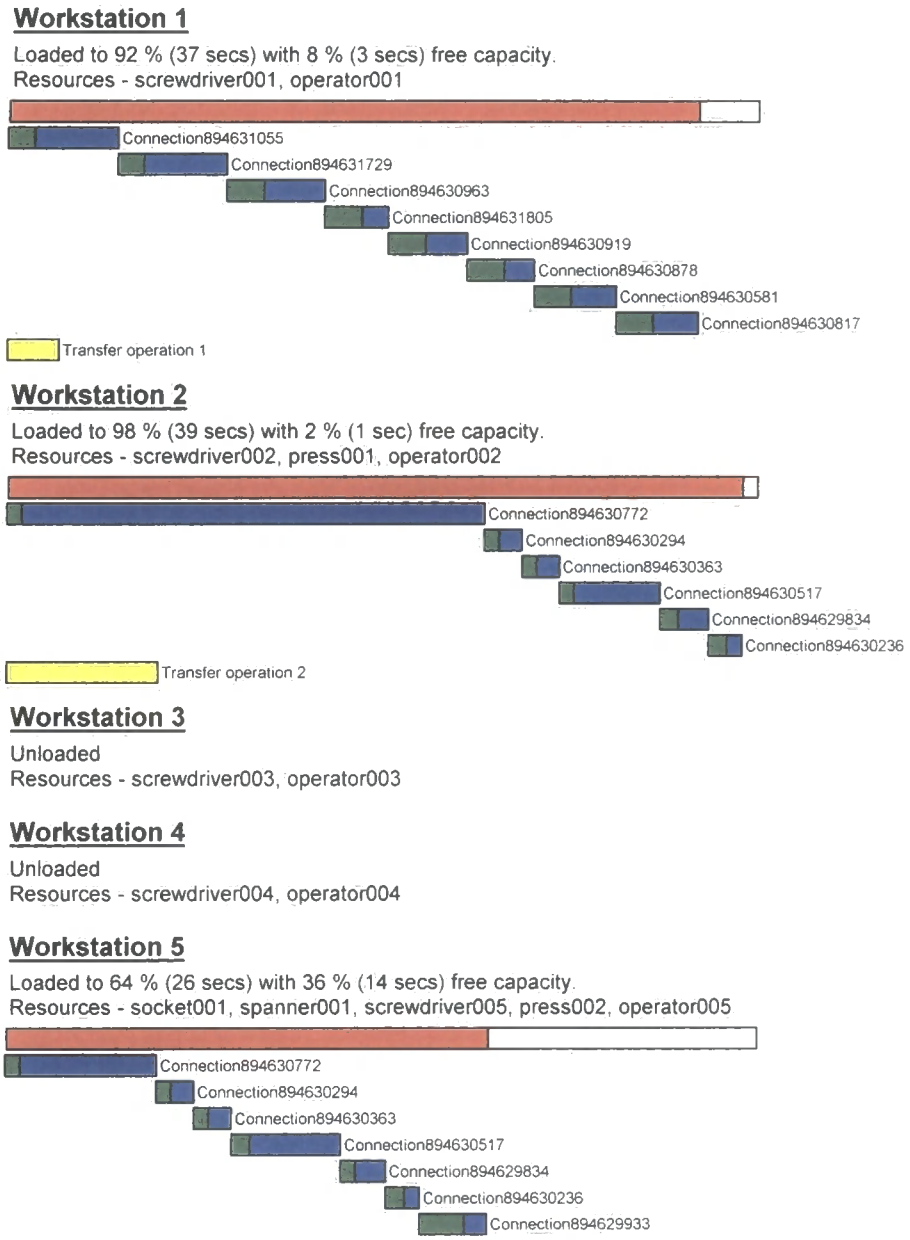


Figure 7-10: Parallel Workstation Loading

Figure 7-10 displays the loading of workstations one to five. From this, we can see that workstation one is loaded to ninety-two per cent of its available capacity, workstation two to ninety-eight per cent, workstations three and four are unloaded, and workstation five to sixty-four per cent of its available capacity. This results in an average workstation loading of eighty-five per cent. The total assembly time for this example is one hundred and two seconds. Adding to this the transfer operation times, results in a total operation time of one hundred and thirteen seconds. The lead time for this

example is one hundred and twenty-eight seconds. We cannot compare these results with other scenarios because different assembly resource process data has been used for this aggregate assembly process plan.

7.4.2.5 Loading New Workstations

During the design of a new product, it is usual that effort will be spent redesigning or designing new assembly lines. Whereas an existing factory has restrictions on the location of assembly tools and machines at each workstation, a new factory can be designed for a specific product to gain the minimum lead time and maximum throughput. This section displays the results of creating new workstations for the Mini-Trimmer product. The production data entered into the system by the user for this aggregate assembly process plan is a volume of fifteen thousand and eight hundred Mini-Trimmers, to be assembled in one week, using two eight-hour length shifts. From these values an assembly line cycle time is calculated to be twenty-five seconds. Resources are stored in a main pool until they are assigned to a workstation, and can take the form of existing, new or world-class assembly resources. For this example, existing assembly resources are utilised, allowing the system to select any resource from the complete Company X factory.

Initially, a new empty workstation is created with an operator. The first operation to be loaded on the new assembly line is connection 894631055, a threaded connection requiring an internal driver tool. The algorithm initially confirms that the new workstation has the available capacity using the ideal assembly operation time. Because the new workstation has not currently been assigned a resource, an internal driver with the desired capability has to be selected from the main resource pool. An internal driver, screwdriver 008 is selected based on the fastest process rate. The 'actual' assembly time is next calculated using the process rate of screwdriver 008. The final check that the algorithm performs is a secondary capacity check using the 'actual' assembly operation time. Because all of these checks have been successful, the assembly connection 894631055 is loaded onto new workstation one. Screwdriver 008 is assigned to new workstation one and removed from the main resource pool. The system now selects the next operation in the sequence, connection 894631729, and attempts to load it onto the same new workstation. Connection 894631729 also requires

an internal driver. As new workstation one has already been assigned an internal driver, this is utilised rather than selecting another tool from the resource pool. Connections 894631729, 894630963, 894631805, 894630919 and 894630878 are also loaded onto new workstation one before it runs out of capacity. Two more new workstations are created to load the remaining connections. New workstation two is assigned a press machine and an operator, and new workstation three is assigned an internal driver tool and an operator.

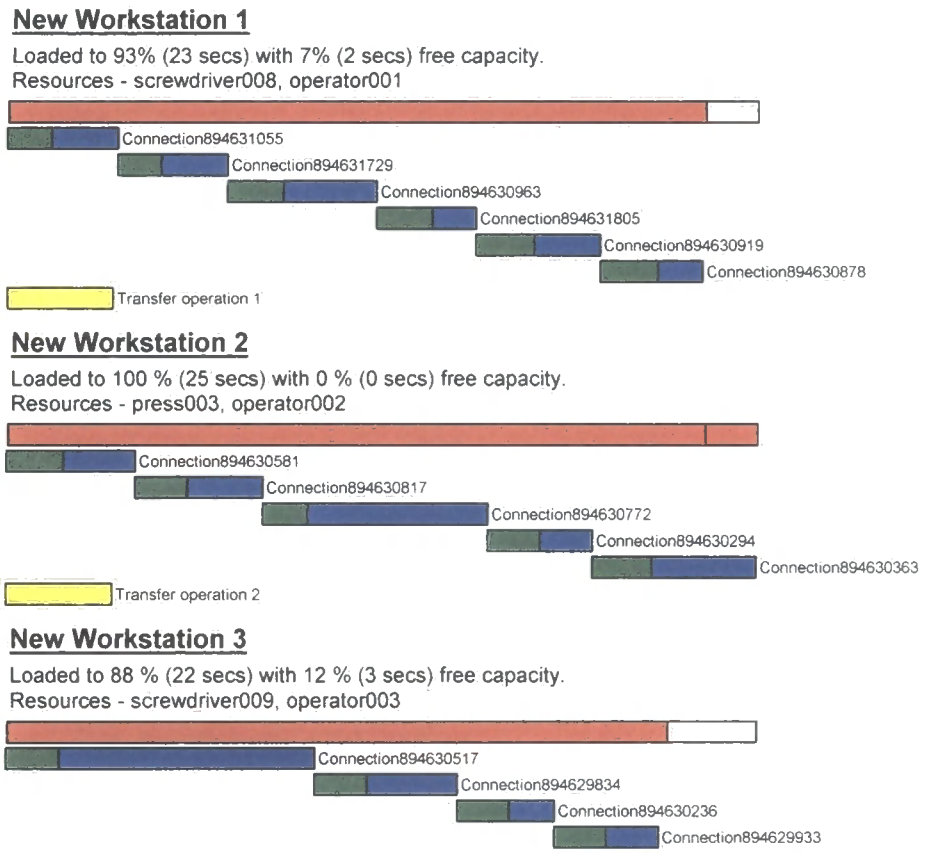


Figure 7-11: New Workstation Loading

Figure 7-11 displays the loadings of new workstations one, two and three. From this we can see that new workstation one is loaded to ninety-three per cent of its available capacity, new workstation two is one hundred per cent fully loaded, and new workstation three is loaded to eighty-eight per cent. This gives an average efficient workstation loading of ninety-four per cent. Table 7-8 shows the contrast between the assembly, transfer, total operation and lead times, the average workstation loadings, and the production volumes for the efficient and new workstation loading scenarios. It can be seen that the total assembly, total operation and assembly lead times are slightly

lower for the new workstation loading compared to the efficient loading scenario. These differences are because the new workstation example selects and assigns the best resources from the complete Company X factory pool of resources to the new workstations, whereas the efficient loading example is restricted to the resources already situated in the existing Mini-Trimmer cell workstations. Table 7-8 also displays the maximum production throughput using three workstations for both the efficient and new workstation loading scenarios.

Earlier, the efficient loading scenario demonstrated that the maximum production volume using three workstations for the Mini-Trimmer cell was fourteen thousand and four hundred. However, by redesigning the assembly line, the production volume can be increased to fifteen thousand and eight hundred Mini-Trimmers whilst still loading three workstations.

Table 7-8: Efficient and New Workstation Loading Results

	Efficient Loading	New Workstation Loading
Total Assembly Time	79.5 seconds	70 seconds
Total Transfer Time	5.5 seconds	8 seconds
Total Operation Time	85 seconds	78 seconds
Assembly Lead Time	89.5 seconds	83 seconds
Average Workstation Loading	94 %	94 %
Production Volume	14400	15800

7.4.2.6 Mixed Workstation Loading

Whilst loading an existing factory, there will be occasions when a high production volume and/or a low number of available workstations will result in the factory resources not having sufficient capacity. If this situation occurs, it is desirable for the system to load the existing factory, and then design and load new workstations for the remaining assembly operations. The final scenario demonstrates a combination of loading existing and new workstations. The production data entered into the system by the user for this aggregate assembly process plan is a volume of twenty-five thousand Mini-Trimmers, to be assembled in one week, using two eight-hour length shifts. From these values an assembly line cycle time is calculated to be sixteen seconds.

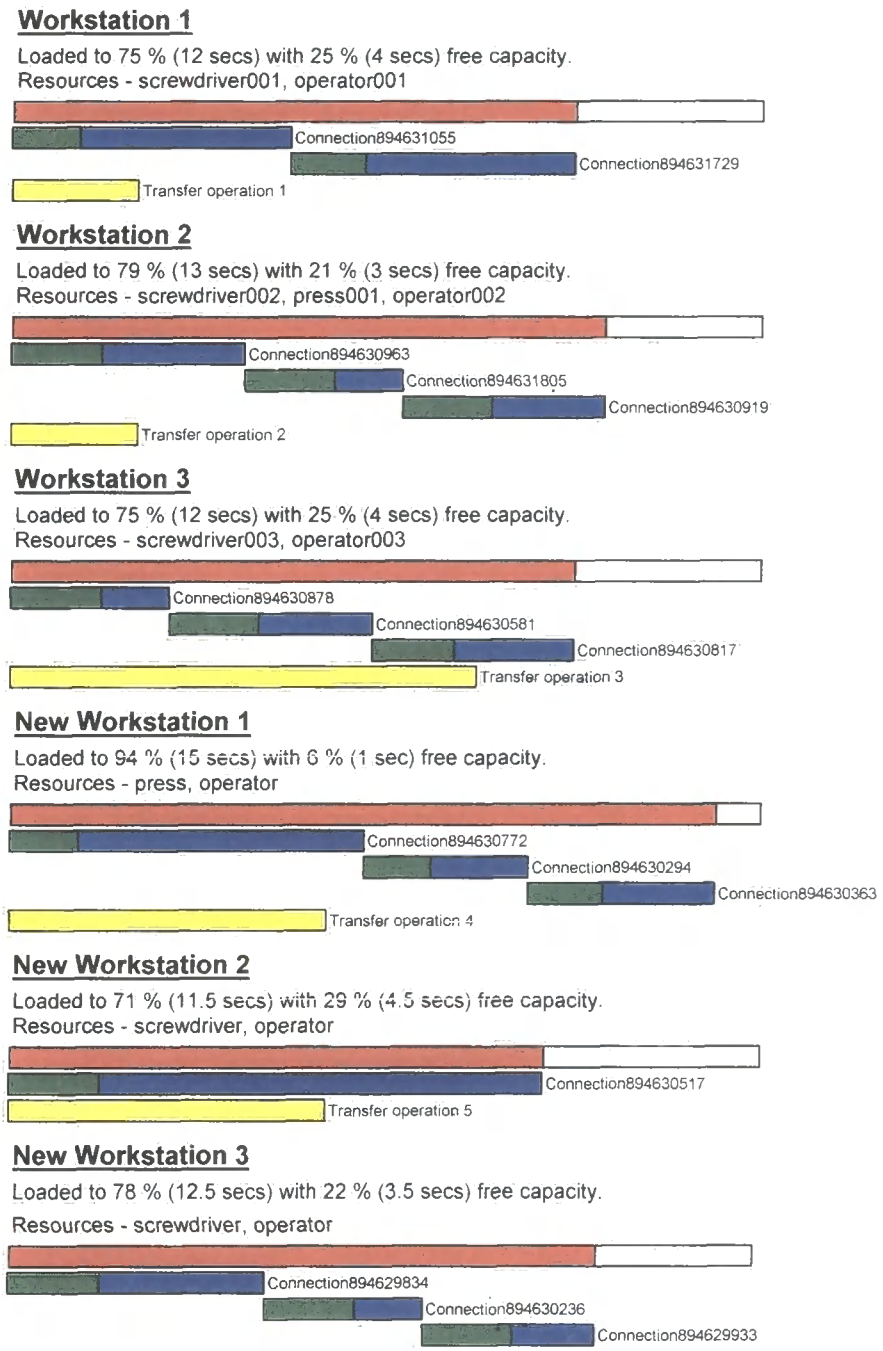


Figure 7-12: Mixed Workstation Loading

Figure 7-12 displays the loading of the existing and new workstations. It can be seen that connections 894631055, 894631729, 894630963, 894631805, 894630919, 894630878, 894630581 and 894630817 are loaded onto existing workstations one, two and three. Workstation four is unloaded, and workstations five and six have been made unavailable for selection. New workstations are created for the remaining assembly operations. Connections 894630772, 894630294 and 894630363 are loaded onto new workstation one, connection 894630517 is loaded onto new workstation two, and

connections 894629834, 894630236 and 894629933 are loaded onto new workstation three. The average workstation loading for this example is seventy-nine per cent. The total assembly time for this example is seventy-six seconds. Adding the transfer operation times to this, results in a total operation time of one hundred and five seconds. The assembly lead time for this example is one hundred and nineteen seconds.

7.5 Further Testing With Real Products

In this section, further testing on three increasingly complex Company X products are outlined.

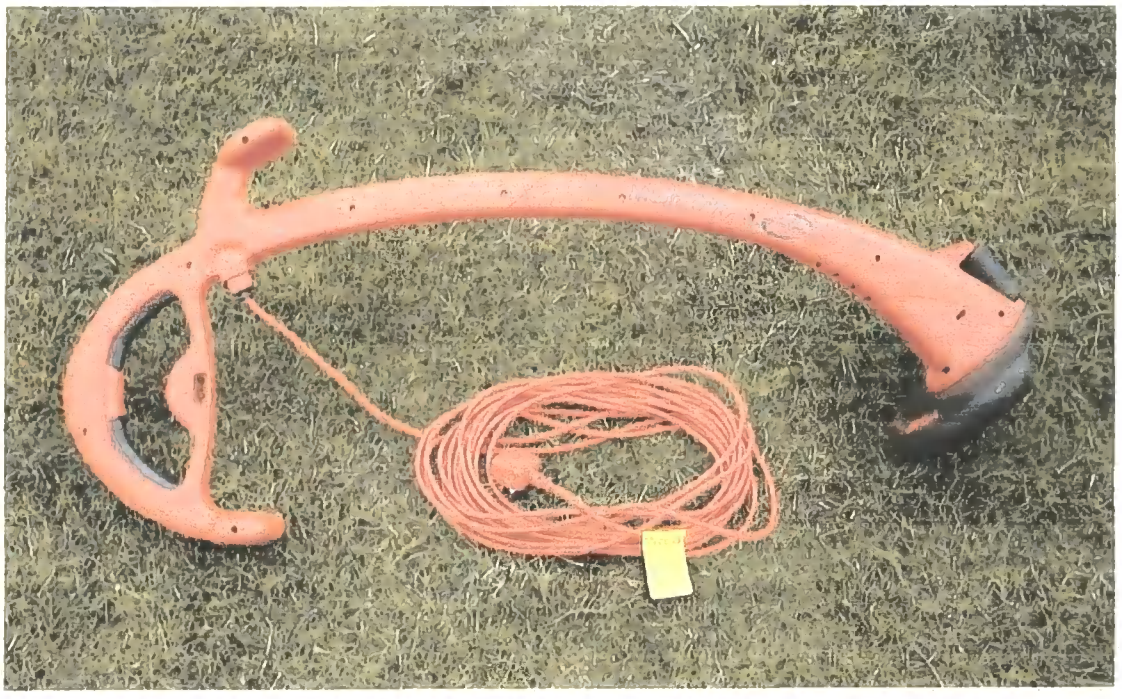


Figure 7-13: Multi-Trimmer Product

7.5.1 Case Study - Multi-Trimmer

This case study shows the aggregate assembly process planning results for the Multi-Trimmer product. The dimensions of the complete product are approximately a length of 900mm, a height of 300mm, and a width of 200mm. The Multi-Trimmer product, as displayed in Figure 7-13, consists of thirty-one components and twenty-eight AFCs. Figure 7-14 shows the overall product structure and the assembly hierarchy for the Multi-Trimmer. At the top of the product model is the final assembled state of the

Multi-Trimmer. At each level below this, the product is broken down using sub-assemblies. The main sub-assemblies are the handle, motor and body. The handle assembly consists of moulded handle components, wires, bolts and a switch. The electric motor, reel holder, wire reel and clip constitute the motor assembly. The Multi-Trimmer body consists of two moulded sections in which the motor sub-assembly locates. The motor, switches, capacitor and bolts are examples of standard parts loaded into the product model from component libraries.

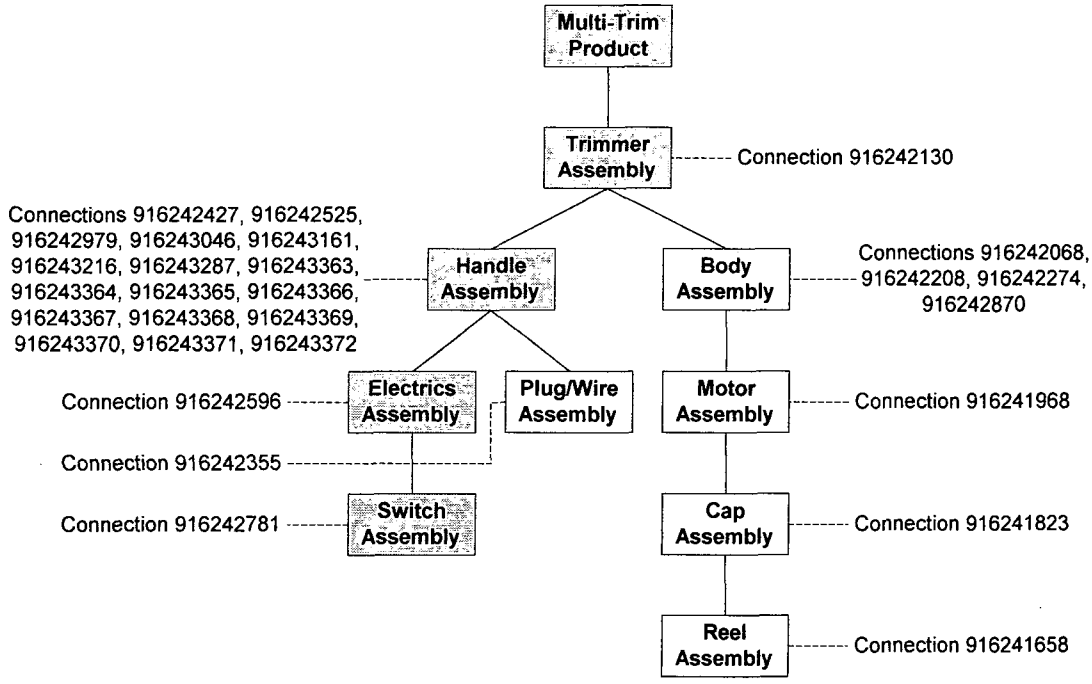


Figure 7-14: AFCs, Assembly Critical Path and the Multi-Trimmer Product Connectivity Model

Figure 7-14 also displays the product connectivity model and AFCs. A variety of AFC types are employed in the model, including placement, plug and target, threaded, snap fit, and wiring. To decrease the size of packaging, storage and transportation costs, the Multi-Trimmer is shipped to the user in two halves. The user is required to undertake a final irreversible assembly operation, snap fitting the top and bottom sections together prior to operating the strimmer.

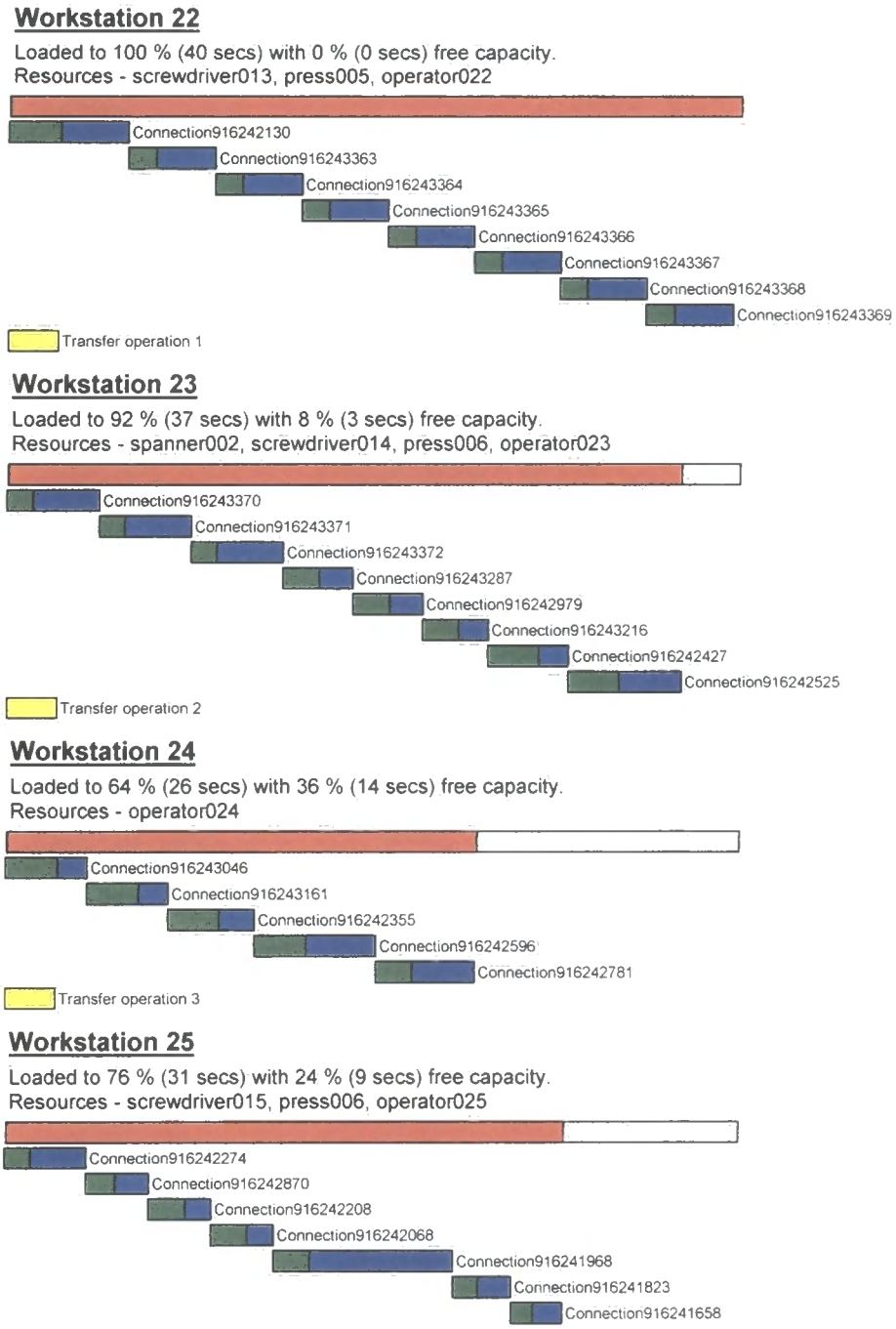


Figure 7-15: Multi-Trimmer Assembly Plan

The first main stage of the aggregate assembly process plan algorithm is the generation of an assembly sequence. Figure 7-15 and Table 7-9 display the outputted assembly sequence in two different formats. Studying the sequence, the results appear to be in a logical and feasible order apart from one operation. The last operation in the sequence is a wiring operation, joining the motor and the main switch on the handle together. This is sequenced after the top and bottom body components have been positioned and screwed together. Unfortunately, after these operations are performed, access would not

be possible to perform the wiring operation. The error in the sequence is probably due to the unusual design of the Multi-Trimmer, which requires a final assembly operation by the user, as mentioned earlier. After this minor modification to the sequence has been made, the sequence generated by the AAMP system is very similar to the actual sequence in which Company X assembles the product.

The factory resource to be loaded was the Multi-Trimmer cell from the Company X factory. This cell consists of six workstations, each with a variety of assembly machines, tools and human resource. A palletised asynchronous assembly conveyor connects the workstations together in the cell. The production data entered into the system by the user for this aggregate assembly process plan is a volume of ten thousand Multi-Trimmers, to be assembled in one week, using two eight-hour length shifts. From these values an assembly line cycle time is calculated to be forty seconds. Figure 7-15 displays the loadings of workstations twenty-two to twenty-five. From the figure it can be seen that each workstation is loaded with a number of operations, with an average efficient workstation loading of eighty-three per cent.

Table 7-9 displays a breakdown of the assembly times and costs for each operation. Summing the assembly times gives a total assembly time for the Multi-Trimmer operations of one hundred and thirty-four seconds. Adding to this the transfer operation times gives a total operation time of one hundred and forty-two seconds. The lead time for this example is one hundred and sixty-eight seconds. Summing the assembly costs for each connection results in a total assembly cost of one pound and twenty-nine pence. The critical assembly path for the Multi-Trimmer is displayed in Figure 7-14, with the grey assembly nodes representing the critical path. Summing the assembly times for all the connections on the critical path results in a critical assembly path time of ninety-six and a half seconds. To verify the accuracy of the outputted assembly times from the AAMP system, it is important to compare these times with actual factory assembly process times. The factory assembly process time for the complete Multi-Trimmer product is measured to be approximately one hundred and fifty-three seconds. This is comparable to the AAMP result of one hundred and forty-two seconds, a difference in times of seven per cent.

Table 7-9: Multi-Trimmer Assembly Times and Costs

Connection	Type	Tool	t_h	t_i/t_p	t_t	Cost
916242130	Wiring - Tag	-	3.0	3.6	6.6	£0.06
916243363	Threaded	Screwdriver 013	1.5	3.3	4.8	£0.05
916243364	Threaded	Screwdriver 013	1.5	3.3	4.8	£0.05
916243365	Threaded	Screwdriver 013	1.5	3.3	4.8	£0.05
916243366	Threaded	Screwdriver 013	1.5	3.3	4.8	£0.05
916243367	Threaded	Screwdriver 013	1.5	3.3	4.8	£0.05
916243368	Threaded	Screwdriver 013	1.5	3.3	4.8	£0.05
916243369	Threaded	Screwdriver 013	1.5	3.3	4.8	£0.05
916243370	Threaded	Screwdriver 014	1.5	3.5	5.0	£0.05
916243371	Threaded	Screwdriver 014	1.5	3.5	5.0	£0.05
916243372	Threaded	Screwdriver 014	1.5	3.5	5.0	£0.05
916243287	Placement	-	1.9	1.5	3.4	£0.03
916242979	Placement	-	2.0	1.5	3.5	£0.03
916243216	Plug and Target	-	1.9	1.8	3.7	£0.03
916242427	Snap Fit	-	3.0	1.9	4.9	£0.04
916242525	Wiring - Tag	-	3.0	3.6	6.6	£0.06
916243046	Placement	-	3.0	1.5	4.5	£0.04
916243161	Placement	-	3.0	1.5	4.5	£0.04
916242355	Plug and Target	-	2.9	1.8	4.7	£0.04
916242596	Wiring - Tag	-	3.0	3.6	6.6	£0.06
916242781	Wiring - Tag	-	1.9	3.6	5.5	£0.05
916242274	Threaded	Screwdriver 015	1.5	3.1	4.6	£0.05
916242870	Snap Fit	-	1.1	1.9	3.0	£0.03
916242208	Placement	-	2.0	1.5	3.5	£0.03
916242068	Placement	-	1.9	1.5	3.4	£0.03
916241968	Plug and Target - Press	Press 006	2.0	8.0	10.0	£0.11
916241823	Snap Fit	-	1.5	1.9	3.4	£0.03
916241658	Plug and Target	-	1.1	1.8	2.9	£0.03

7.5.2 Case Study - Hedge-Trimmer

The next case study used to test the AAMP system is a Company X Hedge-Trimmer, as shown in Figure 7-16. The product is approximately 850mm in length and consists of thirty-six components and thirty-four AFCs. Figure 7-17 shows the overall product structure and the assembly connectivity model for the Hedge-Trimmer. Two moulded case components make up the main body. Within the body sits the motor, fan, gearing and blade components. Also attached to the body is the handle assembly and blade guard. A two-handed switch mechanism operates within the handle assembly as an

extra safety feature. A collection of wires connect all the electrical components together. A number of standard parts are used within the product model, including a motor, capacitor, spring, switches and screws. A full range of AFCs are employed in the model, including placement, threaded, plug and target, wiring, and snap fit types.

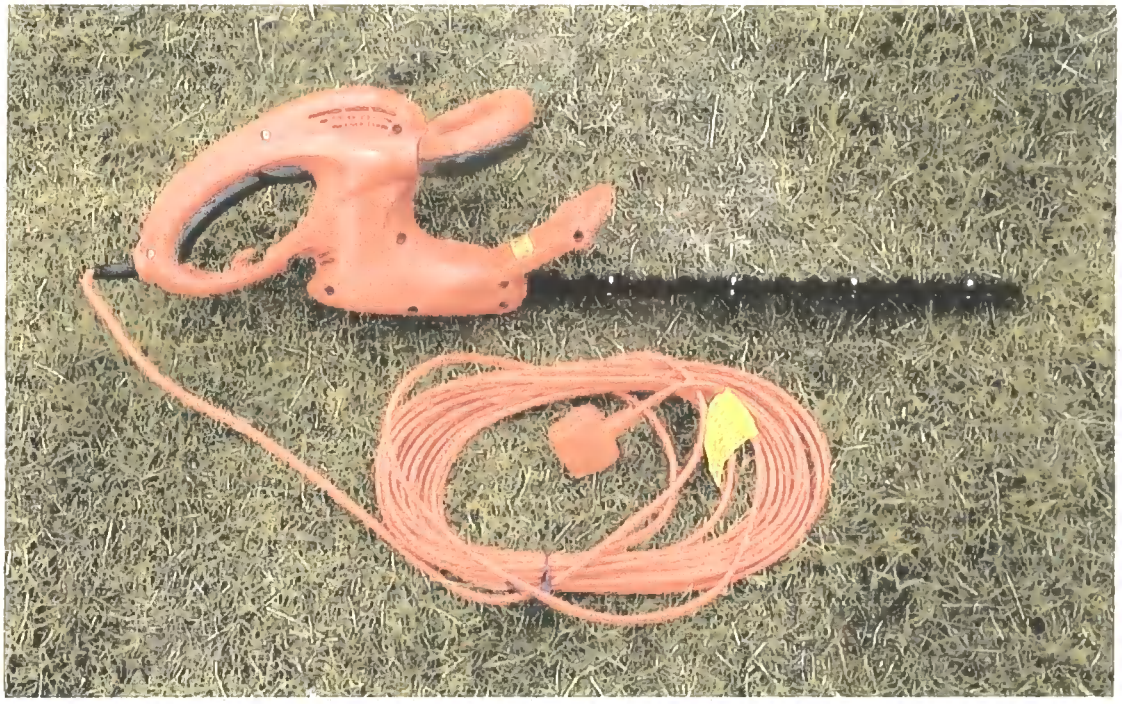


Figure 7-16: Hedge-Trimmer Product

Studying the outputted sequence, shown in Figure 7-18 and Table 7-10, the results appear to be in a feasible and logical order. The generated sequence recommends initially assembling the front and rear sub-assemblies of the handle. This includes building the wire sub-assembly loom and attaching the electrical components. Next the motor sub-assembly is fitted together, which includes pressing the rotor onto the motor axle, and assembling together the motor, gearing and blade components. The handle, wiring and motor sub-assemblies are next fitted into the bottom body section. Finally, the top body section is located onto the bottom body, and screws are placed and tightened to hold the complete Hedge-Trimmer together. Between the generated sequence and the actual sequence in which Company X assembles the Hedge-Trimmer, there are a couple of minor differences in the order that the handle and wiring sub-assembly is built. However, the AAMP generated sequence is expedient, and the differences are mainly due to different preferences of the assembly planner.

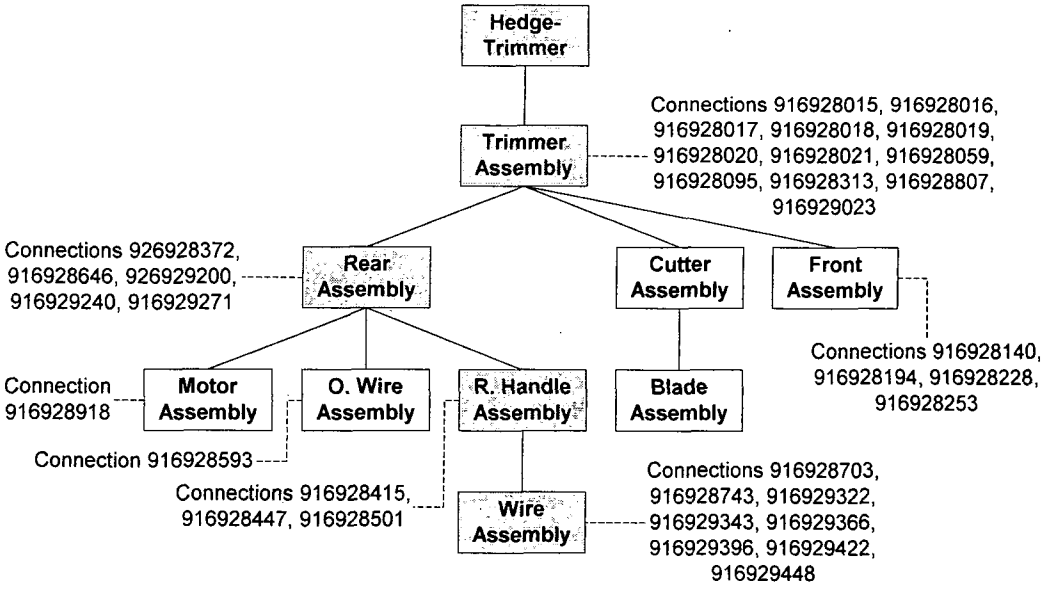


Figure 7-17: AFCs, Assembly Critical Path and the Hedge-Trimmer Product Connectivity Model

The factory resource to be loaded was the Hedge-Trimmer cell from the Company X factory. The production data entered into the system by the user for this aggregate assembly process plan is a volume of ten thousand Hedge-Trimmers, to be assembled in one week, using two eight-hour length shifts. From these values, an assembly line cycle time is calculated to be forty seconds. Figure 7-18 displays the loadings of workstations eleven to seventeen in the Hedge-Trimmer cell. Workstations fifteen and sixteen are unloaded because they do not possess the required press resource. From the figure, it can be seen that each loaded workstation in the cell is well-balanced, with an average efficient workstation loading of eighty-six per cent. Table 7-10 displays a breakdown of the assembly times and costs for each operation. Summing the assembly and transfer operation times gives a total operation time of one hundred and eighty-nine seconds. The lead time for this example is two hundred and sixteen seconds. Summing the assembly costs for each connection results in a total assembly cost of one pound and seventy-eight pence. The critical assembly path for the Hedge-Trimmer is displayed in Figure 7-17, with the grey assembly nodes representing the critical path. Summing the assembly times for all the connections on the critical path results in a critical assembly path time of one hundred and thirty-nine seconds.

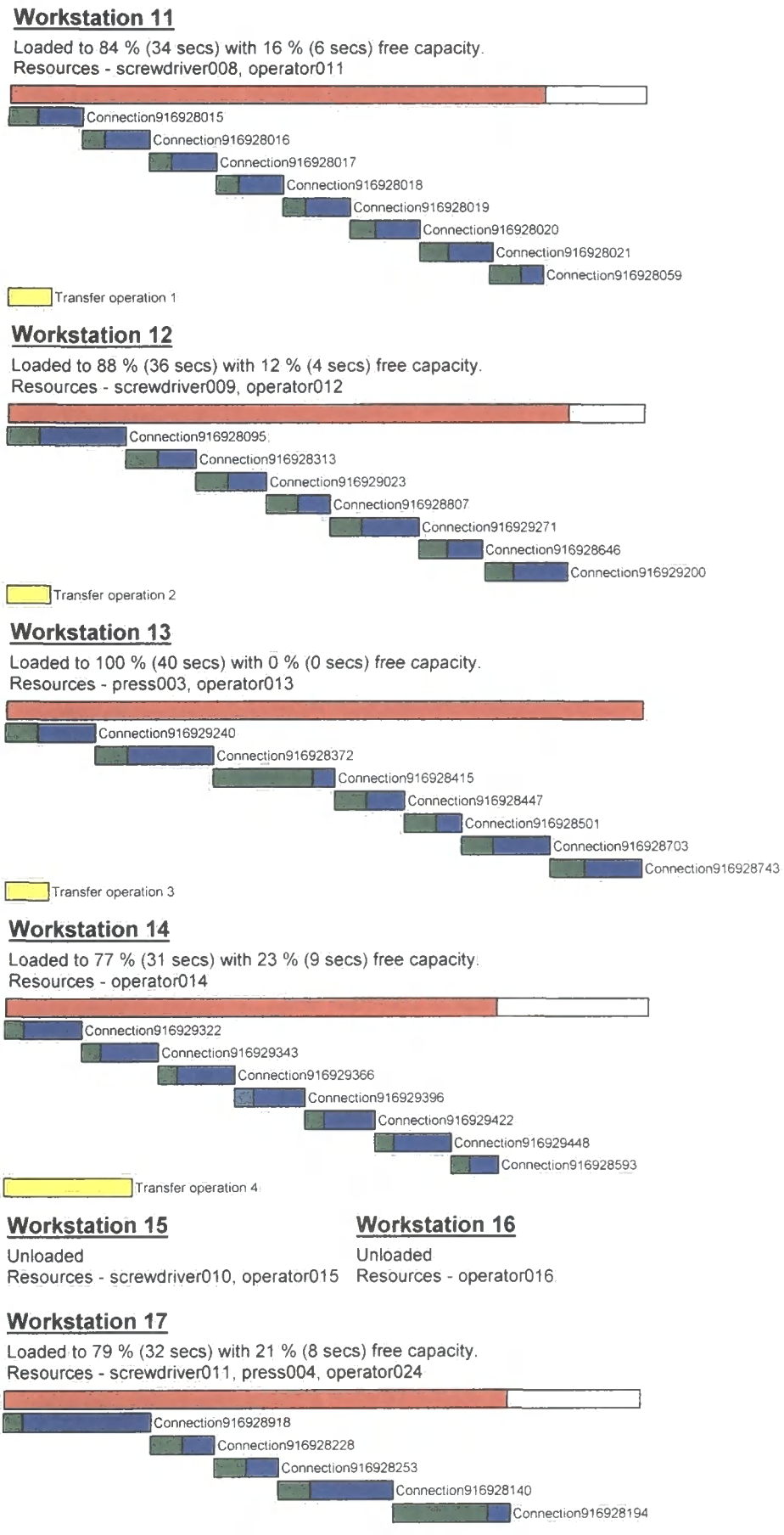


Figure 7-18: Hedge-Trimmer Assembly Plan

Table 7-10: Hedge-Trimmer Assembly Times and Costs

Connection	Type	Tools	t_h	t_i/t_p	t_t	Cost
916928015	Threaded	Screwdriver 008	1.5	2.9	4.4	£0.04
916928016	Threaded	Screwdriver 008	1.5	2.9	4.4	£0.04
916928017	Threaded	Screwdriver 008	1.5	2.9	4.4	£0.04
916928018	Threaded	Screwdriver 008	1.5	2.9	4.4	£0.04
916928019	Threaded	Screwdriver 008	1.5	2.9	4.4	£0.04
916928020	Threaded	Screwdriver 008	1.5	2.9	4.4	£0.04
916928021	Threaded	Screwdriver 008	1.5	2.9	4.4	£0.04
916928059	Placement	-	2.0	1.5	3.5	£0.04
916928095	Placement	-	2.0	5.5	7.5	£0.08
916928313	Placement	-	1.9	2.5	4.4	£0.04
916929023	Placement	-	1.9	2.5	4.4	£0.04
916928807	Plug and Target	-	1.9	2.1	4.0	£0.04
916929271	Wiring - Tag	-	2.0	3.6	5.6	£0.06
916928646	Wiring - Screw	Screwdriver 009	2.0	2.3	4.3	£0.04
916929200	Wiring - Tag	-	1.9	3.6	5.5	£0.06
916929240	Wiring - Tag	-	2.0	3.6	5.6	£0.06
916929372	Placement	-	2.0	5.5	7.5	£0.08
916928415	Placement	-	6.4	1.5	7.9	£0.08
916928447	Placement	-	2.0	2.5	4.5	£0.05
916928501	Plug and Target	-	2.0	1.8	3.8	£0.04
916928703	Wiring - Tag	-	1.9	3.6	5.5	£0.06
916928743	Wiring - Tag	-	2.0	3.6	5.6	£0.06
916929322	Wiring - Tag	-	1.1	3.6	4.7	£0.05
916929343	Wiring - Tag	-	1.1	3.6	4.7	£0.05
916929366	Wiring - Tag	-	1.1	3.6	4.7	£0.05
916929396	Wiring - Tag	-	1.1	3.6	4.7	£0.05
916929422	Wiring - Tag	-	1.1	3.6	4.7	£0.05
916929448	Wiring - Tag	-	1.1	3.6	4.7	£0.05
916928593	Plug and Target	-	1.1	1.8	2.9	£0.03
916928918	Plug and Target	Press 004	1.1	8.0	9.1	£0.10
916928228	Snap Fit	-	1.9	1.9	3.8	£0.04
916928253	Snap Fit	-	1.9	1.9	3.8	£0.04
916928140	Placement	-	2.0	5.5	7.5	£0.08
916928194	Placement	-	6.4	1.5	7.9	£0.08

To verify the accuracy of the outputted assembly times from the AAMP system, it was important to compare the actual factory process times with the generated assembly times for the Hedge-Trimmer. The actual Company X assembly process time is approximately two hundred and five seconds, which is comparable to a generated

system assembly time (including transfer times) of one hundred and eighty-nine seconds, a difference in times of only eight per cent.

7.5.3 Case Study - Cordless Trimmer.

This final case study is a more complex product than the Mini-Trimmer, Multi-Trimmer and Hedge-Trimmer, having more components, assemblies and AFCs. The Cordless Trimmer product as displayed in Figure 7-19, consists of thirty-nine components and thirty-seven AFCs, and is approximately 800mm in length.

Figure 7-19: Cordless Trimmer Product

Figure 7-20 shows the overall product structure and the assembly hierarchy for the Cordless Trimmer. The trimmer can be broken down into three main sub-assemblies. The top assembly contains the power pack, electronics, wiring, switch gear and handle. The bottom assembly holds the motor, rotor, strimmer, roller and guard components. The middle assembly is used to connect the top and bottom sub-assemblies together. The figure also displays the assembly parents to which the AFCs are attached. A wide variety of AFC types are employed in the design, including threaded, placement, plug and target, snap fit and wiring.

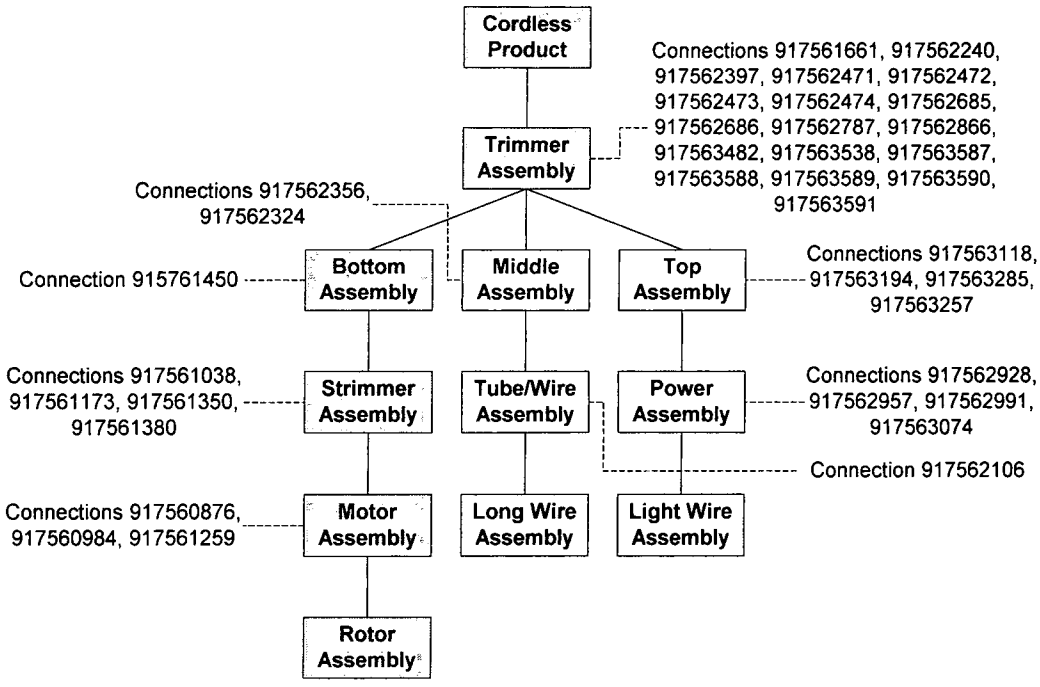


Figure 7-20: AFCs, Assembly Critical Path and the Cordless Trimmer Product Connectivity Model

Studying the outputted sequence shown in Figure 7-21 and Table 7-11, the results appear to be in a feasible and logical order. The generated sequence recommends initially assembling the motor and strimmer sub-assemblies, which includes press fitting the rotor onto the motor axle. This operation is sensibly sequenced prior to the strimmer wire, locking button, spring and cover being assembled onto the rotor. The motor and strimmer sub-assemblies are next located into the bottom body casing. The middle sub-assembly is then assembled, which includes threading wires through the main tube, and this sub-assembly is then fixed to the bottom sub-assembly. The wiring and power sub-assemblies are then constructed and placed into the top body casing. All final wiring connections are next undertaken, and the top sub-assembly is attached to the body of the trimmer. The top and bottom casing lids and battery cover are finally located and secured using screws. Apart from minor differences between the generated order and the actual order in which the trimmer is built, fundamentally both sequences are virtually identical.

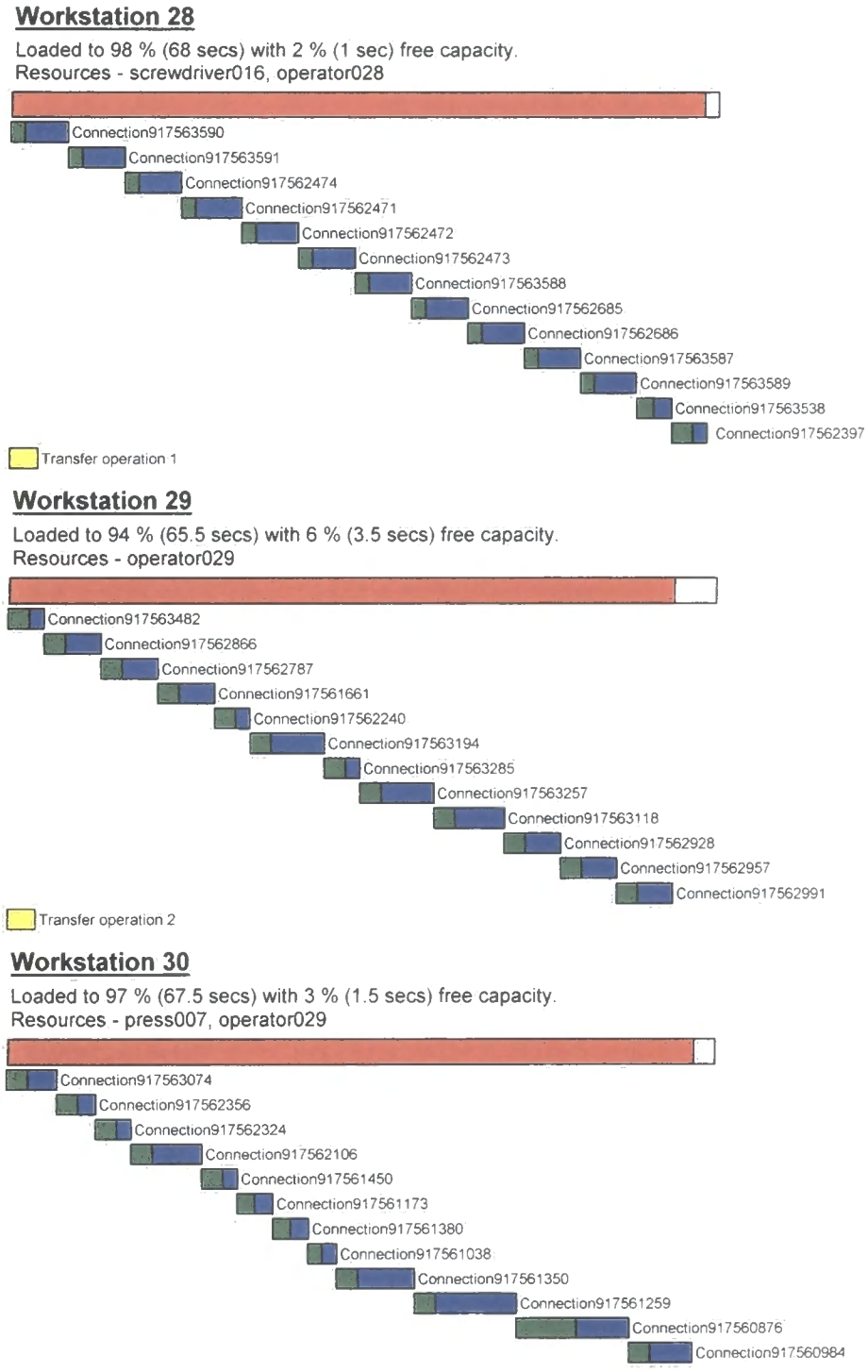


Figure 7-21: Cordless Trimmer Assembly Plan

The factory resource to be loaded was the Cordless Trimmer cell. This cell consists of six workstations, each with a variety of assembly machines, tools and human resource. The production data entered into the system by the user for this aggregate assembly process plan is a volume of five thousand and eight hundred Cordless Trimmers, to be

assembled in one week, using two eight-hour length shifts. From these values an assembly line cycle time is calculated to be sixty-nine seconds.

Table 7-11: Cordless Trimmer Assembly Times and Costs

Connection	Type	Tools	t_h	t_l/t_p	t_t	Cost
917563590	Threaded	Screwdriver 016	1.5	4.1	5.6	£0.06
917563591	Threaded	Screwdriver 016	1.5	4.1	5.6	£0.06
917562474	Threaded	Screwdriver 016	1.5	4.1	5.6	£0.06
917562471	Threaded	Screwdriver 016	1.5	4.1	5.6	£0.06
917562472	Threaded	Screwdriver 016	1.5	4.1	5.6	£0.06
917562473	Threaded	Screwdriver 016	1.5	4.1	5.6	£0.06
917563588	Threaded	Screwdriver 016	1.5	4.1	5.6	£0.06
917562685	Threaded	Screwdriver 016	1.5	4.1	5.6	£0.06
917562686	Threaded	Screwdriver 016	1.5	4.1	5.6	£0.06
917563587	Threaded	Screwdriver 016	1.5	4.1	5.6	£0.06
917563589	Threaded	Screwdriver 016	1.5	4.1	5.6	£0.06
917563538	Snap Fit	-	1.9	1.9	3.8	£0.04
917562397	Placement	-	2.0	1.5	3.5	£0.04
917563482	Placement	-	2.0	1.5	3.5	£0.05
917562866	Wiring - Tag	-	2.0	3.6	5.6	£0.06
917562787	Wiring - Tag	-	2.0	3.6	5.6	£0.06
917561661	Wiring - Tag	-	2.0	3.6	5.6	£0.06
917562240	Placement	-	1.9	1.5	3.4	£0.03
917563194	Placement	-	2.0	5.5	7.5	£0.08
917563285	Placement	-	2.0	1.5	3.5	£0.04
917563257	Plug and Target	-	5.6	1.8	7.4	£0.07
917563118	Placement	-	1.9	5.0	6.9	£0.07
917562928	Wiring - Tag	-	1.9	3.6	5.5	£0.06
917562957	Wiring - Tag	-	1.9	3.6	5.5	£0.06
917562991	Wiring - Tag	-	1.9	3.6	5.5	£0.06
917563074	Wiring - Tag	-	1.9	3.6	5.5	£0.06
917562356	Snap Fit	-	2.0	1.9	3.9	£0.04
917562324	Placement	-	2.0	1.5	3.5	£0.04
917562106	Plug and Target	-	1.9	4.8	6.7	£0.07
917561450	Placement	-	1.9	1.5	3.4	£0.03
917561173	Snap Fit	-	1.8	1.9	3.7	£0.04
917561380	Snap Fit	-	2.0	1.9	3.9	£0.04
917561038	Placement	-	1.5	1.5	3.0	£0.03
917561350	Placement	-	1.9	5.5	7.4	£0.07
917561259	Plug and Target	Press 007	2.0	8.0	10.0	£0.11
917560876	Placement	-	5.6	5.0	10.6	£0.11
917560984	Placement	-	1.9	4.0	5.9	£0.06

Figure 7-21 displays the three workstations loaded in the cell. From the figure we can see that each loaded workstation is well-balanced, with an average efficient workstation loading of ninety-six per cent. Table 7-11 displays a breakdown of the assembly times and costs for each operation. Summing the assembly and transfer operation times gives a total operation time of two hundred and six seconds. The lead time for this example is two hundred and nine seconds. Summing the assembly costs for each connection results in a total assembly cost of two pounds and fourteen pence. The critical assembly path for the Cordless-Trimmer is displayed in Figure 7-20, with the grey assembly nodes representing the critical path. Summing the assembly times for all the connections on the critical path results in a critical assembly path time of one hundred and forty-one seconds.

In contrast to the previously discussed case studies, the Cordless Trimmer was a new product which was being introduced onto the assembly lines during my visits to Company X. Although the process planners had forecasted for higher production volumes, in practice, this was not being achieved. I was asked to model and generate assembly process plans for the Trimmer, and suggest improvements to the sequence and assembly line using the AAMP system. Initally, the Trimmer was being assembled by Company X using four workstations. Unfortunately, these assembly workstations were not evenly balanced, and the assembly lead time was high because of this factor. Using predominantly the same sequence as Company X, a number of assembly process plans were generated using the AAMP system for different production volumes and assembly line layouts. It was found that using three workstations resulted in a smoother line, with a higher efficiency and a significantly lower assembly lead time. This change was implemented by Company X, and within a month of reducing the number of workstations on the assembly line, the assembly lead time was reduced from approximately two hundred and fifty seconds to two hundred and twenty seconds. This new value is comparable to the generated AAMP system assembly lead time of two hundred and nine seconds.

7.6 Testing Overall System Performance

This section addresses the issues of the success of the methodology, as distinct from the computer system. The key question to be answered here is will the AAMP system provide a useful tool to product development, bringing real benefits? The most powerful features of the system are the provision of an automated system for applying assembly process knowledge in order to rapidly evaluate designs, and the provision of an expert knowledge source for assembly planning. It is expected that product designers will benefit from both of these features, because they will bring assembly and processing knowledge to bear on early designs. In addition, assembly process planning engineers will gain the ability to perform assessments more rapidly. In particular, the AAMP system gives the ability to consider multiple product configurations, sequences, assembly line layouts and equipment availability to investigate the effects on assembly times and costs of design changes.

7.7 Conclusions

The testing and evaluation of the proposed methodology has been undertaken through the use of the AAMP system. It has been demonstrated that the system is capable of generating aggregate assembly process plans from the aggregate product model, assembly process models and resource data. It was also illustrated that these plans are both technically feasible and realistic, and can be produced in an acceptable time scale.

Using four increasingly complex Company X products, the assembly sequences, times and costs calculated by the system are comparable with those observed in industry. It was demonstrated that the system could be employed to select and evaluate suitable assembly processes and resources, and to load and balance a variety of factory scenarios. A more thorough testing of the system would require access to more detailed cost breakdowns from industry than were available during this project. In particular, the testing of the system is sensitive to the costing methods applied. However, the times calculated for processing have been shown to be realistic estimates for assembly times. Therefore, it is assumed that the cost calculations are also valid.

Chapter Eight

Discussion and Conclusions

8.1 Discussion

A method for the assessment of assembly options during the early stage of product design has been developed. This has been implemented as the AAMP computer system, which generates aggregate assembly process plans from a product model, and has been realised on a UNIX platform using Smart Elements for X-Windows. The system maintains models of the product design, the production facility, and assembly processes. The various functions of the system are integrated using a GUI to provide a system which is flexible, efficient, and easy to use.

A comprehensive review of published literature has been conducted covering Concurrent Engineering, CAE, DFA, assembly sequencing, assembly product and process modelling, and the disciplines of product development, including design and assembly process planning. From the review there appears to be many proposed methods, but no developed computer system that can handle a realistic product assembly. Some DFA approaches give a qualitative solution to gain an optimum design, but not a quantitative value for the actual assembly process, and do not consider the actual assembly plan. On the other hand, some systems concentrate on simply finding the best assembly process plan for a less optimal design. Aggregate assembly process planning has been identified as the most suitable strategy for overcoming these problems. The adoption of Concurrent Engineering as a product development strategy leads to a requirement for a restructuring of all design, manufacturing and assembly

disciplines. In particular, the computer tools which are available for supporting design and assembly process planning need to be integrated at an earlier stage than is presently possible if the ideal of concurrent working on process plans and design is to be realised.

A number of reasons have been identified for supporting Concurrent Engineering.

- Designers would benefit from a ready review of potential processing options which are available for their latest designs.
- The ability to get detailed feedback on the assembly consequences of design modifications would encourage the consideration of alternative designs during the conceptual and embodiment stages.
- Assembly facility designers would be made aware of the requirements which a new product design will place on the existing factory, and assembly assessment of product designs would include a link to assembly machines, tools and other required resources.

Careful management of the way in which the aggregate assembly process planning function is applied is required to ensure that it is made clear which time, cost and sequence changes are the result of modifications to previously considered design elements, and which are the result of additional refinements to the design. The ability to regularly update the assembly time, cost and sequence of the design as the detail is added should give designers a better understanding of product design, and encourage simpler, more efficient designs.

A flexible aggregate product model is used in the AAMP system, which can represent data over the early stages of product development. All the product information required for aggregate assembly process planning can be stored in this model, including the structure and grouping of assemblies and components, feature geometry and assembly connectivity. Assembly connections are used in the model to allow the representation of assembly joins. Standard part libraries are employed in the system to aid the design process. Aggregate assembly process models have been developed by the controlled simplification of detailed process models so they can function using limited product data available during the initial design stages. This allows the rapid evaluation of

alternative product configurations and assembly processing options at an early design stage, so that the best design options can be developed later. The developed assembly process time models are based on a combination of standard assembly time databases, process equations and operation rates.

Organisational and factory information is stored in a detailed resource model within the AAMP system. The hierarchical model includes data on cells, workstations, assembly machines and tools, labour, and transportation resource. The system allows users to load existing factories, redesign or reconfigure existing layouts, and design new factories. In combination with the product and process models, the effects of changing the product design, assembly sequence, or available resource, can be studied. One of the main functions involved in aggregate process planning is determining the best feasible sequence in which to assemble a product. The AAMP implementation uses an algorithm founded on a knowledge-based approach, using product information and accepted engineering and assembly practice to satisfy numerous constraints. Outputted aggregate assembly process plans and operation indicators are presented to the user in an HTML format.

The complexity of the methodology developed in this thesis is such that it could only be tested by implementing the algorithms as a computer system. A software language which combined a knowledge-based system approach was required to build and manage models of aggregate assembly process planning expertise, whilst the need to maintain a feature-based product model led to a requirement for an object-oriented language. The Smart Elements software development package was selected. A particular benefit of this system is that it is designed as a rapid prototyping tool for software systems, and is therefore eminently suited to the development of research prototypes.

The goal of the development is not to produce a fully-functioning commercial system, but to identify and resolve difficulties in the methodology, and to perform the necessary calculations to test it. It is worth noting, however, that a substantial part of the development time for this project was spent on building a suitable interface to the system so that the aggregate assembly process planning methodology could be demonstrated in the proposed environment of an integrated system.

The developed AAMP system has been tested using four increasingly complex Company X products. This demonstrated that the system is capable of efficiently generating aggregate assembly process plans to an acceptable quality and accuracy. The generated sequences, assembly process times and costs, assembly lead times and workstation loadings were comparable to those observed in industry. Testing of the AAMP system in a working design environment, where real time design changes could be rapidly assessed, was limited. However, Company X have successfully implemented assembly process plans generated by the system, and the feedback from them has been very positive for all aspects of the AAMP system. The company has specifically commented on the usefulness and simplicity of such a comprehensive aggregate assembly planning system, and has been particularly impressed by the speed, accuracy and clarity of the outputted results. This favourable response from industry gives encouragement to the concept that the AAMP system could be a valuable part of an integrated Concurrent Engineering environment during the conceptual design stage.

8.2 Research Issues

This thesis acknowledges the following issues:

- In today's highly competitive global market, it is imperative that products are released to market at the right time and with the desired quality. Hence, there is a need to reduce the product development cycle time by foreseeing manufacturing problems during the early stages of design.

- The Concurrent Engineering methodology requires that the assembly process planning function be initiated earlier in the design cycle, at a stage when less information is available about a design.

- There is a recognised need for a closer link between design decisions and assembly consequences. This can best be achieved by empowering the design engineer with the ability to assess the assembly options available for a product design.

- It is necessary to compare and select alternative designs, assembly processes and sequences, and resource options at an early stage of the design process.

- Human process planners are not able to evaluate the large number of alternative assembly process plans available for complex products, leading to selection based on intuition instead of calculation.

- The rapidly changing nature of product design, planning and manufacture means that there is a need for immediate availability of alternative aggregate assembly process plans.

8.3 Conclusions

The work of this thesis addresses the issues highlighted in the previous section and new contributions to this field of research have been achieved:

- A novel methodology for supporting Concurrent Engineering has been developed to provide a link between the early stages of design and assembly planning. The generation of aggregate assembly process plans gives details of feasible sequences, assembly process times and costs, resource requirements and factory loadings.

- An integrated computer support system operating at an aggregate level, which fills the current gap between product development and assembly process planning, has been developed to implement the proposed methodology. The AAMP system is an automated CAE tool that brings together for the first time, all aspects of product development to consider assembly planning at the conceptual stage of design.

- The minimum information requirements for aggregate assembly process planning during early design have been identified, and a product model encompassing this data has been developed which allows the efficient representation of assembly structure, components and assembly connectivity. An innovative factor of this research is the introduction of AFCs within the conceptual product model.

- A generic assembly process modelling technique has been developed and applied to selected processes to develop methods for accurate calculation of assembly criteria, including time and cost. In order to assess the assemblability of designs, these detailed assembly process models have been generated to function with limited design data. Previous attempts to apply comparative models to assembly planning

during conceptual design, have used over-simplified assembly process models which do not consider all the necessary factors.

- The introduction of a detailed resource model for aggregate assembly process planning enables the system to calculate accurate assembly times dependent on which resources are used within a factory, or even which factory is utilised. This resource model has been developed for the representation of assembly facilities, including organisation, factory and transportation data.

- The generation of two new algorithms has produced a novel routine for efficiently deriving accurate assembly sequences and subsequent factory loadings. The first new algorithm is used to generate a feasible sequence using the structure of the product model, process constraints, and assembly rules. The fundamental objective of the second algorithm is to load all the assembly operations onto workstations, whilst ensuring that the workstations have the capacity and capability.

- The integrated system provides a flexible environment to assess the inter-connected effects of changing product design, assembly process plans and facilities.

- Once a set of aggregate assembly process plans have been generated, the planner may select the most suitable for detailed planning depending on the latest factory conditions. Alternative aggregate assembly process plans could be used as an input to a shop floor planning system.

- The AAMP system has been implemented on a UNIX-based computer. Testing of the methodologies and the developed AAMP system has yielded accurate results, and industrial response has been extremely favourable.

8.4 Recommendations For Further Work

This work has led to the identification of many further avenues of research and development. In this section a number of possible extensions to the work are discussed. Many of the research areas identified during the course of this project have already begun to be researched, and the AAMP computer system is undergoing further development as part of a funded research project.

The AAMP system was developed as a test-bed for the aggregate assembly process planning methodology and to prove the concept of using aggregate assembly process plans to evaluate early product designs. It was not designed to be applied in industry, and therefore, several functions which would be required to turn it into a fully functioning system are not in place. In particular, there is a need for a link to a commercial CAD package on which the designs would be developed. This should be accomplished using the STEP standard. However, a requirement of this link would be that it established a means of extracting from a detailed product model, only that information which is required by the aggregate product model.

The proposed methodology of product development using aggregate assembly process planning has the capacity of assessing all feasible assembly processes for a product. The current AAMP system covers a sub-set of assembly processes, due to the limitations of time. Further work is required to enhance the assembly process model to include additional processes. In particular, the current system has no model for metallurgical or chemical assembly processes, and the use of automation in assembly.

The resource model described in this thesis was primarily developed to test the assembly process planning rules. It is not intended to be a fully comprehensive and definite list of all assembly machine and tool types, and there are opportunities for improving this model in several ways. The class structure developed could be refined by increasing the number of classes to reflect the subtle variations in machines and tools. Also, a fully comprehensive list of assembly machines, tools and transportation could be added to the system.

Whilst the AAMP system generates the theoretical best assembly sequence, the aggregate process planning methodology could be extended to derive numerous feasible sequences and find the optimal one from this set. Some work at Durham University has been conducted in this area, with the development of a methodology for the creation and selection of optimal assembly sequences using a simulated annealing technique (Laguda and Maropoulos, 2000). It is important to note that the generation of an optimal assembly sequence does not in itself imply an optimal assembly plan. An optimal assembly plan can only be realised when the available resources are taken into consideration. The generated optimal assembly sequence essentially provides a suitable

input for optimising the line balancing, and this is also being investigated at present by Laguda.

References

Abdalla, HS and Knight, J (1994), An Expert-System for Concurrent Product and Process Design of Mechanical Parts, *Proceedings of the Institution of Mechanical Engineers- Part B: Journal of Engineering Manufacture*, vol. 208, no. 3, pp. 167-172.

Andreason, MM and Gudnason, CH (1992), Planning and Integration of Product Development, (Gavrial Salvendy ed.), *Handbook of Industrial Engineering*, Second Edition, John Wiley & Sons.

Arai, E and Iwata, K (1992), Product Modelling System in Conceptual Design of Mechanical Products, *Robotics and Computer-Integrated Manufacturing*, vol. 9, nos. 4-5, pp. 327-334.

Arai, E and Iwata, K (1993), CAD System with Product Assembly/Disassembly Planning Function, *Robotics and Computer-Integrated Manufacturing*, vol. 10, nos. 1-2, pp. 41-48.

Baldwin, DF, Abell, TE, Lui, MCM, Defazio, TL and Whitney, DE (1991), An Integrated Computer Aid for Generating and Evaluating Assembly Sequences for Mechanical Products, *IEEE Transactions on Robotics and Automation*, vol. 7, no. 1, pp. 78-94.

Ben-Arieh, D and Kramer, B (1994), Computer-Aided Process Planning for Assembly: Generation of Assembly Operations Sequence, *International Journal of Production Research*, vol. 32, no. 3, pp. 643-656.

Bloch, C and Ranganathan, R (1992), Process-Based Cost Modelling, *IEEE Transactions on Components, Hybrids and Manufacturing Technology*, vol. 15, no. 3, pp. 288-294.

Boothroyd, G (1992), *Assembly Automation and Product Design*, Marcel Dekker Inc., New York.

Boothroyd, G and Alting, L (1992), Design for Assembly and Disassembly, *Annals of the CIRP*, vol. 41, no. 2, pp. 625-636.

Boothroyd, G and Dewhurst, P (1987), *Product Design for Assembly*, Boothroyd Dewhirst Inc., Rhode Island, Kingston.

Boothroyd, G, Dewhurst, P and Knight, W (1994), *Product Design for Manufacture and Assembly*, Marcel Dekker Inc., New York.

Bourjault, A (1984), *Contribution Ö une Approche Methodologique de l' Assemblage Automatise: Elaboration Automatique des Sequences Operatoires*, Thesis to obtain Docteur äs Sciences Physiques, UniversitÇ de Franche-Comte, France.

Bradley, HD (1997), *Aggregate Process Planning and Manufacturing Assessment for Concurrent Engineering*, Thesis for Doctorate of Philosophy, University of Durham, England.

BS 308: Part 3 (1985), *Engineering Drawing Practice: Recommendations for Geometrical Tolerancing*, British Standard Institute.

BS 7000: Part 3 (1997), *Standard for Engineering Management: Guide to Managing Designs of Manufactured Products*, British Standard Institute.

Bullinger, HJ and Richter, N (1991), Integrated Design and Assembly Planning, *Computer-Integrated Manufacturing Systems*, vol. 4, no. 4, pp. 239-248.

Butterfield, WR, Green, MK, Scott, DC and Stoker, WJ (1985), Part Features for Process Planning, *C-85-PPP-03*, CAMI-I Inc., Arlington, Texas, USA.

Case, K (1994), Using a Design by Features CAD System for Process Capability Modelling, *Computer-Integrated Manufacturing Systems*, vol. 7, no. 1, pp. 39-49.

Case, K and Gao, J (1993), Feature Technology - An Overview, *International Journal of Computer-Integrated Manufacturing*, vol. 6, nos. 1-2, pp. 2-12.

Case, K, Gao, JX and Gindy, NNZ (1994), The Implementation of a Feature-Based Component Representation for CAD-CAM Integration, *Proceedings of the Institution*

of Mechanical Engineers- Part B: Journal of Engineering Manufacture, vol. 208, no. 1, pp. 71-80.

Case, K and Harun, WAW (1999), A Single Representation to Support Assembly and Process Planning in Feature-Based Design of Machined Parts, *Proceedings of the Institution of Mechanical Engineers- Part B: Journal of Engineering Manufacture*, vol. 213, no. 2, pp. 143-155.

Chakrabarty, S and Wolter, J (1997), A Structure-Oriented Approach to Assembly Sequence Planning, *IEEE Transactions on Robotics and Automation*, vol. 13, no. 1, pp. 14-29.

Cleetus, KJ (1992), Definition of Concurrent Engineering, *CERC Technical Report Series*, CERC-TR-RN-92-003.

Coad, P and Yourdon, E (1991), *Object-Oriented Analysis*, Englewood Cliffs, Second Edition, New Jersey.

Cross, N (1994), *Engineering Design Methods: Strategies for Product Design*, Second Edition, John Wiley and Sons.

Cutkosky, MR, Engelmore, RS, Fikes, RE, Genesereth, MR, Gruber, TR, Mark, WS, Tenebaum, JM and Weber, JC (1993), Pact - An Experiment in Integrating Concurrent Engineering Systems, *Computer*, vol. 26, no. 1, pp. 28-37.

De Fazio, TL, Edsall, AC, Gustavson, RE, Hernandez, J, Hutchins, PM, Leung, HW, Luby, SC, Metzinger, RW, Nevins, JL and Tung, K (1993), A Prototype of Feature-Based Design for Assembly, *Journal of Mechanical Design*, vol. 115, pp. 723-734.

De Fazio, TL and Whitney, DE (1987), Simplified Generation of all Mechanical Assembly Sequences, *IEEE Journal of Robotics and Automation*, vol. 3, no. 6, pp. 640-658.

Delchambre, A (1992), *Computer-Aided Assembly Planning*, Chapman & Hall, First Edition, London.

Denzel, H and Vosniakos, GC (1993), A Feature-Based Design System and its Potential to Unify CAD and CAM, *IFIP Applications B: Applications in Technology*, vol. 10, pp. 131-144.

Dong, JJ, Parsaei, HR and Leep, HR (1996), Manufacturing Process Planning in a Concurrent Design and Manufacturing Environment, *Computers & Industrial Engineering*, vol. 30, no. 1, pp. 83-93.

Dossett, RJ (1992), Computer Application of a Natural-Language Predetermined Motion Time Study, *Computers and Industrial Engineering*, vol. 23, nos. 1-4, pp. 319-322.

Edan, Y and Nof, SY (1995), Motion Economy Analysis for Robot Kitting Tasks, *International Journal of Production Research*, vol. 33, no. 5, pp. 1213-1227.

Elola, LN, Tejedor, ACP and Menorca, LG (1996), New methods of Evaluating Physical Demand at Work Areas, *Technovation*, vol. 16, no. 10, pp. 595-599.

Evans, S (1990), Implementation Framework for Integrated Design Teams, *Journal of Engineering Design*, vol. 1, no. 4, pp. 355-363.

Fisher, EL and Nof, SY (1989), Knowledge-Based Economic Analysis of Manufacturing Systems, *Journal of Manufacturing Systems*, vol. 2, no. 1, pp. 53-59.

French, MJ (1985), *Conceptual Design for Engineers*, The Design Council, Springer-Verlag, Second Edition, London.

Fu, Z, de Pennington, A and Saia, A (1993), A Graph Grammar Approach to Feature Representation and Transformation, *International Journal of Computer-Integrated Manufacturing*, vol. 6, nos. 1-2, pp. 137-151.

Gandhi, A and Myklebust, A (1989), *A Natural Language Approach to Feature-Based Modelling*, Mobil Research and Development Corporation, Princeton, New Jersey, USA.

Gao, JX and Huang, XX (1996), Product and Manufacturing Capability in an Integrated CAD/Process Planning Environment, *International Journal of Advanced Manufacturing Technology*, vol. 11, pp. 43-51.

Genaidy, AM, Agrawal, A and Mital, A (1990), Computerized Predetermined Motion-Time Systems in Manufacturing Industries, *Computers and Industrial Engineering*, vol. 18, no. 4, pp. 571-584.

Gillen, DJ and Fitzgerald, EM (1991), Expanding Knowledge and Converging Functions, *Concurrent Engineering*, vol. 1, no. 3, pp. 20-28.

Gindy, NNZ, Huang, X and Ratchev, TM (1993), Feature-Based Component Model for Computer-Aided Process Planning Systems, *International Journal of Computer-Integrated Manufacturing*, vol. 6, nos. 1-2, pp. 20-26.

Gui, JK and Mäntylä, M (1994), Functional Understanding of Assembly Modelling, *Computer-Aided Design*, vol. 26, no. 6, pp. 435-451.

Hartley, JR (1990), *Concurrent Engineering, Shortening Lead Times, Raising Quality, and Lowering Costs*, Mass Productivity Press, Cambridge, England.

Henderson, MR (1986), Automated Group Technology Coding from a Three-Dimensional CAD Database, *Knowledge-Based Expert System for Manufacturing*, ASME, pp. 195-204.

Henrioud, JM and Bourjault, A (1992), Computer-Aided Assembly Process Planning, *Proceedings of the Institution of Mechanical Engineers- Part B: Journal of Engineering Manufacture*, vol. 206, no. 1, pp. 61-66.

Hernani, JT and Scarr, AJ (1987), An Expert System Approach to the Choice of Design Rules for Automated Assembly, *Assembly Production* (MM Andreasen ed.), IFS Publications, London, pp. 129-136.

Homem de Mello, LS and Sanderson, AC (1990), AND/OR Graph Representation of Assembly Plans, *IEEE Transactions on Robotics and Automation*, vol. 6, no. 2, pp. 188-199.

Homem de Mello, LS and Sanderson, AC (1991), Representations of Mechanical Assembly Sequences, *IEEE Transactions on Robotics and Automation*, vol. 7, no. 2, pp. 211-227.

Hsu, W, Lee, CSG and Su, SF (1993), Feedback Approach to Design for Assembly by Evaluation of Assembly Plan, *Computer-Aided Design*, vol. 25, no. 7, pp. 395-410.

Huang, GQ and Mak, KL (1999), Design For Manufacture and Assembly on the Internet, *Computers In Industry*, vol. 38, pp. 17-30.

Krause, F-L, Kimura, F, Kjellburg, T and Lu, SC-Y (1993), Product Modelling, *Annals of the CIRP*, vol. 42, no. 2, pp. 695-706.

Kunica, Z and Vranjes, B (1999), Towards Automatic Generation of Plans for Automated Assembly, *International Journal of Production Research*, vol. 37, no. 8, pp. 1817-1836.

Kunz, JC, Luiten, GT, Fischer, MA, Jin, Y and Levitt, RE (1996), CE4 - Concurrent Engineering of Product, Process, Facility, and Organization, *Concurrent Engineering: Research and Applications*, vol. 4, no. 2, pp. 187-198.

Laguda, A and Maropoulos, PG (2000), Automatic Generation of Optimal Assembly Sequences Using Simulated Annealing, *16th International Conference on Computer-Aided Production Engineering*, I.Mech.E. Conference Transactions, pp. 401-410.

Laperrière, L and ElMaraghy, HA (1992), Planning of Products Assembly and Disassembly, *Annals of the CIRP*, vol. 41, no. 1, pp. 5-9.

Laperrière, L and ElMaraghy, HA (1996), GAPP: A Generative Assembly Process Planner, *Journal of Manufacturing Systems*, vol. 15, no. 4, pp. 282-293.

Latif, MN, Boyd, RD and Hannam, RG (1993), Integrating CAD and Manufacturing Intelligence Through Features and Objects, *International Journal of Computer-Integrated Manufacturing*, vol. 6, nos. 1-2, pp. 87-93.

Latif, MN and Hannam, RG (1996), Feature-Based Design and the Object-Oriented Approach, *Journal of Engineering Design*, vol. 7, no. 1, pp. 27-37.

Lee, K and Andrews, G (1985), Inference of the Positions of Components in an Assembly, *Computer-Aided Design*, vol. 17, no. 1, pp. 20-24.

Lee, K and Gossard, DC (1985), A Hierarchical Data Structure for Representing Assemblies, *Computer-Aided Design*, vol. 17, no. 1, pp. 15-19.

Lee, S and Shin, TG (1993), Assembly Coplanner: Co-operative Assembly Planner Based on Subassembly Extraction, *Journal of Intelligent Manufacturing*, vol. 4, pp. 183-198.

Lenau, T and Alting, L (1992), Models of Manufacturing Processes for Design Information Systems, *Manufacturing Systems*, vol. 21, no. 2, pp. 129-135.

Lenau, T and Mu, L (1993), Features in Integrated Modelling of Products and Their Production, *International Journal of Computer-Integrated Manufacturing*, vol. 6, nos. 1-2, pp. 65-73.

Li, RK and Hwang, CL (1992), A Framework for Automatic DFA System Development, *Computers and Industrial Engineering*, vol. 22, no. 4, pp. 403-413.

Lin, AC and Chang, TC (1993a), An Integrated Approach to Automated Assembly Planning for Three-Dimensional Mechanical Products, *International Journal of Production Research*, vol. 31, no. 5, pp. 1201-1227.

Lin, AC and Chang, TC (1993b), 3D MAPS: Three-Dimensional Mechanical Assembly Planning System, *Journal of Manufacturing Systems*, vol. 12, no. 6, pp. 437-456.

Maropoulos, PG (1995a), A Novel Process Planning Architecture for Product-Based Manufacture, *Proceedings of the Institution of Mechanical Engineers- Part B: Journal of Engineering Manufacture*, vol. 209, no. 4, pp. 267-276.

Maropoulos, PG (1995b), Aggregate Process Modelling Technology for Process Planning Applications, *Proceedings of the 11th International Confererence of Computer-Aided Production Engineering*, I.Mech.E. Transactions, pp. 179-184.

Maropoulos, PG, Bradley, HD and Zhihui, Y (1998), CAPABLE: An Aggregate Process Planning System for Integrated Product Development, *Journal of Materials Processing Technology*, vol. 76, pp. 16-22.

Martin-Vega, LA, Brown, HK, Shaw, WH and Sanders, TJ (1995), Industrial Persective on Research Needs and Opportunities in Manufacturing Assembly, *Journal of Manufacturing Systems*, vol. 14, no. 1, pp. 45-58.

Maynard, H, Stegmerten, G and Schwab, J (1948), *Methods Time Measurement*, McGraw Hill, New York.

Mazouz, AK, Souilah, A and Talbi, M (1991), Design of an Expert System for Generating Optimal Assembly Sequences, *Computer-Aided Engineering Journal*, December, pp. 255-259.

Meerkamm, H (1993), Design System MFK- an Important Step Towards an Engineering Workbench, *Proceedings of the Institution of Mechanical Engineers- Part B: Journal of Engineering Manufacture*, vol. 207, no. 2, pp. 105-116.

Miyakawa, S and Ohashi, T (1986), The Hitachi Assemblability Evaluation Method, *Proceedings of First International Conference in Product Design for Assembly*, Production Engineering Research Laboratory, Hitachi Ltd., pp. 1-13.

Mo, J, Cai, J, Zhang, Z and Lu Z (1999), DFA-Oriented Assembly Relation Modelling, *International Journal of Computer-Integrated Manufacturing,* vol. 12, no. 3, pp. 238-250.

Molina, A, Alashaab, AH, Ellis, TIA, Young, RIM and Bell, R (1995), A Review of Computer-Aided Simultaneous Engineering Systems, *Research in Engineering Design*, vol. 7, no. 1, pp. 38-63.

Molloy, E, Yang, H and Browne, J (1993), Feature-Based Modelling in Design for Assembly, *International Journal of Computer-Integrated Manufacturing*, vol. 6, nos. 1-2, pp. 119-125.

Neuron Data (1995), *Smart Elements User Manual*, Neuron Data, Palo Alto.

Nevins, JL and Whitney, DE (1978), Computer Controlled Assembly, *Scientific American*, vol. 238, no. 2, pp. 62-74.

Nof, SY, Wilhelm, WE and Warnecke, HJ (1997), *Industrial Assembly*, Chapman and Hall, First Edition, London.

Oh, JS, O'Grady, P and Young, RE (1995), A Constraint Network Approach to Design for Assembly, *IIE Transactions*, vol. 27, pp. 72-80.

Pahl, G and Beitz, W (1988), *Engineering Design a Systematic Approach*, (K Wallace Ed.), The Design Council, London.

Pawar, KS, Menon, U and Riedel, JCKH (1994), Time to Market, *Integrated Manufacturing Systems*, vol. 5, no. 1, pp. 14-22.

Pratt, MJ (1993), Applications of Feature Recognition in the Product Life-Cycle, *International Journal of Computer-Integrated Manufacturing*, vol. 6, nos. 1-2, pp. 13-19.

Pratt, MJ and Wilson, PR (1985), Requirements for Support of Form Features in a Solid Modelling System, *T-85-ASPP-01*, CAM-I Inc., Arlington, Texas, USA.

Rahman, AR, Harun, W and Case, K (1994), Object-Oriented Feature-Based Design, *Advances in Manufacturing Technology VIII*, (K Case and ST Newman, eds.), pp. 294-298.

Redford, A and Chal, J (1994), *Design For Assembly: Principles and Practice*, McGraw-Hill Book Company.

RS Catalogue, RS Components Ltd, Corby, England.

Sackett, PJ and Hollbrook, AEK (1988), DFA as a Primary Process Decreases Design Deficiencies, *Assembly Automation*, August, pp. 137-140.

Salomons, OW, van Houten, FJAM and Kals, HJJ (1993), Review of Research in Feature-Based Design, *Journal of Manufacturing Systems*, vol. 12, no. 2, pp. 113-132.

Sandvik Cormant (1997), Rotating Tools, *C-1100:5-ENG*, Sandvik Cormant, Sweden.

Santochi, M and Dini, G (1992), Computer-Aided Planning of Assembly Operations: The Selection of Assembly Sequences, *Robotics and Computer-Integrated Manufacturing*, vol. 9, no. 6, pp. 439-446.

SECO Tools AB (1997), *Turning Main Catalogue, ST944739E*, Sweden.

Senin, N, Groppetti R and Wallace DR (2000), Concurrent Assembly Planning with Genetic Algorithms, *Robotics and Computer-Integrated Manufacturing*, vol. 16, pp. 65-72.

Shah, JJ and Rogers, MT (1988), Functional Requirements and Conceptual Design of the Feature-Based Modelling System, *Computer-Aided Engineering Journal*, February, pp. 9-15.

Singh, N and Dengzhou, Q (1992), A Structural Framwork for Part Feature Recognition: A Link Between Computer-Aided Design and Process Planning, *Integrated Manufacturing Systems*, vol. 3, no. 1, pp. 4-12.

Smith, RP (1997), The Historical Roots of Concurrent Engineering Fundamentals, *IEEE Transactions on Engineering Management*, vol. 44, no. 1, pp. 67-78.

Sohlenius, G (1992), Concurrent Engineering, *Annals of the CIRP*, vol. 41, no. 2, pp. 645-655.

Spur, G, Krause, F-L and Armbrust, P (1986), Product Models as Basis for Integrated Design and Manufacturing, *International Journal of Machine Tool Design Research*, vol. 26, no. 2, pp. 171-178.

Subrahmanyam, S and Wozny, M (1995), Survey Paper: An Overview of Automatic Feature Recognition Techniques for Computer-Aided Process Planning, *Computers in Industry*, Vol. 26, pp. 1-21.

Swift, KG (1987), *Knowledge-Based Design for Manufacture*, Kluwer Academic Publishers, Kogan Press, London.

Tönshoff, HK, Aurich, JC, Ehrmann, M and D'Agostino, N (1996), A unified approach to free-form and regular feature modelling, *Annals of the CIRP*, vol. 45, no. 1, pp. 125-128.

Tönshoff, HK, Menzel, E and Park, HS (1992), A Knowledge-Based System for Automated Assembly Planning, *Annals of the CIRP*, vol. 41, no. 1, pp. 19-24.

van der Net, AJ, de Vries, WAH, Delbressine, FLM and van der Wolf, ACH (1996), A Relation-Based Product Model Suited for Integrating Design and Manufacturing, *Annals of the CIRP*, vol. 45, no. 1, pp. 161-164.

Van't Erve, AH and Kals, HJJ (1988), XPLANE: A Generative Computer-Aided Process Planning System for Part Manufacturing, *Annals of the CIRP*, vol. 35, no. 1, pp. 325-329.

Warnecke, HJ, Schweizer, M, Tamaki, K and Nof, SY (1992), *Handbook of Industrial Engineering*, Second Edition, Wiley, New York.

Werling, G and Wild, H (1994), HOM - A Hybrid Object Model for Automated Assembly Planning, *Journal of Intelligent Manufacturing*, vol. 5, pp. 153-163.

Wilson, RH (1995), Minimizing User Queries in Interactive Planning, *IEEE Transactions on Robotics and Automation*, vol. 11, no. 2, pp. 308-312.

Wilson, RH, Kavraki, L, Latombe, JC and Lozano-Perez, T (1995), Two-Handed Assembly Sequencing, *International Journal of Robotics Research*, vol. 14, no. 4, August, pp. 335-350.

Winner, RI, Pennell, JP, Bertrand, HE and Slusarczuk, MMG (1988), *The Role of Concurrent Engineering in Weapon System Acquisition*, IDA-Report R-338.

You, CF and Chiu, CC (1996), An Automated Assembly Environment in Feature-Based Design, *International Journal of Advanced Manufacturing Technology*, vol. 12, no. 4, pp. 280-287.

Zandin, K (1980), *MOST: Work Measurement Systems*, Marcel Dekker Inc., New York.

Zeid, I (1991), *CAD/CAM Theory and Practice*, McGraw-Hill Series in Mechanical Engineering, First Edition, New York.

Zha, XF, Lim, SYE and Fok SC (1999), Integrated Knowledge-Based Approach and System for Product Design for Assembly, *International Journal of Computer-Integrated Manufacturing*, vol. 12, no. 3, pp. 211-237.

Zhang, Y, Luh, PB, Yoneda, K, Kan, T and Kyoya Y (2000), Mixed-Model Assembly Line Scheduling Using the Lagrangian Relaxation Technique, *IIE Transactions*, vol. 32, pp. 125-134.

Appendix A

Feature Set Details

Table 1: Positive Features

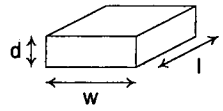
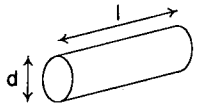
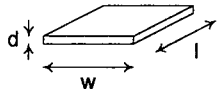
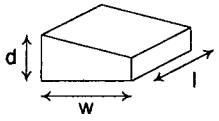
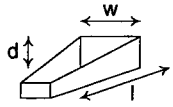
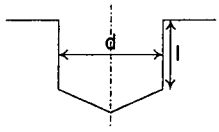
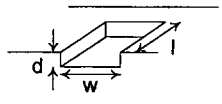
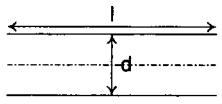
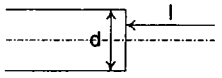
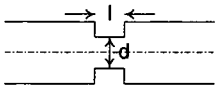
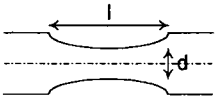
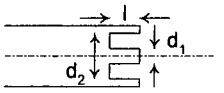
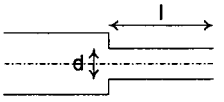
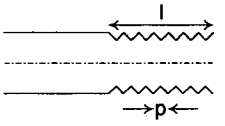
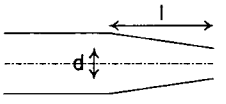
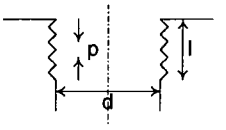
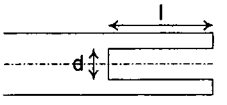
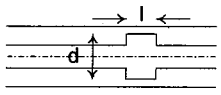
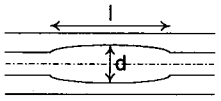
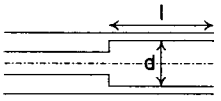
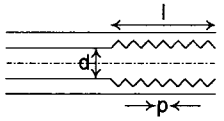
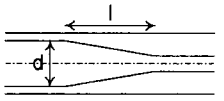
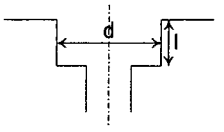
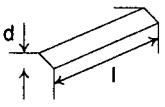
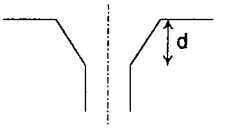
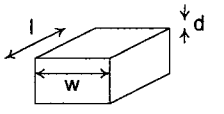
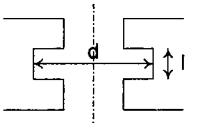
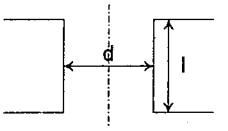
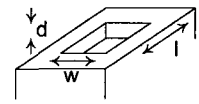
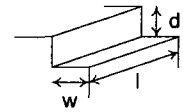
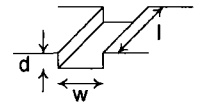
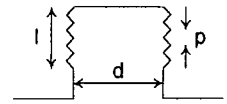
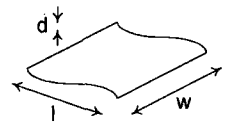
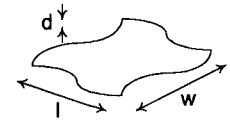
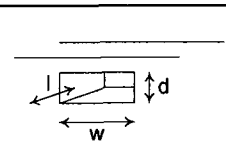
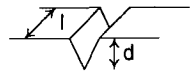
Feature Class		Diagram	Minimum Feature Relations
code	description		
pri	prism		length, width, depth
cyl	cylinder		length, diameter
she	sheet		length, width, depth
wir	wire		length
sol	solid		length, width, maximum depth
mou	moulded		length, maximum width, maximum depth

Table 2: Negative Features

Feature Class		Diagram	Minimum Feature Relations	Optional Feature Relations
code	description			
bho	blind hole		length , diameter	

cst	closed slot		length, width, depth	radius, angle
ecy	external cylindrical surface		length, diameter	
efa	end face on a cylindrical part		length, diameter	
egv	external groove on a cylindrical part		length, diameter	
epf	external profile on a cylindrical shape		length, minimum diameter	
erg	circular groove on the face of a cylindrical part		length, diameter, internal diameter	
esp	external step on a cylindrical part		length, diameter	
etd	external thread on a cylinder		length, diameter, pitch	
etp	external taper on a cylinder		length, diameter, angle	
htd	thread on a non-axial hole		length, diameter, pitch	
icy	internal cylindrical surface on a cylindrical part		length, diameter	

igv	internal groove on a cylindrical part		length, diameter	
ipf	axi symmetrical internal profile		length, maximum diameter	
isp	internal cylindrical step		length, diameter	
itd	axi symmetrical internal thread		length, diameter, pitch	
itp	axi symmetrical internal taper		length, diameter, angle	
pcb	counterbore: a square depression around a hole		length, diameter	radius
pcf	prismatic chamfer		length, depth, angle	
pcs	countersink: a chamfer around a hole		depth, angle	
pfa	prismatic face: any flat surface		length, width, depth	
pgv	cylindrical groove in a hole		length, diameter	
pho	through hole		length, diameter	

ppk	pocket		length, width, depth	
psd	shoulder on a prismatic part		length, width, depth	
pst	slot		length, width, depth	angle
ptd	thread on a cylindrical section of a prismatic part		length, diameter, pitch	
sf2	prismatic curved surface with fixed profile		length, width, depth	minimum radius
sf3	prismatic curved surface		length, width, depth	minimum radius
pky	keyway		length, width, depth	
vst	v-slot		length, depth	angle