# UNIVERSITY OF NOTTINGHAM

School of Mechanical, Materials, Manufacturing
Engineering and Management


# KNOWLEDGE-BASED APPROACH FOR THE FORMATION OF RE-CONFIGURABLE ASSEMBLY CELLS - A USE CASE STUDY

by

Ainul Akmar Mokhtar

2001

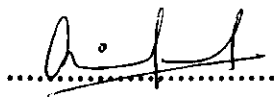
A Dissertation presented in part consideration for
a MSc Degree in Operations Management and Manufacturing Systems

# DECLARATION OF ORIGINALITY

**Title of Dissertation** Knowledge-Based Approach for the Formation of Reconfigurable Assembly Cells - A Use Case Study

This dissertation has been written by me. Material from any outside source which I have used directly or indirectly is acknowledged accurately and I have made clear the extent to which it has been used.

**Name** Ainul Akmar Mokhtar

**Signed** 

**Dated** 5 October, 2001

# ABSTRACT

The current market turbulence has forced the companies to increase their productivity in order to remain in business, not only to remain competitive. Companies that make high volume products involving labour-intensive assembly operation normally use automated assembly since it may reduce the company cost and increase productivity. Improving productivity is focused in the assembly area since it contributes a bigger portion of manufacturing cost.

A key factor for recent development in assembly automation is the need for assembly machines and modules that can be reconfigured and reused to support a wide range of assembly processes and products over a sufficiently long period of time. A re-configurable and reusable assembly system is enabling manufactures to adapt rapidly to changing market conditions by updating products or bringing in next-generation models without investing in new equipment.

The reconfigurability of the assembly system is often determined by its embedded software. The software that controls its overall system should be knowledge based since it has the advantage to be modular, re-configurable and easy to implement, maintain and upgrade. A large amount of domain specific knowledge is necessary in achieving high performance and intelligent behaviour of the knowledge-based software system in the re-configurable assembly cells. To develop domain knowledge for the software, it is necessary to elicit, analyse and document requirements of the software systems and the techniques used to achieve this is a use case technique. Use case modelling provides a convenient description of system behaviour that maps well into implementation and is useful at all stages of the software development process.

This paper will discuss the application of the knowledge-based approach and the use case technique in the formation of a re-configurable assembly system. The goal of the project is to specify functional requirements of the assembly operation using the use case technique and to use these requirements in developing knowledge-based systems aiming for the development of re-configurable assembly systems.

# TABLE OF CONTENTS

ABSTRACT.....i

TABLE OF CONTENTS.....ii

LIST OF FIGURES..... vi

LIST OF TABLES.....viii

ACKNOWLEDGEMENTS.....ix

**CHAPTER 1: INTRODUCTION**

1.1 Background Information.....1

1.2 The Motivation.....2

1.3 The Objectives.....3

1.4 Project Development Method.....4

1.5 Organisation of the Project.....5

**CHAPTER 2: AN OVERVIEW OF KNOWLEDGE-BASED SYSTEMS**

2.1 Introduction.....6

2.2 Knowledge and its Application.....7

2.3 What is a Knowledge-Based System?.....7

2.3.1 Knowledge Base.....9

2.3.2 Inference Engine.....9

2.3.3 User Interface.....10

2.4 Knowledge Acquisition.....11

2.5 Knowledge Representation.....11

2.6 Software in Knowledge-Based Systems.....13

2.7 The Object-Oriented Software Development Framework .....15

**CHAPTER 3: USE CASES**

3.1 Introduction....18
3.2 Definition of Use Cases....19
3.3 The Roles of Use Cases....20
3.4 Advantages of Use Cases....20
3.5 Disadvantages of Use Cases....21
3.6 Types of Use Cases....21
3.7 Use Case Modelling....22
3.7.1 Environment Level....23
3.7.2 Structure Level....24
3.7.3 Event level....25
3.8 Use Case Diagram....26
3.8.1 Actors....26
3.8.2 Scenarios....27
3.8.3 Use Case Relationships....27
3.9 Steps in Developing Use Cases....28

**CHAPTER 4: THE FUNDAMENTALS OF ASSEMBLY**

4.1 Introduction....29
4.2 Definition of Assembly....30
4.3 The Role of Assembly....30
4.4 Assembly Operations....31
4.4.1 Feeding Operation....31
4.4.2 Forming Operation....31
4.4.3 Fastening Operation....31
4.4.4 Inspecting Operation....34
4.4.5 Marking Operation....34
4.4.6 Packaging Operation....35
4.5 Types of Assembly Method....35
4.5.1 Manual Assembly....35
4.5.2 Semi-Automatic Assembly....36
4.5.3 Automated Assembly....36

4.6 Automated Assembly.....36
4.6.1 Types of Automated Assembly.....37
4.6.1.1 Fixed Automation.....37
4.6.1.2 Programmable Automation.....37
4.6.2 Design for Automated Assembly.....38
4.6.3 Advantages and Disadvantages of Automation.....39
4.6.4 Types of Automated Assembly System.....39
4.6.4.1 Robotic Assembly Systems.....40
4.6.4.2 Flexible Assembly Systems.....41
4.6.4.3 Re-configurable Assembly Systems.....42
4.6.5 Knowledge-Based Approach in Designing Re-configurable Automated Assembly Systems.....44

**CHAPTER 5: CASE STUDY - CONNECTOR ASSEMBLY OPERATION**

5.1 Introduction.....45
5.2 Company Background.....46
5.3 What is a Connector?.....47
5.4 Connector Assembly Process Flow.....48
5.5 How does the Assembly Process work?.....50
5.5.1 Contacts in Strip.....50
5.5.2 Housings from Mouldings.....52
5.5.3 Loose Pieces Contacts in Bags.....53
5.5.4 Assembly of Contacts in Strip and Housings.....54
5.5.5 Assembly of Loose Contacts to the Assembled Housings.....55
5.6 Evidence of Good Connector Assembly Operation.....58

**CHAPTER 6: APPLICATION OF USE CASES ON THE CASE STUDY**

6.1 Introduction.....59

6.2 Development of Use Cases.....59

6.2.1 Contacts in Strip.....64

6.2.2 Housings from Moulding.....70

6.2.3 Assembly of Contacts in Strip and Housings.....73

6.2.4 Loose Piece Contacts in Bags.....75

6.2.5 Assembly of Loose Contacts to the Assembled Housings.....77

6.3 Use Cases in Application to the Study.....85

**CHAPTER 7: RECOMMENDATIONS FOR FUTURE WORK**.....86

**CHAPTER 8: SUMMARY AND CONCLUSIONS**.....88

**REFERENCES**.....90

**APPENDICES**

Appendix A: Basic Use Case Template.....94

# LIST OF FIGURES

## CHAPTER 2: AN OVERVIEW OF KNOWLEDGE-BASED SYSTEMS

Figure 2.1: The main components of a knowledge-based system (Hopgood, 1993)
Figure 2.2: An example of a frame-based representation (Hopgood, 1993)
Figure 2.3: Categories of software in knowledge-based systems (Hopgood, 1993)
Figure 2.4: Structure of the object-oriented software development framework (Mak and Lau, 2000)

## CHAPTER 3: USE CASES

Figure 3.1: Concept relations and levels of abstraction (Regnell et al., 2000)
Figure 3.2: Use case diagram for car park information system (Bustard et al., 2000)
Figure 3.3: An example of extend relationship (Fowler and Scott, 2000)

## CHAPTER 4: THE FUNDAMENTAL OF ASSEMBLY

Figure 4.1: An example of robotic assembly line (Goetsch, 1991)

## CHAPTER 5: CASE STUDY–CONNECTOR ASSEMBLY OPERATION

Figure 5.1: Connectors placed by robots (Goetsch, 1991)
Figure 5.2: Connector Assembly (AMP Inc., 1997)
Figure 5.3: An example of a connector (AMP Inc., 1997)
Figure 5.4: Connector Assembly Process Flow (AMP Inc., 1997)
Figure 5.5: Contact Feeding – Dereeling (AMP Inc., 1997)
Figure 5.6: Contact Forming – Cutting (AMP Inc., 1997)
Figure 5.7: Contact Forming – Stamping (AMP Inc., 1997)

Figure 5.8: Housing Feeding (AMP Inc., 1997)

Figure 5.9: Housing Inspecting (AMP Inc., 1997)

Figure 5.10: Mass Inserting (AMP Inc., 1997)

Figure 5.11: Stitching (AMP Inc., 1997)

Figure 5.12: Forming – Bending (AMP Inc., 1997)

Figure 5.13: Inspecting – Pin Count (AMP Inc., 1997)

Figure 5.14: Inspecting Insertion Depth (AMP Inc., 1997)

## *CHAPTER 6: APPLICATION OF USE CASES ON THE CASE STUDY*

Figure 6.1: Use Case Structure Diagram for the Connector Assembly Operation

# LIST OF TABLES

Table 3.1: Functions in Environment Level

Table 3.2: Functions in Structure Level

Table 4.1: Flexibility requirements in different time frames (Heilala and Voho, 2001)

Table 6.1: Association between the principal actors and the scenarios

# ACKNOWLEDGEMENTS

I wish to express my gratitude to my supervisor, Dr. Svetan Ratchev, for his valuable guidance, support and advices given to me in doing this project and his time and understanding in listening to my problems.

I am very grateful to Dr. Mike Byrne, the Course Director, for his willingness to listen to my problems and his supportive help to me in completing the course.

Not to forget, I am very indebted to Mr. Hitendra Hirani for his willingness to listen to my discussions and doubts, his guidance and his innumerable suggestions and comments in reviewing the paper.

I also would like to thank to all my lecturers and friends for the constant and valuable support given directly or indirectly to me and also for sharing their valuable experience with me.

Last but not least, let me thank my beloved husband, Syukri Baharudin, my son, Ilham Syukri, and my loving family who always give motivation to me in completing this study. Without their care and infinite support, the creation of this paper would not have been possible.

# Chapter 1

# INTRODUCTION

## 1.1 Background Information

Today's marketplace is very demanding and challenging. Customers are now looking for products that are lower in price and higher in quality with shorter delivery time. Companies are increasingly facing local and international competition such as increasing product variety, shorter product life cycles and minimising the time from the concept to final product. Companies are forced to increase their productivity as a matter of survival.

To increase productivity, the assembly area needs to be optimised. Increased attention has been given to the assembly processes due to the reason that the assembly accounts for a bigger proportion of the manufacturing cost where in some cases, assembly cost is 50% or more of the manufacturing cost (Goetsch, 1991). Typically, it contributes 20% to 50% of the manufacturing cost (Goetsch, 1991). Moreover, assembly is important not only for making and delivering the products the customers want, but is also the last check on the quality of the manufacturing processes before the customer receives the products and it must be scheduled to meet the delivery requirements of the customers. Adopting automated assembly is one of the ways to increase productivity.

Automated assembly has been widely applied in electronics and mechatronics industry (Ratchev and Hirani, 2001). A key factor for recent development in assembly automation is the need for assembly machines and modules that can be reconfigured and reused to support a wide range of assembly processes and products over a sufficiently long period of time (Ratchev and Hirani, 2001). The development of re-configurable assembly systems has gained considerable attention in the industry. A re-

configurable and reusable assembly system is enabling manufactures to adapt rapidly to market changing conditions by updating products or bringing in next-generation models without investing in new equipment.

The reconfigurability of the assembly system is often determined by its embedded software. The software that controls its overall system should be knowledge based. Knowledge-based approaches have been widely used in industry because they have the advantage to be modular, re-configurable, easy to implement, maintain and upgrade, and fault-tolerant. A knowledge-based system denotes a computer program that can store knowledge of a particular domain and use that knowledge to solve problems from this domain in an intelligent way. A large amount of domain specific knowledge is necessary in order to achieve high performance and intelligent behaviour of the knowledge-based software system in the re-configurable assembly cells. One of the techniques used to capture the necessary system requirements to develop domain knowledge for the software is a use case technique.

Use case modelling is well known within the software engineering and is concerned with system description. Use case analysis has attracted considerable attention over recent years as it provides a convenient description of system behaviour that maps well into implementation. Use cases provide user-friendly documentation at all stages of the software development process. A use case can be defined as a specification of actions, including variants, which a system or other entity can perform, interacting with an actor of the system (Berg and Simons, 1999). An actor is the agent who triggers the functions of the system and it can be a human or an external system.

## 1.2 The Motivation

The project is motivated by a few factors as listed below:

- In today's manufacturing industries, the products must have these characteristics to remain competitive; they need to be lower in price and higher in quality with shorter delivery time; the product variety is increasing; they have shorter life

cycles, which leads to the minimization for the time from the concept to final product so that the products reach the market faster.

- Assembly cost contributes about 50% of the manufacturing cost and in some cases, the assembly cost is more than 50% (Goetsch, 1991).
- Market turbulence drives assembly plants to constantly adjust their production volume of products, variants and quantities (Heilala and Voho, 1997). At the same time, the assembly plant managers must protect the long-term investments in the flexible assembly system (Heilala and Voho, 1997). The best solution is the modular approach, which gives managers possibilities to adjust volume by adding new modules or to automate the manual system step by step (Heilala and Voho, 1997).
- Traditionally, the assembly systems are fixed automation, which can only handle specific types of products with high demand volumes. In order to manufacture small-batch and high-variety products effectively, a flexible automation system is required, which can be easily reconfigured and programmed. This system may allow production to run with a batch size of one in the future.

## 1.3 The Objectives

The aim of the project is to specify functional requirements of the assembly operation being studied for the development of a re-configurable assembly system using the knowledge-based approach. In order to achieve this goal, few specific objectives have been defined and are used as the project guideline. The objectives are listed below:

- To carry out literature review on the concept of knowledge-based systems.
- To understand the concept of re-configurable assembly cells or systems.
- To understand the use case technique for the elicitation, analysis and documentation of requirements on software systems.
- To relate the use case concept to the knowledge-based approach in the formation of a re-configurable assembly cell.
- To analyse the case study of the AMP connector assembly operation.

- To develop related use cases for the case study of the AMP connector assembly operation and link them to specific assembly modules.
- To provide some recommendations for future work related to the topic discussed in the project.

## 1.4 Project Development Method

The technique, which will be employed in the software development process, is the use case technique. A use case is a tool for describing and understanding system functional requirements. Once all the functional requirements are identified, the system designer can use them for developing the domain knowledge in the knowledge-based software system, which has the advantage of being modular, re-configurable, easy to implement, maintain and upgrade, and fault-tolerant. A re-configurable assembly system also enables companies to act fast to market changes so that products can be update easily and new product can be produced without investing in new equipment.

## 1.5 Organisation of the Project

This presentation of the project is divided into two main parts, which are the literature review and an industrial case study. For the first part, sources were obtained from the university library, the project supervisor, one of the PhD students who is doing research related to this topic and the electronic journals. Sources used are books, articles from journal, previous dissertations and related website. For the second part, a case study used is the assembly operation of AMP connector. The concept found in the study is being illustrated using this case study.

This paper consists of eight chapters:

- Chapter 1 introduces the project and the topics being discussed in this paper. It gives a brief introduction to the topic of knowledge-based systems as an approach for the formation of re-configurable assembly cells or systems.
- Chapter 2 covers the literature review on the knowledge-based systems.
- Chapter 3 describes the use case technique, which is a useful, powerful technique in the software development process for the object-oriented based system. Use cases have become the norm for system functional requirements capture in object-oriented projects leading to the development of the domain knowledge for the system software.
- Chapter 4 discusses the fundamental of the assembly system and the concept of re-configurability.
- Chapter 5 describes a case study of the AMP connector assembly operation.
- Chapter 6 presents the application of use case development using the case study of the AMP connector assembly process. This chapter is the main core of the project.
- Chapter 7 focuses on the recommendations and points need to be considered for future work.
- Chapter 8 concludes the paper with a summary of the material covered and conclusions of the project.

# Chapter 2

# AN OVERVIEW OF KNOWLEDGE-BASED SYSTEMS

## 2.1 Introduction

At today's stage, we are living in the knowledge-intensive manufacturing industry where computers help human experts for data processing and decision-making process. The tendency to replace the power of human brain by computers is developing into decision-making automation, which is based on the technology of knowledge processes. Knowledge-based approaches have been widely used in manufacturing industry because they offer the advantage to be modular, scalable, re-configurable, distributed, easy to implement and maintain and fault-tolerant.

Knowledge-based systems, also known as expert systems, are computer programs embodying knowledge about a narrow domain for solving problems related to that domain (Pham and Pham, 1999). Knowledge-based systems are based on the technology of knowledge processes. Knowledge-based systems have been applied in many areas, for instance, finance, chemistry, geography, medical field and engineering. Examples of the application of knowledge in engineering are selection of materials, machine elements, tools, equipment and processes, signal interpreting, condition monitoring, fault diagnosis, machine and process control, machine design, process planning, production scheduling and system configuring.

This chapter will discuss what knowledge-based systems are, the important elements in the knowledge-based systems, the meanings of knowledge acquisition and knowledge representation and the existing software in the knowledge domain. A framework for the development of an object-oriented system is also presented here.

## 2.2 Knowledge and its Application

Knowledge can be defined as any information that is useful for the task being performed (Ranky 1990). In the knowledge-based systems scope, knowledge is the information that a computer program must have to behave 'intelligently'. In developing a highly complex structure of a re-configurable assembly system, the application of knowledge is employed to translate the user or system requirements into a set of design rules and potential cell configurations.

## 2.3 What is a Knowledge-Based System?

In a very broad sense, knowledge-based systems are programs that use human-like reasoning processes, which rely on experimental human knowledge or expertise represented in a knowledge base for a specific problem (Ranky, 1990). A knowledge-based system denotes a computer program that can store knowledge of a particular domain and use that knowledge to solve problems from this domain in an intelligent way. When knowledge-based systems approach the performance levels of a human expert, they are called expert systems (Ranky, 1990). The knowledge-based system approach is straightforward. The system should provide the following features (Kriz, 1987):

- Explanation of its behaviour on request by the user;
- User-friendly dialog that takes the user and the kind of the application into account;
- Application-oriented knowledge representation language; and
- Possibility to easily modify and extend the knowledge base during the lifetime of the system. The knowledge is represented explicitly in the knowledge base, not implicitly within the structure of a program (Hopgood, 1993). Therefore, the knowledge can be altered relatively easily.

In the simplest case of the knowledge-based systems, the systems consist of two main components, namely, the knowledge base and the inference engine (refer to Figure

2.1). The knowledge base is the knowledge module that consists of a set of facts and/or rules and the inference engine is the control module, which implements a finite set of mappings, called rules of inference.

The explicit separation of knowledge from control makes it easier to add new knowledge either during program development or in the light of experience during the program's lifetime (Hopgood, 1993). There is an analogy with the brain, whose control processes (the inference engine) are approximately unchanging in their nature, even though individual behaviour is continually modified by new knowledge and experience (updating the knowledge base) (Hopgood, 1993).

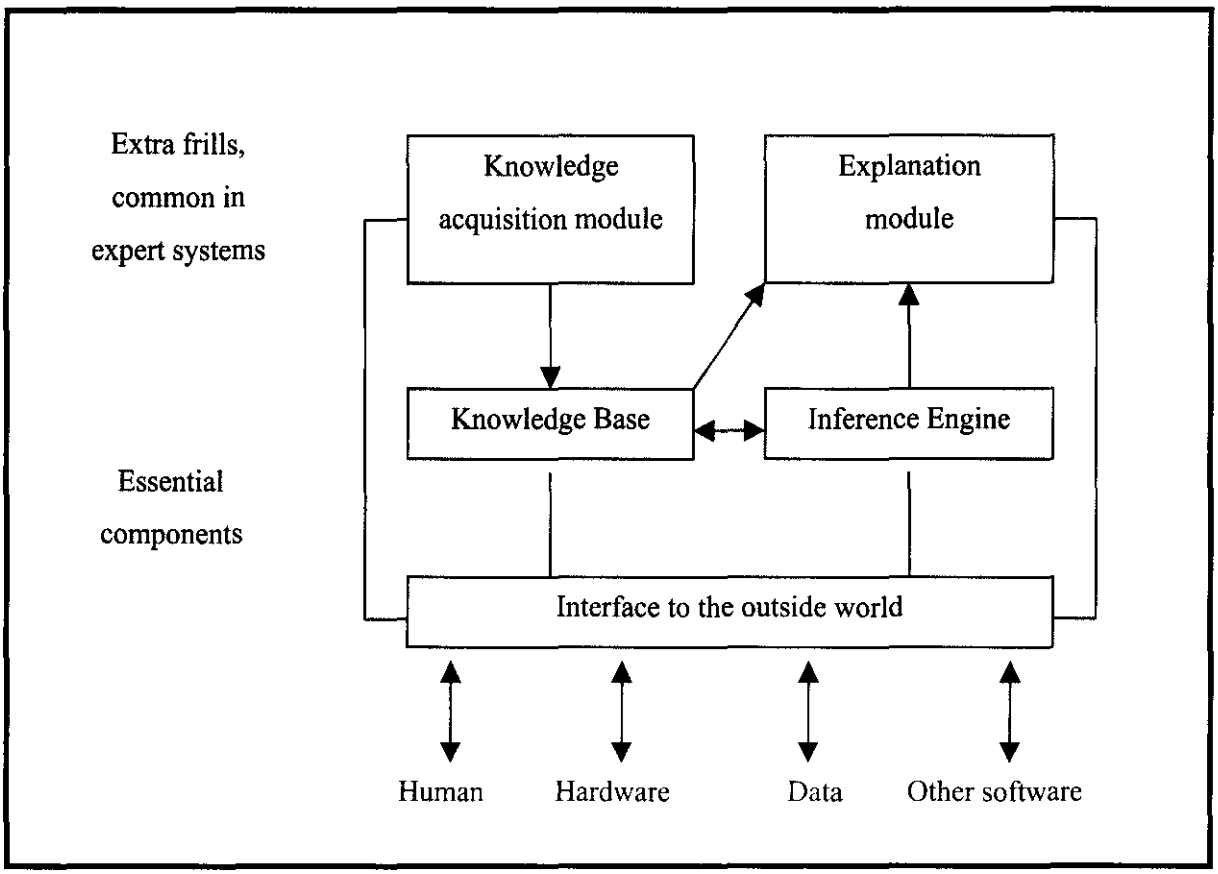



Figure 2.1: The main components of a knowledge-based system (Hopgood, 1993)

### 2.3.1 Knowledge Base

The knowledge base, in which the domain-specific knowledge is stored in form of facts and rules, represents knowledge of facts, general information and heuristics[1] such as judgements, intuition and experience. Normally, the knowledge base contains the following:

- A collection of basic facts, which often referred to a database;
- A set of rules that describe relations, which called as a rule base; and often
- Methods for solving problems in the given domain.

During a consultation with the expert system, the rules are used in conjunction with the facts in order to deduce or infer new facts relevant to the solution of any particular problem that the expert system is working on (Ranky, 1990).

### 2.3.2 Inference Engine

Decision-making by the inference engine is a fundamental part of the execution of a knowledge-based system. The inference engine, also known as an inference procedure, is a computer program that is capable of accessing and using the knowledge base. The inference engine is responsible to draw conclusions by using data, rules and relationships from its knowledge base, and whatever it has learned to justify the conclusions using conditional statements for its reasoning process.

An inference process consists of a number of inference steps, each step creating additional knowledge. A set of inference rules in the knowledge-based system can

[1] A heuristic is a rule of thumb, a trick, a strategy, a simplification, a clever short cut or other method that aids in the solution of complex problems (Ranky, 1990). There are two kinds of heuristics, namely special and general. A special heuristic is the kind that applies to a particular problem, whereas a general heuristic is a process or philosophy that can be applied to a wide range of problems.

manipulate information in the knowledge base. An inference mechanism interprets the knowledge in the knowledge base and performs logical inferences and deduces new knowledge by applying rules to facts. The knowledge-based system can accept a number of input facts and based on these facts, together with other facts and rules stored in the knowledge base, the system can infer new facts, which are not explicitly stored in the knowledge base. Typically, the control structure is in the form of

IF...THEN...ELSE...

Or

FOR X FROM A TO B.

There are two important types of inference engine, which are forward chaining[2] (or data-driven) and backward chaining[3] (goal-driven). A knowledge-based system working in data-driven mode takes the available information (the "given" facts) and generates as many derived facts as it can (Hopgood, 1993). Thus, the output is typically unpredictable.

An important feature of the inference engine is its ability to deal with incomplete or uncertain information (Ranky, 1990). A variety of techniques can be employed, for instance, the use of certainty factors, degrees of belief, Bayesian probabilities and measured based fuzzy logic.

### 2.3.3 User Interface

The user interface provides the communication vehicle between the users and the system in order to provide the user with some insight into the problem solving processes carried out by the inference engine (Ranky, 1990).

[2] In forward chaining, a certain true situation triggers the action or consequent part of a rule. An example in process control is a sensor sending a new value to the system, which may trigger alarms, infer a solution and take corrective action (Ranky, 1990).

## 2.4 Knowledge Acquisition

Knowledge acquisition is the process of extracting, structuring and organising knowledge from some sources (Ranky, 1990). The acquired knowledge is stored in the knowledge base. There are three different methods in acquiring the relevant knowledge for a particular domain (Hopgood, 1993):

- The knowledge is teased out of a domain expert;
- The builder of the knowledge-based system is a domain expert; or
- The system learns automatically from examples.

## 2.5 Knowledge Representation

Knowledge representation is the process of structuring knowledge about a problem in a way that makes the problem easier to solve (Ranky, 1990). In order to achieve high performance and intelligent behaviour of a knowledge-based system, a large amount of domain-specific knowledge is necessary. There are several special methods for knowledge representation such as frame, object and rules. These methods allow the representation of well-established and formalized knowledge about a particular area.

- ***Frame Based***: Frame is a knowledge representation method that associates features with nodes representing concepts or objects. A frame, also known as a schema, is a data structure, which contains various properties arranged in slots. Slots are variable-sized memory areas and may contain standard attributes about the object, hypotheses that relate to the programs function or rules about application area situations and actions to take under certain conditions. Slots may contain subprogram that point to and link the slots of one frame to other frames, and slots in one frame may also be independent frames, with their own complete

[3] In backward chaining, a specific goal is established and a hypothesis of how to achieve the goal is made. The system backtracks through the rules to find evidence to support the hypothesis.

sets of slots of attributes, hypotheses, rules and processes, forming a hierarchy of relationships. Figure 2.2 shows an example of a frame-based representation.

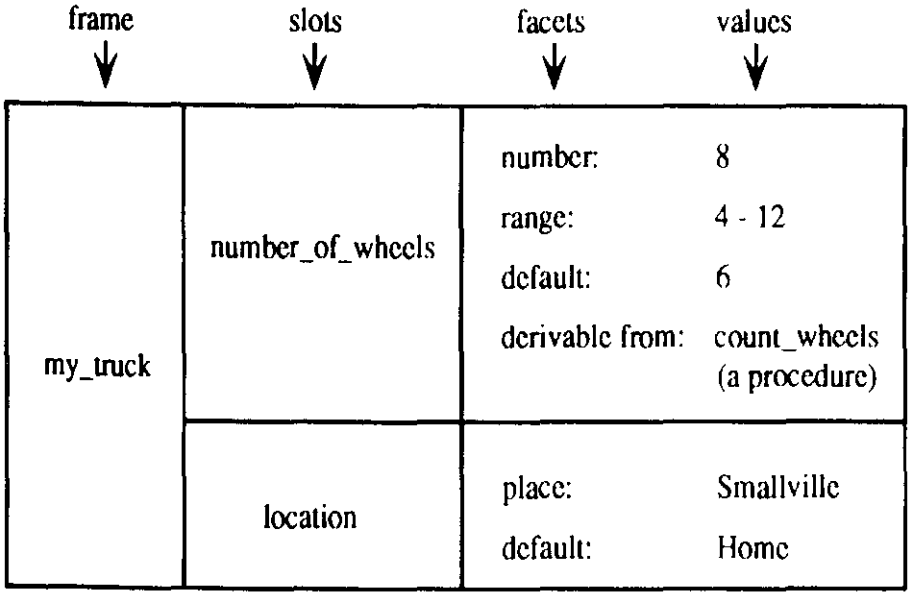



Figure 2.2: An example of a frame-based representation (Hopgood, 1993)

- ***Object Based***: Object is any item on a layout. Layout objects can be active or passive. Active objects perform an action whereas passive objects are illustration only. Object based representation can be represented as nodes and lines. The nodes represent objects, actions, or events whereas the lines represent the relationship between objects since they connect nodes together. The nodes can have procedures or methods attached to them and are activated when a message is sent to the object. The message is actually the name of a method or procedure. Object base representations can be accommodated within a frame.
- ***Rule Based***: A rule is a conditional statement that specifies an action that is supposed to take place under a certain set of conditions. In the rule based languages, a sequence of steps for a program is not required. The languages require conditions indicating some situation to be described, specify the action that should occur if the situation is true and then dump the whole set of the structured rules into the program. Rules can be integrated into frame or object based representations.

Several different knowledge representation techniques are normally used such as formal mathematical logic, state-space-search, procedural, semantic nets, production systems and frames.

## 2.6 Software in Knowledge-Based Systems

There are several categories of software in knowledge-based systems (refer to Figure 2.3), namely, expert systems, rule-based systems, blackboard system and object-oriented and frame-based systems.

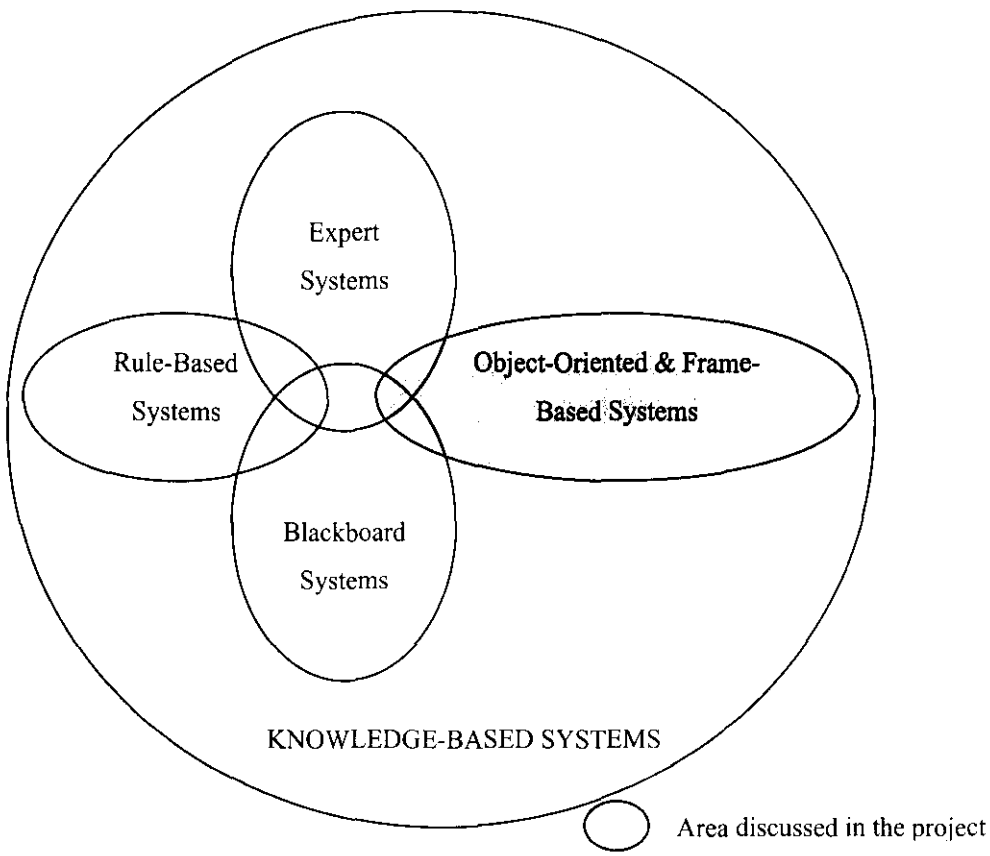



Figure 2.3: Categories of Software in Knowledge-Based Systems (Hopgood, 1993)

- *Experts Systems*: An expert system is a computer program, which behaves like a human expert in some usually narrow domain of application (Hopgood, 1993). An expert system should have the capability of dealing with uncertain and incomplete

information and should have the ability to explain its decision-making reasoning process.

- *Rule-Based Systems*: A rule-based system is a knowledge-based system where the knowledge base is represented in the form of a set (or sets) of rules (Hopgood, 1993). Rule represents knowledge that can be used by the computer without specifying how and when to apply that knowledge. The simplest type of rule is called a production rule and takes the form:

  IF <condition> THEN <conclusion>

  An example of a production rule is:

  IF <the tap is opened> THEN <the water flows>

- *Blackboard Systems*: A blackboard is a database accessible to independent knowledge sources and used by them to communicate with each other. The information they provide relates to intermediate results of problem solving. A blackboard system might provide an ideal way of breaking down a problem into subtasks. Blackboard systems allow each subtask to be handled using an appropriate technique, therefore they may contain several different knowledge bases, each with its own inference engine.
- *Object-Oriented Systems*: Object-oriented programming offers a way of representing the things that are being reasoned about, their properties and the relationships between them. Decomposing a problem using object-oriented programming helps in the software design and makes it more maintainable, adaptable and recyclable.
- *Frame-Based Systems*: Frame-based systems are closely allied to object-oriented systems. Frame-based systems evolved from artificial intelligence research whereas an object-oriented programming developed as 'a better way to program' (Hopgood, 1993).

## 2.7 The Object-Oriented Software Development Framework

The object-oriented system development framework provides a systematic approach in capturing the features and behaviour of the system. The framework has the following features (Mak and Lau, 2000) (refer to Figure 2.4):

- Object-oriented techniques to document a system;
- Heuristics to guide the system software representations to transform from the abstract to the concrete; and
- A scheme to check the correctness of such transformations.

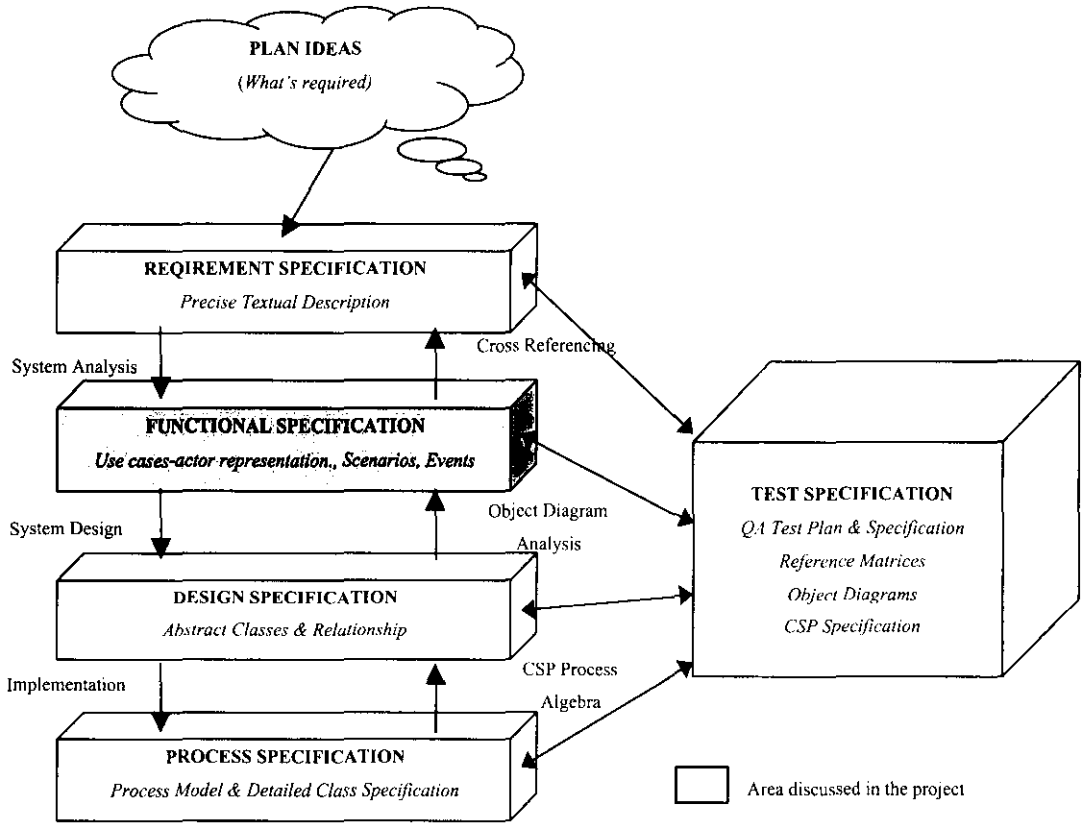



Figure 2.4: Structure of the object-oriented software development framework (Mak and Lau, 2000)

The framework consists of five system specifications to quantify software production as well as refinement procedures and verification procedures to monitor and control the overall software development. The five system specifications are defined as the following (Mak and Lau, 2000):

- *Requirement specification* captures the requirement of the software with reference to customer's requirement.
- *Functional specification* formalises and structures the requirement definition in an object-oriented notation.
- *Design specification* documents the refined entities and their relationships specified in a functional specification.
- *Process specification* consists of detailed class specifications, a process model that specifies the software architecture and a list of implementation attributes.
- *Test specification* consists of cross reference matrices, object diagrams and formal specification of processes.

There are two major activities in the software development framework namely, refinement and verification. The refinement process transforms the requirement specification into actual software through various specifications in a traceable manner. There are three main steps in the refinement process, which are system analysis, system design and implementation (Mak and Lau, 2000).

- *System Analysis*: System analysis transforms the requirement into a functional specification by identifying essential functionality and information. It also defines the scope of the system. Actors and use cases are being identified and grouped under corresponding scenarios. This is the area where the project will be focused on.
- *System Design*: Design specification is produced by system design. The design process refines the functionality specification with the objective of identifying the primary entities in the form of abstract classes to make up the software from the scenarios.

- *Implementation*: In implementation, the design specification is refined further into a solution, which can be directly implemented by incorporating constraints to the abstract classes. These classes are then grouped into the physical processes.

The verification procedures provide a scheme to verify the correctness, completeness, consistency and functionality of the refinement process at every stage of its development. The procedures used for the verification processes are cross-referencing matrices, object diagrams and CSP process algebra (Mak and Lau, 2000).

- *Cross-Referencing Matrices:* Cross-referencing matrices are used to establish the completeness and consistency between the requirement specification and the functional specifications.
- *Object Diagrams:* The system design correctness is verified using object diagrams that represent the particular scenarios. In order to prove that a design specification is complete in terms of functional specification, the object diagrams are used to show that the classes identified can perform all the system functionality.
- *CSP Process Algebra:* CSP process algebra is used to specify the process model. Formal verification can be performed to predict the system behaviour and, in turn, to deduce the correctness of the implementation with respect to the design specifications as well as the original requirements.

The area of interest in this project is particularly in the system analysis where the knowledge is being extracted here. Knowledge is used to transform the system requirements into functional specifications. A tool that will be used in extracting the knowledge is the use case technique where it captures all the necessary system requirements for the software development by identifying the actors and use cases and grouping them under corresponding scenarios. The use case technique will be further discussed in the next chapter, Chapter 3.

# Chapter 3

# USE CASES

## 3.1 Introduction

To achieve the goal of producing re-configurable assembly systems, the software used in the assembly cell should be compatible for re-configuration, maintenance and upgrading. A large amount of domain specific knowledge in the software is necessary in order to achieve high performance and intelligent behaviour of the knowledge-based software system in the assembly cells. Without proper capture of the overall system requirements for the development of domain knowledge in the software, the software projects may face a high probability of failure. Use cases have become extremely popular as an effective software development technique due to its capacity for capturing functional requirements[4].

The use case driven approach was originally introduced by Jacobson (Kim and Carlson, 2001). Use case modelling was first presented as part of the Object-Oriented Software Engineering (OOSE) methodology for software development (Bustard et al, 2000). The use case technique has also gained widespread acceptance within methods and notations such OMT (Object Modelling Technique), ROOM (Real-Time Object-Oriented Modelling) and UML (Unified Modelling Language). Use case analysis has attracted considerable attention over recent years as it provides a convenient description of system behaviour that maps well into implementation and which has

[4] Requirements may be functional or non-functional. Functional requirements concern with those things that the system must do. System functional requirements can be defined as the descriptions of what the system is to do, for example, what information needs to be maintained and what needs to be processed. Non-functional requirements are mostly constraints on the system design; for instance, the system must be operated on certain types of electrical power.

now been integrated into the UML standard for object modelling (Ratcliffe and Budgen, 2001).

The purpose of this chapter is to present the use case technique, which has become the norm for functional requirements capture in object-oriented projects. To illustrate the application of the use case technique, a case study of the AMP connecter assembly operation will be used. This case study will be presented in Chapter 5. Chapter 6 will demonstrate how the use case concept is being modelled in the case study of the connector assembly operation.

## 3.2 Definition of Use Cases

A use case tells a story of how a goal succeeds or fails or a set of stories of both getting and failing a goal. In the literature review, there are several definitions of uses cases found such as:

- A use case is a specification of actions, including variants, which a system or other entity can perform, interacting with an actor of the system (Berg and Simons, 1999).
- A use case is defined as a sequence of ordered events that illustrate the behaviour of the system and its components when a service is invoked (Tuok and Logrippo, 1998).
- A use case is a description of one or more end-to-end transactions involving the required system and its environment (Rolland and Achour, 1998).
- A use case is a description of system usage, documenting transactions or sequences of interrelated events initiated by an actor (Bustard et al., 2000).
- A use case is a complete description of all possible normal and non-normal flows of action (Rolland & Achour, 1998).

## 3.3 The Roles of Use Cases

Use cases are useful at all stages of the software development process. At the early stage of the development process, use cases are used to specify requirements and to guide system design. At the final stage, they are useful in verifying the correctness, completeness, consistency and functionality of each stage of the system development process.

The elicitation, analysis and documentation of requirements on software systems are a crucial and non-trivial task (Regnell et al., 2000). Well-defined concepts and methods are needed when constructing specifications that represent requirements in an unambiguous, consistent and complete manner. It is also important to have representation of requirements that are easily understood by different stakeholders that take part in requirement analysis (Regnell et al., 2000). Use cases are widely regarded as powerful tools for describing and understanding system requirements because they facilitate human understanding. Use cases are also conductive to creativity needed for performing requirement elicitation, analysis, documentation and validation.

## 3.4 Advantages of Use Cases

The use case technique offers several advantages such as:

- Use cases provide user-friendly documentation at all stages of the software development process;
- Use cases can be changed when functionalities or system requirements are added or changed; and
- Use cases address functional requirements in an easy-to-read and easy-to-track text format.

## 3.5 Disadvantages of Use Cases

However, using use cases also gives some drawbacks, for instance:

- It is difficult to ensure consistency and completeness of the set of use cases throughout the lifecycle;
- Changing one use case may require to visit all other use cases in order to determine whether the change will affect them or not;
- The use case model does not show the starting and terminating contexts of the use cases;
- Deriving a large set of use cases manually is time consuming; and
- Use cases show only the functional requirements. They do not capture performance requirements and do not show interface requirements.

## 3.6 Types of Use Cases

There are five major types of use cases, namely, common use cases, variant use cases, component use cases, specialised use cases and ordered use cases (Berg and Simons, 1999).

- *Common use cases*: Common use cases are factored out so that these can be used or reused by other use cases without repeating the description.
- *Variant use cases*: Alternatives to the normal use case behaviour are captured in variant use cases. They are also used for exceptions.
- *Component use cases*: Parts of use cases are further defined in component use case, which lead to a hierarchical decomposition of use cases.
- *Specialised use cases*: Use cases may be categorized in more specialised version.
- *Ordered use cases*: Ordered use cases deals with situations where the completion of one use case is required before the following use case can be executed (Berg and Simons, 1999).

## 3.7 Use Case Modelling

Use case modelling is well known within the software engineering and is concerned with system description. Use case modelling is a requirement engineering[5] technique that identifies the system activities and is normally driven by the needs of the system's users.

A use case model or diagram describes a collection of related use cases where each use case describes a system function. The use cases determine the order of the events and define the possible alternatives in the flow of events (Regnell et al., 2000). Each use case covers a set of scenarios (Regnell et al., 2000). In use cases, there are three different abstraction levels, namely, environment level, structure level and event level (refer to Figure 3.1).

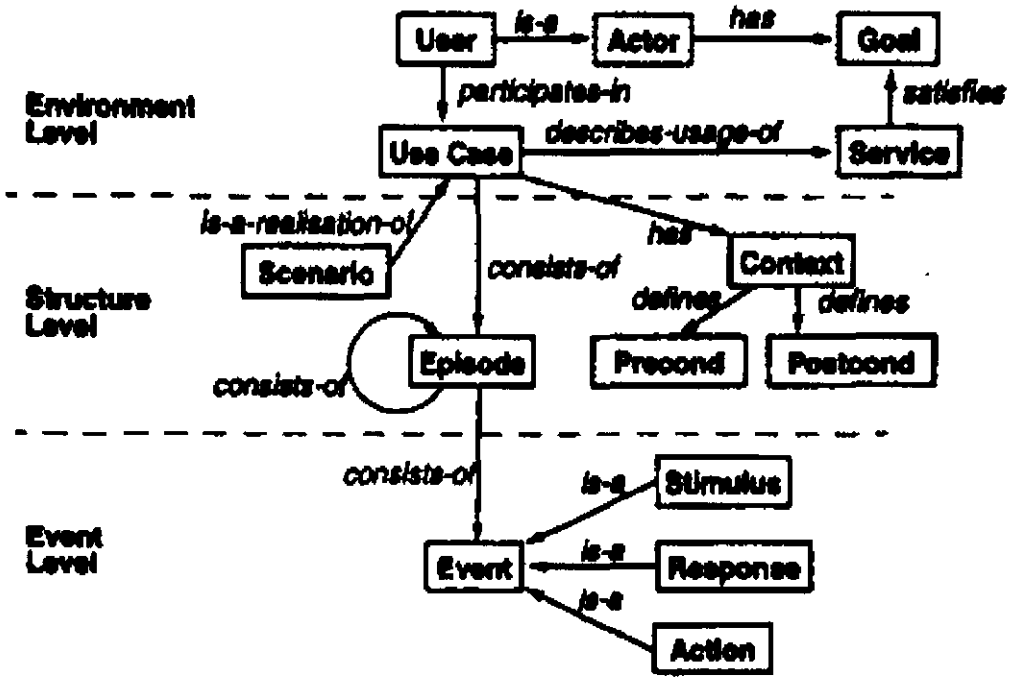



Figure 3.1: Concept relations and levels of abstraction (Regnell et al., 2000)

[5] Requirement engineering is the heart of software development process that is concerned with identifying the purpose of a software system and the contexts in which it will be used. There are four steps in requirement engineering, namely, requirement elicitation, requirement analysis, requirement specification and requirement validation.

### 3.7.1 Environment Level

The environment level consists of users, actors, goals, services and use cases. Table 3.1 below describes the functions of each element:

| Element | Description |
|---|---|
| User | Users can be either humans or other software or hardware based systems. Users belong to the intended target system's environment. |
| Actor | An actor is also called a *user type* or an *agent*. It represents a set of users that have some common goals and triggers the functions of the system. Actors can be human or otherwise. |
| Goals | Goals are the objectives or the aims that the users have when using the services of the target system. Goals are often used to categorize users into actors. |
| Service | In the intended target system's environment, there are a number of services. A service is a package of functional entities or features offered to the users in order to satisfy one or more goals that the users have (Regnell et al., 2000). |
| Use Case | A use case represents a usage situation where one or more actors use one or more services of the target system aiming to accomplish one or more goals. A use case may contain an unlimited number of scenarios including alternatives and repetitions. |

Table 3.1: Functions in Environment Level

### 3.7.2 Structure Level

The structure level includes concepts relating to the use cases structure such as different variants and parts of a use case. The structure level consists of scenarios, contexts, pre-conditions, post-conditions and episode. Table 3.2 below describes the functions of each element.

| Element | Description |
|---|---|
| Scenario | A scenario is a specific and bound realisation of a use case described as a sequence of a finite number of events. A scenario may either model a successful or an unsuccessful accomplishment of one or more goals. |
| Context | A context limits the scope of the use case and defines its pre-conditions and post-conditions. |
| Pre-condition | A pre-condition is a property of the environment and the target system that need to be fulfilled in order to invoke the use case. |
| Post-Condition | A post-condition is a property of the environment and the target system at use case termination. |
| Episode | In use cases and scenarios, it may be possible to identify coherent parts, called episodes (Regnell et al., 2000). Similar event sequences may occur in several use cases and episodes can be used as a modularisation mechanism to encapsulate use case parts and create a hierarchical use case model (Regnell et al., 2000). |

Table 3.2: Functions in Structure Level

### 3.7.3 Event level

An event is the lower abstraction level of use cases, scenarios and episodes. There are three types of an event, namely, a stimulus, a response and an action (Regnell et al., 2000).

A *stimulus*: A message from the users to the target system.

A *response*: A message from the target system to the users.

An *action*: A target system intrinsic event, which is atomic in the sense that there is no communication between target system and the users that participate in the use case.

Stimuli and responses can have parameters that carry data to and from the target system.

## 3.8 Use Case Diagram

Use cases can be viewed in a diagram where it illustrates a set of use cases for a system, the actors and the relationship between the actors and the use cases. The use case diagram is a part of the UML. Figure 3.2 below shows the use cases of the information system for the car park.

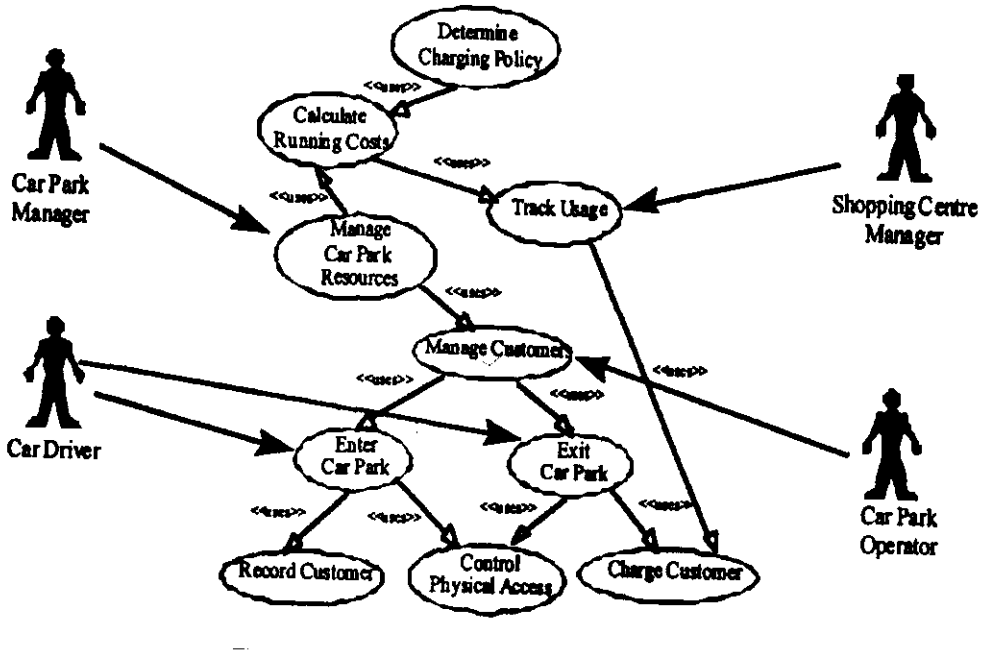



Figure 3.2: Use case diagram for car park information system (Bustard et al., 2000)

### 3.8.1 Actors

Actors are the agents (human or otherwise) who trigger the functions of the system. An actor can also be an external system that needs some information from the current system. As mentioned before, an actor is also called a *user type* or an *agent*. It represents a set of users that have some common goals. A use case may have one actor or several actors performing it. Conversely, a single actor may perform several use cases. For instance, by performing a use case analysis on the car park information system, all corresponding use cases and four actors have been identified. The actors

are Car Park Manager, Car Driver, Car Park Operator and Shopping Centre Manager (refer to Figure 3.2).

### 3.8.2 Scenarios

The concept of 'scenarios' that describes system use has been employed by engineers of all kinds to aid the conceptualisation of a problem and its solution (Ratcliffe and Budgen, 2001). Scenarios are fundamental to the description and comprehension of systems of any type (Bustard et al., 2000). Scenarios can be used in system development at different logical levels for different purposes in different ways (Bustard et al., 2000).

A scenario is a 'story', involving people and activities. A scenario, also known as a use case instance, is a specific sequence of actions as specified in a use case that is carried out under certain conditions. Scenarios can be viewed from four perspectives (Bustard et al, 2000):

1. *Process perspective*, focusing on actions and events rather than individual acts;
2. *Situational perspective*, describing concrete problematical situation;
3. *Choice perspective*, examining alternative options, only one of which must be selected; and
4. *Use perspective*, examining an artefact from a user, designer or stakeholder's viewpoint.

### 3.8.3 Use Case Relationships

Several types of relationships can be shown between use cases. They are include-relation, generalization-relation, extend-relation and precede-relation.

- *Include*: Include-relation is also known as uses-relation. A use case may depend on other use cases. In this case, the include-relation is used. One use case uses other use cases in order to avoid repetition.

# Chapter 4

# THE FUNDAMENTALS OF ASSEMBLY

## 4.1 Introduction

Assembly is an important stage for product development, which affects the product quality. Assembly in manufacturing process consists of putting together all the component parts and sub-assemblies of a given product, fastening, performing inspections and functional tests, labelling, separating good assemblies from bad and packaging and preparing them for final use (Goetsch, 1991).

Assembly has traditionally been one of the highest areas of direct labour cost. In some cases, assembly accounts for 50% or more of manufacturing costs and typically 20% to 50% (Goetsch, 1991). However, close cooperation between design engineers and manufacturing engineers in evaluating a product design has resulted in reducing and, in some cases, eliminating the need for assembly. Design for assembly (DFA) is commonly being practised nowadays due to potential production savings, better quality and improved product reliability.

The purpose of this chapter is to provide a basic understanding on assembly by giving a definition of assembly, the role of assembly in manufacturing processes, examples of assembly operation and types of assembly method. Then, the chapter proceeds with a discussion on automated assembly operations and types of automated assembly systems. The concept of re-configurability will also be discussed here.

## 4.2 Definition of Assembly

Assembly is defined as follows by DIN 8593 (Lotter, 1989):

> *Assembly, often also defined as connecting, is the joining together of two or more workpieces of a specified regular geometrical form or similar workpieces using a formless material. In doing so, a bond is formed locally which generally enlarges. Accordingly, the act of placing together and filling is defined as assembly. The assembly of various parts of one and the same object, e.g. a ring, falls within the definition of assembly. On the other hand, the application of layers of a formless material on to workpieces falls within the main group of coating.*

The main objective of assembly is to form a part of higher complexity with specified functions in a specific period of time from the individual parts.

## 4.3 The Role of Assembly

The importance of assembly is not only to make and deliver the products the customers want, but also (AMP Inc., 1997):

- Assembly is the last checking point for the quality of the manufacturing processes before the customer receives the products.
- Assembly has to fit together parts from other manufacturing processes which may be on the edge of allowable tolerances.
- Assembly must avoid adding more value to non-conforming components, which cannot produce conforming parts.
- Assembly must schedule to meet the delivery requirements of the customer
- Assembly uses the components available in slightly different combinations to make a much larger variety of products to fill customer needs.

## 4.4 Assembly Operations

Assembly operations are processes of assembling together. The goals of assembly are to put together all the component parts and sub-assemblies of a given product, fastening them, performing inspections and functional tests, labelling, separating good assemblies from bad and packaging and preparing them for final use. The common assembly operation in manufacturing process are feeding, forming, inserting, fastening, inspecting, marking and packaging.

### *4.4.1 Feeding Operation*

The feeding operation must fulfil the functions of storing, ranging, feeding or placing in a magazine of individual parts. The feeder unit is available in several forms such as a vibratory spiral conveyor, a centrifugal bowl feeder, pallet feeding, strip feeding, inclined conveyors and many others.

### *4.4.2 Forming Operation*

Forming is the operation to make physical changes to the fed components. Forming operation may include cutting, profiling, bending, kinking, punching, drilling and tapping.

### *4.4.3 Fastening Operation*

Fastening can be classified into two main groups, namely, permanent fastening and semi-permanent fastening. Below are some common permanent fastening techniques in assembly operations (Owen, 1984):

- *Crimping*: Crimping involves the deformation of one component into a groove or hole in another component.

- *Stitching*: This technique is used under the following circumstances:
  - → When it is desired to fasten together certain dissimilar metals or non-metallic material to metals.
  - → When high speed fastening is required.
  - → When elimination of pre-drilled or pre-punched holes is desired.
  - → When it is necessary to join coated materials with minimum disturbance to the coating.
- *Staking*: This generally refers to the use of lanced tabs on one components being inserted into slots in another component and then being deformed resulting a permanent joint. This techniques usually used in the sheet metal industry.
- *Welding*: There are five types of welding process, namely
  - → *Arc welding*. This method of welding applies high energy and a filler metal to join the components.
  - → *Friction welding*. This method is accomplished by the relative rotation of the components to be joined. This results in frictional heat that causes plastic flow and the two components are forged together with an axial force.
  - → *Laser welding*. Laser welding uses a laser to heat a local area on a component such that a molten zone is formed resulting a solid-state weld.
  - → *Spot welding*. This method involves the application of an electrical current and a pressure to a localized area such that the components are metallurgically joined together.
  - → *Ultrasonic welding*: This welding technique uses ultrasonic vibrations to the components being joined so that solid-state joints are made. It applies to both metals and plastic.
- *Riveting*: This generally refers to a one-piece fastener that is used to secure components by the deformation of its headless end resulting in a large clamping and frictional forces. Rivets can be classify into two basic categories:
  - → Rivets that requires access by personnel and/or tooling to *both* sides of the elements to be joined together. This is the traditional method of riveting. It applies to a whole range of rivet types, configurations and sizes.
  - → Rivets that can be inserted and deformed from *one* side of the elements being riveted. Special tooling is required and it can only be performed with a specific style of rivet.

- *Soldering / brazing*: Soldering and brazing are two similar processes where two elements are joined together using a "filler" metal. The filler metal is normally in molten form. It penetrates the interface void between the two elements by means of capillary flow and when it has solidified, a joint has been formed. There are a number of automatic machines available now for these two processes, for example, equipment for soldering printed circuit board.

Examples of semi-permanent fastening operation are (Owen, 1984):

- *Screwing*: Screws are used for joining components together. Screws are threaded devices that are less than 6.00 mm in diameter. Screws are available in two different types, namely, self-tapping[6] and standard threading[7].
- *Pinning*: In pinning operation, a dowel or spring pin is used to hold two components together. A hole is required to be drilled into both components before a pin can be inserted.
- *Bolting*: This fastening method utilizes large diameter bolts which are screwed into a threaded hole of the component generating sufficient magnitude of a clamping force so that the two components hold together.
- *Clipping*: Clipping operation uses the elastic properties of material requiring the automatic elastic deformation of one or both components. The deformed components snap into recesses such that they cannot be disassembled without the application of special tooling.

[6] Self-tapping screw requires only a drilled hole since the screw cuts its own thread as it is screwed into position (Owen, 1984).

[7] Standard threading screw requires a tapped hole in a component, such that a clamping force can be generated between the underside of the screw head and the top surface of the component with the clearance hole (Owen, 1984). This type of screw is generally not used with a nut.

### 4.4.4 Inspecting Operation

The purpose of inspecting operation is to ensure the product is being made correctly. In an inspection station, the part attributes are being inspected, compared to a standard and the rejected parts are segregated from the good ones. Rejecting parts will follow by rejecting actions and there are two rejecting actions usually take place (AMP Inc., 1997):

- Reject the non-conforming part so that it does not reach packaging
- Stop machines so the operator can resolve the problem

This operation can be manual or automated. For highly automated inspecting operation, the non-conforming parts may not be rejected and segregated earlier. The control system may track the non-conforming parts until they reach a shared reject station, then they are segregated.

The types of inspecting operations found in assembly fall into three basic categories (AMP Inc., 1997):

- Physical inspection of presence or position by touching the product
- Optical inspection using vision or beams
- Electrical circuit testing

### 4.4.5 Marking Operation

Marking functions to identify the products. The marking goal is to put the desired marking on each part in the desired location. Several types of marking are used on products such as:

- Name or other identifying product name or mark
- Part number
- Control information such as date, plant and lot number
- Functional markings such as circuit number, colour coding or instructions.

Marking operations found in assembly can be classified into three basic categories:

- Markings made by some kind of type set in a fixture;
- Those which can write any kind or marking through a programmable control; and
- Those which attach the mark.

Examples of the first category are hot stamping, cold stamping and ink stamping. The second category includes laser marking, ink jet marking and thermal printing. Labels, decals and colour added to a feature on the parts are included in the third category.

### 4.4.6 Packaging Operation

Packaging functions to place the products in the specified container and then deliver to the specified customer. Packaging also is used to place the product to the best container to transfer to another process. Packaging operation can be a manual or an automated operation. There are many types of packaging used such as tape and reel, tube packaging, bagging and many others.

## 4.5 Types of Assembly Method

Assembly method can be classified into three main types, which are manual assembly, semi-automatic assembly and automated assembly.

### 4.5.1 Manual Assembly

Some assembly operations are very difficult to automate, for example, assembly using bolts and nuts. In this case manual assembly is required. As the word "manual" implies (Lat.: manus = hand), the assembly operations are performed by human or people. By simply using his hands, dexterity, sense organs and intelligence, assembly operations are performed by using instruments such as tools, fixtures and gauges. Some instruments may be powered by air or electricity and some are manually

activated. The output capacity in this case will depend on a number of factors such as work point and space arrangement, climate, noise and company morale.

### 4.5.2 Semi-Automatic Assembly

In semi-automatic assembly cell, both human and machinery perform the assembly operations where the machines must be manually fed. In some cases, the machines run continuously while the operators manually loading the components and removing the final product at the pace set by the machine. Other cases require the operators to initiate the cycle of the machine.

### 4.5.3 Automated Assembly

Refer to the next section for a detailed discussion on automated assembly.

## 4.6 Automated Assembly

New technology provides opportunities for the automation of complex assembly operations. Automation can be defined as automatic handling between machines combined with continuous automatic processing at machines. Automation, therefore, requires that several operations be interlocked and coordinated for continuous production without human assistance. There are many interconnected processes that are incorporated in the automated systems, for example, automatic materials handling within and between workstations, automatic inspection, automated assembly, automatic testing and even automated packaging.

Companies that make high volume products involving labour-intensive assembly operation normally use automated assembly since it may reduce the company cost and increase productivity. The increasing use of automation can be explained due to two main reasons, which are the reliability and the flexibility of its output. Mechanization and automation are sometimes being used interchangeably. Mechanization of an

another (Waters, 1996).

Industrial robots are often used in this system, either as multiple workstations or a single robot at one station. Both cases involve complex tasks; therefore, the robots are programmed to deal with multiple product styles so that a mixed model production system is achieved. Due to the inherent flexibility of such system, small batches are economical.

### 4.6.2 Design for Automated Assembly

There are some recommendations and principles that can be applied in product design to facilitate automated assembly (Groover, 1996):

- *Use modularity in product design.* The reliability of the assembly system may increase due to the increasing number of separate tasks that are accomplished by an automated assembly system. To reduce the reliability problem, it is recommended that the design of the product be modular in which each module or sub-assembly has a maximum of 12 or 13 parts to be produced on a single assembly system. The sub-assembly should also be designed around a base part to which other components are added.
- *Reduce the need for multiple components to be handled at once.* The preferred practice for automated assembly system is to separate the assembly operation at different stations instead of simultaneously handling and fastening multiple components at the same workstations.
- *High quality components.* High quality components added at each workstation are essential for high performance of an automated assembly system. Poor quality components might cause jams in feeding and assembly mechanisms resulting in machine downtime.
- *Use of snap fit assembly.* Using snap fit assembly eliminates the need for threaded fasteners. Using snap fit assembly requires the parts to be designed with special positive and negative features in order to facilitate insertion and fastening.

### 4.6.3 Advantages and Disadvantages of Automation

It is worthwhile to consider several aspects associated with the implementation of automated assembly such as practicality of the process for automation, simulation for economic considerations and justification, management involvement and labour relations. However, research have shown that assembly automation offers several potential advantages to a manufacturing plant such as the following (Goetsch, 1991):

- Conformance to specification greatly improves as well as consistency in product quality by reducing or eliminating human errors; hence, results in fewer rejects, a high degree of product reliability and a reduction of liability and warranty cost.
- Reduced manufacturing costs resulting from decreased both direct and indirect labour requirements and increased productivity.
- Improved safety and better working conditions by removing operators from hazardous operations. Some operations can be injurious to health and automation may then be the only solution.
- Efficient production scheduling and reduced inventory requirement due to the ability of automated assembly systems to respond immediately to production demands.
- Reduced floor space requirements.

However, in order to implement assembly automation, more skilled workforce usually is required.

### 4.6.4 Types of Automated Assembly Systems

The assembly system is defined as a manufacturing system in which some stations perform assembly operations, by which two or more parts or subassemblies are bought together to form a single unit (Jeong and Kim, 2000). There are several types of automated assembly system. The ones that will be discussed here are a robotic assembly system, a flexible assembly system and a re-configurable assembly system.

### 4.6.4.1 Robotic Assembly Systems

In a robotic assembly system, industrial robots are used to perform a variety of tasks. A robotic assembly line may consist of only one robot performing the assembly operation or a number of robotic stations linked by a conveyor. An effective robotic assembly system requires consideration of the delivery of components to the workstations, component feeding and orienting, robot end effectors[8], sensing requirements and system controls (Goetsch, 1991). Several robot characteristics need to be considered for assembly applications including the following (Goetsch, 1991):

- High accuracy[9] and repeatability[10] in point-to-point and path conformance;
- Reliability, flexibility and dexterity;
- Capacity for a large number of inputs and outputs;
- Sensory communications and system communication capability;
- Off-line programmability with adaptability to high level language; and
- Memory capacity for program storage.

[8] An end-effector is a gripper located at the end of the robot arm.

[9] Accuracy in robotic assembly systems is the precision with which the robot moves from its home position to a designated coordinate location and it is usually expressed as the difference in the distance between the point where the robot manipulator actually is and the point indicated in the program. Several factors affect the robot accuracy such as speed, payload, the approach direction, the inherent mechanical inaccuracy in the robot and many other factors. A claimed accuracy of +/- 0.004 inches or 0.10 mm is common and +/- 0.002 inches or 0.05 mm is becoming available (Goetsch, 1991).

[10] Repeatability is a measure of the robot precision when it returns to the predefined, programmed point, cycle after cycle and it is often expressed as a distance differential when the robot manipulator returns to the taught point within a certain range of speeds and payloads. A repeatability of +/- 0.002 inches or 0.05 mm is common and +/- 0.001 inches or 0.03 mm is becoming available (Goetsch, 1991).

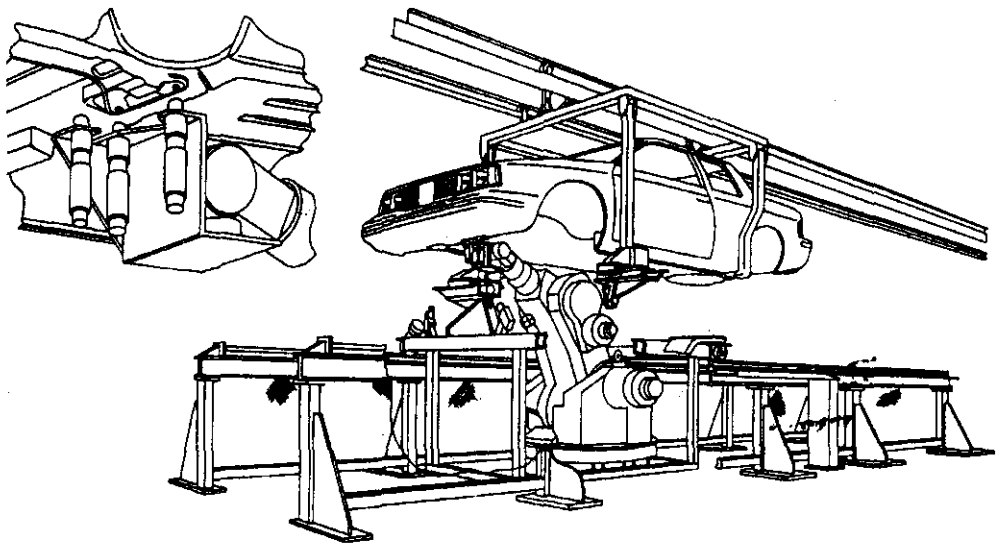


Figure 4.1: An example of robotic assembly line (Goetsch, 1991)

### 4.6.4.2 Flexible Assembly Systems

A flexible assembly system implies an assembly device in which different product variants of particular product can be assembled in an arbitrary manner (Lotter, 1989). A flexible assembly system is a computer controlled automated assembly system for converting raw material and purchased components into products of a known and desired functional quality (Owen, 1984). Theoretically, the system is very flexible due to its ability to adapt to changes in product design and its capability of assembling more than one product model without tooling changeover. The system flexibility allows any product within the functional, power and geometric constraint of the assembly system to be processed. The flexibility of an assembly system can be considered both internally and externally (Tzafestas and Stamou, 1997).

- *Internal flexibility* is the ability of the system to deal with environmental variations and compensate for uncertainties introduced by the sensors, etc.
- *External flexibility* is the ability of the system to adapt to variations of the product to be assembled or its consisting part.

A flexible assembly cell consists of two main components, which are hardware and

controlling software. The hardware includes assembly robots, storage and material handling systems. The controlling software can comprise several distinct components such as software for assembly task planning, software for sensory data processing and interpreting, software for object recognition and location, control software for motion planning and control, software for communication and software for error recovery.

#### 4.6.4.3 Re-configurable Assembly Systems

The optimum assembly system for today's market turbulence is a re-configurable and reusable, modular system. The concept of re-configurability in assembly cell has gained considerable attention in the manufacturing engineering research. The development of re-configurable assembly system is due to the need of assembly machines and modules that can be reconfigured and reused to support a wide range of assembly processes and products over a sufficiently long period of time. A re-configurable and reusable assembly system is enabling manufacturers to adapt rapidly to changing market conditions by updating products or bringing in next-generation models without investing in new equipment. The re-configurable components of the assembly system make it easier and more cost effective for companies to make small production batches. Companies are also able to protect their long-term investments. Examples of products that are using this system are computers, mobile phones and industrial and consumer electronics.

A re-configurable assembly system should have the following characteristics:

- *Physical flexibility*. Physical flexibility is achieved by changing the number of assembly stations or by modifying their features (Heilala and Voho, 1997). The system physical flexibility enables the assembly stations to be added with a new system or change the existing system after it has been designed, built and debugged off-line.
- *Logical flexibility*. Logical flexibility comes from the use of the intelligence pallets (Heilala and Voho, 1997). As a result, the material flow is very flexible and multiple tasks can be carried out at one station (Heilala and Voho, 1997).
- *Capability flexibility*: The capability flexibility refers to the system's ability to

react to changing market demands in term of the product variants that is asked for (Heilala and Voho, 2001). It can further be divided into (Heilala and Voho, 2001):

- → The system's flexibility in assembling products that belong to one product family;
- → The system's flexibility in assembling products that belong to a number of product families; and
- → The system's flexibility in assembling products that do not belong to the product family or families the assembly system was developed for.

- *Capacity flexibility*: The capacity flexibility refers to the system's ability to react to changing market demands in terms of the quantities asked for.
- *Error recovery flexibility*: The error recovery flexibility refers to the system's ability to react to internal disturbances occurred during assembly and endangered the operation of the system.
- *Modularity*: The system contains many modules that support a wide range of assembly processes and products. The system can benefit from the design-create-file-update sequence.

The difference between a flexible assembly system and a re-configurable assembly system is mainly on the flexibility requirements in different time frames. A flexible assembly system reacts according to the product life-cycle phases. However, a re-configurable assembly system is required to react faster in a very short period of time. Table 4.1 below summarizes the difference between a flexible assembly system and a re-configurable assembly system.

| | Flexible Assembly System | Re-configurable Assembly System |
|---|---|---|
| Time to react | Product life-cycle phases | Very short |
| Flexibility | Static or physical flexibility | Dynamic or logical flexibility |
| Why | Production volume changes | Mass customisation, lot size one, assembly-to-order |
| | New variants in the same system | Disturbances, machine breaks |
| | New products in the same system | Repair work, rush order |
| | Demand fluctuation | Demand fluctuations |
| How | Layout modifications | Control tasks and resources |
| | Size and degree of automation | Use of information technology |
| | Re-configurability, re-utilization | Change of control programs, routines |
| | Modularity, expandability | Sorting and routing |
| | Scalable | Robotics, flexible automation |
| | Exchange of system module/sub-module | Human intelligence and skills |

Table 4.1: Flexibility requirements in different time frames (Heilala and Voho, 2001)

### 4.6.5 Knowledge-Based Approach in Designing Re-configurable Automated Assembly Systems

The development of re-configurable assembly systems can take advantage of the knowledge-based systems where a knowledge-based approach is applied. The advantages of the knowledge-based approach is that it is modular, re-configurable, easy to implement, maintain and upgrade, and fault-tolerant. In order to achieve high performance and intelligent behaviour of the knowledge-based software system in re-configurable assembly cells, a large amount of domain specific knowledge is necessary.

# Chapter 5

# CASE STUDY - CONNECTOR ASSEMBLY OPERATION

## 5.1 Introduction

One of the areas where automated assembly has been widely applied is electronics assembly. Most electronic assemblies involve placing components onto or inserting components into printed circuit boards (PCBs). The components include resistors, capacitors, diodes, inductors, transistors, transformers, switches, connectors and many others.

In the past, the printed circuit board assemblies were performed manually. However, with the advance of technology, they have moved to fully automatic insertion machines and robotic systems where all types of components are now being inserted sequentially through a continuous array of in-line machines or groups of machines bridged by material handling equipment. The printed circuit board assembly system is also equipped with automated test equipment and automated inspection and both controlled by computers. Figure 5.1 shows how the connectors are being placed onto the printed circuit board by a robot.

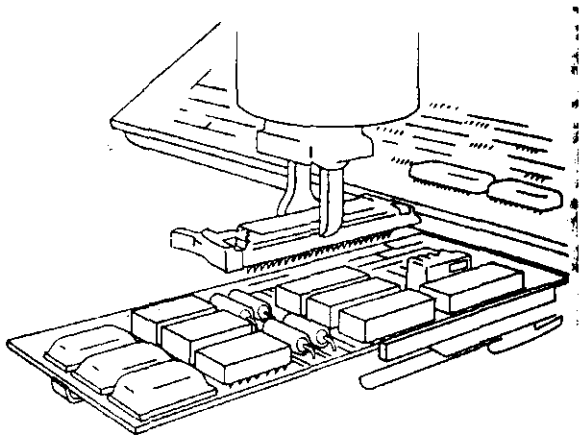


Figure 5.1: Connectors placed by robots (Goetsch, 1991)

This chapter will discuss the connector assembly process operation in AMP Incorporated, the world's leading supplier of electrical and electronic connectors and interconnection systems including cutting-edge technologies in fibre optics, wireless and sensors. This case study will be used to develop use cases for the software development in the next chapter, Chapter 6.

## 5.2 Company Background

Established in 1941, AMP Incorporated is one of the well-known, established suppliers in the connector and interconnection systems industry. AMP is recognized as the premiere brand of electrical and electronic connectors and interconnection systems. AMP is also recognized for its innovative products of highest quality. AMP products include electrical and electronic connectors and interconnection systems, IC sockets, fibre optic components, relays and modules, circuit protection devices, wire and cable, switches, wireless components, sensors, printed circuit boards, touch screens and application tooling. AMP also provides products to insulate, protect, hold, bundle and identify high-performance electrical harnesses.

## 5.3 What is a Connector?

When a specified number of contacts are inserted into a housing in order to make it functional, the assembly is usually referred to as a connector. A contact is an electrically conductive item designed for use in a multi-circuit connector and for convenience in making multiple electrical connections whereas a housing, sometimes referred to as "block", is an insulating encapsulation for contacts. However, a connector assembly includes more parts than just a housing and contacts (refer to Figure 5.2). It usually consists of a housing (with contacts), or a shell (with modules or inserts and contacts), and the necessary hardware to hold the assembly together and make the assembly a functional connector.

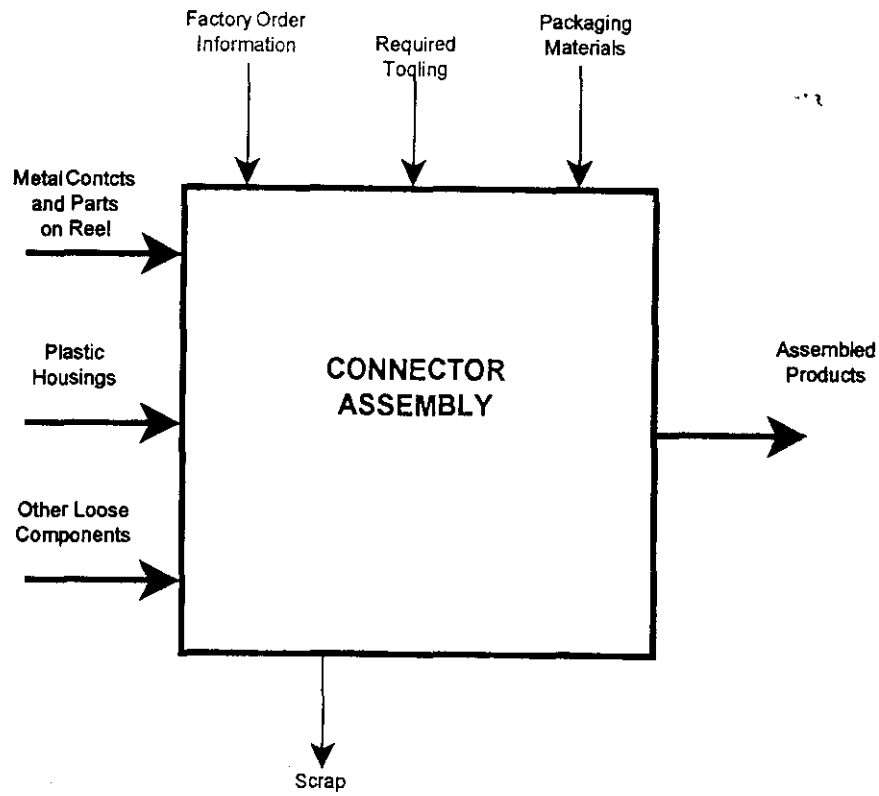



Figure 5.2: Connector Assembly (AMP Inc., 1997)

A connector is used to connect conductors of one circuit with those conductors of another circuit. A connector is also used to provide rapid connect or disconnect mating with a printed circuit board or another connector. Figure 5.3 shows an example of a connector.

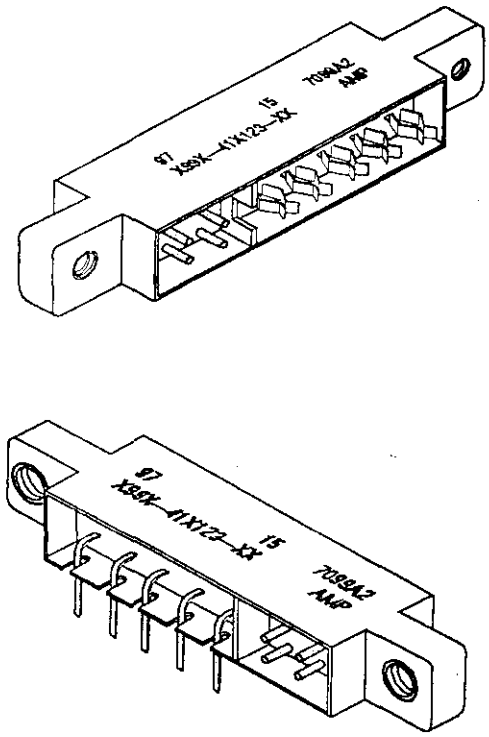


Figure 5.3: An example of a connector (AMP Inc., 1997)

## 5.4 Connector Assembly Process Flow

Figure 5.4 shows the AMP connector assembly process flow, which can be categorized into eight main functions, namely,

- ***Contact Feeding***. Contact feeding supplies contacts at the rate required by inserting.
- ***Housing Feeding***. Housing feeding supplies housings in proper orientation at the required rate by inserting.
- ***Forming***. Forming changes the contacts from the way they are supplied to the form they are needed for the product. The forming operation for contacts are usually cutting, shaping and bending whereas for housings, drilling, tapping and cutting are the examples of the forming operation.
- ***Inserting***. The function of inserting is to take contacts and housings and combine them into a product.

- ***Inspecting***. Inspecting is the operation where the quality of some product attributes is checked. Parts that are failed to meet the requirement specification or the quality standard are being rejected for either rework or scrap.
- ***Marking***. Marking functions to identify the products. The marking goal is to put the desired marking on each part in the desired location. The marking could be part number, lot number, manufacturing codes, line identification, date of manufacture and other important information.
- ***Packaging***. Packaging places the product in the specified containers or packages for delivery to the customer or for transfer to another process. Examples of packages are boxes, bags, trays, tubes, taped strip and bins.
- ***Controlling***. The main goal of controlling is to automatically manage some machine actions to reduce the amount of operator attention required.

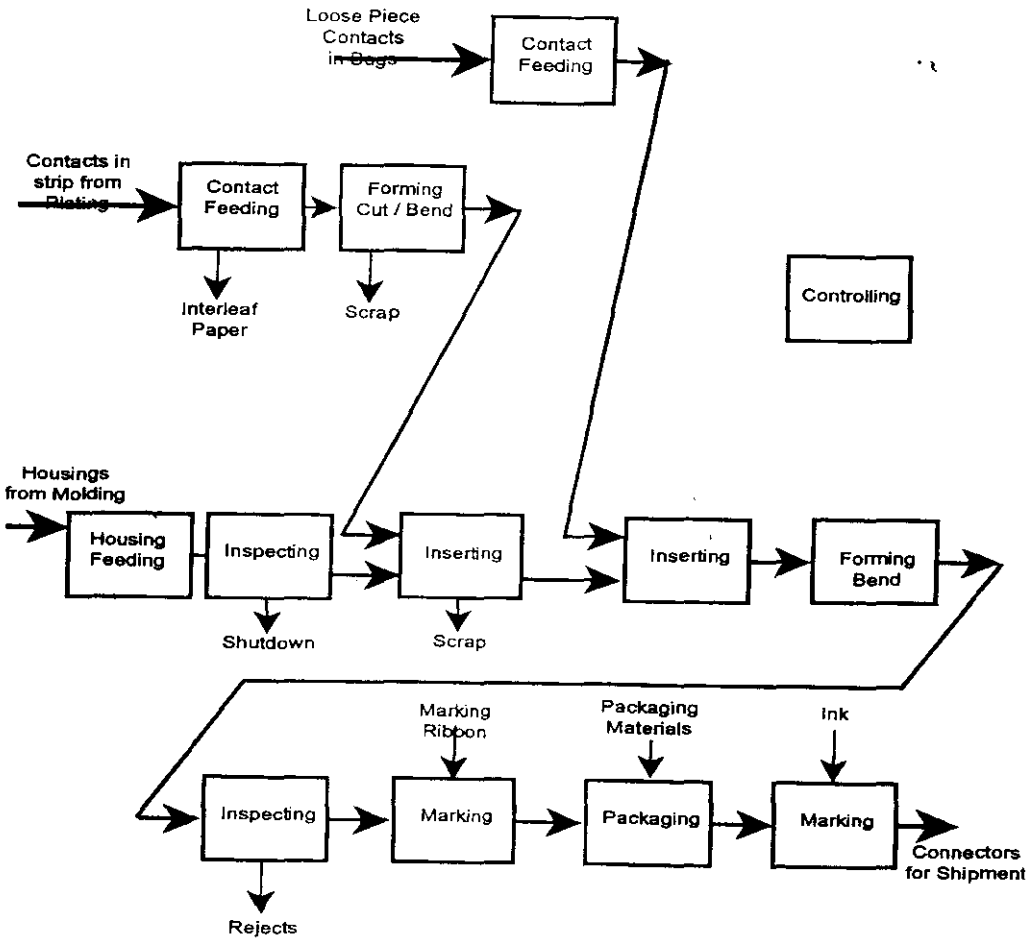



Figure 5.4: Connector Assembly Process Flow (AMP Inc., 1997)

## 5.5 How does the Assembly Process work?

To simplify the AMP connector assembly process, it is assumed that only a specified number of contacts in strip and loose pieces contacts in bags are being inserted into a housing in order to make a functional connector. The connector assembly process will be discussed sequentially according to the processes.

### 5.5.1 Contacts in Strip

***Contact Feeding from reel***: The connector assembly process begins with the contact feeding operation from a reel. The function is to feed a continuous strip of contacts using a dereeler (refer to Figure 5.5).

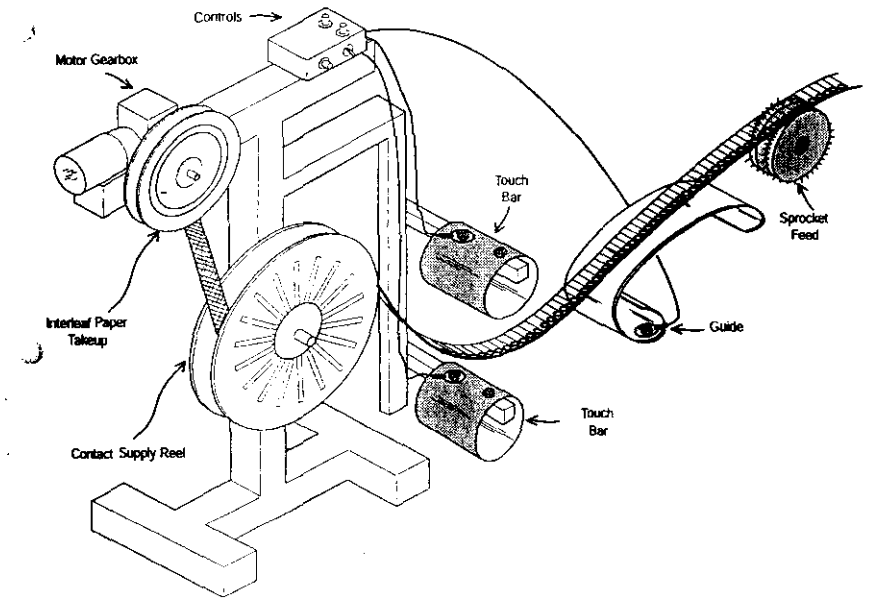



Figure 5.5: Contact Feeding – Dereeling (AMP Inc., 1997)

***Forming (Cutting and Stamping)***: Then, the strip goes through the forming operation. There are two main functions in the forming operation, namely, cutting and stamping. In the cutting operation, one side of the contacts carrier is being removed, cut away as waste and segmented the waste carrier into short lengths for ease of handling (refer to Figure 5.6). Then, the contact goes through the stamping operation

in order to give the contact its final shape (refer to Figure 5.7). The tin-plated contacts are then spot plated with gold after the stamping operation.

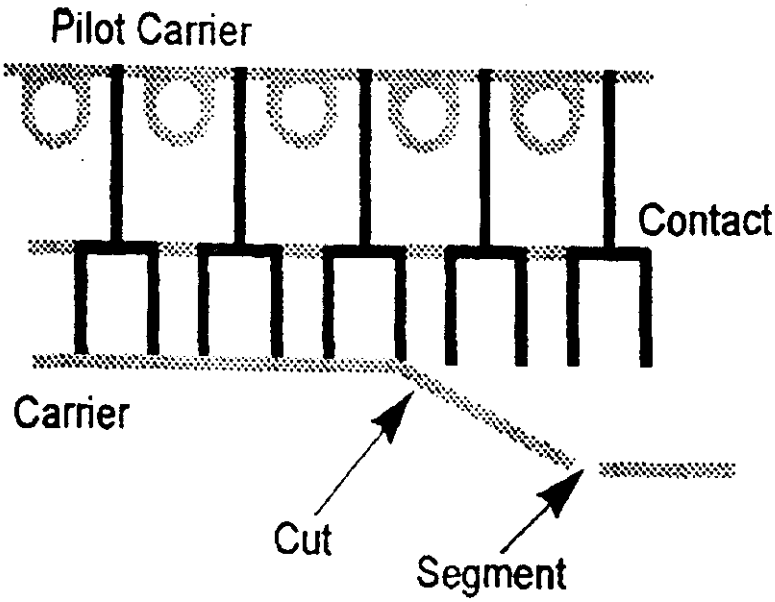



Figure 5.6: Contact Forming – Cutting (AMP Inc., 1997)

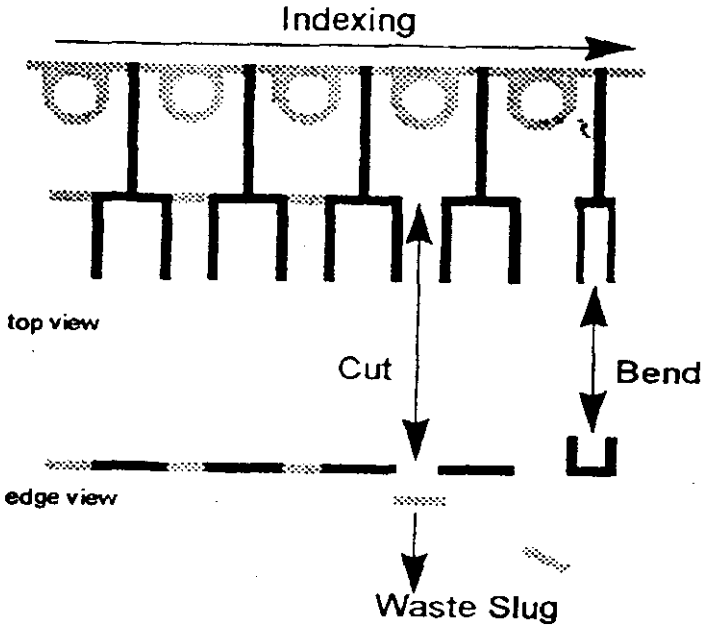



Figure 5.7: Contact Forming – Stamping (AMP Inc., 1997)

## 5.5.2 Housings from Mouldings

***Housing Feeding***: At the same time, the moulded housings are fed from a magazine. Prior to this operation, the housings have gone through a separate semi-automated assembly process, which added two threaded inserts in each housing. Then, the housings are oriented properly and packed into tubes to transfer over to the assembly process (refer to Figure 5.8).

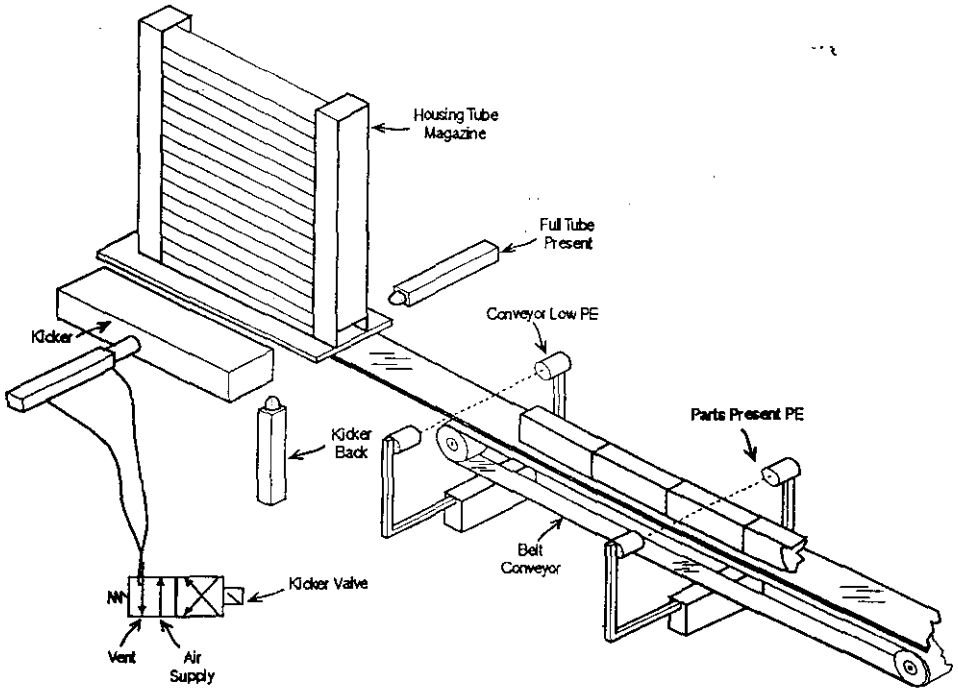



Figure 5.8: Housing Feeding (AMP Inc., 1997)

***Inspecting***: Due to the possibility of misoriented housings in the housing feeding operation, the housings are inspected for correct orientation using a vision system. The machine is stopped if misorientation of housings is found to prevent tool damage (refer to Figure 5.9).

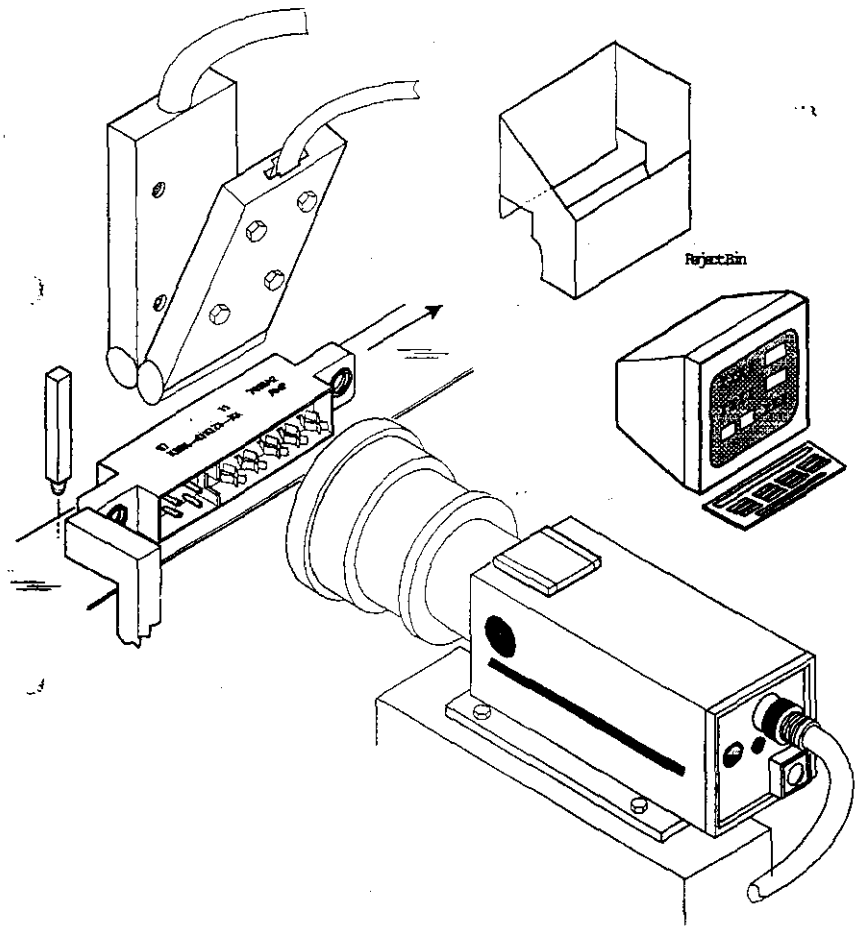



Figure 5.9: Housing Inspecting (AMP Inc., 1997)

### 5.5.3 Loose Pieces Contacts in Bags

***Contact Feeding from bulk***: The function is to feed loose contacts that are supplied by vendors in bags of 2500 using a vibrating bowl feeder.

### 5.5.4 Assembly of Contacts in Strip and Housings

***Mass Inserting***: The function of mass inserting is to place a group of 5 formed, strip fed contacts are put into the housing (refer to Figure 5.10).

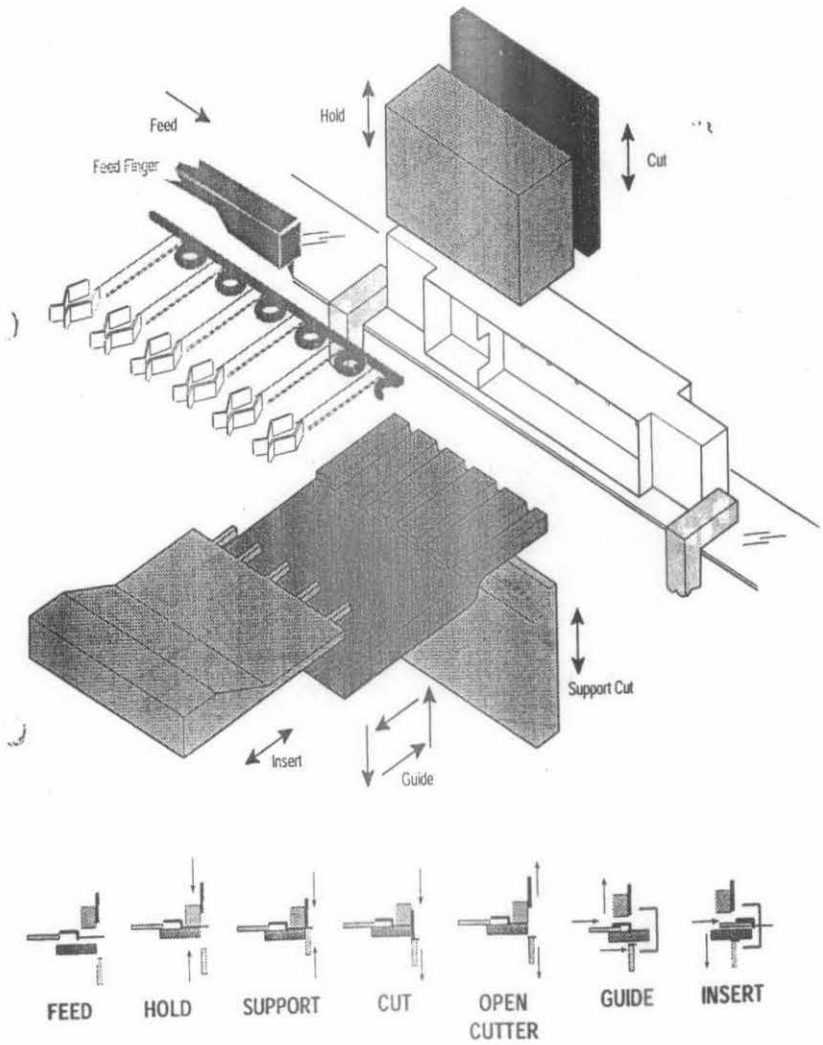



Figure 5.10: Mass Inserting (AMP Inc., 1997)

## 5.5.5 Assembly of Loose Contacts to the Assembled Housings

***Stitching***: Four loose contacts, received from the contact feeding vibrating bowl, are then inserted into the same housing (refer to Figure 5.11). Stitching is also called a single contact inserting operation.

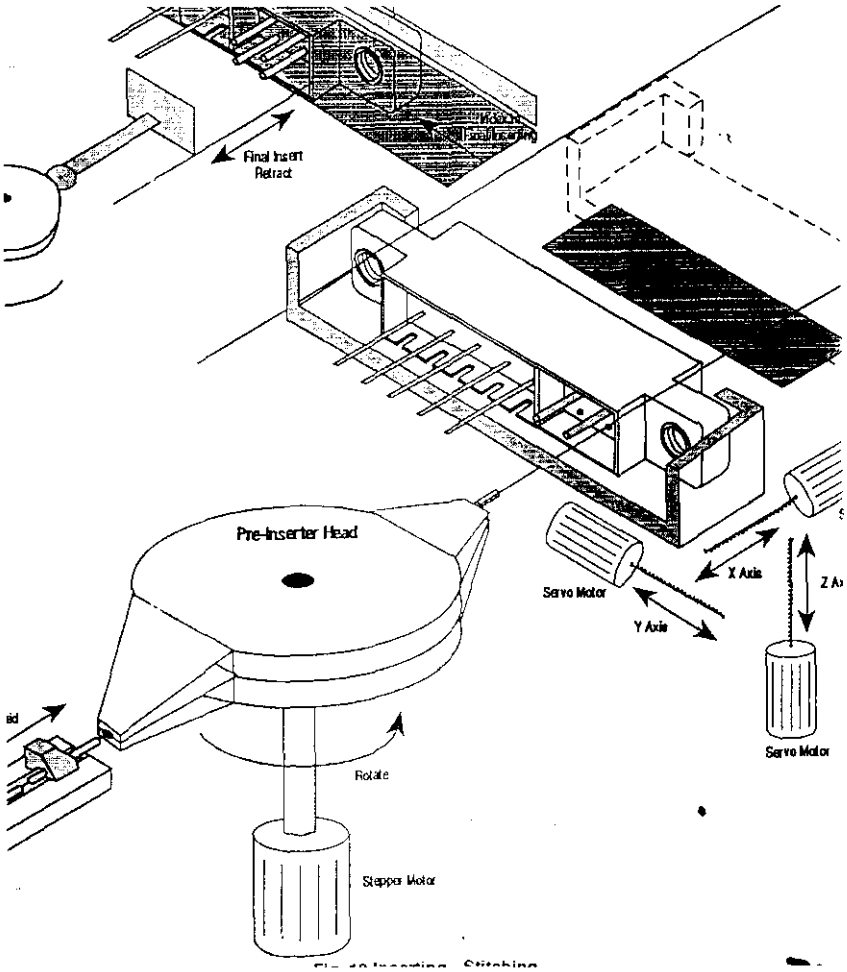



Figure 5.11: Stitching (AMP Inc., 1997)

***Forming (Bending)***: The straight contacts placed into the housing are then bent to form legs to fit into a printed circuit board (refer to Figure 5.12).

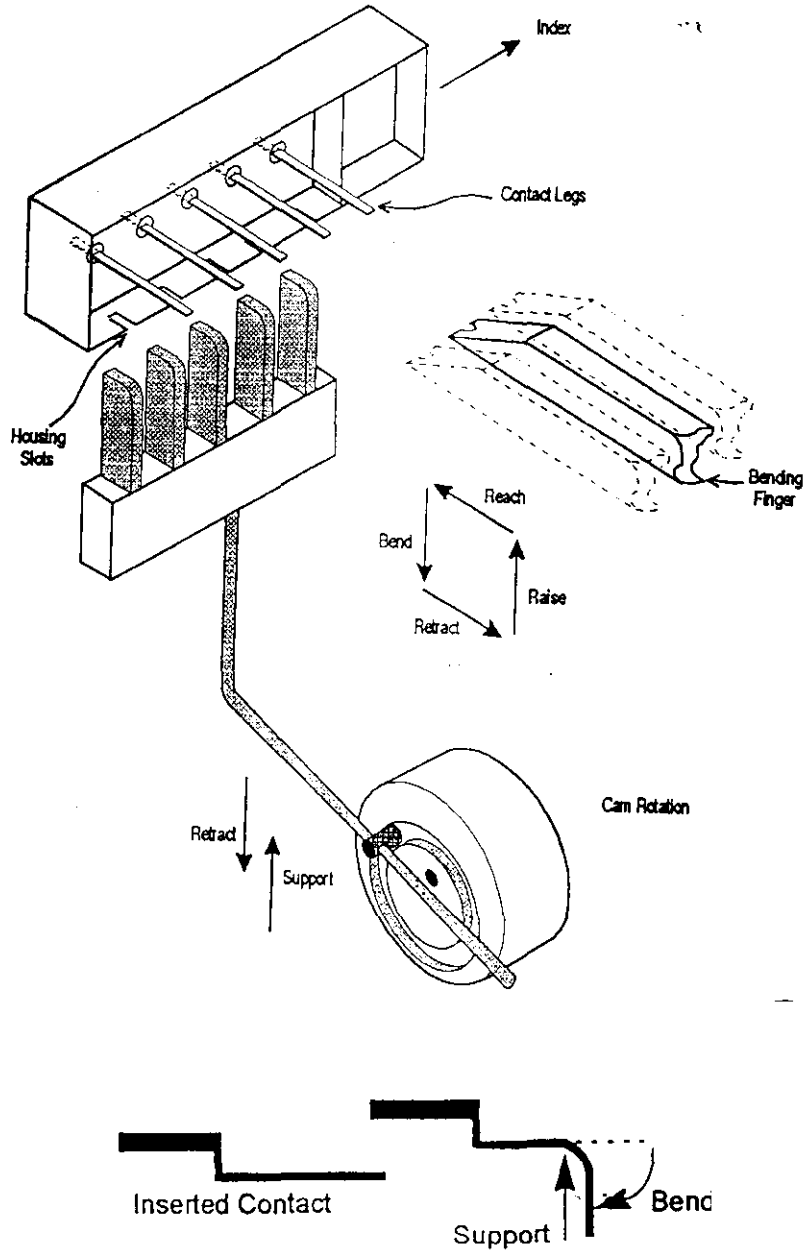



Figure 5.12: Forming – Bending (AMP Inc., 1997)

***Inspecting***: The assembled housings are then inspected again for the correct number of pins count (refer to Figure 5.13) and the correct depth of pin insertion (refer to Figure 5.14). In this operation, any non-conformance part is rejected.

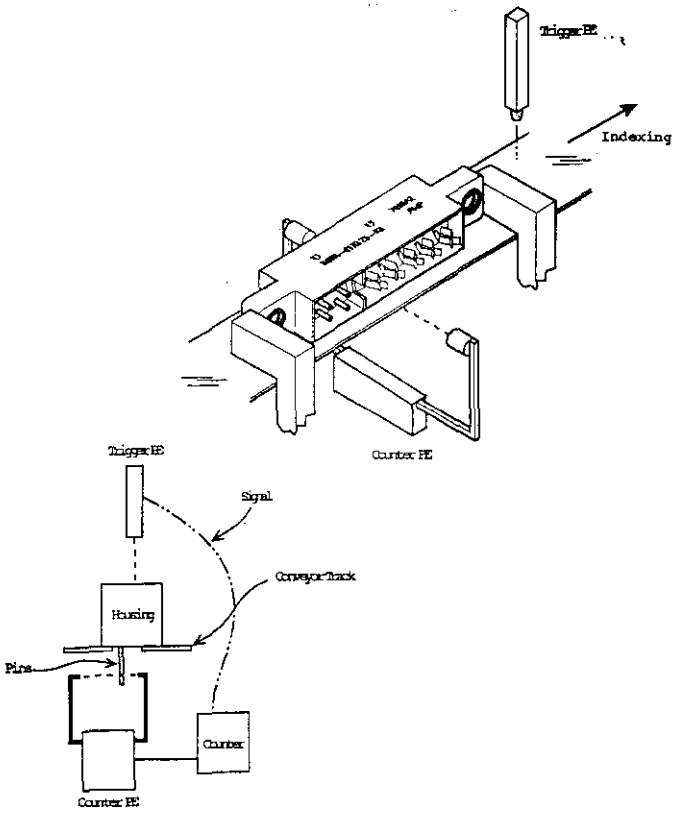



Figure 5.13: Inspecting – Pin Count (AMP Inc., 1997)

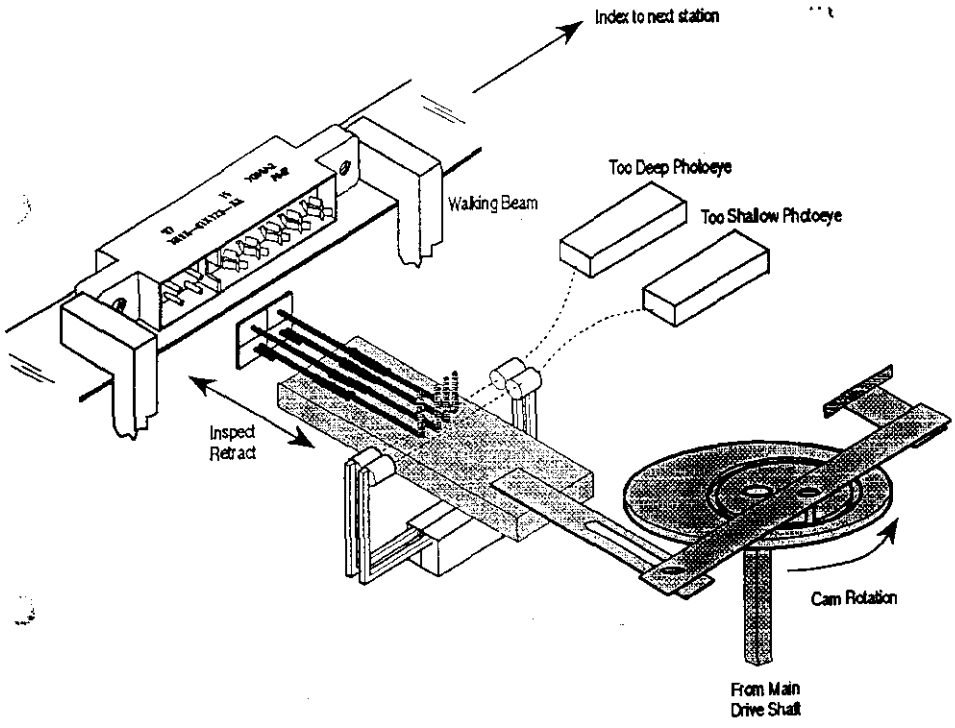



Figure 5.14: Inspecting Insertion Depth (AMP Inc., 1997)

***Marking (Hot-Stamping)***: There are two marking station for this operation. At the first station, the part number is stamped in colour on the housing in to order to meet customer requirements. At the second station, the date, line and shift are marked using a laser.

***Packaging***: Then, the connectors are packed individually into trays for shipment to the customer.

***Marking***: The order identification is marked in ink on the packages.

***Controlling***: The main goal of controlling is to automatically manage some machine actions to reduce the amount of operator attention required. The controlling function serves four different purposes:

1. Connecting and timing the assembly functions into an integrated process, which is called indexing.
2. Interface between the operator and the machine control systems, for example, the touch screen.
3. Personnel safety such as light curtain, proximity detector, emergency stop, guards, doors and door switches.
4. Machine protection such as clutches.

## 5.6 Evidence of Good Connector Assembly Operation

Some typical attributes that are being looked into to determine the quality of the connectors are:

- *Correct pin count for the part*. However, it does not necessarily mean that all the circuits in the housing are filled.
- *Correct insertion depth*. The contacts are being inserted to the correct depth.
- *Correct position*. The contacts are in the correct position. This is important to ensure the contacts will fit well into the circuit boards or mating components.
- *Correct marking*. The markings must be correct and legible.

# Chapter 6

# APPLICATION OF USE CASES ON THE CASE STUDY

## 6.1 Introduction

Use cases are the main reference base for the development of the software system where they are used in the elicitation, analysis and documentation of requirements. Use cases are important in developing the domain knowledge of the system. A larger amount of domain knowledge is necessary to achieve high performance and intelligent behaviour of the knowledge-based software system for the re-configurable assembly cell.

This chapter demonstrates the application of use case development using the case study of the AMP connector assembly process, which was discussed in Chapter 5.

## 6.2 Development of Use Cases

The AMP connector assembly operation is an automated assembly system. To develop an assembly system that is re-configurable, it is important to identify and capture the operational requirements of the system and transform them into functional specifications. In order to do so, the use case technique is being employed. The main goals of developing use cases for the case study of AMP connector assembly operation are:

- To define the scope of the software system being developed.
- To identifying essential functionality and information as a guide in designing the system.

- To verify the correctness, completeness, consistency and functionality of each stage of the system development process.

Use cases in the connector assembly operation can be classified as *ordered use cases* since the completion of one use case is required before the following use case can be executed. In other words, the first use case must complete first before the second use case can be executed. In the ordered use case, the "precede-relation" association is usually employed.

Figure 6.1 shows the proposed use case structure. The use case diagram describes a collection of related use cases in the connector assembly operation where each use case describes the system function. In this case, scenarios can be viewed from the *process perspective* where operator actions and machine actions are being focused. Scenarios are being developed using a successful accomplishment of in assembling the connector. Machine actions will typically consist of the equipments that acts as actors. Actors in this context are also the operators. Table 6.1 shows the association between the principal actors and the scenarios, which are identified in this project.

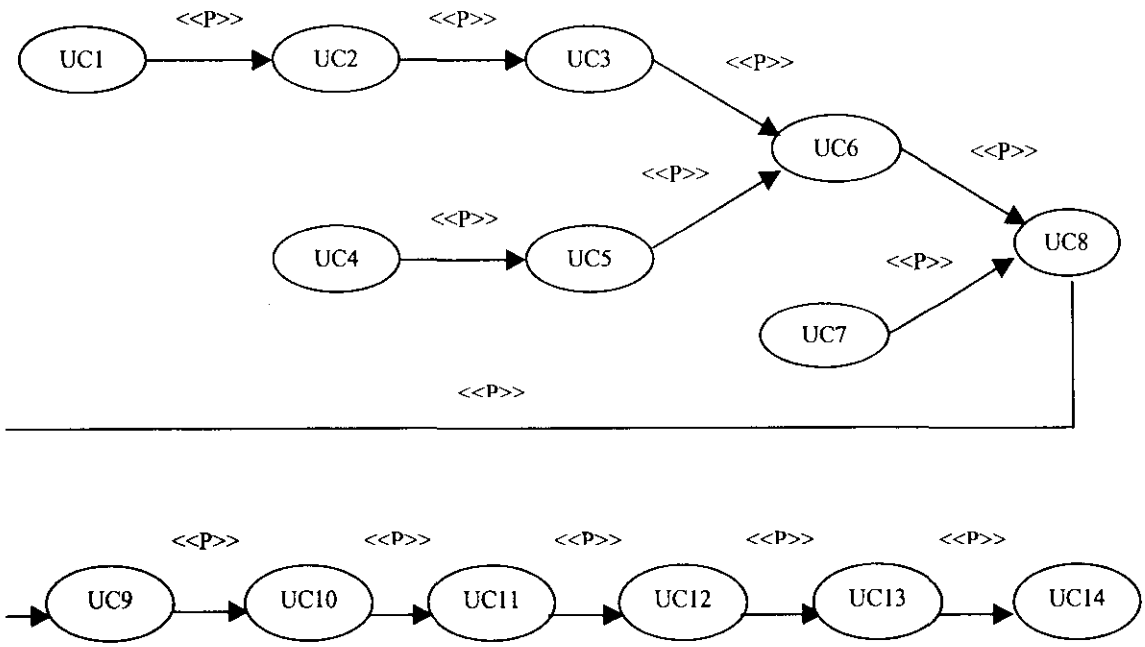



Figure 6.1: Use Case Structure Diagram for the Connector Assembly Operation

**Contacts in Strip**

| Use Case # | Scenario | Actor | Description |
|---|---|---|---|
| UC1 | Feeding Operation for contacts in strip | Operator | Feed a continuous strip of contacts to be inserted to the housings at the rate required by mass inserting operation |
| | | Vertical dereeler | |
| | | Sprocket feeder | |
| | | Touch bars | |
| | | Guide | |
| UC2 | Contact Forming – Cutting Operation | Three Rotating Wheels | Take contacts strip from contact feeding and remove the carrier from one side before feeding to the stamping operation |
| | | Rotating Cutter | |
| UC3 | Contact Forming – Stamping Operation | Compacut Stamping Machine | Take contacts in strip from cutting and give their finals shape. |

**Housings from Moulding**

| Use Case # | Scenario | Actor | Description |
|---|---|---|---|
| UC4 | Housing Feeding Operation | Magazine Housing Feeder | Provide housings in the proper orientation for contacts to be inserted |
| | | Accumulator Conveyor Low Photoeye | |
| | | Parts Present Photoeye | |
| | | Accumulating Conveyor | |
| UC5 | Inspecting Operation for housings | Vision Inspection System | Inspect housings before inserting in order to detect a misoriented or non-conforming housing |

**Assembly of Contacts in Strip and Housing**

| Use Case # | Scenario | Actor | Description |
|---|---|---|---|
| UC6 | Mass Insertion Operation | Walking Beam | Insert a group of 5 formed, strip fed contacts into housing |
| | | Feed Finger | |
| | | Hold and Guide Blocks | |
| | | Cutter and Support Blocks | |
| | | Pusher and Guide Blocks | |

**Loose Piece Contacts in Bags**

| Use Case # | Scenario | Actor | Description |
|---|---|---|---|
| UC7 | Feeding Operation for Loose Piece Contact | Operator | Feed loose piece contacts in bags of 2500 to be inserted to the housings at the rate required by stitching operation |
| | | Vibrating Bowl Feeder | |
| | | Track and Gates | |
| | | Conveyor | |

**Assembly of Loose Contacts to the Assembled Housings**

| Use Case # | Scenario | Actor | Description |
|---|---|---|---|
| UC8 | Stitching | Housing Carrier | Insert four straight, loose piece contacts into the connector |
| | | Feed Block and Vibrating Conveyor | |
| | | Gripper Fingers on Pre-Inserted Head. | |
| | | Final Inserter | |
| | | Walking Beam | |
| UC9 | Contact Forming – Bending Operation | Bending Machine | Bend contact legs down at 90 degree so the connector can be mounted to a printed circuit board |
| UC10 | Inspecting Operation – Pin Count | Photoeye Detector | Inspect correct number of contacts being inserted into the housing and segregate the good ones from the non-conforming ones |
| UC11 | Inspecting Operation – | Mechanical Contact | Inspect depth of the contacts being inserted into the housing |

| | | | |
|---|---|---|---|
| | Insertion Depth | | and segregate the good ones from the non-conforming ones |
| UC12 | Marking Operation – Hot Stamping | Hot Stamping Machine. | Mark product identification in colour at the desired location on the housing |
| UC13 | Packaging | Pick and Place Robot | Place connectors in specified containers for delivery to the customer |
| UC14 | Ink Jet Marking | Ink Jet Marking Machine | Mark required identification at the desired location on the packages |
| UC15 | Controlling | Computer System | Manage machine |

Table 6.1: Association between the principal actors and the scenarios

There are two types of use case templates, which are a template in plain text and a template in table form (refer to Appendix A for detail explanation and examples). In developing the AMP connector use cases, the template in plain text will be used. The following section will describe in details each use case.

### 6.2.1 Contacts in Strip

**Use Case: 1 Feeding Operation for Contacts in Strip**

---

CHARACTERISTIC INFORMATION

Goal in Context: To feed a continuous strip of contacts to be inserted to the housings at the rate required by the mass inserting operation.

Scope: System

Level: Primary Task

Preconditions: The contacts are in the strip form, not in other kind of forms.

Success End Condition:

- The contacts are fed at the rate required by mass inserting operation.
- The contacts are fed at the interval required by mass inserting operation. The interval here is set by the pitch of the progressive stamping.
- Contacts are in good conditions such as the correct contact size and length and free from any bent and contamination.

Failed End Condition:

- Bad quality of parts are being supplied.
- Incorrect feeding rate is being set.

Primary Actor:

- *Operator:* manually puts the contact strip onto the supply reel.
- *Vertical dereeler:* consists of:
    - *Contact supply reel:* is used to supply the contact strip.
    - *Interleaf paper winding:* is used to move the contact supply reel.
    - *Brake:* is used for preventing the supply reel from turning too far.
    - *Dereeler motor:* is used to run the dereeler.
    - *Control panel:* has a power switch, alarm light and reset button.
- *Sprocket feeder:* is used for pulling and indexing the contact strip into the assembly process at the rate required by the inserting operation.
- *Touch bars:* are used to control the dereeler motor. Two touch bars are located above and below the contact strip between the supply reel and the sprocket feeder.
- *Guide:* An electrical current is passed through the contact strip from the guide at the sprocket feeder.

Trigger: Feeding operation starts based on the production schedule.

---

MAIN SUCCESS SCENARIO

1. A continuous strip of contacts is manually put onto the contact supply reel.
2. The contact strip is in contact with the top touch bar.
3. The contact strip is in contact with the guide.

4. The dereeler motor runs when the contact strip is in contact with both the top touch bar and the guide since the electrical circuit is completed.
5. The dereeler continues running until the time runs out.
6. The contact strip is indexed and pulled into the assembly process by the sprocket at the rate required by the inserting operation.

---

EXTENSIONS

2a. The contact strip is not in contact with the top touch bar.

2a1. The contact strip is in contact with the bottom touch bar.

2a11. The dereeler motor will not run.

2a12. An alarm will sound to alert the operator.

2a121. The end of the reel.

2a122. Problems with the dereeler.

3a. The contact strip is not in contact with the guide.

3a1. The dereeler motor will not run.

4a. The dereeler motor stops.

4a1. The contact strip is not in contact with the top touch bar.

4a2. The contact strip is not in contact with the guide.

5a. The dereeler stops running.

5a1. The timer reaches the set time and the top touch bar is not contacted.

5a2. The stop button is pressed.

5a3. The bottom touch bar is contacted.

---

SUB-VARIATIONS

1. Use
   - Horizontal dereeling[11].
   - Larger reels[12]. may be used without the need for interleaf paper winding that can cause nesting of the contacts.

[11] Horizontal dereeling allow for the reels to be supplied in a stack on the pallets in order to reduce handling.

[12] Larger reels may be used without the need for interleaf paper winding that can cause nesting of the contacts.

## Use Case: 2 Contact Forming – Cutting Operation

---

CHARACTERISTIC INFORMATION

Goal in Context: To take the contacts strip from contact feeding and remove the carrier from one side before feeding to the stamping operation.

Scope: System

Level: Primary Task

Preconditions:

The strip should go through a splice detector to detect welded splices, which may present in the strip and may damage the tooling in the forming operation. The strip passes through the cutting machine should be free from welded splices.

Success End Condition:

The carrier strip from one side of the contacts is removed, cut and segmented for ease of handling.

Failed End Condition:

The carrier strip from one side of the contacts is not removed, not cut or not segmented.

Primary Actor:

- *Three Rotating Wheels*: are used in cutting where two wheels to support the strip and one wheel to shear the carrier strip from the contact strip.
- *A Rotating Cutter*: is used in segmenting which breaks the waste strip into short length for ease of handling.

Trigger: Feeding Operation for Contacts in Strip (Use Case 1)

---

MAIN SUCCESS SCENARIO

1. The machine takes the contact strip from contact feeding.
2. The carrier strip from one side of the contacts is being removed.
3. The contacts are feed to the stamping operation.
4. The waste carrier strip is cut.
5. The waste carrier strip is segmented into short length for ease of handling.

## Use Case: 3 Contact Forming – Stamping Operation

---

CHARACTERISTIC INFORMATION

Goal in Context: To take the contacts in strip from cutting and gives their finals shape.

Scope: System

Level: Primary Task

Preconditions:

- The contacts complete the contact forming cutting operation.
- The strip is indexed through a compacut machine one contact per stamping. The contacts must feed through the stamping operation at the rate that the contacts are being inserted. Due to the reason that the stamping operation is only forming one contact per stroke and the inserting operation is inserting 5 contacts, the compacut must run at a rate 5 times the main line index rate.

Success End Condition:

- The contact arms are bent up at 90 degrees.
- Small pieces of metal called slug are removed.

Failed End Condition:

- The contact arms are not bent up at 90 degrees.
- Small pieces of metal called slug are not removed

Primary Actor:

Compacut Stamping Machine

Trigger: Stamping operation begins once the cutting operation finishes.

---

MAIN SUCCESS SCENARIO

1. The compacut machine takes the contact strip from cutting operation.
2. A series of rollers bend the strip opposite the curl introduced by coiling on the reel.
3. Straightening takes the coil set out of the contacts so that they will move properly through the stamping operation.
4. Indexing advances the strip and positions it properly for forming.

5. Cutting removes small pieces of metal called slugs. All metal removed from the contact strip is collected for recycling.
6. The two arms of the contact are shaped and bent up at 90 degrees in 30 degrees increment.

## 6.2.2 Housings from Moulding

### Use Case: 4 Housing Feeding Operation

---

CHARACTERISTIC INFORMATION

Goal in Context: To provide the housings in the proper orientation for the contacts to be inserted.

Scope: System

Level: Primary Task

Preconditions: All the machine specifications must be input prior to the operation.

Success End Condition:

- The housings are fed in the correct orientation.

Failed End Condition:

- The orientation of the housings is incorrect.

Primary Actor:

- Electro-Pneumatic Magazine Housing Feeder
- Accumulator Conveyor Low Photoeye
- Parts Present Photoeye
- Accumulating Conveyor

Trigger: Trigger by the production schedule.

---

MAIN SUCCESS SCENARIO

1. The operator manually loads the housings packed in tubes into a magazine that holds up to 10 tubes.
2. The housings slide out of a tube where the tube is positioned in line with the discharge track by gravity.
3. The empty tube is discharged.
4. A full tube drops into the feed position by gravity.
5. The accumulating conveyor is running.
6. The accumulator conveyor low photoeye detects the housing.
7. The parts present photoeye detects the housing and determine whether the accumulating conveyor is full or not.

8. When the accumulating conveyor is not full, the housings slide out to the conveyor by gravity.
9. The cycle repeats when there is room on the belt conveyor for another tube of housings.

---

EXTENSIONS

5a. The accumulating conveyor is not running when the inserting operation is off and shutdown condition exists.

6a. When the accumulator conveyor low photoeye does not detect the housing for 2 seconds and the accumulating conveyor is running, the alarm will sound.

8a. When the accumulating conveyor is full, the magazine feeder waits.

---

SUB-VARIATIONS

1. Use
    - A centrifugal bowl feeder[13].
    - Strip feeding[14].
    - Tray packaging[15].
    - Manual feeding[16].

---

[13] In a centrifugal bowl feeder for housing feeding, instead of vibrating, the bowl rotates to provide conveying motion. This type of feeder requires fewer parts in cycle, which makes it more suitable for small orders. The advantage of using a centrifugal bowl feeder in this case is that it can supply parts at a higher rate than vibrating bowls. However, it is not able to do as much selection as the vibrating bowl feeder.

[4] Strip feeding is sometimes used for housings, either from a strip moulding, or an over-moulding process, or by additional packaging such as tape (AMP Inc, 1997). In this type of operation, the housings are on pitch.

[5] When orientation of the housings is a problem in the process, tray packaging is sometimes being used. In this case, a robot removes the housings from the mould and places them in a tray. The parts may be fed to assembly directly in the tray, from the tray to an accumulating conveyor by stripping as in the magazine feeder, manually, or a pick and place robot (AMP Inc., 1997).

[6] In the manual feeding operation, removing one housing from a bulk container, orienting it properly and placing it on the inserting equipment are done by operators.

## Use Case: 5 Inspecting Operation Using Vision System

---

CHARACTERISTIC INFORMATION

Goal in Context: To inspect the housings before inserting in order to detect a misoriented or non-conforming housing.

Scope: System

Level: Primary Task

Preconditions: Housings should be in proper orientation for mass inserting and stitching operations.

Success End Condition:

A conforming housing is being segregated from the bad ones.

Failed End Condition:

A non-conforming housing, or an escapee, is not being detected by the vision system and is used for the connector assembly.

Primary Actor:

Vision Inspection System, which consists of:

- Video Camera and Lighting (for imaging)
- Image Processor, Video Screen and Control Panel (for comparing)

Trigger: A walking beam triggers the presence of the housings from housing feeding operation.

---

MAIN SUCCESS SCENARIO

1. A video camera looks at the housing to be inspected.
2. The images are captured when the walking beam is advancing the housings.
3. Two images are taken, one in the first section of the housings, the other in the second section of the housing.
4. The video images are compared to a standard for each targeted attribute.
5. The image processor will determine whether the housing is accepted or rejected.
6. An accepted housing will move to the next process.
7. A rejected housing will move to the reject bin.

---

EXTENSIONS

'a. The process is stopped, allowing the operator to determine if corrective action is ieeded without removing part from the process.

## 5.2.3 Assembly of Contacts in Strip and Housings

### Jse Case 6: Mass Insertion Operation

---

CHARACTERISTIC INFORMATION

Goal in Context: To insert a group of 5 formed, strip fed contacts into the housing.

Scope: System

Level: Primary Task

Preconditions:

- The housing should be properly oriented.
- The contacts in strip have been formed.

Success End Condition:

Five formed, strip fed contacts are inserted into the housing.

Failed End Condition:

Incorrect numbers of formed, strip fed contacts are inserted into the housing.

Primary Actor:

- *Walking Beam*: for housing indexing.
- *Feed Finger*: for contact indexing.
- *Hold and Guide Blocks:* for holding machine action.
- *Cutter and Support Blocks:* for cutting machine action.
- *Pusher and Guide Blocks:* for inserting machine action.

Trigger: Both housing inspecting and contact forming stamping operations completes.

---

MAIN SUCCESS SCENARIO

1. The housing is indexed into position for insertion by the walking beam.
2. The contacts are indexed by a feed finger, which engages the pilot carrier and advances five contacts for every machine cycle.

3. When the contacts reach the full advance, the holding block drops to clamp the contacts to the guide block, which is in fully up position.
4. The cutting support block raises to match up with the guide block to support the contact legs and serve as an anvil for the cutter.
5. The cutter drops to cut away the pilot carrier.
6. The cutting support block drops away as the cut is made.
7. The cutter raises once cut.
8. The slugs of metal cut away drop through the support block into a container.
9. The pusher block moves forward to take control of the contacts.
10. The holding block raise out of the way.
11. The guide block moves forward into the housing to put the protruding legs of the contacts into the circuit holes in the housing.
12. The pusher block follows, holding the contacts in position.
13. The guide block drops out of the way as the legs of the contacts enter the housing.
14. The pusher completes the insertion.
15. The guide block and pusher block retract together.
16. The guide block raises to catch the next indexing of contacts.
17. The housing is indexed out of the insertion station by the walking beam as the next housing is brought in.

## 5.2.4 Loose Piece Contacts in Bags

### Use Case: 7 Feeding Operation for Loose Piece Contacts

---

CHARACTERISTIC INFORMATION

Goal in Context: To feed loose piece contacts in bags of 2500 to be inserted to the housings at the rate required by stitching operation.

Scope: System

Level: Primary Task

Preconditions: The contacts are in the form of loose contacts in bulk supply, not in other kind of forms, for example in strip.

Success End Condition:

- The loose contacts are fed at the rate required by stitching operation.
- The loose contacts are fed at the placement required by stitching operation. In this case, the placement is the availability of a contact pin every time the stitching head reaches for one.

Failed End Condition:

- Misorientation of parts that are being supplied.
- Incorrect dimension of parts that are being supplied.
- Incorrect feeding rate is being set.

Primary Actor:

- *Operator:* manually feeds the loose contacts in bags.
- *Vibrating Bowl Feeder:* is used for contact feeding.
- *Track and Gates:* are used at orienting and selecting stations.
- *Conveyor:* is used to convey parts to the next operation.

Trigger: Feeding operation starts based on the production schedule.

---

MAIN SUCCESS SCENARIO

1. Loose contacts from bags or some kind of bulk supply are manually put into the feeder bowl.

2. The feeder bowl is vibrating in an oscillating motion at a natural harmonic frequency.
3. Contacts are oriented initially as they migrate from the centre of the bowl to the wall.
4. Contacts are conveyed up the wall of the bowl.
5. Contacts are then moved to the track, which have several gates, and these gates present sufficient stations to eliminate every possible misorientation of the parts.
5. Parts then are moved to the conveyer to get ready to the next operation.

---

EXTENSIONS

2a. The feeder bowl is not oscillating at the natural harmonic frequency.

2a1. Reset the frequency by tuning the feeder bowl (use case on tuning the feeder bowl).

---

SUB-VARIATIONS

1. The contact feeding operation may use other types of feeders such as a bandolier machine, which is a combination of dereeler strip feeding and loose pin contacts.

## 5.2.5 Assembly of Loose Contacts to the Assembled Housings

### Use Case: 8 Single Contact Inserting / Stitching Operation

---

CHARACTERISTIC INFORMATION

Goal in Context: To insert four straight, loose piece contacts into the connector.

Scope: System

Level: Primary Task

Preconditions: Five formed, strip fed contacts are already inserted into the housing.

Success End Condition:

Four straight, loose piece contacts are inserted into the housing.

Failed End Condition:

Incorrect numbers of straight contacts are inserted into the housing.

Primary Actor:

- Housing Carrier
- Feed Block and Vibrating Conveyor
- Gripper Fingers on the Pre-Inserted Head.
- Final Inserter
- Walking Beam

Trigger: The stitching operation starts when the mass inserting operation completes.

---

MAIN SUCCESS SCENARIO

1. The housing carrier moves in the X-axis and Z-axis to take the housing off the walking beam track.
2. The contact is fed one at a time from a vibrating conveyer, which brings them from the vibrating bowl.
3. The contact, presented by the feed block, is gripped by the gripper fingers on the pre-inserted head.
4. The pre-inserted head then rotates at 180 degrees so that the contact is facing the housing.

5. The housing carrier moves in the Y and Z direction to align the circuit and contact.
5. The carrier moves in the X-axis to insert the contact part way. This process is called pre-inserting
7. Repeat Step 3,4,5 and 6 until the desired number of contacts is filled into the housing carrier, in this case is four contacts.
3. Once all the four contacts are inserted, the housing carrier moves in the X-axis to return to the walking beam track.
). The connector is moved by the walking beam to the final insertion station.
0. The final inserter pushes all the contacts into the housing to the required depth during the time the walking beam is not moving the housing.

## Use Case: 9 Contact Forming - Bending Operation

---

CHARACTERISTIC INFORMATION

Goal in Context: To bend the contact legs down at 90 degree so the connector can be mounted to a printed circuit board.

Scope: System

Level: Primary Task

Preconditions:

- The contacts, both part A and part B, must already be inserted to the housing.
- The entire bending operation must occur when the housing is not moving.

Success End Condition:

- The contact legs are bent down at 90 degrees with free of slivers, burrs, scratches or incorrect dimensions.

Failed End Condition:

- The contact legs are not bent down at 90 degrees and the legs have slivers, burrs, scratches or incorrect dimensions.

Primary Actor:

Bending Machine consists of supporting fingers and bending fingers.

Trigger: The bending operation starts when the stitching operation completes.

---

MAIN SUCCESS SCENARIO

1. Supporting fingers move up through the slots in the housing requiring accurate housing location.
2. Supporting fingers hold the contacts up and provide a guide for bending a smooth even curve.
3. Bending fingers pushes the contacts down to the desired location. The bending fingers move downward only when the supports are fully raised or the contact is bent incorrectly.

## Use Case: 10 Inspecting Operation – Pin Count

---

CHARACTERISTIC INFORMATION

Goal in Context: To inspect the correct number of contacts being inserted into the housing and segregate the good ones from the non-conforming ones.

Scope: System

Level: Primary Task

Preconditions: Both loose contacts and contacts in strip are already inserted to the housings.

Success End Condition:

- The connector has the correct number of contacts being inserted to the housing.
- The non-conforming connectors are being rejected from in the process.

Failed End Condition:

The non-conforming connectors are still in the assembly line.

Primary Actor:

Photoeye detector

Trigger: Pin count inspection starts after the contact forming bending operation completes.

---

MAIN SUCCESS SCENARIO

These scenarios will not be detailed in this project.

## Jse Case: 11 Inspecting Operation – Insertion Depth

---

:HARACTERISTIC INFORMATION

Goal in Context: To inspect the depth of the contacts being inserted into the housing ınd segregate the good ones from the non-conforming ones.

Scope: System

Level: Primary Task

Preconditions: Both loose contacts and contacts in strip are already inserted to the ıousings.

Success End Condition:

- The contacts are being inserted to the correct depth.
- The non-conforming connectors are being rejected from in the process.

Failed End Condition:

The non-conforming connectors are still in the assembly line.

Primary Actor:

Mechanical Contact

Trigger: Insertion depth inspection starts after the pin count inspection completes.

---

MAIN SUCCESS SCENARIO

These scenarios will not be detailed in this project.

## Jse Case: 12 Marking Operation – Hot Stamping

Jse Case: Marking – Hot Stamping

---

CHARACTERISTIC INFORMATION

Goal in Context: To mark the product identification in colour at the desired location on the housing.

Scope: System

Level: Primary Task

Preconditions: All contacts are already inserted to the housing.

Success End Condition:

Correct identifications are marked at the desired location on the housings.

Failed End Condition:

- Correct identifications are marked at the wrong location on the housings.
- Incorrect identifications are marked at the desired location on the housings.
- Incorrect identifications are marked at the wrong location on the housings.
- No marking at all on the housings.

Primary Actor:

A hot stamping machine.

Trigger: The hot stamping operation starts when the connectors are already been nspected.

---

MAIN SUCCESS SCENARIO

1. The housing is ready to be stamped.
2. A mechanical type, mounted into a fixture, is heated by an electric resistance heater to melt the housing material.
3. The colour ribbon, used to make the mark, is threaded so that it is between the type and the housing.
4. Once the desired location is identified, the type will stamp on the colour ribbon to make the mark on the housing.
5. The completed housing will move to the next process.
6. The colour ribbon is advanced approximately the width of the type after the stamping action.

---

EXTENSIONS

1a. If a housing is not present, the air cylinder used to move the hot stamp fixture will not be actuated. A photoeye is used to detect the presence of the housings.

2a. A temperature controller monitors the temperature of the fixture using a thermocouple and turns the heater on and off to maintain the desired temperature.

## Use Case: 13 Packaging

Goal in Context: To place the connectors in the specified containers for delivery to the customer.

Scope: System

Level: Primary Task

Preconditions: The assembly processes are completed.

Success End Condition:

The conforming connectors are packed in the correct packages at the proper orientation as required.

Failed End Condition:

- The conforming connectors are packed in the correct packages at the wrong orientation.
- The conforming connectors are packed in the wrong packages at the proper orientation.
- The non-conforming connectors, or escapees, being packed for the customer.

Primary Actor:

Pick and Place Robot

Trigger: The marking operation or hot stamping operation on the housings is completed.

---

MAIN SUCCESS SCENARIO

These scenarios will not be detailed in this project.

## Use Case: 14 Ink Jet Marking

---

CHARACTERISTIC INFORMATION

Goal in Context: To mark the required identification at the desired location on the packages.

Scope: System

Level: Primary Task

Preconditions: The connectors are already packed.

Success End Condition:

Correct identifications are marked at the desired location on the packages.

Failed End Condition:

- Correct identifications are marked at the wrong location on the packages.
- Incorrect identifications are marked at the desired location on the packages.
- Incorrect identifications are marked at the wrong location on the packages.
- No marking at all on the packages.

Primary Actor:

Ink Jet Marking Machine

Trigger: The marking operation starts when the packaging operation completes.

---

MAIN SUCCESS SCENARIO

These scenarios will not be detailed in this project.

**Jse Case: 15 Controlling**

---

CHARACTERISTIC INFORMATION

Goal in Context: To automatically manage machine action in order to reduce the mount operator attention required.

Scope: System

Level: Primary Task

Primary Actor:

Computer System

---

MAIN SUCCESS SCENARIO

These scenarios will not be detailed in this project.

## 5.3 Use Cases in Application to the Study

The development of the AMP connector assembly process from manual processes to a ully automatic assembly system requires knowledge to support the system evelopment. In the study, the application of use case is useful in capturing all the ecessary system requirements to develop the requirement specification knowledge nodel for the assembly system, which consists of domain knowledge and inference nowledge. The application of use case technique gives the benefits of user-friendly ocumentation at all stages of the system development process. It also helps to verify ne correctness, completeness, consistency and funtionality in each stage.

. vast amount of knowledge is essential and necessary in order to achieve high erformance of automated assembly systems, which can be re-configured and reused,. . large amount of information also makes the system to behave "intelligently". Re-onfigurable assembly systems enable production to run variety of products and allow olume to be adjusted only by adding new modules to the systems.

# Chapter 7

# RECOMMENDATIONS FOR FUTURE WORK

The purpose of this chapter is to focus on the recommendations and points need to be considered for future work.

Every project has its own limitations. In this project, time is one of the limitations where the project has to be completed in three-month time frame. This chapter will focus on the recommendations and points need to be considered for future work.

Based on the object-oriented software development framework, which was discussed in Chapter 2, the project concentrates in the process of refining the system specifications. To be more specific, the project only focuses in the system analysis, one of the steps in the refinement activities for software development. System analysis transforms the requirement specifications into functional specifications using the use case technique. Using the use case modelling, the system scope has been defined and actors and use cases are identified and grouped under corresponding scenarios.

Doing the followings can further expand the project:

Completing the refinement activities.

Verifying the refinement process.

## *Completing the refinement activities.*

The project can be further expanded by performing system design and implementation, the other two steps in the process of refining system specification.

- *System design*: In the system design step, the abstract classes of the software and the relationships among them are defined. The relationships between classes are identified through scenarios and the abstract classes (Mak and Lau, 2000).
- *Implementation*: Implementation involves refining the abstract classes and defining the software's architecture (Mak and Lau, 2000). By incorporating the physical constraints and the pre-defined solutions given by the system attributes, and by introducing the details for implementation, such as algorithms and language syntax, detailed class descriptions are generated (Mak and Lau, 2000).

## *Verifying the refinement process.*

It is important to verify the correctness, completeness, consistency and functionality of the process of refining system specification. In doing so, three procedures need to be performed, namely, cross-referencing matrices, object diagrams and CSP process algebra.

- *Cross-Referencing Matrices:* At the system analysis stage, cross-referencing matrices are used to establish the completeness of the functional specification with regards to the requirement specification (Mak and Lau, 2000). The major system functionality is connected to the identified scenarios, use cases and actors.
- *Object Diagrams:* At the system design stage, object diagrams that represent the particular scenarios are verified. In order to prove that a design specification is complete in terms of functional specification, the object diagrams are used to show that the classes identified can perform all the system functionality (Mak and Lau, 2000).
- *CSP Process Algebra:* CSP process algebra is used to specify the process model. Formal verification can be performed to predict the system behaviour and, in turn, to deduce the correctness of the implementation with respect to the design specifications as well as the original requirements (Mak and Lau, 2000).

# Chapter 8

# SUMMARY AND CONCLUSIONS

n today's manufacturing industries, the products must have these characteristics to emain competitive:

the products need to be lower in price and higher in quality with shorter delivery time.

the product variety is increasing.

the products have shorter life cycles, which leads to the minimisation of time from the concept to final product.

hese factors have driven the manufacturers to improve their productivity and to onstantly adjust their production in terms of product volume and variation of roducts in within short time frames. This situation leads to the formation of re-onfigurable assembly cells to support a wide range of assembly processes and roducts over a sufficiently long period of time.

ssembly is the focus area for productivity improvement since it contributes a bigger ortion of manufacturing cost. Moreover, assembly is important not only for making nd delivering the products the customers want, but also assembly is the last check on ie quality of the manufacturing processes before the products are delivered to the ustomer and assembly must be scheduled to meet the customer delivery equirements.

he development of re-configurable assembly systems has gained considerable ttention in the industry. A re-configurable and reusable assembly system is enabling anufactures to adapt rapidly to changing market conditions by updating products or ringing in next-generation products without investing in new equipment. Its mbedded software often determines the reconfigurability of the assembly system. he software that controls its overall system should be knowledge based.

Knowledge-based approaches have been widely used in industry because they have he advantage to be modular, re-configurable, easy to implement, maintain and ıpgrade, and fault-tolerant. A large amount of domain specific knowledge is ıecessary in order to achieve high performance and intelligent behaviour of the nowledge-based software system in re-configurable assembly cells. One of the echniques used to capture the necessary system requirements to develop domain nowledge for the software is a use case technique.

The use case modelling is a powerful technique in the software development process, nainly for the object-oriented based system. Use cases have become the norm for ystem functional requirements capture in object-oriented projects. The use case echnique aims to assist the software designer in designing the assembly system.

This paper has discussed the knowledge-based approach in the formation of a re-onfigurable assembly cell. A use case technique is presented in this paper since the echnique is useful in capturing the assembly system requirement for the development f domain knowledge in the system software. This paper also demonstrates the pplication of the use case technique using this case study of the AMP connector ssembly operation.

# REFERENCES


AMP Incorporated, ***AEM-0300, Connector Assembly Process Operation***, Manufacturing Technology Strategies, Inc. (MTS), United States of America, 1997.

Amyot, D. and Logrippo, L. (2000) "Use Case Maps and LOTOS for the prototyping and validation of a group call system", ***Computer Communications***, Vol. 23, pp. 1135 – 1157.

Berg, K. G. and Simons, A. J. H. (1999) "Control-flow semantics of use cases in UML", ***Information and Software Technology***, Vol. 41, pp. 651 – 659.

Bustard, D. W., He, Z. and Wilkie, F. G. (2000) "Linking soft systems and use-case modelling through scenarios", ***Interacting with Computers***, Vol. 13, pp. 97 – 110.

Cockburn, A., ***Basic Use Case Template***, Humans & Technology, Salt Lake City, Utah. (http://members.aol.com/acockburn)

Cockburn, A., ***Use Cases in Theory & Practice***, Humans & Technology, Salt Lake City, Utah. (http://members.aol.com/acockburn)

Fowler, M. and Scott, K., ***UML Distilled: A Brief Guide to the Standard Object Modelling Language***, Addison-Wesley, 1999, Chapter 3.

Koetsch, D. L., ***Modern Manufacturing Processes***, Delmar Publishers, Albany, N. Y., 1991, Chapter 16.

Groover, M. P., ***Fundamentals of Modern Manufacturing: Materials, Processes and Systems***, Prentice-Hall, Upper Saddle River, N.J., 1996, Chapter 31.

Heilala, J. and Voho, P. (1997) "Human touch to efficient modular assembly system", ***Assembly Automation***, Vol. 17, No. 4, pp. 298 – 302.

Heilala, J. and Voho, P. (2001) "Modular reconfigurable flexible final assembly systems", ***Assembly Automation***, Vol. 21, No. 1, pp. 20 – 28.

Hopgood, A. A., ***Knowledge-Based Systems for Engineers and Scientists***, CRC Press, 1993.

http://www.amp.com/about/

Jeong, K. C. and Kim, Y. D. (2000) "Heuristics for selecting machines and determining buffer capacities in assembly systems", ***Computer & Industrial Engineering***, Vol. 38, pp. 341 – 360.

Jordan, S. (1997) "Modular assembly: a process not an engineering technique", ***Assembly Automation***, Vol. 17, No. 4, pp. 282 – 286.

Kim, J. and Carlson, C. R. (2001) "Design units – a layered approach for design driven software development", ***Information and Software Technology***, Vol. 43, pp. 539 – 549.

Lochan, A. (1997) "Modular assembly: not just for phones but also for taps", ***Assembly Automation***, Vol. 17, No. 4, pp. 295 – 297.

Mariz, J. ***Knowledge-based expert systems in industry***, Ellis Horwood, Chichester, 1987, Chapter 1.

Motter, B., ***Manufacturing Assembly Handbook***, Butterworths, London, 1989.

Mak, K. L. and Lau, H. Y. K. (2000) "An object-oriented specification of a flexible manufacturing cell", ***International Journal of Operation & Production Management***, Vol. 20, No. 5, pp. 534 – 548.

Najjari, H. and Steiner, S. J. (1996) "An Intelligent Software System for a Robotics Assembly Cell", ***Journal of Material Processing Technology***, Vol. 61, pp. 18 – 23.

Najjari, H. and Steiner, S. J. (1997) "Integrated Sensor-Based Control System for a Flexible Assembly Cell", ***Mechatronics***, Vol. 7, No. 3, pp. 231 – 262.

Owen, A. E., ***Flexible Assembly Systems: Assembly by Robots & Computerized Integrated Systems***, Plenum Press, London, 1984.

Pham, D. T. and Pham, P. T. N. (1999) "Artificial intelligence in engineering", ***International Journal of Machine Tools & Manufacture***, Vol. 39, pp. 937 – 949.

Tanky, P. G., ***Manufacturing Database Management and Knowledge Based Expert Systems***, CIM Ware Limited Guildford Surrey, 1990.

Tatchev, S. M. and Hirani, H. J. (2001) "Requirement Engineering for Re-configurable Assembly Cells", ***Proceedings for the CIRP Anals***, Cardiff, United Kingdom.

Tatcliffe, M. and Budgen, D. (2001) "The application of use case definitions in system design specification", ***Information and Software Technology***, Vol. 43, pp. 365 – 386.

Tegnell, B., Runeson, P. and Wohlin, C. (2000) "Towards integration of use case modelling and usage-based testing", ***The Journal of Systems and Software***, Vol. 50, pp. 117 – 130.

Tolland, C. and Achour, C. B. (1998) "Guiding the construction of textual use case specifications", ***Data & Knowledge Engineering***, Vol. 25, pp. 125 – 160.

'uok, R. and Logrippo, L. (1998) "Formal specification and use case generation for a mobile telephony system", ***Computer Network and ISDN Systems***, Vol. 30, pp. 1045 – 1063.

'zafestas, S. G. and Stamou, G. B. (1997) "Concerning automated assembly: knowledge-based issues and a fuzzy system for assembly under uncertainty", ***Computer Integrated Manufacturing Systems***, Vol. 10, No. 3, pp. 183 – 192.

Vaters, F., ***Fundamentals of Manufacturing for Engineers***, UCL Press, London, 1996, Chapter 10.

Vang, C. H. and Paredis, C. J. J. "On-Line Planning of Flexible Assembly Systems: An Agent-Based Approach", ***Institute for Complex Engineered Systems Carnegie Mellon University***.

ha, X. F., Du, H. J. and Qiu, J. H. (2001) "Knowledge-based approach and system for assembly oriented design, Part I: the approach", ***Engineering Application of Artificial Intelligence***, Vol. 14, pp. 61 – 75.

# Appendix A

# BASIC USE CASE TEMPLATE

This use case template is taken from by the website http://members.aol.com/acockburn. The use case template has the following sections: name (which is the goal), goal in context, scope, level, trigger, pre-condition, post-condition, main course, extensions, sub-variations, and other characteristic data for the use case. The base template is presented twice, in simple word-processing format and in table format.

This document has the following parts:

- Template in plain text
- Example in plain text
- Template in table form
- Example in table form

## Template in plain text

Use Case: <number> <the name should be the goal as a short active verb phrase>

---

CHARACTERISTIC INFORMATION

Goal in Context: <a longer statement of the goal, if needed>

Scope: <what system is being considered black box under design>

Level: <one of: Summary, Primary task, Sub-function>

Preconditions: <what we expect is already the state of the world>

Success End Condition: <the state of the world upon successful completion>

Failed End Condition: <the state of the world if goal abandoned>

Primary Actor: <a role name for the primary actor, or description>

Trigger: <the action upon the system that starts the use case, may be time event>

---

MAIN SUCCESS SCENARIO

<put here the steps of the scenario from trigger to goal delivery, and any cleanup after>

<step #> <action description>

---

EXTENSIONS

<put here there extensions, one at a time, each referring to the step of the main scenario>

<step altered> <condition> : <action or sub use case>

<step altered> <condition> : <action or sub use case>

---

SUB-VARIATIONS

<put here the sub-variations that will cause eventual bifurcation in the scenario>

<step or variation # > <list of sub-variations>

<step or variation # > <list of sub-variations>

---

RELATED INFORMATION (optional)

Priority: <how critical to your system / organization>

Performance Target: <the amount of time this use case should take>

Frequency: <how often it is expected to happen>

Super-ordinate Use Case: <optional, name of use case that includes this one>

Subordinate Use Cases: <optional, depending on tools, links to sub use cases>

Channel to primary actor: <e.g. interactive, static files, database>

Secondary Actors: <list of other systems needed to accomplish use case>

Channel to Secondary Actors: <e.g. interactive, static, file, database, timeout>

---

OPEN ISSUES (optional)

<list of issues about this use cases awaiting decisions>

---

SCHEDULE

Due Date: <date or release of deployment>

...any other schedule / staffing information you need...

## Example in plain text

Use Case: 5 Buy Goods

---

CHARACTERISTIC INFORMATION

Goal in Context: Buyer issues request directly to our company, expects goods shipped and to be billed.

Scope: Company

Level: Summary

Preconditions: We know Buyer, their address, etc.

Success End Condition: Buyer has goods, we have money for the goods.

Failed End Condition: We have not sent the goods, Buyer has not spent the money.

Primary Actor: Buyer, any agent (or computer) acting for the customer

Trigger: purchase request comes in.

---

MAIN SUCCESS SCENARIO

1. Buyer calls in with a purchase request.
2. Company captures buyer's name, address, requested goods, etc.
3. Company gives buyer information on goods, prices, delivery dates, etc.
4. Buyer signs for order.
5. Company creates order, ships order to buyer.
6. Company ships invoice to buyer.
7. Buyers pays invoice.

---

EXTENSIONS

3a. Company is out of one of the ordered items:

3a1. Renegotiate order.

4a. Buyer pays directly with credit card:

4a1. Take payment by credit card (use case 44)

7a. Buyer returns goods:

7a. Handle returned goods (use case 105)

---

SUB-VARIATIONS

1. Buyer may use

phone in,

fax in,

use web order form,

electronic interchange

7. Buyer may pay by

cash or money order

check

credit card

---

RELATED INFORMATION

Priority: top

Performance Target: 5 minutes for order, 45 days until paid

Frequency: 200/day

Superordinate Use Case: Manage customer relationship (use case 2)

Subordinate Use Cases:

Create order (use case 15)

Take payment by credit card (use case 44)

Handle returned goods (use case 105)

Channel to primary actor: may be phone, file or interactive

Secondary Actors: credit card company, bank, shipping service

---

Channels to Secondary Actors:

---

OPEN ISSUES

What happens if we have part of the order?

What happens if credit card is stolen?

---

SCHEDULE

Due Date: release 1.0

# Template in table form

| | | |
|---|---|---|
| USE CASE # | < the name is the goal as a short active verb phrase> | |
| Goal in Context | <a longer statement of the goal in context if needed> | |
| Scope & Level | <what system is being considered black box under design><br><one of : Summary, Primary Task, Sub-function> | |
| Preconditions | <what we expect is already the state of the world> | |
| Success End Condition | <the state of the world upon successful completion> | |
| Failed End Condition | <the state of the world if goal abandoned> | |
| Primary, Secondary Actors | <a role name or description for the primary actor>.<br><other systems relied upon to accomplish use case> | |
| Trigger | <the action upon the system that starts the use case> | |
| DESCRIPTION | Step | Action |
| | 1 | <put here the steps of the scenario from trigger to goal delivery, and any cleanup after> |
| | 2 | <...> |
| | 3 | |
| EXTENSIONS | Step | Branching Action |
| | 1a | <condition causing branching> :<br><action or name of sub use case> |
| SUB-VARIATIONS | | Branching Action |
| | 1 | <list of variation s> |

| RELATED INFORMATION | <Use case name> |
|---|---|
| Priority: | <how critical to your system / organization> |
| Performance | <the amount of time this use case should take> |
| Frequency | <how often it is expected to happen> |
| Channels to actors | <e.g. interactive, static files, database, timeouts> |
| OPEN ISSUES | <list of issues awaiting decision affecting this use case > |
| Due Date | <date or release needed> |
| ...any other management information... | <...as needed> |
| Superordinates | <optional, name of use case(s) that includes this one> |
| Subordinates | <optional, depending on tools, links to sub use cases> |

# Example in table form

| USE CASE 5 | Buy Goods | |
|---|---|---|
| Goal in Context | Buyer issues request directly to our company, expects goods shipped and to be billed. | |
| Scope & Level | Company, Summary | |
| Preconditions | We know Buyer, their address, etc. | |
| Success End Condition | Buyer has goods, we have money for the goods. | |
| Failed End Condition | We have not sent the goods, Buyer has not spent the money. | |
| Primary, Secondary Actors | Buyer, any agent (or computer) acting for the customer. Credit card company, bank, shipping service | |
| Trigger | Purchase request comes in. | |
| DESCRIPTION | Step | Action |
| | 1 | Buyer calls in with a purchase request |
| | 2 | Company captures buyer's name, address, requested goods, etc. |
| | 3 | Company gives buyer information on goods, prices, delivery dates, etc. |
| | 4 | Buyer signs for order. |
| | 5 | Company creates order, ships order to buyer. |
| | 6 | Company ships invoice to buyer. |
| | 7 | Buyers pays invoice. |
| EXTENSIONS | Step | Branching Action |
| | 3a | Company is out of one of the ordered items:<br>3a1. Renegotiate order. |
| | 4a | Buyer pays directly with credit card:<br>4a1. Take payment by credit card (use case |

| | | |
|---|---|---|
| | | 44) |
| | 7a | Buyer returns goods:<br>7a. Handle returned goods (use case 105) |
| SUB-VARIATIONS | | Branching Action |
| | 1 | Buyer may use<br>phone in,<br>fax in,<br>use web order form,<br>electronic interchange |
| | 7 | Buyer may pay by<br>cash or money order<br>check<br>credit card |

| | |
|---|---|
| RELATED INFORMATION | 5. Buy Goods |
| Priority: | top |
| Performance | 5 minutes for order, 45 days until paid |
| Frequency | 200/day |
| Channel to actors | not yet determined |
| OPEN ISSUES | What if we have part of the order?<br>What is credit card is stolen? |
| Due Date | release 1.0 |
| ...any other management information... | |
| Superordinates | Manage customer relationship (use case 2) |
| Subordinates | Create order (use case 15)<br>Take payment by credit card (use case 44) |