A NEW MODEL FOR THE DEVELOPMENT OF INFORMATION SYSTEMS

BY

SHWAN MOUBARAK

A Doctoral thesis submitted in partial fulfilment of the requirements
for the award of the degree of Doctor of Philosophy
of the Loughborough University of Technology
January 1991

Supervisor: Professor E. Edmonds
Department of Computer Studies
Loughborough University of Technology

A NEW MODEL FOR THE DEVELOPMENT OF INFORMATION SYSTEMS

SHWAN MOUBARAK

ABSTRACT

The most commonly used systems specification and design techniques in commercial computing are described and compared; Information Engineering as proposed by James Martin, A Framework for Information Definition-Multiview proposed by Wood-Harper et al, Real-world Modeling as described by Jackson, Structured Analysis and Design as in Demarco, Yourdon and Constantine and Output-Oriented Structured Requirement Definition proposed by Orr. In addition, system prototyping is discussed, including the role of prototyping in large software development projects and as a tool for the design of human-computer interfaces. Other areas described and discussed include decision support systems (DSS) and knowledge based management support systems. The context is in the design and development approaches for DSS, prototyping for DSS, expert system for DSS and the integration of DSS and information system. The design and development of human-computer interface is also discussed in relation to user interface complexity and adaptive interfaces. Further, the important issue of user involvement and support within the development process is discussed. Thus, weaknesses of current approaches to the system development process are identified and a new model for the development of information system is proposed. In proposing the model, data and functional analysis structured method and methodology for decision support systems (DSS) development is presented including guidelines for the development of knowledge based DSS. The new proposed model is put to test in the design, development and implementation of large integrated commercial systems including DSS. Results and discussion on the use of the model is reported with special consideration to the users' and developers' view of the model.

Finally the objectives of this research program are examined in relation to what has been achieved during this program of research. The prospect of using the model for the development of information systems are concluded with references to current and future goals.

ACKNOWLEDGEMENT

I would like to thank my supervisor, Professor E. Edmonds for his guidance and encouragement throughout this program of research.

I am grateful to Mrs. Beryle Meus, Mrs. Rosemary Linnell and Mrs. Lyn Abbey for organising the typing of this thesis.

I would also like to thank the many researchers and others involved in the design and development of information systems, particularly those connected to the methodologies and techniques reviewed in this thesis.

I would like to express my gratitude to the members of the business systems development team, the selected users at Massey-Ferguson Parts-Company who took part in our study and the dealers responding to the survey.

January 1991

Shwan Moubarak

CONTENT

Chapter 1 Information, Methodology and Systems,1

1.1 Introduction,2

1.2 Information System Methodologies,8

1.3 Decision Support and Knowledge based management support Systems,38

1.4 The Design and Development of Human-Computer Interfaces,44

1.5 The Need for User Involvement and Support,52

1.6 Conclusion,62

Chapter 2 The Systems Development Process and a New Model,67

2.1 Information Architectures,68

2.2 Methodology for Decision Support Systems Development,75

2.3 Incorporating Interactive Dialogue Networks within the Systems Development Process,89

2.4 The new systems Development Life Cycle Model,91

2.5 System Development Productivity Tools,132

2.6 Conclusion,135

Chapter 3 The Role of Information Computer Systems in a Major Business Initiative,138

3.1 Place of Strategy in Shaping and Guiding a Company,139

3.2 Information Systems for Competitive Advantage,141

3.3 The Corporate Business Environment,142

3.4 Conclusion,151

Chapter 4 The New System Development Life Cycle Model in Use,153

4.1 From Concept to Practice,154

4.2 Case Studies -Portfolio of Examples Application,155

4.3 Evaluating the New Model-Evaluation Methodology,174

4.4 Conclusion,199

Chapter 5 Results and Discussions on the use of the New Life Cycle Model,200

5.1 The New Life Cycle Model and System Success,201

5.2 Project Risk Assessment-Results,203

5.3 Control Specification Assessment-Results,204

5.4 The End-Users' View of the Model,204

5.5 The Developer's View of the Model,217

5.6 Development Productivity Gains-Results,225

5.7 Implementation as an agent of change,228

5.8 Conclusion,233

Chapter 6 Project Review, Future Research and Conclusions,234

6.1 Summary of Proceeding chapters,234

6.2 Suggestions for further Research Work,255

6.3 Conclusion,257

References,260

Appendix

CHAPTER 1

INFORMATION, METHODOLOGY AND SYSTEMS

1.1 Introduction

What Goes Wrong in Systems Development?

During the late 1970's, it was recognised at the author's own organisation, Massey-Ferguson, that large and complex systems were being built with virtually no management control, resulting in high cost overruns and delivery of the systems, if delivered at all. Furthermore, systems that were delivered often did not meet the user's requirements and were supported by poor or non existent documentation, which made systems maintenance and enhancement a nightmare. The users and developers alike wondering whether or not there was more fruitful approach to developing these systems even when the systems can be cost justified. The need for a radically different approach to the development of information processing systems became apparent. With this need came a critical evaluation of existing systems design practices and the realisation that those practices were inadequate. Clearly, the effective and efficient control of information resources was attracting a lot of interest. However, not only the quality of the design product, but even more fundamental key questions were asked about their investment in information systems. Typical questions asked included the following:

- The needs of the commercial user: Are the Professionals and the Managers getting the right information, on the right timescale, to make effective decisions?

- Is the organisation developing the right systems in the right sequence based on business strategy to achieve a meaningful competitive advantage?

- Does the current systems development strategy lay the foundation for later business and systems development or is it just based on a short term solution to problems?

- Does the current systems development strategy take the best advantage of latest state of the art technology?

To address these issues, we have to bring together those with knowledge of the business and those with the expertise in the information and systems fields. We have to identify and eliminate possible causes of misunderstanding and ensure that the nature and needs of the business can be communicated between staff with different skills and experience. Therefore to gain a clear understanding of a business situation, current practices and problems, we need three points of view. The expert in that aspect of the business, another person with a broader view of the business and the systems designer/developer. Damodaran and Eason (1981) have argued that computer systems stand or fall on the ability and willingess of users to make effective use of them. They went on to argue that systems designers may feel they have created a perfect system for the user, but if the users are not also convinced, their effort may have been wasted. Hekmatpour and Ince (1986) have argued that the earlier an activity occurs in the systems development software project, the poorer are the notations used for that activity. Secondly, the earlier an activity occurs in the software project, the less we understand about the nature of that activity. Finally, the earlier an error is made in the software project, the more catastrophic are the effect of that error. The significance of defining information requirements prior to proceeding on to the design phases of an information system development was identified by Boehm (1976, 1980, 1981). He has argued that early errors in requirements and specification have cost a hundred to a thousand times as much as those errors made during programming. Wasserman et al, (1983) have stated "a key assumption of modern software development practices is that increased effort in the earlier stages of development will result in a better system". Gallier and Lyons (1984) discovered that senior management believe a crucial factor contributing to the successful implementation of a management information system is the proper identification of their information requirements in the first instance. Their observation is in line with the results of research undertaken by Taggart and Tharp (1977) and Carter et al, (1975). The absolute need for requirements analysis and specification techniques was, therefore, generally accepted and invented for example (Bell et al, 1977, Ross and Schoman 1977, Teichroew and Hershey 1977, Lehman and Yauneh, 1985). Other notions proposed included mathematical methods of software development which claims for more reliable systems meeting user needs Jones (1980), Silverberg (1981) and Musser (1979).

The importance of requirements definition at an early stage of the systems development and user involvement was also identified at the author's own organisation, Massey-Ferguson. As a result, the systems development life cycle standards and structured systems development methods for developing information computer systems were introduced Moubarak (1989). Management at Massey-Ferguson found that now they have control over systems development and insisted that these standards be strictly followed. This meant that any systems being planned, developed and implemented had to follow the standard project plan and produce the standard outputs or deliverables. This proved to be both beneficial and a burden for management. It was beneficial because the standards gave management tight control over the total process. The Auditors and the Quality Control staff found it useful because at last they could demand specific documentation in specific formats and containing specific information. It was a burden because the standards were directed at a particular design, the design of large mainframe information systems with no consideration given to decision support systems. Furthermore, the computer specialist applying these standards found it difficult to describe in general terms the characteristics that make a problem appropriate for a specific type of computer system development. A typical computer systems designer/developer needs to consider information and/ or decision support systems including knowledge base decision support systems. In addition, the Management Information Services Department sees its problem as one of constant overload. Moreover, There is a growing demand for new systems, but available resources with specific required skills are seldom adequate. Furthermore, technical solutions to productivity problems address symptoms, not fundamental issues. The fundamental issue of increasing productivity does not depend on software tools alone. True productivity gains come from identifying and developing systems that are critical to the organisation's business and ensuring that inadequate design approaches are not employed during the systems development process.

The need for a framework:

Based on notions discussed above, there appear to be several goals that we must attain to improve the system design and development process. What is needed is a framework containing guidelines which allows the designer/developer and the user to work together within the systems development process. A framework that supports the design and development of information systems, decision support and knowledge based decision support systems. To this end, the aim of this program of research has been to focus on the systems development process with specific emphasis on the construction of a new system life cycle model with the objective of providing guidelines for designing and developing information systems and decision support systems including knowledge based decision support systems in commercial computing environment incorporating dialogue networks within the process.

The program of research was divided into two phases of study and investigation. The first phase addressed by the M.phil thesis Moubarak (1989) included the following:

(i) An investigation into the design of interactive systems with special consideration given to models of interaction, design methodology, information flow and categories of human-computer dialogue. The context was in the design and development of large integrated commercial systems.

(ii) The state of the art was surveyed in the area of human-computer interfaces; this was accomplished with particular references to selected papers published. Other areas surveyed included decision support systems and expert systems for decision support.

(iii) The interface system, "an environment for the design of interactive dialogue networks" was developed. The system casts interactive dialogues in the form of transition networks. The network is analysed using facilities provided by the interface system to monitor the network and user performance during the interaction.

(iv) The use of dialogue networks as a framework for studying user interface issues as a communication activity, the psychology of the user and finally the design of the user interface itself was discussed. The discussion was based on observation and experiences of user sample of diversity of background when creating dialogue networks at the author's own organisation, Massey-Ferguson.

(v) Finally, the role of dialogue networks within the systems development process in commercial computing was examined and benefits of applying it to each stage within the development process identified.

The second phase of the program of research focused on the new systems development life cycle model; the subject for this thesis. The material in this thesis is presented in six chapters, together with twelve appendices. Following the introductory discussion in this chapter, which described the evolution and objective of the program of research, the sections ahead seek to survey and review the appreciation of major systems specification, design and development strategies. A broad set of strategies including the following:

- **Information system development methodologies.**

- **Decision support and knowledge based management support systems.**

- **The design and development of human computer interfaces.**

- **The need for user involvement and support in the development process.**

Chapter 2 concentrates on the systems development process. Notions discussed in Chapter 1 are used to develop the "new systems development life cycle model", together with detail discussion on structured data and function analysis. in addition, guidelines for developing knowledge-based decision support systems and role of dialogue networks within the systems development process are discussed.

Chapter 3 discusses the role of information systems in the corporate and business unit strategic direction, the context is in the author's own organisation, Massey-Ferguson.

The subject for Chapter 4 is quite specific in that it uses the new systems development life cycle model described in Chapter 2 in the design and the development of live commercial computer systems at the author's own organisation, Massey- Ferguson. Results and discussions on the use of the life cycle model are discussed in Chapter 5. The discussions are based on a view which sees the model as an agent for managing change in the business corporate strategy and in the development of business systems for the competitive edge. Technical and implementation misalignments encountered during the development and implementation of the live commercial systems are discussed. In addition, the role of user involvement and support in th design and development of these live systems are discussed. The discussion is in the context of four major system indicators. System quality, system usage, user attitudes, and user information satisfaction.

Finally, Chapter 6 reviews the objectives of the program of research in relation to what has been achieved and future research and development goals.

1.2 Information System Methodologies

Before proceeding to the practical aspects of systems specification, design and development, it is worthwhile considering briefly the term "information system" and what is involved in design and development of such a system.

Zorkoczy (1985) described "information systems" as a non-standard term covering computer based service which provide information in response to specific request from users. Wood-Harper et al, (1985) defined the term "information system" as a system capable of collecting, processing, storing, transmitting and displaying information. Another author in the field of information systems, Connor (1985) describes the term to imply a system that provides management with decision making information. This definition excludes other types of information such as operational systems, audit control systems and finally decision support (what if) type systems. As for the main purpose of information systems it aims to satisfy user's information needs. The way in which these needs are expressed largely determines the principles of operation of particular information systems. The information systems may be operated by organisations for their own benefits and/or offered to other third party customers. An example of this is the author's own organisation. At Massey-Ferguson, information systems are providing services for the internal company organisation and third party customers. Increasingly, the organisation is looking to information system to gain a competitive edge. To deliver this business advantage, information systems must reflect the business needs, not outmoded or outdated operating procedures. But more significantly, these systems must evolve rapidly to support changes in the business and markets it serves. To achieve this responsiveness requires a solution to the problems that traditionally face systems designers and developers. Due to these problems the process of developing information business systems has been under study by many academic institutions and commercial organisations and there are many school of thought about the methodology that should be used to develop a well designed business system. Some methodologies are concerned primarily with technical issues, others with organisational issues in relation to the strategic business direction and change management in the nature of clerical and professional work.

A well designed business system needs to consider the technical as well as organisational viewpoints. In Chapter 2, we will show the way this is addressed within the new systems development life cycle model. As for the development process Moubarak (1989) has reported that the process of developing a computer system can be broken into a series of steps or activities, which create a set of pre-defined deliverables. The sequence in which the activities occur is almost identical for all systems development projects and this fact can be exploited by developing a framework for the system development life cycle and is a high level definition of the system development process. There are several approaches for developing computerised information business systems. One such approach was described in detail within the systems development life cycle standard Moubarak (1989). In addition the complete system development process must contain guidelines and techniques for the specification and design of computerised information systems. There are a variety of specification and design techniques. We will review several of these techniques below. The techniques vary in scope and each of them has its own strengths and weaknesses and we will be discussing these in the sections a head.

1.2.1 Information Engineering

Information engineering is a term used by James Martin, the pioneer of information engineering methodologies, to describe the set of inter-related disciplines which are needed to build effective information systems in support of corporate strategic requirements. The basic premise of information engineering is the need for data to lie at the centre of modern information processing. Information engineering proposes that basic data in an organisation is stable, while procedures tend to change, hence data orientated development techniques succeed if correctly applied, whereas procedure orientated techniques have failed. Another basic premise of information engineering methodology is that the users of information must be thoroughly involved in the information systems development process. These users take responsibility for the nature of their systems by becoming directly involved in processes which enable them to specify their own requirements. The methodology also provides techniques to

support the development of corporate information systems and decision support systems. In this section we will discuss information engineering cocept in the form of a set of building blocks, each reflecting the stages of the information engineering methodology. In addition we will be discussing additional information engineering techniques and we will argue the case for information engineering.

(i) **Areas of Information Engineering:**

Information engineering methodology as proposed by James Martin consists of an integrated set of staged techniques. Each stage is dependent upon the one beneath it, although the blocks can be assembled in different ways to suit the needs of particular organisations. The areas of information engineering are:

- **Strategic requirements planning:** is the base on which all the others rest. It determines the objectives of the enterprise and the information needed to accomplish these objectives. The information systems can be built without strategic requirements planning, but to do so involves risk taking as final quality product can be assured.

- **Information analysis:** is a cataloguing of the types of data that must be kept of the business activities that use them and of how they relate to one another. Information analysis may be carried out across an entire enterprise or only one division, subsidiary, factory. It creates an information model at corporate level.

- **Data modelling:** extends the information analysis. It creates a detailed logical data model based specifically on the views of the users and makes it as stable as possible. Automated aids simplify the consolidation and refining of user views.

- **Procedure formation:** identifies actions that change or use the data. It uses simple diagrammatic techniques to group the actions into access diagrams. The methodology claims that these procedures can be converted into program specifications written in

structural English.

- **Data use analysis:** provides a formal way of collecting and illustrating information on the usage path, the volumes of use and response time needs for physical database design.

- **Distribution analysis:** addresses the questions of whether the data model may be split for implementation in separate databases and whether processing power should be distributed. Factor analysis and usage of distribution analysis provide the basis for taking appropriate decisions and relating them to political and geographical consideration.

- **Physical database design:** uses the logical data model and usage information to develop a detailed database design that will satisfy the needs of all of its intended users whilst being consistent, secure and maintainable.

- **Program specification synthesis:** identifies and groups all changes to data and formally develops program logic for the required procedures. This is done using structured English which replaces conventional program specifications and can be read and inspected by end users.

- **Programming (coding):** The use of a new language and which enables results to be obtained without the use of professional programmers. However, conventional programming is still proposed in many installations.

(ii) **Additional Information Engineering Techniques:**

Macdonald (1986) identified additional information engineering techniques which support analysis and design activities. They are:

- **Data analysis diagramming:** Shows data items and the associations between them. They are used in the business area analysis stage to analyse fields in current systems or fields required in business process. They are then used in business systems design to confirm that designed user views are compatible with the business' data model. They are also used to represent the user's views of layouts (reports and screen layouts).

- **Stage transition diagramming:** They are used to document details of the life cycles of things such as entity types and dialogue screens which are changed by the activities with which they are involved. This technique is used in business area analysis for entity life cycle analysis and in business system design to show interactions between various screens and procedures which take place to support the screens.

- **Data flow diagramming:** They are used to illustrate associations between designed activities - procedures program modules or systems. They can be used by strategy planners and analyst to show current systems architecture. Data flow diagrams show the flow of data between each procedure in the design area. They also show flow of data between procedures and data stores and flows across the boundaries of design area.

- **Data structure diagramming:**These diagrams are used by designers to prepare the preliminary data structure during business system design and refined data structure during technical design. The diagrams show the structure of data used in database or file systems. They illustrate the record types included in the design, the types of linkages between those record types and any direct entry mechanisms.

- **Action diagramming:** These diagrams express the logic of a process, procedure or program. Action diagrams are used in business area analysis, where process action diagrams describe the logic of processes; in business system design, where procedure action diagrams define the logic procedures and dialogue steps; in technical design, where program action diagrams define the logic of programs and modules.

(iii) **The Case for Information Engineering:**

Connor (1985) gave an overall picture of advantages and disadvantages of information engineering and have reported that systems designed with information engineering techniques are designed based on data and data relationships derived from management objectives and current data in use rather than data flow allowing the systems analyst, designers and users to concentrate their attention on data and hence save considerable time. Further, Data normalisation the process which refines the initial set of relations into an optimum set, sometimes called a relational model with the objective is to eliminate data redundancy (and hence simplify update processing) and maximise extensibility and understanding allowing for efficient data dictionary. Other benefits obtained are Stable, logical databases or files are obtained which can easily be converted to physical databases or files. These logical databases have been designed in conjunction with users. As a result, the users completely understand and relate to them. Output procedures are separated from file maintenance procedures. Maintenance procedures are developed based on events that add, delete, modify and retrieve data. As for disadvantages of information engineering Connor (1985) has reported that information engineering is suited to systems designed on the basis of a static data model which emphasises data storage and data update and it is not suited to development of systems with strong time dimension. In section 1.2.3 a suitable method for time dimension systems development.

In section 1.2.6 we will be comparing the information engineering methodology to other methodologies reviewed in this chapter. In chapter 2, we will discuss information engineering methodology in relation to the new system development life cycle model.

1.2.2 Multiview: A Framework for Information Systems Definition

Multiview addresses problems associated with the analysis and design activities of information systems definition. It is a methodology to structure the tasks for the analysts and users during the analysis and design activities. Multiview incorporates five different views which are appropriate to the progressive development of an analysis and design project, covering all aspects required to answer the critical questions of users. The methodology supports both technical and human requirements. In this section we will describe the five stages of Multiview as presented by Wood-Harper et al, (1985) and consider the case for the multiview approach.

(i) The Stages of Multiview Approach

The methodology exhibits five different views, all of which can be emphasised, reduced in scale, or even omitted, according to circumstances. The five stages as presented by Wood-Harper et al (1985) are;

- Analysis of Human Activity Systems

The term human activity is used to cover any sort of organisation which might consider using a computer. The stage is based on the analysis of human activity systems carried out by (Checkland and Griffin, 1970, Checkland, 1984). It will form the basis for describing the systems requirements and will be carried forward to further stages in the methodology. The technique proposed for analysing the human activity systems is the rich picture. The "picture" represents a subjective and objectives perception of problem situation in diagrammatic and pictorial form, showing the structural of the process and their relation to each other. It is a pictorial representation of an organisation. The picture showed all the major people activities, tasks and issues and reveals major worries and potential conflicts. The picture is achieved by getting hold of as much background information as possible from all sources. The picture should include elements of structure, process and

relationship between them. In addition, the picture shows all the information about the problem, the role of the analyst and members of the team. Finally, the picture should show both the primary tasks of the situation and the issues surrounding those tasks.

- **Information Modelling:**

This stage consists of analysis of entities, functions and events of the system described, independent of any consideration of how the system will eventually develop. This stage consists of two phases. The creation of function model and the entity model. The function model is to identify the main function. This is then broken down into subfunctions until the functions cannot be broken down further. In constructing the entity model, the analyst (problem solver) identifies the entities from the area of concern. Relationship between the entities are established. The entity model is then constructed. The model is represented as a chart with entities shown in boxes and relationships indicated by lines and arrows.

- **Analysis and Design of Socio-Technical System:**

The philosophy behind this stage is the "user involvement" in the design and development process. Multiview argues that user participation in the analysis and design of systems that they will be using will enhance implementation, acceptance and operation of the system. Multiview proposes that human considerations, such as job satisfaction, task definition, are just as important as technical consideration.

- **Design of Human-Computer Interface:**

This stage is concerned with technical design of the human computer interface and makes specific decisions on the technical system alternatives. Once the human-computer interfaces have been defined, the technical requirements to fulfill these can be designed. In Section 1.4, we will discuss human-computer interface design in detail.

- **Design of the Technical Subsystem:**

This stage is concerned with detail technical design with specific consideration to efficient design and the production of a full system specification.

(ii) **The case for Multiview**

Wood-Harper et al notes that many design methodologies are prescriptive not only of what must be done but of the order in which it has to be done. In the real world, decision are often made before all the facts have been gathered. Frequently this is due to time constraints. This is just as true of the system designer as of the general or the entrepreneur. No-one could pretend that this is the ideal way of decision making, but Multiview can help to make decision which are supportable in these situations. The above five stages, all of which can be emphasised, reduced in scale, or even omitted, according to circumstances. The final outputs of the methodology as defined by Wood-Harper et al are the social systems, the role set and people tasks, the human-computer interface and the technical specification, including the information to design, implement, operate and maintain a more complete system in both human and technical terms. In Section 1.2.6, we will be comparing the Multiview approach to other methodologies reviewed in this chapter. In Chapter 2, we will relate this approach to the new development life cycle model proposed in this thesis.

1.2.3 Jackson System Development (JSD)

JSD is a method for specifying, designing and implementing software systems. It was developed by Michael Jackson and his colleague John Cameron. It has been in use since 1981. McNeile (1986) gave a brief account of JSD. He described it as a method technically in flavour guiding the developer through the stages of creating a detailed specification of requirement, design of programs, modules data storage and finally the software itself.

However, the method does not address areas of user interview techniques, cost benefit analysis, user interface design quality assurance and project management. Jackson (1983) has argued that "a JSD system may be regarded as a simulation of the relevant parts of the real world outside the system; system functions are regarded as providing outputs derived from the behaviour of this simulation". The emphasis is on the "real world outside the systems which is dynamic, with events occurring in time ordered sequence". The real world provides the subject matter for the system: it contains the engines to be controlled, the employees to be paid, the customers and suppliers whose transactions are to be accounted for. The JSD model simulates the real world and in the process conveys the scope of the system to the user in clear terms. The user then has the opportunity to add, delete or modify this scope. The system functions are added to the model and provide the required outputs. The frequency of the outputs is taken into account and the system is implemented. Within JSD the primary distinction is between specification and implementation. The JSD development procedure has six steps, of which the first four are concerned with creating a specification of the required system and the last two with imoplementing the specification. The design is has largely been absorbed into the implementation part of JSD. To understand JSD approach to systems development we need to discuss the JSD specification medium, the JSD development steps, structure clash between data streams in JSD and finally the case for using JSD.

(i) <u>The JSD Specification Medium</u>

McNeile (1986) used the term "implementation by transformation" to describe the conversion of JSD specification into a working software. This involves the creation of a mapping whereby the specification, which has been developed in a form which reflects the structure of the subject matter of the system, is transformed into a new shape which reflects the structure of the implementation environment. The transformation produces the programs, transactions database structure. The specification medium is in form of a network of communicating sequential process. A sequential process is an execution instance of a program text. The text is a tree structure of sequence, selection, iteration

and executable components. Both diagrammatic and textural representatives are used for these tree structure are used for the tree structures. There are two means by which sequential processes can be made to communicate; data stream and state vector communication. One process may communicate with another process by operations on a named data stream. The operations on a data stream are write and read. The data system is considered to be an unbounded buffer in which writing and reading obey the FIFO rule. The state vector communication consists of its local variables, together with its text pointer. The value of a process state vector can be changed only by execution of the process itself.

(ii) **Development Steps**

The development steps for the construction of a JSD specification consists of six development steps. They are:

- **The entity action and structure steps:** A JSD entity is an entity that exists in real world outside the system. Examples of these are customer account number, part and market. An action must take place at a point in time. An example of an action is the function of picking and packing into a case. In the entity structure step, we construct an abstract model of the real world. The model represents the entities and the actions defined in the entity action step. The model is represented by structure diagram.

- **The initial model step:** In this step, the developer specifies the system to be built by specifying a simulation of the real world. The structures defined in the entity/action and entity structure step are used to define a set of sequential processes to model the real world. The first task in this step is to state hoe the real world process is to be connected to the model process. The system specification diagram is used to represent the connection between the real world and the model.

- **The functions steps (interactive and information):** In this step, the developer specifies the system functions in terms of the model. The model is extended to provide required outputs. This is done by specifying events or a combination of events in the model. Structure text is then written for the detailed specifications of functions. An example of this step is given by McNeile of a system which keeps track of debts. The model process for a debit process might be based on the action issue-invoice, receive-payment and write off.

- **System timing step:** The purpose of this step is to specify and constraints on the implementation that may be required to ensure that it is reasonable. Two issues need to be considered. First the speed of execution of the system, defined in terms of elapsed time and secondly relative scheduling, defined in terms of how far one process may advance relative to another.

- **System implementation step:** Connor (1985) described this step as a task of identifying how many real or virtual processors are to be used; where the number of processors is far smaller than the number of processes, how the processes will be distributed over the processors; and where one processor has to handle more than one process, how the time will be allocated to each. The developer has to consider other factors, such as programming language, teleprocessing transaction monitor.

(iv) **Structure Clash**

Connor (1985) has described the structure conflicts arising from two or more data streams which merge as input to a process, or are output from a process, may not have the same data structure or be in the same sequence. This is referred to by Jackson as structure clashes. He divides them into three classes: boundary clash, ordering clash and interleaving clash. Boundary clash occurs when the data input to a process have a different boundary from the data output and are in the same sequence. An ordering clash occurs when the data are in a different sequence from the data output. Interleaving clash

occurs when two or more data streams can contain the same data and the data can vary from transaction to transaction. Jackson solves these conflicts through the use of subroutines, tables in primary storage or on-line files and processes that reflect the data structure.

(v) The case for JSD

Jackson (1983) has argued that JSD is used to develop systems whose subject matter has a strong time dimension. In a information system for a bank, the subject matter is the real customer, their loans, their payments..etc. The emphasis is on the time dimension. A loan must be granted before repayment begin; repayment are credited before interest charges are applied. Because of this emphasis on the time dimension JSD is used for the development of systems with a strong time dimension such as operating systems, program generations, compilers, elevator control systems and missile guidance systems. However, JSD can also be used to design and develop any form of business system on-line and batch.

From a project management point of view, a JSD project has three main phases. In the first phase an abstract description of the real world is made consisting of entity action and entity structure steps. The second phase, a process model representing the initial function and system timing steps including currently known functions. The third phase converts the specification into practical set of executable programs. The major benefit claimed by Jackson is that major check-points should occur at the end of each phase. He argues that the end of the first and second phase, the check is concerned to establish the fit between the specification and the users' needs. At the end of the third phase, the check is primarily technical, addressing questions of convenience and efficiency of the system execution and the correctness of the implementation with respect to the specification. Jackson argues that it demands a clearer separation of skills on the part of the development team. The first two phases are focused on the user. The developer needs skills to be able to communicate well with users and to reach quickly an understanding of the users needs.

The third phase in JSD provides a demanding technical task that must be carried out by people whose interest lie in the technical area:

The major disadvantage of JSD, argues Connor (1985), is that JSD as a development tool needs a very competent person to use it effectively and it can consume large amounts of development time. In addition JSD is not "Top-Down". The fundamental idea of top-down development is that the object to be developed, whether it is a small program or a large system, can best be regarded as a hierarchy. Starting at the highest level of hierarchy and the decomposition into a small number of large objectives, then each of these decomposed further until the lowest level is reached. Top-down is a reasonable way of describing things which are already fully understood. Jackson however, argues that top-down is not a reasonable way of developing, designing or discovering anything. In his view the earliest stages of top-down development deal by definition, in unknowns. The decision to decompose the hierarchy is inevitably imprecise and vague, because those parts will not become precise until the development is completed. We do not share this view and will show how a top-down supported by checking out with bottom up method can result in good systems supporting user requirements.

In Section 1.2.6, we will compare JSD with other techniques discussed in this section. In addition, in Chapter 2 we will discuss JSD again in relation to techniques used in the new systems development life cycle model.

1.2.4 Structured Analysis and Design

Demarco (1979) defines structured analysis as the use of data flow diagrams, data dictionary, structured English, decision tables and decision trees to construct the structured specification. These tools provide the mechanism for the computer specialist and the users to record the specification with the objective of constructing the software for implementation from the design. Yourdon and Constantine (1979) define structured design in two ways. First as "the art of designing the components of a system and the interrelationship between these components in

the best possible way". Secondly as "the process of deciding which components interconnected in which way will solve some well specified problems". Connor (1985) has argued that both these definitions are very general and is applicable to any effective system design technique. He has proposed a more precise definition that is "structured design is a strategy used to convert the target document obtained from structured analysis into an implementable computer system using the principles and tools of structured design". He has described structured analysis and design as a technique based on system hierarchy. The analysis stage and the system processes are described graphically as boxes with inputs and outputs linked in networks. The network is called the data flow diagram. Each box is shown as a bubble and the inputs and outputs are data flows. The total system is viewed as a single bubble which is exploded into a bubble network and each bubble is exploded further into separate networks until a "primitive" bubble level is reached. These bubbles are described in language which is based on Bohm and Jacopini's (sequence, iteration and selection). The primitives are termed minispecifications. Data flow diagrams are used to record the present physical system, the present logical system, the proposed physical system. The data flow diagrams are supported by a data dictionary and minispecification. All of these are used to develop structure charts. In this section we will discuss steps in structured analysis and design followed by a discussion on the case for using the structured analysis and design techniques.

(i) **Steps in Structured Analysis**

The principle tool of structured analysis is the data flow diagram which illustrates the seven steps involved in structured analysis approach.

- **The current physical environment:**

This step concerns itself with recording the flow of information through the current system - no attempt to distinguish between logical and physical structure is made. The output from this step is the data flow diagrams of the current system with all data flows and files recorded in the data dictionary. The scope and all the external interfaces are shown on

the data flow diagrams.

- **The logical representation of the current environment:**

  In this step, all the physical features of the system is stripped and normalisation of the data in the current files takes place. Foe example a form in the previous step may contain one or more records in this step the form is elimented and only the record data is accounted for. In addition the normalisation will eliminated data redundancies in these files.

- **The creation of the new logical environment:**

  In this step, the users new requirements are added to the logical representation developed in the previous stage. The new requirement may result in the addition of new data and processes or the deletion of existing data or processes.

- **The creation of the new physical environment:**

  This step concerns itself with the task of identifying different options to provide the new system. Each option consists of different bubbles in the data flow diagram, depending on the degree and type of automation involved. In each case, the diagram is partitioned and option described. These partitions distinguish between people function and machine functions including the man-machine interface. The output from this step are the physical option.

- **Selecting the design option:**

  In this step, the option best suited to user needs in terms of cost/benefits and the development times is chosen.

- **Packaging option and supporting documents into the structured specification:**

In this step, four major activities are performed. First the key interfaces are highlighted. Second the user specification guide to help the users are prepared. Thirdly the Material to augment the specification are prepared. Finally details that have been deferred are now been recorded (e.g. error messages, startup and shutdown procedures).

(ii) **Steps in Structured Design**

Before we proceed and describe the steps involved in structured design, we need to discuss where structured design fits into the system life cycle. Structured design has input to it from the structured analysis phase in particular structured specification and the hardware/software configuration requirements. with the structured design there are four major steps. They are:

- Design of the structured chart based on the logical data **flow diagram:**

This chart is a hierarchy of modules which call one another and pass data and control between them. Structured charts are the mechanism for converting the data flow diagrams into a series of instructions that can be executed on a computer. High level modules are the only modules represented by this chart.

- **Refine the structured chart:**

The objective of this step is to examine the chart and the modules to obtain minimum of interdependence between modules and the maximum of module cohesion (the process of sticking together tightly the elements in a module to execute the function or functions within the module). To convert the data flow diagrams to structured charts, two tools are used. Transform analysis and transaction analysis. Transform analysis deals with the conversion of data flow diagram for sequential use. Transaction analysis deals with the conversion of transactions.

- **Module specification:**

In this step,instructions in the logical modules that makes the structured chart is specified.

- **Packaging physical module:**

This step concerns itself with the identification of manageable set of process instructions which perform related functions. The output from this step is a set of physical modules and programs that can be coded during implementation.

(iii) **Implementation**

Implementation follows and heavily influenced by the structured design. Three options are available for the implementer to develop and test the system. The first option is to start at the top and build a framework of high level modules where he can test their interfaces and add on modules at lower levels until all the system is tested. The second option is to start at the bottom and keep adding high level modules. The third option is to start from both ends.

(iv) **The Case for Structured Analysis and Design**

Connor (1985) has argued that the key feature of structured analysis and design is the step by step evolution of the analysis and design. This evolution starts from the present physical system, converted to the present logical system including data normalisation eliminating data redundancy. This logical system is then converted to proposed logical system containing new system requirements. This system is then used to specify the new physical system and become the basis for the new system design which in turn drives to testing and implementation. The second feature is that the process within the methodology deals with manageable pieces of information which can be effectively

controlled. The hierarchical design approach provides a well organised and manageable system. Structure charts provides modules which are independent and functional.

The major disadvantage highlighted by Connor is that the large amount of data is recorded and controlled. The weakness of the methodology is the duplication of data. The same data is defined in many ways (due to action taken on data values) which is the fundamental principle of structured analysis and design. This problem applies to structured charts which due to large amount of data and controls being passed from one module to another, making the charts messy and hard to maintain. Changes in the process causes changes to the data model. This could cause problem when designing complex systems and large files or databases. All time-orientated features of the system are only addressed during the logical module packaging. Finally, the principle of data conversion requires lower level data flows to be children of high level data flow and this conversion task is time-consuming and difficult in which errors can easily occur, even if an effective data dictionary software package is used.

In Section 1.2.6, we will compare the structured analysis and design to other methodologies described and discussed in this chapter. In Chapter 2, we will be comparing the structured analysis and design to the methodology used within the new systems development life cycle model.

1.2.5 Structured Requirements Definition

Orr (1977) defined structured systems development as "output orientated design". The outputs the user needs to do his job. Orr (1981) expanded this definition further. He has argued that "it is not always possible to elicit the required outputs from the user". Based on these two notions, Orr identified the need for "a requirements definition process that works from a definition of the problem to the definition of the outputs. But even this is not enough. It is also necessary to develop a planning and definition phase that moves from an initial problem statement (symptoms) to correct definition of the problems and scope (systems identification)." Orr,

therefore, proposes the notion that system specification and design are derived from output requirements. The tools used in structured requirements definition are the Warnier-Orr diagram and the entity diagram. Structured requirements definition is divided into two phases. The logical definition and the physical definition phase. Connor (1985) reviewed structured requirements definition technique including advantages and the disadvantages of the technique including tools used. In this section we will discuss these tools, requirements, documents and phases of structured requirements techniques. In addition we will discuss the case for using structured requirement techniques.

(i) **Tools Used in Structured Requirement Definition**

Tools used are Warnier-Orr diagram and the entity diagram. Warnier-Orr diagram is an analysis and design tool which uses braces to display hierarchical systems (organisation, a computer system, a computer program, a data structure or a set of manual procedures). It resembles a family tree, but instead of being vertical, it is horizontal. It is based on the principle that a single parent can have many children and a child can have only one parent. There are three application areas for Warnier-Orr diagrams. Data structure diagram, system and process diagrams and assembly line diagram. In the first application we only interested in the data structure representation using a Warnier-Orr diagram notations. The representation takes the form of a single parent (e.g orderv header), order -item and other details are the children. In the second application the Warnier-orr diagram is used to represent the process (e.g processing a customer order from entry,validation to print order details). The third diagram the assembly line diagram is very powerful application of the Warnier-Orr diagram. These diagrams desribe the input, output and the process including time frames governing the subsystems.

(ii) **Requirements Document**

We described structured requirements as output orientated design. The output expected from the process are the Layouts, samples, data structure, process logic, volumes,

frequencies and response times. Data dictionary for logical files, logical records, data items and data structures. Definition of the assumptions and constraints involved in the system including risk and benefits of the various approaches.

(iii) **Phases of Structured Requirements Techniques**

The total process consists of two phases. The logical and physical. The logical consists of three major steps. First step is the definition of the application context, e.g. user-level entity diagram for each user, combined user-level entity diagram, application-level entity diagram and the definition of the objectives. The second step is application functions which consists of four steps. Defining the "mainline" functional flow (linking the principle processes together in a stream), scope, subsystem definition: Each subsystem can now be decomposed into a series of tasks or procedures and decision support functions. The third step is the application results in the form of principal outputs. This step concerns itself with identifying and defining the principal outputs. In addition, organisational cycles are defined. As for the physical definition phase, the logical definition of the system provides a detailed requirements of what the system must do. In the physical definition phase the logical is converted to physical specification of how the system will be built.This is done in five steps. The first step is to define constraints on building the system. These are limitations on building the systems (computer processing capacity and storage capacity are examples of constraints). The second step is to define the alternative physical solutions. Examples of alternatives to be considered includes "do-nothing" or "purchase a software package". The third step is to identify the benefits and the risks for each alternative solution. The fourth step is the selection of a solution and the recommendation of course action. The development team guide management on its choice. The fifth and final step is the creation of the final requirements definition document.

(iv) **The Case for Structured Requirements Definition Technique**

Connor (1985) identified three advantages of structured definition. The first is that it results in systems which are a direct reflection of the user needs (the outputs). The second, the hierarchical design structure, followed in the functional flow diagrams, ensures that logical subsystem design and the decomposition of the subsystems into processes satisfy the time frames of the outputs. Thirdly, the use of the Warnier-Orr diagram ensures a one parent to many children relationship, which in turn ensures the production of single outputs from multiple inputs. Structured requirements definition have two major disadvantages. The first is that the emphasis on the outputs tends to create data duplication. Orr (1981) does not discuss the design of systems where the requirement is for logical database which is constantly interactively updated and does not involve movement of information between entities.

In Section 1.2.6, we will compare structured requirements definition technique with other methods discussed in this chapter. In Chapter 2, we will relate this methodology to method used in the new systems development life cycle model.

1.2.6 System Prototyping

Boehm (1974) has reported that in some large systems, up to 95% of the code had to be rewritten to meet user requirements. Leibrandt and Schnupp (1984) have argued that formal improved techniques and notions for user requirements analysis and specification are not fully addressing the problem reported by Boehm. Furthermore, the transition from user conceptual model of a system to the specification of a proposed software system is an informal process. Hekmatpour and Ince (1986) have reported many researchers have concluded that software development, particularly during its early stages, is a learning process and that this process involves both the system developer and the user. For this process to be efficient, it requires close co-operation between the developer and the user and can only be successful when it is

based on the actual working system. These notions have been supported by Lehman (1979) who has argued that software system can be regarded as an evolutionary product rather than an entity subjected to a series of development activities followed by a distinct and separate activity known as software maintenance. A number of techniques that are based on these ideas have emerged. The techniques are known under the generic term prototyping.

In this section we will be discussing system prototyping. In particular prototype development, categories of prototyping and the role of prototyping in large software project development.

(i) **Prototype Development**

Draper and Norman (1985) have argued that the processing and user interface of a software system should be regarded as separate entities and designed as such. This view is supported by Hekmatpour and Ince (1986). They classify technical approaches to prototyping into those that are relevant to prototyping the functional aspects of the system and those that are relevant to user interface prototyping.

When discussing the subject of prototype development, it is important to define prototyping within the context of software engineering. Budde and Sylla (1984) and Gehani (1982 b) have defined it as a mocked-up initial version of a system which is thrown away after use. Gilb (1981) and Patton (1983) have suggested that the term prototype should be used to refer to the throw-it-away approach and the term evolutionary development to be used when a prototype "evolves" to become the final system. The question of whether a prototype should become a final system was addressed by Hekmatpour and Ince, (1986). They have argued that two important questions need to be answered. The first is how the prototype should be constructed and secondly when it can be accepted as the final product. Based on this, they have argued for the classification of prototyping techniques and have identified three categories of prototyping. We will now describe each category:

- **Throw-it-away prototyping:**

This type is referred as specification prototyping Keus (1982) or as specification by example Christensen and Kreplin (1984). Hekmatpour and Ince have argued that this type corresponds to the most appropriate use of the term "prototype". It consists of building a version of a system which lacks some of its intended functionality or where some constraints are relaxed. (Dearnley and Mayhew (1981)) and (Kraushaar and Shirland (1985)) have stated that such a system is normally never used after user requirements have been clarified. Floyd (1984) pointed out that what is important about throw-away prototyping is the process itself and not the product. Hekmatpour and Ince have proposed that the primary part of the prototyping effort should go into the critical evaluation of the prototype rather than its developmental design. For this to happen, facilities for rapidly producing software must be available to the developer. Throw-it-away prototyping are also used for examining designs and as test comparators, (Dearnley and Mayhew (1984)), (Bonet and Kung (1984)) and Weyuker (1982). In chapter 4 we will show the use of this approach in designing reports and screen layout for a live commercial user interfaces.

- **Evolutionary prototyping:**

This is used throughout a project. Nauman and Jenkins (1982) proposed the development of an inefficient version of the proposed system for requirement clarification and then gradually refine it into production system. Nosek (1984), Gilb (1981) and Patton (1983) have all proposed the incremental production of different versions of a system which has increased functionality. Rzevski (1984) identified the advantage of evolutionary prototyping to be minimising the impact of change on the software project. Other advantages of evolutionary prototyping identified by Hekmatpour and Ince (1987) include the following: the impact of early errors is much less serious. It can be used as a tool for training end-users. Development can be organised on a small team basis where each

team produces a release of the software. Finally, it allows for more efficient use of developer time. This is achieved by reducing communication complexity between team members. The final product is the output from the evolutionary prototyping process. Parnas (1972, 1979) has argued for modularity as design principles so that no extra work needs to be carried out to produce the final product. Munson (1981) has argued for modifiability into the software from start if the evolutionary approach is to succeed. In chapter 4 we will show how this approach is used in the design and development of live commercial user-computer interface.

- **Incremental prototyping:**

Baldwin (1982) described this type of prototyping as a type allowing systems development one section at a time. The design responds to changes in requirements as they occur. Bally et al, (1977) and Taggart and Tharp (1977) have all proposed the use of conventional project management techniques to control incremental prototyping environment.

(ii) **Large Software Project Development and Prototyping**

Hekmatpour and Ince (1987) and experiences at the author's own organisation, Massey-Ferguson, have shown that prototyping can play a big part in providing a solution to overcome a number of problems which surface in large software projects. The first problem arises from the fact that the designer/developer and the end user have cultural differences which lead to user requirements being at best an approximation. The second problem relates to the use of documents. Large projects result in bulky documentation. As a result, the end user finds it difficult to visualise the final product. The third problem occurs when the end user realises that his original perception of what he requires, as expressed in a requirements specification, does not meet his needs. A fourth problem which occurred in the author's own organisation is that in medium to large projects the nature of end-user requirements change due to the company's internal reorganisation. The changes in requirements that can happen are no fault of the end-user of a specific

department. The computer system (final product) has to operate in a real world which is dynamic and which has to reflect the real world reflecting the organisation. All four problems can be addressed by evolutionary prototyping. A prototype will be available throughout a project from start to implementation. The first two problems can be solved by throw-it-away prototyping. Introducing the prototype in the early stage in the project and using it as a tool for training.

(iii) **Prototyping and the systems development process**

Based on the notions discussed in this section it appparent that prototyping has a major role to play within the systems development process. We will discuss this role in section 1.2.6 where we will address the question which is often asked. Which technique to use? The discussion is in the context of prototyping in relation to other methodologies and techniques discussed in this chapter. Furthermore, in Section 1.4, we will be discussing user interface prototyping. In Chapter 2, we will describe the role of prototyping within the new systems development life cycle model and in Chapter 5, we will report on the results of prototyping in designing real life commercial systems.

1.2.7 Systems Specification and Design Techniques - Choosing the Right One?

In the previous section, we discussed six systems specification and design techniques. Choosing the correct system design technique is important and only the systems analyst/designer can say which is best suited to his needs. Each approach has its good and weak points. Each using its own conceptual framework, each based on its own philosophy, affecting systems development work. A particular technique may help more with one class of problems and situations than with another. Maddison et al, (1983) has argued that comparative judgements on the merits and features of two or more methodologies can not easily be obtained from methodology authors and practioners, since most people's experience is limited to a single methodology. Floyd (1986) argues for the development of methods for the investigation of methods, concepts for the description and comparison of methods and criteria

for their evaluation and assessment.

Information systems developers are interested in selecting systems specification and design techniques that will be appropriate in their particular environment, address the problems that they have and improve productivity during the development process. Connor (1985) proposes combination features from different techniques to support the developer designer in achieving his objectives. This view is supported by author's own experience in developing information systems at Massey-Ferguson. In this case, combination features of techniques from different methodologies developed to create a framework for designing and developing information systems in the commercial computing environment. This framework is described in Chapter 2 in detail. However, the subject for this section is choosing the "right" methodology containing the design technique in relation to the five techniques discussed. Three important questions need to be answered. First, which of the techniques discussed is the best? Second, is one technique really better than the other? Third, can we combine two or more techniques to answer these questions. We discussed the advantages and disadvantages of each technique previously. We will now briefly compare the techniques with each other.

(i) **Information Engineering, Structured Analysis and Design, Structured Requirements Definition and JSD:**

Connor (1985) proposes that the common factor between these techniques is the "normalised data model". In information engineering, the data model is developed before procedures are identified. In structured analysis and structured requirements definition, the model is developed based on the logical files developed from the procedural flow. This is acceptable as long as the data in the data model are restricted to a particular system being developed. Where the data model needs to support the corporate both organisationally and system levels, then information engineering supports this requirement efficiently rather than structured analysis and design including structured requirements definitions. The data model created in information engineering can then be used to interface with structured analysis and design and with structured requirement definition.

The requirement for corporate level data model broken down to organisational and system levels have also been identified at the author's own organisation. In Chapter 2, we will be discussing how this is achieved. Floyd (1986) points out from experiences with structured analysis and design technique that the syntax of the language is easy to learn. Competence in the method itself requires extensive practice. The method leaves a lot to the developer's discretion in spite of the detailed rules about how to proceed. Floyd argues that despite the various shortcomings of the method, the method considered a reasonable approach to start out with.

In the case of information engineering and JSD, information engineering argues Connor (1985) is a database orientated, static data model which presents relationships between records but does not represent data flow. We defined JSD as a technique technical in flavour for defining and designing dynamic, time orientated systems. Database design is addressed at implementation stage in JSD. The common factor in both systems is the entity having the same meaning in both techniques. Entity definition actions performed on the entities, structured entity mode and actions during entity life cycle products of JSD gives the systems analyst and user a good understanding of the business. It will also simplify the task of developing the data model. Furthermore, as we stated previousley JSD is not a top-down method and in our view top-down supported by bottom-up can result in good systems. In chapter 2 we will be discussing a specific method which supports this view. Finally, Floyd (1986) states "system development according to JSD is possible, but implies great additional effort with doubtful benefits".

(ii) **System Design Techniques and Prototyping**

Experience at author's own organisation shows that the introduction of prototyping software has given the impression to the end-user community that any system that the user requires can be modelled (e.g. reports and screens, etc.) without any additional information provided to create the internal and external databases. This assumption implies that systems can be prototyped with no assistance from specification and system

design techniques we discussed. Connor (1985) proposes the use of prototyping software and good systems design techniques for effective and fast development of system. Prototyping can work in harmony with information engineering, structured analysis and design, structured requirements definition and Jackson System Development (JSD) techniques argues Connor. File record structures, screen layout and other principle outputs from information engineering is constructed using prototyping software. In structured analysis and design, the specification of a logical data flow diagram of the user's requirements which is converted to a physical data flow diagram which in turn becomes the basis for the system design using the structure chart as a design tool. Connor argues that the data flows defined in the final physical data flow diagram could provide the output, input definitions and with availability of normalised data structure prototyping is possible. The problem argues Connor is that we will not be using structured design and in particular structured chart. Instead, we are using the prototyping software to produce databases and output programs. This will result in developing a "model" and the real system can then be developed using structured design and structured charts.

Prototyping can also be used with requirement definition techniques. The input, output definitions and transactions which is the result of the mainline functional flow diagrams and application results principle outputs from structured requirements definitions needed to be prototyped. The missing key feature is the file structure. Connor argues that prototyping can be used in conjunction with structured requirements definition, but only after functional flow diagrams, file definitions and outputs are defined.

JSD and prototyping have little in common argues Connor. JSD is a technique for defining and designing dynamic time orientated systems. Database and file structure design in the later steps within JSD steps. Before this step, it is assumed that every entity exists by itself and is processed on its own independent processor. The process within JSD step is defined in considerable detail in procedural language. Therefore, any attempt to prototype these steps will result in wasted development effort.

(iii) Information System Definition: The Multiview Approach

A multiview is a methodology to structure the tasks for the analysts and users to address problems associated with the analysis and design activities. Wood-Harper et el, (1985) have argued that multiview approach can help to make decisions which support situations causing departure from the ideal methodology (decision making before all the facts are gathered by the systems analyst/ designer) as taught in text books. Methodologies that specify what must be done, but of the order in which it has to be done, e.g. information engineering, JSD, structured analysis and design and structured requirement definition. The multiview methodology combines important aspects of some of the major methodologies and thus offering the systems analyst/designer and the end user a broad understanding of the whole process of analysis and design. The five stages of multiview described previously are appropriate to the progressive development of information system. The order of the five stages are important from the general to the specific, from the conceptual to hard fact, and from issue to task. As for the role prototyping with the multiview approach, the socio-technical systems, the human-computer interface and the technical subsystems can all be prototyped.

1.3 Decision Support and Knowledge Based Management Support Systems

Ginzberg and Stohr (1982) attempted to define and provide an orientation to the DSS field. They note that a number of non-intersecting definitions of DSS have been offered and that each definition represents the special interest of author proposing it. They argue that the definition of DSS must reflect the central concern of the DSS field (support and improve the decision making in the organisations). The earliest definitions included the following:

Gorry and Scott Morton (1971) identify DSS as systems supporting managerial decision making in an unstructured or semistructured decision situation. Recent definitions of DSS offered by Moore and Chang (1980), Bonczek, Holspple and Whinston (1980) and Keen, (1980). Moore and Chang define DSS as extensible systems, systems capable of supporting ad-hoc data analysis and decision modelling, systems orientated towards future planning and finally as systems used at irregular, unplanned intervals. Bonczek, Holsapple and Whinston defined DSS as a computer system consisting of three major components. Language system (facilities to provide communication between the user and components of the DSS). Knowledge system - we will discuss knowledge based decision support systems in Section 1.3.2. The third component of DSS identified is a problem processing system. Keen (1980) gives a full definition. He argues that it is a product of development process in which the DSS user, the DSS builder and the DSS itself are all capable of influencing one another and resulting in evolution of the system and the pattern of its use. Sprague (1980) has argued that DSS is not an evolutionary advancement of information systems. It is a class of information system that draws on transaction processing systems and interacts with the other parts of the overall information system to support the decision making activities of managers and other knowledge workers in the organisations. However, there are differences between DSS and traditional information systems. These systems require a new combination of information systems technology. The way these technologies interact are important and in section 1.3.4 will discuss the integration of information systems and decision support systems. There are many views of DSS. Ralph and Sprague (1980) have reported three views of DSS. The first view is "DSS application" with characteristics that make it significantly different from a typical data processing application. In this case, DSS is the "software" that allows a specific decision maker

or group of them to deal with a specific set of related problems. The second view of DSS is the "DSS generator". A package of related software which provides a set of capabilities to quickly and easily build a specific DSS. Executive Information Systems (EIS) is an example of DSS generator. The third view is DSS tools which facilitate the development of a specific DSS or DSS generator. This category of technology has seen the greatest amount of development specifically in the area of expert system technology for decision support. Expert systems for decision support is discussed in section 1.3.2.

1.3.1 The Design and Development Approach for DSS

Traitional information systems and methodologies have different emphasis than those needed for DSS. The emphasis is on well defined tasks, low uncertaintly, and slow changing user requirement and data structurees. DSS systems involves considerable volatile user requirements. Arinze (1989) has argued for the development of viable frameworks for understanding and describing real world decision making and guidelines for transforming decision models into set of requirements and specification for DSS. Arinize discusses relevant issues for decision support and two important points. The first is that the DSS builder must take of the primary, secondary and wider environments. The primary environment contains the decision maker with required skills. The environment will hold resources available to decision maker for carrying out both the decision making process and task performance.
The second point is how a methodology can incorporate the descriptive formallsms in order to create useful DSS. The important link between description of user requirements and development of DSS specification. In chapter 2 we will disscuss a methodology for DSS development within the new system development life cycle model. Further, we will briefly discuss the design of man-machine decision systems as proposed by Gerrity (1971) below we will also discuss prototyping in the context of DSS. Sheinin (1980) proposes two approaches to the design of DSS. A universal approach and a problem orientated approach. The first approach is based on decision making theory and theory of choice. Sheinin argues that managers face a set of alternatives when going through decision making process. The second approach proposed is problem-orientated approach. This is appropriate to decision making

process consisting of two stages, generating the alternatives and choosing between alternatives.

(i) **Design of man-machine decision systems (MMDS):**

Gerrity (1971) argues that the fundamental problems in building successful MMDS are methodological rather than technological. He proposes a general framework for decision centered MMDS design methodology. The framework which consists of four major stages. They are:

- **Decision system analysis and problem finding:**

Four activities are identified. First decision system bounding and goal setting. Defining the system and what it aims to accomplish. The second descriptive modelling. This activity entails the analysis and the description of the current man-machine decision system structure and behaviour. The third is normative modelling. In this activity, an abstract model of an idealised process for achieving the goals of the decision system is constructed. The fourth activity - recognition of problems as gaps between the normative and descriptive models.

- **New decision system specification:**

The activities in this stage consist of predicted decision system behaviour and decision system functional model. The first entails modelling of predicted man-machine decision system behaviour. The latter involves the specification of decision system functions that should be programmed into the machine to alleviate the gaps and provide a more effective decision process.

- **New decision system design:**

In this stage, two activities are identified. The design of the MMDS (e.g. hardware, programs, data structures and operating procedures). The second is the design of the decision control system. Examples of activities within this design phase include the design of measures and mechanisms for testing the hypotheses implicit in the productive decision system model.

- **Decision system development and evolution:**

Implementation and control and adaption are the two activities within this step. Implementation of an activity includes the installation of the hardware and the training of end-users of the system. Control and adaption include the control and measurement of actual system behaviour to allow for redesign and evolution.

(ii) **Prototyping for DSS**

Henderson and Ingraham (1982) proposed prototyping as an effective approach for developing and implementing DSS. They have argued that empirical research has shown this design strategy is effective in establishing meaningful user involvement and high user satisfaction. The benefit in prototyping DSS is the availability of a crude, but usable immediate system that can support the user during the extended system development life cycle.

Henderson and Ingraham (1982) tested their notion, prototyping a case study implementation of a model-based DSS. The case study suggested that the highly convergent design process may be detrimental in that significant information needs may be missed. Secondly, the influence of individual difference is strong - the features reflect the biases of the individual and may lose the potential creativity associated with effective group design process. Finally, they identify the need for an integrated programming

system product that combines both data orientated features with model-based features.

1.3.2 Expert Systems for Decision Support

The generic DSS framework proposed by Bonczek (1981) sees decision support systems as having three components. These are its language system, knowledge system and problem processing system. Fox, Alvey and Myers (1983) have argued that knowledge based techniques may facilitate decision making methods which are not possible with traditional mathematical approaches. Knowledge based techniques allow any number of decision making strategies to be combined. Reitman (1982) has argued that good decision support systems are powerful tools. However, compared with the support provided by a competent human staff, they are still limited. Many decision problems involve interactions among intelligent agencies or organisations. Decision support systems not employing knowledge based techniques cannot deal with these directly. Instead, argues Reitman, their projection mechanisms require that these interacting intelligence be replaced by numeric models. These will approximate the consequences of the interactions as the system moves through time. Further, even with the best decision support systems, the user has to specify each of the individual alternatives to be considered. Reitman (1982) suggests that because they cannot make use of knowledge about goals, resources, options and constraints in an interactive framework, they are unable to selectively generate good alternatives. They can only project forward, in a non-interactive, non-selective fashion, the outcomes of options chosen for consideration by a human user. By contrast, an intelligent expert system, DSS can collect, organise and use the knowledge in ways that enable it to create good alternatives entirely on its own.

Based on these notions, it follows that DSS have a major contribution in the development of expert systems. This contribution is based on major demands from the decision support systems users. The fundamental requirement identified (Benchimol et al, (1987)) is ease of use as well as clarity of screens and dialogues. Another quality of decision support systems is often their ability to operate at two levels; an advanced level for the experienced user and a simple system for the less experienced user. Finally, decision support systems often interface with

large corporate databases which can be achieved with and of expert systems technology. Benchimol et al, (1987) argue that the decision support approach provides inspiration for the designers of expert systems and to allow them to produce systems accessible to their users. As for expert systems technology itself, Moubarak (1989) has described the evolution and the interplay between principles and practice in expert systems including knowledge representation and acquisition techniques employed. In addition specific expert systems and shells were reviewed. Specific systems and shells reviewed included: Prospector, Yorktown MVS Manager, RI, Savoir, Counsellor, XI and Mycin. In Chapter 2, we will address knowledge based decision support systems development.

1.3.3 The Integration of Information Systems and DSS

Experience at the author's organisation, Massey- Ferguson, has shown that small decentralised systems give the end-users much more control over their own requirements. However, the consequences of many users doing their own thing using non standard software and hardware can be counterproductive. The centralised mainframe information and operational systems collect, store and present the information to the end-users in a standard format. The problem with these systems as seen by the end-user is that they often are not very responsive in meeting management's need for reporting the data. Decentralised decision support systems generally make the most efficient use of computers with the advantage of releasing the systems developer to concentrate on more strategic systems development for the corporation. This integration can only be achieved through communication between individuals who understand the development of computer systems and those who understand the requirements of decision support systems. Traditionally, the users of business information systems are supporters of the decentralised approach and established systems developers are supporters of the centralised approach. Mattern (1981) argues that the most important distinction between these two groups of people is the way in which they visualise the information that is processed by their system. Further, he argues that looking at information in terms that are understandable by both groups. From a common understanding, developing the other aspects of interactive business systems should come much easier. The common element in the notions discussed is that a conceptual

integration of the corporate total business information system provides a significant step towards giving end-users easy access to company information by using terms they understand. This integration is achieved with help of a framework, incorporating the design and development of decision support within the systems development process. In Chapter 2, we will be discussing this framework.

1.4 The Design and Development of Human-Computer Interfaces

Sutton and Sprague (1978) report that, on average, 59% of the program code accounts for the user interface. This view is supported by experiences in developing commercial decision support and information systems at the author's own organisation, Massey-Ferguson. Sutcliffe (1988) reports that in on-line systems, it is the interface which is the critical and physically the largest part and that good design is important. Edmonds (1982) has suggested that the design of human computer interface is central to the design of an interactive system. Long (1986) states, "Design should be understood more widely as development. Although design in the limited sense might be thought to involve principally software engineering, development in the wider sense more obviously involves both human factors and software engineering." Following this, Edmonds (1987) has argued that the hardest task is to understand what is the "human factor" and that it can be understood in terms of the concept of "usability" - the ability of users to operate the system. Schackel (1984) states, "We have much to learn about usability so as to be able to produce valid guidelines. Extensive research studies of different types of users, doing different types of task, with different hardware and software tools to establish a comprehensive understanding of the parameters of usability." Hekmatpour and Ince (1986) have argued that user interface design is an inherently difficult task and that designing an interface so that it appeals to all users is not a simple task. Shneiderman (1979) has stated, "The complexity of the requirements for a user interface often results in conflicting design goals which necessitate a compromise." At the author's compromise." At the author's own organisation, Massey-Ferguson, computer systems were designed in the early 1970's with the assumption that the user should adapt to the system. This assumption is no longer acceptable by the user community. Good et al, (1984) have argued that it is unreasonable to require all

users to spend considerable time learning how to use a system and that the computer system should be made to adapt to it. We will discuss adaptive and intelligent human computer interfaces in Section 1.4.2. Ten-Hagen (1980) proposed the design of interface as the first part of the system to be designed within the systems development process. The systems development process contains a number of fairly universally accepted stages. Moubarak (1989) has reported that analysis and design as two most closely related areas. Both the systems analysts and the designers are concerned with the user requirements. The difference being the systems analyst is concerned with data and functional analysis, while the designer is concerned with the user interaction with the system. Gould (1987) has argued that designing useful, usable and likeable computer systems is hard and there is a need to bring designers and developers closer to users. Following Gould, Sutcliffe (1988) reports that a common theme within interface design is concern and involvement with the user. Methods have been developed within the human computer interaction and within the area of systems science with the objective of improving the user involvement in the system development process. In Section 1.5, we will discuss the subject of user participation in the systems development process in detail. There are several topics which relate to research issues in the design and development of interface design. First, interface design is discussed in the context of evaluating user interface complexity, which is followed by a discussion on prototyping the user interface as an alternative approach. Finally, adaptive and intelligent user interfaces, dialogue specification are reviewed, concluding with a discussion on human computer interface design and this program of research.

1.4.1 Evaluating User Interface Complexity

Karat et al, (1987) reported that considerable effort has been directed towards development of formal user-interface analysis tools and that important steps are now being taken in this area. Work in this area includes the following: Newall and Card (1985), Moran and Newell (1983) and Kieras and Poison (1985). Notions proposed by these authors have the objective of improving the design process for the user interface development. Sutcliffe (1988) has argued that design should be evaluated as early as possible in the development life cycle and later when

implemented. In designing an interface system, the designer is faced with a number of design decisions. What is available to the designer is general guidelines or standards. In order to develop guidelines for design methodology of interactive human-computer interface, the members of the group (Guedj et al, 1980) discussed and presented several models of interaction. These models were described as defined by their authors previously Moubarak (1989). Karat et al, (1987) reported that there has been movement to change the basic style of interaction with computer systems from interface dominated by command languages to "direct manipulation" interfaces. In the command language type dialogue, the user types fixed format command strings, an example of this type of command language interface is the "command language grammer (Moran 1981)". We described CLG previously Moubarak (1989). In direct manipulation interfaces, the user manipulates displayed objects. Shneiderman (1983) has argued that direct manipulation systems offer the user the satisfying experience of operating on visible objects. The pleasure in using these systems stems from the capacity to manipulate the object of interest directly and to generate multiple alternatives rapidly. We described direct manipulation for human-computer dialogue in detail previously Moubarak (1989). Karat et al, (1987) have examined both types of dialogue and argue that "given that command language and direct manipulation systems are clearly different, a key question is whether or not there are formal analysis techniques which would help the designer decide between each alternative. One question examined here is whether or not a production system analysis of the two systems would predict one as superior to the other." Experimental results reported by Karat et al, (1987) suggest large differences in performance between the command language and direct manipulation systems which favour direct manipulation. Another approach to the design of human-computer interfaces is that of simulation Clark (1981). Hekmatpour and Ince (1986) have argued that simulation is a powerful means for studying user behaviour and the effectiveness of proposed systems. Transition networks and production rules are two alternative approaches used to construct interactive dialogues. Each approach has its advantages and its disadvantages. Networks are amenable to task and dialogue separation design concept proposed by Edmonds (1982). Interfaces designed with this philosophy have the advantage that the resulting system can be easily changed as a result of user experience with the system and that effective monitoring and evaluation facilities can be implemented

within such systems. Alty (1982) proposed the use of path algebras technique for analysing dialogues created with transition networks. Dialogue networks, however, can be large and inflexible. Production rules approach is more flexible, but difficult to monitor and evaluate. Guest (1982) examined the effectiveness of the production rule and network approaches by requiring the student to produce interfaces using both approaches. He concluded that with the network approach, all the students managed to produce working systems, even when not supplied with adequate documentation. Students who had indepth knowledge of parse and translation writing systems using production rule approach were less successful in producing the dialogue systems. These motions have formed the basis of one aspect of this program of research, in particular in the area of interactive dialogue network implementation in a commercial computing environment Moubarak (1989). Further, in designing a software tool "the interface system" a dialogue delivery vehicle with specific facilities for evaluation of user and dialogue network performance. Another way of prototyping user interfaces is via facilities provided by screen generators and tools. These facilities allow the designer to actually produce the format on a VDU display and carry out a repeated process of modification until the user and developer agree with the presentation. Hekmatpour and Ince (1986) report two categories of screen prototyping tools. A high level notation for screen definition. Christensen and Kreplin (1984) implemented a processor which converts screen definitions to a prototype version of the screen display. Dixon (1985) and Sale (1985) have proposed an alternative to Christensen and Kreplin. A package of library routines accessible from programming language. A second category makes use of screen editors to produce screen layouts interactively. This facility is used extensively in the author's own organisation where PSF panels using "TSO", the time sharing option powerful systems development editor, is used to generate interactive screen demos. Mason and Carey (1983) have devised a technique known as architecture based methodology. The designer starts an external view of the system and works inwards. During this process, the designer ensures that the system is acceptable and understandable to the user. This approach supports previous notions discussed which support the view that systems development should start with the user interface. Other recent approaches which make use of facilities provided by programming languages such as report generation facilities provided by programming language APL and LISP for prototyping

command languages (Levine 1980). These facilities have the advantages of freeing the programmer from the task of having to deal with the very low level detail found in programming human-computer interfaces.

The above approaches to evaluation support the notion that it is advantageous to evaluate a design before it is built. However, there are approaches to evaluation which assume that the product or a prototype does exist. Broadly, three approaches have been reported by Sutcliffe (1988). First, diagnostic analysis which identifies poor design features in an interface design. The second approach would be to monitor the interface for error rates, frequency of command use and duration of usage. These facilities are used extensively in the commercial computing build within teleprocessing software packages to monitor user interaction with the system. Finally experimental analysis, designed to test empirically two different interface designs or two different features of design. Data can be recorded using a variety of techniques. Sutcliffe (1988) reports five techniques for gathering evaluation data: system logging, video recording, direct observation, protocol analysis - users are asked to think about what they are doing in terms of mental activity, decisions and reason for decisions and questionnaires - this technique is useful for correcting subjective data about user characteristics.

The selection of a particular approach and analysis techniques depends on what the evaluator requires to measure. Shackel (1984) states "evaluation is an important design procedure; indeed some would say that design is nothing but test and try again. Certainly the complexity of information systems and the speed of technological change is such that design must be a very flexible and interactive process, with evaluation at each stage."

1.4.2 Adaptive Human-computer Interface

One of the central dilemmas of interface design reported by researchers in the field is how to satisfy the conflicting demand of different users. Users are seen by the developers as novices and experts. Novices require easy-to-use dialogues supported by help facilities. Experts on the other hand need quick, efficient dialogues with less support or with limited help facilities.

However, with practice, novices will become experts. Damodaran and Eason (1981) have emphasised the desirability of a complete user support system which would provide help and tutorial assistance and include a local expert - highly knowledgeable of the system. Benyon (1985) argues that it is often overlooked that an expert at one piece of software may be quite unfamiliar with another, hence this must be viewed in respect of each task performed. The problem of adaptivity is described by Maguire (1981). He states "each specific application for an interactive situation is likely to consist of a unique combination of attributes which determines the nature of the dialogue required." Benyon (1985) asks the question "how can we provide a dialogue which adapts to different users performing different tasks at different times?"

Edmonds (1981) defines the concepts of system adaptivity as a system containing a mechanism for containing a mechanism for providing negative feedback interfaces. Evaluation is proposed as a design method for adaptive interfaces. Another method in use reported by Edmonds is simulation at an early stage of the design process. As for the implementation method and tools for the adaptable interfaces, Edmonds (1981) has argued that in order to make an interface adaptable by a computer specialist, it is necessary to provide a high level implementation facility - extensions to existing computer languages and examples of such facilities. In Edmonds' model of user interface, the designer amends the interface according to data obtained from a resident monitor module. Innocent (1982) argues for systems capable of continuous change by suitable agents, the agent described as a "designer" and a "self adaptive interface system". Innocent goes one step further; he proposes the replacement of the "designer" in Edmonds' model by an "expert system". Based on Innocent's model, Benyon (1985) has developed "MONITOR" to provide a self-adaptive user interface. It has been developed as an knowledge-based system which will monitor, modify and maintain details of the user, the task performed and the tasks available to the system. A prototype system in the area of computer aided learning has been implemented. Another system which is based upon a combination of transition networks and production rule system is the "CONNECT" system developed by Alty (1981). The dialogue between the user and the system is specified as a network of nodes. Adaptability is achieved by allowing each arc to be enabled or disabled by an expert system, which continuously monitors arc transitions. Alty

proposed the use of path algebras for analysing networks. He has argued that any dialogue network will have a set of different labelling schemes, which can be conveniently represented by an adjacent matrix displaying the set of labels between all possible node connections for any particular label set. The idea is to represent the network by the adjacent matrix containing 0 and 1 elements, 1 for valid arc and 0 where no transition exists. Path algebra states Alty "gives us the representation of the network and the way of considering it despite its complexity".

Sutcliffe (1988) states "that adaptation remains an issue of contention and that it is the subject of considerable research activity. How much adaptation is a good thing and how well adaptation can be linked to the user's abilities are problems still to be solved." The key issue identified by researchers is whether the user interface system should be allowed to automatically adapt as a result of user activity. Alty (1984) has argued that adaptability will only serve to confuse the user's model of the system. The classic problem of "hunting" may occur while the user is starting to adapt to the system, while the system is trying to adapt to him. Sutcliffe (1988) proposes adaptive interfaces as a potential solution to the mixed user population dilemma, although the consistency of the interface has to be maintained as it adapts.

In this program of research, we identified the requirement to investigate the possibility of automatically including the user's own performance during the interaction with the "interface" system developed within this project Moubarak (1989).

1.4.3 The Interface System for the Design of Human-Computer Interfaces

Before describing the interface system developed in this program of research we will briefly discuss and model of human-computer interfaces. Moran (1980) considers the construction of interfaces that matches the user's mental model of the system. He proposed three component for the interface. The I/O processor, dynamic processor and the background tasks. The dynamic processor determines the actions that computer system should take at any given moment. The background processor performs tasks, some of these tasks pass information to

the dynamic processor without it being requested. The dynamic processor can be implemented as a set of production rules proposed by Hopgood and Duce (1980) or as a recursive transition network implementation. Denert (1977), Edmonds (1981) and Alty (1982). The interface system developed in this program of research casts interactive dialogue in the form of transition networks. Our objectives in designing the system have been to eliminate complexity . This is achieved by hiding complexity behind sophisticated software and to allow for the separation of conversational aspects (the dialogue) from task aspects (the work to be done) of interactive system. A secondary objectives for the interface included the capability of examining the dialogue network created for effectivence and correctness and the identification of path fragmentation between nodes within the networks. In addition, dialogue network systems created with the aid of the interface system can be considered from three view points. The linguistic, psychological and the design view. The linguistic view sees dialogue network systems as a combination of production rules and network transition. The psychological view sees the interface system having the capability of constructing the user's model of the system. The design view sees the system as a high level tool for helping the designer to construct dialogue network systems. In chapter 2 we will be discussing the role of dialogue networks within the systems development process.

1.5 The Need for User Involvement and Support

The need for user participation in the system development process has long been recognised in the commercial computing environment and by many research and academic institutions. Lucas (1975) has argued that lack of user participation in the system development process contributes to system failure. Biggs, Birk and Atkins (1980) have pointed out that a lack of sufficient user interaction between the project team and user management in organisation, planning, scheduling, reporting or review will result in the creation of a system that belongs to the project team and not the user. Glasson (1985) reported that surveys of the techniques available for analysing and defining the information of user requirements carried out by Synnott and Grubber (1981) and Taggart and Tharp (1977) shows that the techniques are systems developer driven. Glasson discusses the work of McKeen (1983). McKeen has suggested changes to the process of development to include "implementing procedures to ensure meaningful and ongoing user participation throughout the entire development life cycle. This can often be accomplished with parallel activities such as assigning users the responsibility of user training, which the system designers attend for technical aspects such as programming." This approach was used at the author's own organisation, Massey-Ferguson. We will be discussing how it was implemented in Chapter 4. Glasson (1985) proposes a systems development model written from the user perspective with which the user can work. Such a model can be based on three major steps. The first step is to define the several user roles or user actors who need to be involved in the development cycle. The second step is to define several tasks that each user actor might need to perform during the system evolution. The third step is to develop and define the appropriate procedures, methods, tools or techniques needed in carrying out one or more of those tasks.

Damodaran and Eason (1982) have argued for the case against user involvement within the systems development process. It is argued that the user role is unnecessary for two reasons. First because the design team contains all the expertise it needs. The second is user involvement is costly and impractical within the time and resources restraints of the project. Damodaran and Eason report a Norwegian systems manager, Groholt (1980). Groholt argues that user involvement has rarely been successful because neither the designer nor user have

access to procedures suitable for this purpose. Ives and Olson (1983) report two areas of theory considered particularly relevant to user involvement - participative decision making and planned organisational change. The objectives of participative decision making is to increase inputs of subordinate into management decisions that are related to their jobs. Locke and Schweiger (1979) identified improved productivity and job satisfaction of users as major benefits of participative decision making. They found mixed evidence of relationship between participative decision making and improved productivity including improved decision making quality. However, they have found stronger evidence relating to higher job satisfaction, particularly increased acceptance of change and participative decision making. Following Locke and Schweiger (1979), Ives and Olson (1983) have argued that user involvement can be considered as a special case of participative decision making in which users and systems designers substitute for superiors and subordinates. As for the theory of planned organisational change, Ginzberg (1979), Schultz and Slevin (1975) and Zand and Sorenson (1975) have argued that implementation user acceptance and use of the new system is considered to be dependent on the quality of the implementation process. Ginzberg (1979) has argued that planned change assumes the change entails joint effort between the manager and the "change agent". Zand and Sorenson (1979) see the planned change as negotiation between the manager and "change agent". The designers/developers are the "change agent" and the manager representing the users. Zand and Sorenson (1975) have stated that "a process of open interaction and joint evaluation is expected to assume both high acceptance and a high quality solution. Ives and Olson (1975) support this view and report that the quality of the resulting system is dependent on the relationship between the staff and the user. Based on these notions and from descriptive models of user involvement previously proposed by Edstrom (1977), Lucas (1973), Swanson (1974) and Zmud (1981), the major components of the descriptive model can be constructed. The model shows two clauses of conditional variables. First the involvement roles; who should participate and what their role should be. The second class of variables represents the development conditions. Two classes of development conditions may affect user involvement. The first is the type of system being developed and secondly the stage in the development process. For certain systems, user involvement is more critical than for others. There are also systems for which user involvement is not appropriate.

Examples of such systems are systems requiring considerable expertise or where the final output is transparent to the users. A strong case for user participation is the development of decision support systems. The developer can only produce an effective decision support system with knowledge provided by the user and because acceptance is critical due to the voluntary nature of decision support system use.

As for the second development condition; Edstrom (1977), Franz (1978), Ginzberg (1981), Vanlommel and De Brabarder (1975) and Zmud (1979) have all proposed that there are stages of systems development in which participation may be more important than others. The type of user participation may vary from direct to indirect involvement. In the sections ahead, we will discuss the types of user involvement, measure of user involvement and finally user support.

1.5.1 Types of User Involvement

Damodaran and Eason (1982) have reported that there are different routes to user involvement in the systems development process and that these different routes will lead to different forms and degrees of involvement. One approach is based on designers retaining all decision making powers and users providing information requested of them. Another approach is that the whole concept of systems "experts" taking the decisions is under challenge. Groholt (1980) has proposed several forms of user involvement. They are:

(i) **One way involvement (systems development team driven):** This approach will have the user under the control of the development team. The development team will keep the users abreast of systems development through communication media such as meetings, newsletters. The users are used as consultants where the developers seek to elicit the expert knowledge from users. Training is a one-way process from trainer to trainee and are relatively safe and straightforward to conduct.

(ii) **User representatives:** In this approach, user representatives join the development team and supplement the team with business expertise. The major advantage of this approach

is that the developers have user representatives acting as co-ordinator between the users and the system after implementation.

Hedberg (1975) argues against this approach and suggests that the users will be influenced more by other designers than by his user origins and become less representative of users. Another problem identified by Damodaran and Eason is that the developers may not see the necessity to contact and consult other users. Furthermore, developers believe that if they get the support of the users representative, then they have the support of the rest of the user community. They suggest a solution to the problem identified is to have the representative as a part-time member of the team, but this may cause role conflict. Finally, many user representatives who are not trained in systems analysis techniques find it difficult to understand entity modelling and flow charts. The above user involvement discussed relates to the participation in detail design specification. However, there are wider business and organisation issues which require high level management decisions. These issues are usually addressed by steering committees and usually include user representatives. At this level, the user community see the strategic issues related to the project. Damodaran and Eason (1982) have reported that representatives in steering committees have raised a number of issues - lack of executive control over the design team and whose views tended to be ignored when they were contrary to design team objectives. The second issue is the membership of the committee. To ensure that views of all the users are presented, there is a need for wider user population to be represented, but frequently senior members of departments are expected to represent the user community. Damodaran and Eason argue that the user may represent his own view and the users, whilst he may become enthusiastic about the project. Uninvolved user community may remain apathetic or hostile. Another major problem highlighted by Damodaran and Eason is that new systems may result in redundancy, retraining, loss of job skills and demarcation. These issues are usually dealt with by most organisations through formal machinery and the relationship between this machinery and the steering committee can be a problem.

(iii) **Participative Design**

Damodaran and Eason (1982) have argued that the failure of one-way and representative techniques is that user expertise is not fully utilised and full user community commitment not gained. To address this failure, various forms of participative design are now being employed which seek to bring all users to the design process (Clausen (1978), Hedberg (1980) and Muford (1980)). Participative design proposes the notion that all users should be involved in selecting the job they will do, how the task will be allocated between them and the task relationship that will exist between the users and the computer systems. Further, it is only after the user agreement has been obtained, that the technical design and systems specification can be started. Users are then consulted during the design and construction cycle. Mumford (1980) reports successful systems development using this approach with users greeting the system with enthusiasm and commitment. Damodaran and Eason report problems with applying this approach. The first is that the designer and developer play a minor role in the early stages of the project due to the fact that users are specifying the requirements and that it can be a lengthy process. Secondly, it is difficult to manage a design process of this kind when there are large numbers of users. Thirdly, participative design requires skillful management by people who are able to provide information and concepts in the right form to be useful without precluding users from making their own decisions. This requirement is based on the user needs for help in conceiving of different forms of work organisation and to consider which is most appropriate to them. Finally, using this approach in a systems development project may have a very successful beginning and then can end in failure when developers and designers disagree to implement the agreed specification. Hedberg (1980) has reported a case study in which design problems led to a radical change proposed by users. To address these problems, Eason (1977) and Damodaran (1980) propose a continuous process of trials, consultations and pilot schemes.

(iv) **Users Design Expert Advise**

This approach gives the users the power and responsibility for systems design and the designer developers acting as advisors and consultant. This approach in shifting power to users is being brought about by legislation. Examples of this is that there are a variety of laws in Sweden which affect system design. Another source is union action. Trade unions argue that computer systems have widespread implications for their members and they are making major efforts to protect the interests of their members.

1.5.2 Measures of User Involvement

Measures of user involvement refer to general involvement in the systems development process. Ives and Olson (1983) have reported the three most common strategies for investigating user involvement. In the first strategy, a survey of different systems across multiple organisations. Research work carried out in this area includes work by the following researchers: Alter (1978), Edstrom (1977), Ference and Uretsky (1976), Franz (1979), Fuerst (1979), Guthrie (1972), Igershein (1976), Kaiser and Srinivasan (1980), Maish (1979), Olson and Ives (1981), Powers and Dickson (1973), Robey and Farrow (1982), Schewe (1976), Thurston (1959) Vanlommel and De Brabander (1975). In this strategy, users are identified by the Information Systems Manager and are administered a questionnaire containing measure of user involvement with reference to either a specific project or to user's involvement in all past or ongoing systems development projects. Ives and Olson (1983) argue that the disadvantages of this approach is that reliance on the Information Systems Manager to select appropriate users biases the results. Survey questions may be interpreted differently for different systems. Further, the choice of systems and the choice of users participating in the study are under the control of the researcher. In the second strategy, user involvement in a single system is investigated. Again, the Information System Manager identifies an appropriate system and a user - a specific system development or to the user's involvement in all past or ongoing projects. This approach suffers from the same problems identified in the first strategy.

In addition, the results obtained are rarely generalisable to other organisations and other systems. Ives and Olson (1983) report six case studies of single systems - Gallagher (1974), Lucas (1975, 1976), Sartore (1976), Spence (1978) and Swanson (1974). The third strategy reported by Blake and Ives is "controlled experiments" strategy. King and Rodriques (1978, 1981) conducted experimental study of a simulation of management decision making using three groups of subjects. The first group participated in the design of the systems and subsequently used it. The second group used the system, but did not participate in design. The third group served as control. In another experiment, Boland (1978) varied the method of user-design communication in a systems development process.

Ives and Olson (1983) have reviewed the results of these studies and measured system success under four major indicators - system quality, system usage, user behaviour/ attitudes and the user information satifaction. They have reported that studies investigating the relationship between user involvement and system quality provide little justification for user involvement. However, they report positive results for user involvement and system quality from work carried out by Boland (1978) and Gallagher (1974). Six studies examined the relationship between user involvement and system image. Swanson (1974) reported significant results, finding user estimates of "a prior involvement" to be related to "inquiry involvement". King and Rodriques (1981) found that user participation affects the nature of the usage, but not the amount. Lucas (1975) reports mixed results. Seven other studies investigated the link between user involvement and user's behaviour or attitudes towards the system only. Igersheim (1976) reported significant results. He found user involvement to be positively correlated with job satisfaction, job skill, job opportunity, job originality, job status and job salary in one or more of five organisations investigated. The concept of user information satisfaction can be traced to the work of Cyert and March (1983) report Ives et al, (1983). Cyert and March have suggested that an information system which meets the needs of its users will reinforce satisfaction with the system being developed for them. If the system does not provide the needed information, the user will become dis-satisfied and look elsewhere. Ives and Olson (1983) report that an information satisfaction measure was used in 12 of the 22 studies reviewed. Five found a positive relationship between user involvement and information satisfaction. Four other studies found mixed results. However, Ives and Olson argue that the

results obtained suffer from problems in either construct measurement or methodology which may have contributed to the positive result finding. Their view is supported by Powers and Dickson (1973) who have concluded that user involvement is crucial to system success. This is based on user managers' perception of their participation. No relationship was found between user satisfaction and the existence of a project team that includes user personnel.

1.5.3 The Case for User Involvement and No User Involvement

There is a strong convicition throughout the information systems development industry that proper user involvement will result in better systems. But which user should be involved in which task and to what end? There is a multitude of views on this subject and, as yet, no organisation has discovered a recipt for user involvement which guarantees success. Ives and Olson (1983), having reviewed studies carried out by researchers discussed above, have concluded the following:

Research on user involvement is not based on strong theory.

Empirical research has not convincingly demonstrated the benefits of user involvement.

The majority of studies on user involvement have been methodologically flawed to the extent that few conclusions can be made about user involvements relationship to system success.

In Chapter 2, we will be discussing the role of user involvement within the new systems development life cycle model in detail. The role will be discussed in the context of developing live systems development project in the commercial computing. In Chapter 5, we will review the results obtained from our approach to user involvement within the new life cycle model. The review will be based on the four major system success measures discussed above; system

quality, system usage, user attitudes to the system and the measurement of user information satisfaction.

User Support

Damodaran and Eason (1982) have reported that user involvement and support are closely linked processes which both influence the effectiveness of any computer application. They have examined the purpose of user support and have argued that its primary function is to compensate for the limitations and shortcomings of the system. The user has to learn how to initiate and terminate interaction. He must learn to interpret system messages and input to the system and output from the system. The second purpose of user support is to promote continued viability of human-computer interaction. Damodaran and Eason call this "evolutionary user support". They argue that this is complex and varied since it is concerned with an evolving computer system applied in a changing environment through the activities of a human user, who is also changing as a result of his learning from experience. Research into user support reveals a variety of ways of meeting the two major objectives of user support. Damodaran and Eason (1982) have identified training, documentary support and human support mechanisms as methods of user support. Training needs are the most commonly acknowledged user support needs. This is due to the fact that systems cannot become operational until the minimum requirements are met. Documentary support is widely recognised as a method of user support. It takes the form of a user manual or user guide. The document is written for developers and operational personnel to assist when problems arise. It is also written for users so that they can understand the system they are interacting with. There are three approaches to human support mechanisms. The first is the system-centre support personnel.

They are responsible for training users, resolving problems when they arise, providing assistance with hardware and software failures and developing further computer facilities. In some organisations, this group is known as "the maintenance team". The second approach is to use user-centred support

personnel called the "human-interface". This term refers to individuals who operate a system on behalf of other indirect users. The final method of user support is organisational representatives - a special category of human interface. Bank cashiers and airline reservation clerks are examples of organisational representatives.

As for the design of a user support system, Damodaran and Eason (1982) have argued that to design a user support system requires an appraisal of the range of support needs. These needs have implications for the design of software, hardware, documentation, human support and organisational policy. Furthermore, they have emphasised the close relationship between user involvement and user support. They state "There is tacit acknowledgement that the precise statement of relevant design criteria can only come from involvement of the user in the design process". In Chapters 2 and 5, we will discuss user support procedures employed within the new systems development life cycle model including users reaction to support provided. The discussion is in the context of providing user support for live systems project development and implementation in commercial computing environment.

1.6 Conclusion

The foregoing sections of this chapter have presented a broad discussion on information systems methodologies containing techniques and approaches needed to design, construct and implement powerful information and decision support systems to respond to users' needs. Most of the methodologies use the same well understood and proven techniques such as entity modelling, data flow diagrams and normalisation. Choosing the right technique is important and systems developers are interested in selecting a specific technique that will be appropriate to their own environment, address the problems that they have and enhance their productivity. Further, the advent of prototyping, decision support systems and special emphasis on the human-computer interface design have been causing problems for the rather rigid and orderly view of the systems development process, which forms the basis of most methodologies currently used. We have given a brief description and have discussed the advantages and disadvantages of six major systems specification and design techniques. Information engineering, Multiview - a framework for information systems definition, Jackson system development (JSD), structured analysis and design, structured requirements definition techniques and finally prototyping. Each technique has its good and weak point. Each using its own conceptual framework with its own philosophy. These techniques are used in the real world to address problems encountered during systems development to satisfy the user communities. Maddison et al, (1983) have argued that most methodologies rely heavily on intuitive approach to analysis and design. They are described in terms of "what" should be done and rarely in terms of "how" they should be done. Two practitioners of the same methodology often derive completely different solutions, models and entity diagrams by applying the same steps. Maddison et al, (1983) have argued that the extent of the failure of any specific methodology depends on the skills and experience of the practitioner. Downes (1989) reports that time and time again this identifies the basic needs to understand and control the data.

Experience at author's own organisation, Massey-Ferguson, supports these views. Further, with shortage of skilled resources available to practice these methodologies, many organisations have been encouraged by the suppliers of these methodologies to turn to their end user community.

It is claimed that their methodology can be used by end users with no specific skills in data and functional analysis techniques. Our experience suggests that most of the methodologies do require skilled practitioners in data and functional analysis. In addition, our experience shows that these techniques aim to provide guidelines for analysis, design and construction of systems and do not address the fundamental area of project management. In our view, the systems ultimate success depends on skills required for managing the project.

As for the evidence of a relationship between end user involvement and system quality and acceptance, Ives and Olson (1982) reported no strong evidence to relate the two. They have, however, reported that common sense notions discussed previously can be supported by strong theory and can be proven by carefully executed empirical analysis. Our experience at Massey-Ferguson shows that the end users have always participated in the development process. The participation, however, has been limited to early stages of the systems development process. What is required in our view is a formal approach to user involvement, identifying appropriate stages where user involvement is beneficial. The extent of the involvement should have the objectives of the final system developed and implemented to reflect more accurate user requirements and to have a high rate user acceptance. Further, our experience shows that the more accurate information supplied to the end user community via friendly human-computer interfaces, the more information is needed. It is also generally accepted at the author's own organisation that information is the key to increasing an organisation's ability to act effectively, anticipating customer needs and competitor moves. This awareness has created user requirements for the freedom to access information held in corporate databases.

The freedom to enquire and analyse the information will itself give rise to the need for more easy to use decision and knowledge-based management support systems. However, for these systems to be effective, they have to be integrated with the mainframe corporate computer systems in our view. Doukidis et al, (1989) have proposed that the technology based approach of the knowledge-based management support systems have to be integrated with the human activity systems they serve. Doukidis (1989) reports on the emerging trend of knowledge-based decision support systems towards practice through applications in real situations. Hawgood (1985) stressed the need for providing the managers with the capability of constructing their own decision support systems.

Other important considerations in the design and development of information and decision support systems, which in our view have not been addressed explicitly by methodologies discussed previously, are the three areas of distributed systems approach, systems security and controls and the application of productivity tools within the life cycle. . The objective of a distributed information system is to match computer resources with organisational structure and needs. The major issue in the management of a distributed information system is control, which must be exercised at executive and technical levels. Cost reduction, reliability, responsiveness and extensibility are the benefits of distributed systems. The notion of distributing the computing facilities of an organisation to match its organisational structure has been accepted since the late 1970's at the author's own organisation, where a distributed information system is achieved through the synergism of data communications and dispersed computer resources. The attractive notion of placing resources where they are needed and allowing them to communicate is a potentially powerful means of operating a total computer facility. We believe that guidelines are required to assist the designer and developer in applying this important notion. Another important area to be considered is that of data secyrity this subject has been given considerable attention by many commercial organisations, computer manufacturers, software companies and government agencies. Threats to data security have been identified and effective counter measures have been established. Techniques for security include computer hardware features, software in terms of programmed routines and manual procedures, as well as the usual physical means of safeguarding the environment with security

personnel, locks, keys and badges. What is needed, however, is guidance as to where attention should focus on the key areas in a computer environment that have proven in the past to be critical to system control. Information systems controls form part of the entity's internal control system. Internal control comprises "the plan of organisation and all the co-ordinate systems established by the management of the enterprise to assist in achieving management's objectives of ensuring, as far as practical, the orderly and efficient conduct of its business, including the safeguarding of assets, the reliability of accounting records and the timely preparation of reliable financial information", Rosen et al, (1986). For computer control guidelines to be truely effective, it was to be integrated with the systems development process. It should be clear by now that the ephasis must always be on meeting business requirements according to business priorities. It is true however, that work must sometimes be done which does not lead to immediate business benefits, but even this can and should be justified in business terms. An important aspect of project justification is making sure that dependencies are thoroughly understood. Supporting projects, such as those developing the information system infrastructure, can only be justified in this way. It is seldom that the first approach to solving a particular business problem is the only one possible. We have choices in technology, in selection of development approach and indefining the scope of the problem.

We set our goal in this program of research to propose a new model for the systems development life cycle, allowing the management of systems project through the whole cycle of activities needed to develop information including decision support computer systems. A model which supports cross-communications between different skill types and stopping the serial utilisation of user to analyst, analyst to designer, designer to application programmer. A model that allows for the study of user interface issues. A model that brings different skill levels to play their part as the system evolves from idea to installation. A model that supports design environment, that addresses design issues arising out of the task while the system addresses the technology on which it is built, and interests of users who use it. Additionally, a model that delivers significant productivity gains to appropriate stages within the development cycle. The testing of the model is carried out through involvement in real world problems. Real world problems at author's own organisation, Massey-Ferguson. Wilson (1984) reports that there is a

distinction between problems in the real world and problems in the laboratory. In the laboratory, the analyst has the freedom to define the problem and to control the environment. Thus he can allow certain variables to affect the process under investigation and to constrain others so that they do not. In the real world, such freedom does not exist, the analyst has to accept whatever influences the situation under investigation. The method we have chosen is "action science". Argyris et al, (1985) described action science as an enquiry into how human beings design and implement action in relation to one another. It is a science of practice. Action science is based on the observation that science in becoming experimental has itself become a model of directed practical doing.

CHAPTER 2

SYSTEMS DEVELOPMENT PROCESS

AND A NEW MODEL

2.1 Information Architectures

The purpose of architecture in information processing is to ensure that we have a sound basis on which to proceed. We would never consider starting a building without a clear picture of the results; even before that we would need a clear understanding of the purpose of the building, and of any likely requirements for extension. In the building industry it is expensive to demolish a building and replace it simply because the requirement has changed somewhat. This is equally expensive, perhaps more so, in information processing. We need a stable foundation,and a corporate information architecture can provide this. Experience at the author's own organisation has shown that the data required by the organisation by virtue of the nature of its business, and the things with which it deals, is much more stable than the process performed on the data. If then, we use the entity/data model as the basis for planning, we shall not have to make changes unless there is a fundamental shift in the perceived mission of the corporation. So long as we stay at ahigh level of abstraction, we can also expect a fair degree of stability in the corporate function model, and we can use this plan, monitor and control application development. An information architecture also gives us a basis for reviewing the current position, so as to establish significant deficiencies and, perhaps more important, identify areas where existing systems are adequate and need only minor enhancement. It can also enable us to identify areas of data and functions which can safely be treated in isolation; this will enable the rapid development of separate solutions to some urgent problems without prejudicing the planned system structure. Further, we shall be able to see the consequences of various possible sequences of development and thus to make sensible phased development plans. We propose that information architectures must cover two main areas: data architecture and applications architecture. In both cases the architecture must be practical; while based on the results of high-level business analysis, incorporating the business priorities and the awareness of what is practically possible, identifying the need for methods of analysis which will allow us to focus on the real business needs. As we have seen, an important aspect of this analysis is the consideration of the data and function required to support the business needs. Before proceeding to the practical aspects and discussion on a specific technique for the data and function analysis, it is worthwhile considering briefly what data and function analysis is. Data analysis is the process of analysing and documenting the data associated with an area of business and to build a business data model of the company, an enterprise or organisation.

The objectives of data analysis can be summarised to be:

First to design databases which will be less subject to change than has been the case in the past and which will consequently reduce maintenance. Although data analysis is not necessarily associated with database software, the two become closely related as data analysis itself results in a structured representation of data where the records tend to be small, but the number of record types large. Downes (1989) argues that this is a general approach which may be termed the "database rationale". Using database rationale helps to achieve less data duplication, sharable data, consistent data, control over data validation, better enquiries, data independence from process changes and finally better security and recovery of data files.

The second objective is to provide complete, consistent data descriptions, independent of specific database management software and the associated performance comprises resulting in more productive application development and improved end user query facilities. Data dictionary is a major aid in documenting the output of data analysis.

Based on these objectives, data analysis attempts to establish the data requirements and the relationships between data elements and then to analyse the processes that use the data argues Downes (1989). In function analysis, the main function of the business, company, an enterprise or organisation is identified. This main function is then broken down progressively into subfunctions. The data or systems analyst will continue the process until a sufficient comprehensive level is achieved. This process is called functional decomposition. In Chapter 1, we discussed various data and functional analysis techniques. In this section, we will be describing a specific method for analysing data and function, the CACI method. In Section 2.4, we will be showing how the data and function analysis techniques of the CACI method fit into the new systems development life cycle model. In Chapter 4, we will be discussing the way the method is used within the new life cycle model in live commercial projects at the author's own organisation, Massey-Ferguson.

2.1.1 Principles of CACI Method

CACI method employs six important principles. They are:

Firstly, the principle of separating the data and function. The method emphasises the need to separate the treatment of the data from that of the functions and build systems with a foundation of data, rather than application. This separation is done in each phase from strategy through to construction. However, in each phase, the separate results are combined for cross checking. By adopting the approach of a change from providing application orientated system to data orientated systems two benefits are obtained. The data is subject to strict definition, design and control. Another benefit is that the application to use the data may be controlled less strictly (e.g. end user query language may be used).

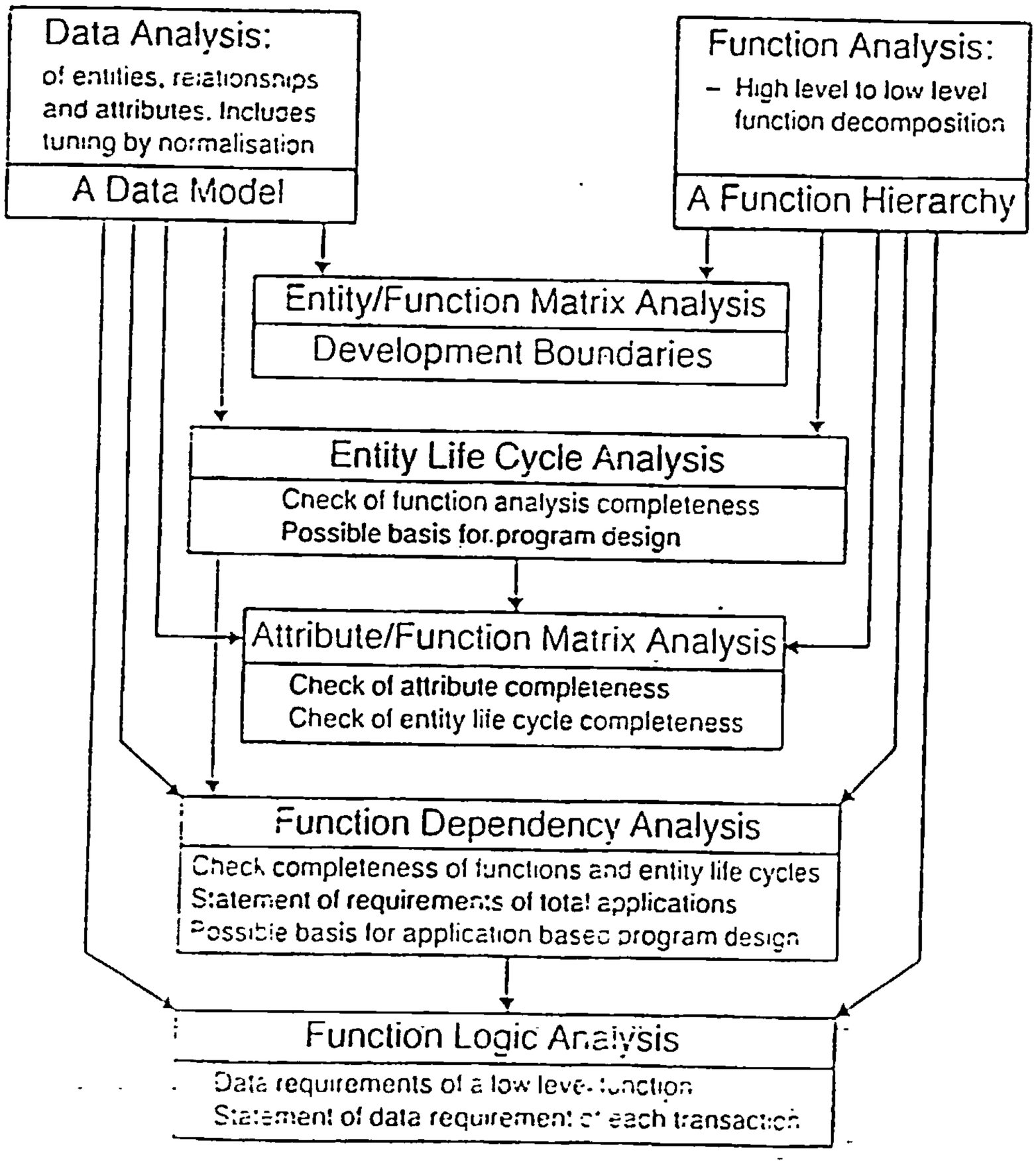
Secondly, the top down method of analysis. This method of analysis starts with the production of wide, strategic views of an enterprise's current and planned data and functions. At this level, top management can be involved in approving the understanding of the business as shown by the analysis and in selecting which areas need further investigation and/or new systems. As the area being studied gets smaller, the complete view is split up into self contained subsets, the level of detail increases and the level of management decreases. The most detailed view of the enterprise, usually obtained from user experts, are checked with the broader views obtained earlier. The main benefit obtained from this approach is that the whole process of successive narrowing and detailing of the areas being analysed ensures that the detailing of the areas being analysed ensures that the detailed development work fits together in the framework that the top management require. The effect of any alternations to the development plan or to an individual development can then be measured against the overall plan.

Thirdly, the principle of making each stage the use of pictorial models developed for previous stage. The advantage of using models is that they can show just the level of detail required. The more strategic the view, the less the detail. Another advantage is that if a model is produced at each stage of the top down approach, those analysing the next level down can all be seen to start with the same understanding of the context they will be working in.

Finally, the other principle proposed by the CACI method is the checking out of the results with the bottom up method. However, there are major faults with bottom up analysis. The amount of detail for an enterprise of any reasonable size is vast and the task of analysing all the data and processing is far too big. There is also no easy way to check that all areas are covered. This depends on the adequacy of existing documentation. Further, examining the current details understanding of what it is trying to do. The how will change as a result of the development.

2.1.2 Overview of CACI Data and Function Analysis Techniques

There are three types of techniques proposed by the CACI methodology. Techniques that analyse data, function and techniques that cross check the results of data and function analysis. The diagram below depicts the overview of CACI data and functional analysis techniques.



The purpose of entity analysis is to determine the fundamental nature of an enterprise's data and then to organise and document all the relevant facts concerning this data. This analysis process identifies all the relationships between the entities. Any ambiguities, inconsistencies and lack of definition will be formally recognised. Discovering the fundamental nature of data requires very careful work. Sufficient data about the enterprise's data must be collected so that the systems designer(s) can choose and/or design the appropriate hardware and software without having to perform the analysis task again. In this process, the emphasis is on the need to seek out implementation(s), only then the designer can make reasoned decisions between differed designs. The most important result of doing data analysis is the improved understanding it brings of the enterprise's data. To ensure that such understanding is reached and as a result of the process, four major outputs are obtained from entity analysis process.

First is the pictorial model of the entity types and their inter-relationships. The model provides a good way of raising such questions of one or more than one and the model's backup documentation can then be used to analyse the data requirements of the functions discovered during the functional analysis. CACI methodology argues that the cross checking between the two will lead to improvement in both the entity and function model. Iteratively improved until they can support all the requirements.

The second result is the detailed documentation of entity, relationships and attribute types. The analyst will document the entity definition. The definition is important as definition of an entity is "anything of significance to the enterprise about which information needs to be known or held", "any named thing". However, it is important to note that an entity can vary from enterprise to enterprise. Other definitions documented are the entity occurences and their entity type. Forms or a data dictionary are used for documenting the information obtained. Examples of information recorded and documented include the following: entity name, synonyms, definition, identifying attributes, volume details - when documenting volume details it is important to distinguish between permanent and transient entities, ownership, creators, deleters and entity sub- and super-types. As for allocating attributes to entities, one of the attribute rules is that an attribute can only belong to one entity. Thus it is necessary to allocate each attribute to one entity. The processing of doing this is alled "normalisation" and the attributes processed are described as being "normalised". There are two techniques for normalisation. The first is by

Inspecting the entity model. In this method, each attribute is examined in turn and the fundamental question is asked "to which entity does this attribute belong?" If the question cannot be answered, then there are three reasons for this. First, the entity model is wrong, the second is the attribute is incorrectly defined and finally both the entity model and the attribute definition are wrong. In all cases, the appropriate action is taken. The major benefit of this technique is that the most important attributes can be handled first and lesser ones added as and when investigation warrants.

The second technique is what is called the relational approach to normalisation. The technique was developed by codd, devised so that the resulting data structures could be mathematically normalised (relational algebra). Normalisation by the relational procedure consists of three major steps. From un-normalised data as found, the repeating groups are removed which results in the "first normal form". The next stage is to ensure that each attribute depends on the whole identifier, which is the "second normal form". The third step is to remove all attributes which really depend on attribute(s) not in the identifier at all, which is the "third normal form".

2.1.2 Analysis of Function

We said earlier that there are two broad activities within data analysis - entity analysis and functional analysis. Entity analysis was discussed in a previous sub section and broadly it concentrates on the things and information the business needs to operate. Function analysis on the other hand concentrates on the functions the business carries out in order to operate. These two activities are normally undertaken in parallel, as complementary activities each providing the other with a better insight into the business. The primary task of function function analysis is to start at a high level business function and breakdown the high level function into progressively more detailed functions. This process is known as the "function decomposition". It is presented pictorially as a function hierarchy and each function is formally documented. The function hierarchy can be checked for completeness and all the functions in which an entity type is involved are summarised. The result of this process is, therefore, the function model of the business which complements the entity model produced by entity analysis. Other important outcomes of function analysis are: a good understanding of what the enterprise does before any design takes place, a firm base for selecting which functions to

implement including system architecture and program and database design. Further, we will have an improved entity model and important pictorial representation of interactions between functions and data.

2.1.3 Entity and Function Model Interaction

After building the function hierarchies, it is necessary to examine the entity and function model in conjunction with each other, with the objectives improving the analyst's understanding of both data and function. This examination may lead to redefinition of some entities and functions.

There are three techniques for analysing and studying the interaction of function and data. The first technique is function logic analysis. This involves drawing a function logic model for each elementary function showing the entity types and relationship types needed to support the function. Function logic analysis provides estimates of entity and relationship usage that are essential for database design and provides a firm basis for transaction design. Other techniques are function dependency diagrams and the attribute usage matrix. Function dependency diagrams depict the entity types and attributes which are created or modified by each function and used by subsequent functions. The attribute usage matrix summarises how much attribute of one entity type is used by each function.

2.2 Methodology for Decision Support System Development

Arinze (1989) has argued that methodology is the key to decision support. Methodology will enable the use of decision models in a systematic way and the selection and tailoring of relevant technologies and techniques to specific decision making situations. Arinze argues that a methodology for creating DSS needs to be iterative. This is based on the fact that users of information systems are better at specifying changes to an existing system than specifying a proposed system requirement. Experience in developing DSS at the author's own organisation supports this view. Thus a DSS methodology will lean towards the experimental approach (prototyping). In Chapter 1, we broadly discussed development and design approaches including prototyping for DSS. In this section, we are presenting a methodology for DSS development and discuss experimental approach in the design of DSS. In Section 2.4, we will see how this methodology fits within the new life cycle model.

2.2.1 Stages of DSS Methodology

The first area that needs to be considered is the "technology" to be used. The designer/developer of the DSS and the end user require an appreciation of the technical feasibility and costs of relevant models and systems interfaces for extracting information from the mainframe systems in order to make a pragmatic decision and produce a proposal for approval. User involvement is important at this early stage to support the central issue of DSS, that is the output from DSS should assist the decision maker in order to achieve his objectives.

Arinze (1989) have proposed six stages of an outline methodology. They are:

Entry: In this stage, the analyst/designer/developer of the DSS is introduced to the problem. It is important at this stage to identify key decision makers, the environments, resources and constraints. Two important aspects of decision making situation will be identified during the investigation - the internal and external aspects. The internal aspects consist of cognitive style, attitudes and beliefs, existing objectives and existing constraints. The external aspects include the description of the three major environments. The primary where the decision making takes place, in addition tasks and internal/external communication to other entities are described. The secondary environment entails the

description of the management decision hierarchy, policy guidelines and controls that are proposed by this hierarchy. The third environment is the "wider" environment which includes legislative, technological, social and economic environments. Each involving a description of effects of these on the decision making environment.

- **Description:** This stage is concerned with describing the organisational, information and decision elements of the decision making situation. The main activity is to describe fully the organisational structure in terms of hierarchy, communications and controls in relation to the key decision maker. The effort at this stage is to define transformations and flows in the current decision making situation. Decision making and information processing are the key transformations that will be identified.

- **Modelling:** The objective in this stage is to produce the DSS model using systems concepts discussed previously. The study of the information required will decide whether the DSS will be standalone or if interfaces to other systems are necessary. The output of the modelling stage will be the description of the DSS model and information in terms of structure, content, flows and other relevant characteristics. Other information contained in the DSS includes frequency, accuracy, content, structure, timelines and relevant process. Data dictionary can be used to record the information.

- **Design:** The objective of the design stage is to provide a full description of the data and processes that are implied and derived from decision enquiries. Arinze identified two steps for the design stage. The first is the process of defining the data and process models. They can be both represented by the entity relationship model and data flow diagrams employed by structured methodologies we discussed in this chapter. The second stage is to produce a DSS logical specification. Details of procedures, data variables and interrelationship needed to support the logical model are described. The output from both steps is, therefore, the logical model and a system specification for the DSS.

- **Experimentation (prototyping):** Prototyping is viewed as a vital step in the development of DSS by Arinze. This is based on the argument that DSS is harder to conceptualise than other more structured systems. In Chapter 1, we discussed the subject of prototyping and prototyping for DSS.

- **Implementation:** The final stage of the methodology proposed by Arinze is two types of implementation process. The first type is a DSS that is implemented in the conventional I/S environment. The second type is "throw away" DSS, the system will be used in a one off manner and then discarded. In the event of continued usage, the DSS will be required to evolve with environmental changes.

2.2.2 Experimental Approach in the Design of DSS

Our interest in this approach stems from the notion that an experimental approach to information systems development may have a role to play in DSS design. Guitierrez (1989) has suggested that experimental approach is a natural fit with DSS type application based on notions discussed previsously and that it will be more effective if applied at the early stages of the development. Guitierrez (1989) reports that the concern of the experimental approach in the design of DSS is in establishing the conditions for learning and communication about decision processes. Two major objectives identified by Guitierrez. The first objective is the pursuit of a change in attitude in the performance of development tasks and processes and consequently in the use of the system. The second objective attempts to create conditions in which learning and fact finding can take place. The experimental approach, therefore, allows for Guitierrez (1989):

- Conditions for learning among the parties can be established.

- Communication problems between user and DSS can be addressed.

- A tangible working model can be used to satisfy the learning and communication needs which are imposed upon a particular situation.

To understand the role of experimental approach to information systems in the design of DSS, we need to briefly discussed the issues involved in the experimental approach to information system development, the relationship between this approach and different problem types and finally the role of this approach in the context of DSS requirements analysis and design.

To understand the role of experimental approach to information systems in the design of DSS,

we need to briefly discuss the role of an experimental approach in the context of DSS requirements analysis and design strategy formulation. Guitierrez (1989) has argued that if the role of an experimental approach to DSS design can be found in the early stages of project developments, then what is involved in the analysis of information requirements experimentally? Determining decision process and information requirements involves four main tasks. The first is to observe, select and report on decision processes and organisational tasks. The second is the interpretation and transformation of the processes into an understandable means of communication. The third task is the verification process of the requirements. The idea is to verify that such requirements have been incorporated into an experimental working model. Finally, assuming that the support system will be satisfactory to its users. These tasks form the base for Guitierrez's model of requirements analysis experimentally. He proposes that requirements analysis can be sub-divided into four separate sub-processes. They are:

(i) **Data collection and elicitation of information needs:**

This involves the observation, recording by the user of decision rules, norms and processes that take part in the execution and flow of decision tasks, including the necessary data structures. In the context of an experimental approach, the process of observing or eliciting information allows for exploring and clarifying the needs of users, especially those for which a complete set of requirements cannot be specified. Information needs collected should also include characteristics of decision support situations. Guitierrez argues that the traditional mode of analysis may not be enough for decision support problems where the data collection process may have to be an experimental one. The traditional mode of data or information collection normally is associated with the revision of existing documentation, interviews, meetings and data collection forms.

(ii) **The abstraction process and modelling:**

This process is concerned with arranging the collected data including the interpretation of the relevant data and information so that a specification understandable guideline is produced. In the context of experimental approach means, we are producing a working model and refining the model to finally producing a set of system specifications.

(iii) **The verification process:**

This process concerned with ease of understanding and communication about an information system's needs. The process verifies the working model for reliability in relation to the abstraction process discussed. The process deals with this in two ways. First it verifies the model for the statement of requirements and secondly it deals with the transformation carried out during the experimental stage - how the specification incorporates results obtained from a working model.

(iv) **The assurance process:**

The process attempts to prove the specification through the use of the model in the real world. Further, it attempts to ensure that the end product will be effective when implemented and assess the impact involved in determining which aspects in the system's environment need further attention in order to ensure effective implementation.

2.2.3 Knowledge Acquisition and Knowledge Representation in DSS

We have previously described DSS as systems having the objectives of supporting the professionals in the execution of the day- to-day decision making process and that knowledge based techniques can enhance the decision making process. A DSS requires not only that a variety of pertinent data to be made easily accessible to decision makers, but also that the implications of such data may be readily explored by such decision makers. What is required to provide support for decision making and problem solving activities is the elicitation of relevant knowledge and knowledge representation in the DSS framework so that the DSS is capable of answering "why?" as well as "what if" traditionally available to the decision maker. In knowledge based DSS development, much attention has been given to problems of knowledge acquisition and representation. The two fundamental issues concerned with knowledge representation need to be addressed. First, what knowledge of a problem domain is to be represented and secondly, how to encode and reason with that knowledge. There are several classes of notations currently employed for knowledge representation. The three most widely used methods are rules, semantic nets and frame based methods. Rule based methods consist of facts and rules. Frame based representation uses a network of nodes connected by relations and organised into hierarchy. Each node represents a concept that may be described by attributes and values associated with the node. Nodes low in hierarchy automatically inherit properties of higher level nodes.

As for knowledge acquisition, researchers have identified three major methods for knowledge acquisition. Learning by being instructed, learning by induction from examples and learning by observation and discovery. We discussed these methods previously Moubarak (1989). These methods highlight the fact that the central problem of knowledge acquisition is the transferring knowledge from expert into the systems knowledge base. This is carried out by specialists known as Knowledge Engineers.

Edmonds et al, (1989) report that "much effort is directed towards identifying and interpreting expert knowledge from interviews with experts followed by mapping directly from verbal data to a chosen from representation. Knowledge acquisition is carried out at very beginning of the need to understand how to structure the knowledge and the inferencing mechanism." At the most basic level, knowledge based technology offers techniques to computer professionals that will aid application development, maintenance and ease human-computer interaction. On the broader scale, it invites more human-computer interaction than traditional systems

solutions, at the level of system designer and the end user. The building of a knowledge based system for everyday use is a process involving considerable amount of trial and error. Designers should consider the following as major features of any knowledge based decision support system:

Major features:

- Expert knowledge should not be forced into an inappropriate format. The format for the knowledge should allow general rules and complex facts.

- Alteration and enhancement of the knowledge base created should be made simple by providing easy to use human-computer interfaces.

- The system should be able to justify and explain its actions. Any system with problem solving capability should be able to decide whether some item of information is missing, true or false. Keravnou and Johnson (1987) stated that "a computer system that appears to lack commonsense knowledge will not be trusted to reach valid decisions on critical matters". The system should have the capability of summarising the rules and explanation instead of just listing the rules recorded within it.

When developing knowledge based DSS we need to address these issues and consider human-computer interface issues arising from the development and use of knowledge based DSS. We need to consider knowledge acquisition, representation, dialogue structure and other fundamental facilities such as volunteering, explanation and reasoning including intelligent handling of data by integration of commonsense reasoning proposed by Keravnou and Johnson (1987). task for the knowledge engineer is to understand the problem that is suitable for computer application development first and then address the question of what aspects of the problem will lend itself to knowledge based techniques. Once this task is completed, the knowledge engineer can then proceed to build his system. In the sections ahead, we will consider guidelines for developing knowledge based DSS and stages of knowledge based DSS development methodology. These guidelines and stages will form the base for our new systems development life cycle model described in section 2.4, which encompasses information and decision support systems development including knowledge based DSS.

2.2.4 Guidelines for Developing Knowledge Based DSS

Before we develop knowledge based systems, we need to have a strategy. There are four strategies that can be considered either individually or in combination with each other. These strategies include the use of knowledge based expert systems technology in the following way:

The four strategies:

(i) **As a software tool aiding the computer professionals in the area of application analysis, design and specification:**

A number of methodologies exist for defining and analysing application requirements. We have in the preceding sections of this chapter discussed a specific method for data and function analysis. Software tools exist for analyzing, displaying, as well as editing a set of application requirements related to the design of databases and programs operating in a specific computing environment. Once the design is complete, a number of programming code generators are used to convert the design to a specification and onto running code. To date, no software tool is available to provide a full solution to convert a design to a running program code. However expert systems technology provides the potential for an application software tool which can bridge the gap between the analysis and the specification stages with the systems development process. Such a tool will provide fundamental productivity gains for the systems designers and applications programs and hence address the productivity issues highlighted in chapter 1. Another area of interest for the knowledge based systems technology within the development process is the common problem of program conversions from one computer system to another. Every management services department needs at times to convert an application system written from one specific computer language to another, or from one operating environment to another. A knowledge based system having the capabilities described previously, should make the task of converting programming code less labour-intensive. An example of a software tool capable of this conversion is the IBM COBOL structuring facility (COBOL/SF). Another example is the design and development of an expert system to assist in the analysis of crash dumps by Jackson (1983).

(ii) **As a tool to improve conventional mainframe application:**

A decision to focus on this strategy allows the introduction of knowledge based systems technology within well understood current application. This provides a low risk approach to knowledge based systems development within the current management information services. An example of this type of system is a system developed by IBM called "YES/MVS" described Moubarak (1989). The system controls the operations of large computer complexes.

Another area of application for knowledge based systems development is human-computer interface to the mainframe databases using natural language ordinary english access to these databases. An example of this type is "INTELLECT" a product from the Artificial Intelligence Corporation. This product provides direct english natural language access to IBM database management systems.

(iii) **As a tool to develop systems, on micro, midsize or workstation application:**

Symonds (1986) described a knowledgeable application as a a computer program that performs decision support functions in a way that exhibits some of the properties associated with human intelligence. These types of systems are easy to use systems that non-computer professionals can work with. Small knowledge based systems building tools are, in fact, comparable to spread-sheet packages - for example LOTUS 1-2-3. Spread-sheet packages allow the end user to enter knowledge and lets the application manipulate it.

(iv) **The fourth and final strategy is to implement major new strategic applications that will impact the way the organisation does business:**

This is a high risk approach that has the advantage of increasing profits and competitive gains and the disadvantage of requiring the resource necessary to develop such systems.

The requirements for an expert:

From what we have discovered so far in Chapter 1 and this chapter, it seems that

knowledge based systems development identifies an important requirement which is the existence of genuine experts in the relevant field of the application. These experts are people generally acknowledged to have an extensive high level of expertise in the problem area and they generally agree on the solution proposed. These experts usually have expertise needed in many locations or may be needed in hostile environments. They have scarce expertise. The other requirement for knowledge based systems development is that the problem areas investigated have the following properties:

The properties:

- The problem solving situations where humans suffer from cognitive overload. Knowledge based systems are well suited for tasks requiring cognitive skills and not physical.

- There are many combinations of relevant information to be evaluated. Many humans cannot simultaneously manipulate all the information available to them to obtain optimal solutions. When examining the information, it is difficult to define exhaustive relationships among the different sources of information and the corresponding conclusions to be reached. These relationships are more easily represented in terms of rules, each of which draws its own conclusion from a small subset of the information.

- Many problems encountered by humans are poorly understood this is due to continuous modification and enhancement of the tasks to be performed as a result of user feedback or changing requirements. The knowledge systems approach is well suited to a problem that can be solved naturally by manipulating symbols and symbolic structures. Waterman (1985) has stated that most real world problems require symbolic reasoning. Mathematically-oriented problems requiring numerical methods and algebraic techniques are usually not suitable for knowledge base systems.

- The problem area investigated is heuristic in nature; they require rule of thumb to achieve an acceptable solution. This is perhaps the most important difference between the traditional information computer system and knowledge based systems where an algorithmic approach is used rather than rule of thumb.

2.2.5 Stages of Knowledge Based DSS Development

Suppose that the problem meets all the criteria discussed in the previous section. What steps are required to build the knowledge based system. Knowledge based system builders have found that an evolutionary development is the most effective way to develop such systems. However the inherent complexity of expert system development process requires well defined steps assisting the system builder.

The first task a knowledge engineer must perform is to identify and understand the problem domain and represent the problem in knowledge systems terms; i.e design the component of system. These systems consist of a knowledge base, which embodies a human expert knowledge and a consultation system which enables the user of the system to make use of the knowledge by means of question and answer session. The knowledge base of a typical knowledge based system consists range of different elements. The major elements are: entities, facts, rules, explanation, reasoning, knowledge acquisition and possibly natural language translation facilities. The process of designing and developing an effective knowledge base for a knowledge based system involves many stages. They are:

(i) **The breakdown of a problem into manageable sub-problems:**

Waterman (1985) argued that a problem must not be too easy, it should be difficult, and it should be a serious problem in a domain in which it takes a human years of study to achieve. Waterman went on to argue that the problem should have the proper scope. It should be sufficiently broad to ensure that the problem has some practical interest.

A problem that is too broad or general is not suitable for successful knowledge based decision support systems development.

(ii) **Design a method for acquiring the knowledge from the experts:**

As we stated previously, one of the major components of a typical knowledge based system is the knowledge base and it is clear that they are useful only insofar as they contain knowledge. Walker (1986) has argued that for a human it is an open question how much knowledge is innate and how much is acquired. For a computer, however, the answer is clear; all knowledge must be acquired. Moubarak (1989) described three methods of knowledge acquistion by which a computer can acquire knowledge from the expert. They are basically learning by being instructed, learning by induction from examples and learning by observation and discovery.

(iii) **The identification of solutions in terms of the knowledge representation aspects of the facts and the rules:**

To use the knowledge contained in the system knowledge base, we must first choose a way of representing the knowledge. As we mentioned previously that there are many knowledge representation schemes. Johnson and Keravnou (1986) described four predominant schemes; predicate calculus, associative networks, frames and rules. They went on to argue that we need to distinguish between knowledge representation schemes and knowledge representation language schemes. In section 2.3 we will be proposing the use of transition networks as a tool for knowledge representation for certain types of knowledge based DSS.

(iv) **Design the facilities to record the knowledge acquired:**

A software tool is required to record the knowledge representation scheme used to build the knowledge base. To develop this software tool requires time, money and specialist computer resources, all of which influence the choice of a tool. As we stated previously two choices are available to us, the use of shells or special programming languages. Examples of both types of facilities were given Moubarak (1989). In the section 2.3 we will be discussing the potential use of the interface system developed in this program of research Moubarak (1989) as a tool for recording the knowledge represented in form of a transition networks.

(v) **Design the Human-computer interface for the knowledge based system:**

Kidd and Coper (1983) have argued that the effectiveness and acceptability of an knowledge based system is critically dependent on its human-computer interface. There are many parameters which need to be considered when designing the human-computer interface for an system. These include knowledge acquistion and representation, consultation, volunteering, explanation, dialogue structures and reasoning strategy. We have discussed the area of knowledge acquisition and representation previously. Explanation, reasoning, consultation, volunteering are important features of expert systems and are badly implemented in most systems currently in use. To make these systems more usable and to extend the use of such systems in critical applications, systems builders need to incorporate these features.

Explanation:

Explanation is important and knowledge engineers and systems builders should investigate the end-user requirement i.e. what the end-user wants to be explained, and the background of the end-user asking for the explanation (whom the user is?). A typical example would be the explanation provided by the system for a GP and a patient. The GP will be expecting explanation in the form of a medical language, whilst the patient would be totally confused with medical terminology. Hence the functions and tasks to be performed by the system should be defined before the style of the explanation is designed. Hughes (1986) has argued that the classification of question types and the awareness of the user's environment, can promote their thinking about different explanation types. The approach taken by Hughes consisted of adopting the existing ways of classifying questions to produce an approach that is of use with knowledge based systems developments. She divides questions up into the broad camps of temporal - those which relate to cause and effect process, and atemporal - to do with filling in slots in a description and understanding concepts. She also splits up questions concerning the state of something, and an act which has occurred. She observes conventional "how" and "why" identifies only two question types amongst many which the user can ask. Alexander Ian (1986) proposed the use of "which" as well as "how" and "why". Other facilities available to aid explanation are question forms and the use of diagrams which can offer better explanations than text.

Reasoning:

As for reasoning strategy, Keravnou and Johnson (1986) have argued that any intelligent problem solving system should be able, given the known data on a case, to decide whether some item of information is true, false or unknown. They went on to argue that to have an system adequately incorporate commonsense knowledge, an explicit representation of structural and strategic knowledge is needed. In order to do this, data subject classifications, attribute information for data subjects and inference networks of data have to be considered. They identified four reasoning strategies: a matching strategy operating on the attribute information, an inferencing strategy operating on the data inference networks, a generalizing restricting strategy operating on the data subject classifications and a default reasoning strategy operating on the default information for attributes. Based on these strategies, Keravnou and Johnson have designed "Decide-Status". Its architecture is based on a separation and explicit representation of the strategic and structural knowledge concerned. This separation enables the explication of links that exist between the strategic and structural knowledge. "Decide-status" uses the notion proposed by Clency (1986) and Murdoch and Johnson (1987). The notion proposed the separation of problem solving and data management functions with an expert system. A "Data expert" locates and resolves conflicts in deciding whether the information is known to be true or false for the case and can be interfaced with more than one "problem solver".

Another implication for human-computer interface design in knowledge based systems is the separation of interviewer and reasoner. Gearing et al, (1982) have identified the problem that knowledge based systems often suffer a basic conflict between their computationally intensive nature and their need for responsiveness to a user. They have proposed a notion which calls for partitioning the system into two modules; one model which performs all the functions required for the interaction between the user and the system and a second model which carries out all the computation.

Volunteering:

Volunteering is an important feature of an effective knowledge based system. Where possible the knowledge engineer and the system builder should employ natural language interface. Kidd and Coper (1983) have proposed the use of graphical interface by the user to

communicate with the system. To volunteer information, the user could point to an appropriate assertion space and a menu of input options displayed for that assertion space.

2.3 Incorporating Interactive Dialogue Networks within the Systems Development Process

The role of dialogue networks as a dialogue design tool was discussed previously. In this section, we will discuss the role of dialogue networks within the development process and as a tool for acquiring knowledge from expert As a productivity tool, we have seen previously Moubarak (1989) dialogue networks with the aide of the interface system developed in this program of research offer the potential to achieve greater productivity gains in the areas of external design, data modelling, internal design and as a tool for supporting data dictionary facilities.

The external design is addressed by the creation of user views - user formatted output such as screens and report layout with the aid of dialogue networks. Dialogue networks allow the designer and end user to construct and maintain the data model. This includes the creation of the entity diagram which represents the business entities and their relationships. Each entity is represented by a node and each relationship is represented by an arc connecting the two appropriate entities. By using dialogue networks, better quality data model design iteration capability, improved productivity when establishing complex entity diagrams and relationships. Other features include the ease of maintenance - new entities are added to the net without changes to the rest of the networks representing the data model.

The internal design is concerned with the specification of database, program and operation procedure. Interactive dialogue network can be used as a dialogue delivery vehicle at the construction stage, in particular in the area of program coding within the development process.

As for the use of dialogue network as a tool for knowledge acquisition and representation, the network created consists of three major elements. First the description of the dialogue, secondly data relationship within the dialogue and finally path fragments within the dialogue. The above three elements represent the user's knowledge base, which correspond to important features of certain knowledge based DSS.

The advantages of using transition networks as a tool for representing knowledge in DSS, is based on the fact that most people can understand dialogue networks, when they describe flow of process or known procedures with hardly any instruction. These procedures and processes can form the knowledge base for certain expert systems. The network is a vehicle for constructing the dialogue network (knowledge base) and in passive mode records the path fragments for driving the rules of certain knowledge based DSS.

2.4 The New Systems Development Life Cycle Model

Why a new model?

From experience in developing information and decision support systems that have given rise to the notions expressed in Chapter 1 and this chapter, it can be seen that there are a number of issues which need to be addressed and a number of overlapping elements which must be reconciled and brought together so that an effective system is designed and developed to satisfy users and provide a sound base for future expansion. Based then on these notions and ideas put forward, we are proposing a new systems development life cycle model. In proposing this model, we have used the systems development standard Moubarak (1989) as a base and developed it further to create the new model.

Major features added include the following:

- **Integration of corporate strategic planning within the development Life Cycle:**

Experience of various systems projects at the authors own organisation, Massey Ferguson, have highlighted the need for the integration between corporate strategic studies and the implementation of individual projects. Even though the need for strategic planning is increasingly accepted individual projects still exist in isolation rather than as part of cohesive strategic framework.

- **The incorporation of decision support systems (DSS) development within the development process:**

The standards described previously Moubarak (1989) were created with mainstream operational systems in mind with no consideration given for the design and development of DSS including knowledge based DSS. These systems need to be integrated with existing corporate information systems, they must address important issues of security, accessibility if they are to be accessed through the existing corporate networks. The new model addresses design and development requirements of DSS including knowledge based DSS.

Distributed Functions Requirements:

In recent years there has been a trend towards the distribution of data and it is processing in all but the smallest companies. In part this is due to the growth in the profit-centre management approach, which leads managers to control of their own data. This is particularly true where the part of a business have substantial autonomy from one another and from Head Office, and where there is a significant geographical spread. In the past some major corporations including the author's own organisation Massey-Ferguson have opted for large data processing centres, with a single central database and terminals spread over a wide area, or even worldwide. This approach incurs considerable communication costs, and potentially also exposes the business to the risk of communication failures or delays. Experience at the authors own organisation shows that some form of distributed approach can reduce this risk, increase the resilience of systems and then improve the responsiveness of the applications.

- **Data Security:**

The term "security" is used with many different meanings. In this chapter we take it to mean the ensuring of the availability of correct data to those with a right to access it. It can thus be split into three aspects, which we will consider in turn in this chapter; ensuring the availability of data; ensuring its correctness, and controlling access to it. These objectives of security can be remembered as reliability, integrity and privacy. It is important therefore that designers and developers follow guidelines for designing and developing distributed systems based on the requirement definition for distributed functions.

- **Applications Package Evaluation:**

Reviewing available application packages against the company's requirements can only be carried out when the business requirements are identified during the analysis phase and during design activities. However, it is a matter of judgement how far this evaluation should be taken within the phases of the development process. Application package evaluation is addressed with this in mind.

- **Developing Processing Controls:**

CICA (1986) have defined general control in computer systems environment to be "General management information" systems controls are controls over activities and resources of the information systems development, information systems processing and information system support functions. They include segregation of incompatible functions and information systems security activities. General control procedures that provide security and control over the computer programs, data and hardware facilities within the information systems processing, development and support environment. The new Life Cycle model we are proposing addresses the need to ensure that effective processing controls are properly identified, agreed and developed as an integral part of a system's development is discussed and proposed within the activities and deliverables.

- **Walkthrough Procedures:**

The new model promotes "walkthrough" procedures as a most effective way of ensuring that a deliverable incorporates collective brainpower. The emphasis is on system quality and improved project communication.

- **Change Control Procedures:**

The need to operate an effective change control procedure during the development cycle of a system in order to control a project's schedule and cost is addressed.

- **Application of system development tools within the development process.**

Integrated computer aided software engineering development environment will provide a complete and integrated set of tools that covers the entire development process. At present however, no integrated tool exists to support the development process. The new model proposes the use of a combination of tools within the development process to address quality communication and productivity issues.

The new life cycle model proposed in this section is proposed for guidance purposes. They are not intended to promote slavish attention to detail at the expense of observance of the principles and objectives behind them. The schematic on the following pages illustrates the project phases. Subsequent sections a head describe each stage in detail. Details discussed include the objectives of each stage, activities, deliverables and the players (skill level) required. Typically, there will be instances where "phase overlap" occurs. This is both natural and sensible. However, the overlap does not apply to more than two phases at a time and there must be a conscious awareness that some risk is involved because if the systems project were not to be approved at the end of the earlier stage being overlapped, then the investment in the activities of the later stage will be wasted. The discipline consists of an evolving set of techniques, activities and deliverables which have grown out of the successful systems development and implementations. We believe that the new model will form a useful framework for a systems developer, or a manager, or a user of systems.

THE NEW SYSTEMS DEVELOPMENT LIFE CYCLE MODEL

OVERVIEW

PROJECT STAGE	<----------DELIVERABLES---------->		
	USER	MIS / DCO	PROJECT APPROVAL
PREPARATORY			Project Plan
BUSINESS ANALYSIS			Project Proposal
DETAIL ANALYSIS	Business System Specification	Technical System Requirements	Pre-Design Approval Request
DESIGN	Business System Design	Technical System Design	Pre-Construction Approval Request
CONSTRUCTION	User Implementation Package	DCO Handover Components	Installation Plan
INSTALLATION	Installed System		
AUDIT			Project Audit Report

2.4.1 The Preparatory Phase

The objectives of the preparatory phase is to prepare the project environment necessary to ensure project success. Eight major tasks are identified for this phase. They are:

(i) **The definition of scope and objectives of the project:**

Three areas need to be specified. The first, the areas to be covered by the study. The second, areas to be excluded from the study and finally areas which will require partial investigation. In this phase, a high level function hierarchy representing the organisation and functions which are included in the study, excluded from the study and functions which require partial investigation.

(ii) **Organising the project team.**

There are more than a hundred significant activities to be considered in the development and installation of a new system for it to be successful. These activities span a range of different skills which are unlikely to be possessed by any single individual. This is further accentuated as the size of a system increases and a system of same complexity is considered. The various phases which are identified in the new life cycle model reflect the need for different skills to play their part as the system evolves from idea to implementation. If the required skills are not provided at the correct time, then the system project will probably incur extraordinary cost to accommodate them at a less opportune time, or even worse, proceed without their specific contribution. It is thus essential that there is a commitment from all concerned to provide the specialist skills required at the appropriate stages of the system project.

Organising the project team entails the identification of the team members from the appropriate user community, management service staff and external consultancy service if required. A project manager, user project leader, system project leader and the development team are appointed. In addition, user representatives from appropriate departments are appointed.

The role of the project manager is to oversee the project and be responsible for overall management of the project, appoint project team members, agree plans, review progress, ensure that problems are resolved and provide direction and interfaces with senior level of management.

The role of the project leader is to plan the project or sub projects (tasks identification, target date setting), identify and resolve conflict and problems, monitor progress (project review and co-ordinate meetings), report progress and problems to project manager, monitor quality of work by the use of standards and procedures and finally co-ordinate the team's effort and results ensuring global models are maintained and that there is no overlap of the team's functional areas.

The role of team members depends on the specific phase within the life cycle. Team members include systems analysts, data analyst, knowledge engineers, system designers, database designers, computer programmes and operational staff and, in some cases, networking specialists. In order to ensure sufficient appropriate skills are present within the team, an external specialist consultancy service is employed.

(iii) **Establish methods and standards:**

Agree conventions and methods, including specific data and function analysis techniques appropriate to the project. In addition, deliverables for each of the selected techniques are specified.

(iv) **Planning the project:**

For the whole of the project and for each phase of the systems project, planning is necessary at various levels. First, the planning in detail of the current phase which, as events proceed, will move into an equally detailed planning of the next phase. This requires the identification of the precise activities which are pertinent to a specific project, determination of the skills required for each task, estimation of the resources to complete each task and the verification of the availability of that skill, when required. Further, target date deadlines/milestones are published.

(v) **Training and education:**

This is an important task and is often ignored. Team members are trained in the appropriate data and functional analysis techniques, system design and construction techniques using specific productivity tool and finally training in the use of the specific project control package to be used. As for educating the user community participating in the project, an overview of the techniques and documentation needs to be given. One way of handling this is to organise a special course designed for users only or to conduct informal on-the-job training.

At the end of this phase, the preparatory phase, a high level document is published containing the objectives, scope, deliverables, project team, project plan and deadlines, standards and analysis and design techniques used. This document is submitted for approval and agreement should be obtained before proceeding with next phase.

2.4.2 The Business Analysis Phase

Traditional approaches to systems analysis has always been taken as their primary concern the processes that the system is to carry out. Any consideration given consisted of the specific application data requirements. Downes (1989) has argued that in this view point, the data has been seen as "a means to an end". In the business analysis phase, we are concerned with the organisation business environment in which the system is to be installed. There are several major activities within the business analysis phase. These activities need to be consciously considered, given that this is arguably an important phase within the systems development project. We propose six major activities and one deleverable for the business analysis phase. They are:

Major Activities:

(i) Strategic Planning Activity

Experience at Massey-Ferguson shows that the most important critical success factors is how effectively they can identify potential opportunities to support strategic business objectives, exploring strategic alternatives, establish a manageable and attainable long range plan supported by detailed resources and skill level profiles and consider the impact on the business environment of new system capabilities. These systems may be in the form of viewdata, electronic mail, expert systems, decision support systems or information systems.

Strategic planning stage determines the objectives of the organisation and the information that is needed to enable it to accomplish its objectives. Strategic planning should include the following:

- Understand and clarify the business objectives of the organisation. This should cover a five year horizon with built-in criteria for regular reviews.

- Identification of the business goals and strategies, the business information needs for the effective operation of the business and appropriate priorities.

- Investigation of the current status of systems already developed and implemented.

- Derive a plan which should specify project and their phases, taking account of current technical direction, covering; hardware, software, distribution policy, application systems structure, telecommunications.

- Identification of the new organisational structure to be formed out of their strategy. This will include resources skill profiles, economic evaluation, the need for maintenance and improvement of service levels and maximising the benefits from current expertise and investment as well as the long term improvements.

(ii) Agree Business Objectives

The strategic planning stage documents the organisation's business objectives and future direction. It does not follow, however, that these are immediately translatable into investigation procedures. Furthermore, business and systems planning should be integrated so that they jointly address the goals and objectives of the organisation. In most organisations and at the author's own organisation, Massey-Ferguson, there is usually a greater demand by users for new systems and revisions to old systems than there are skilled resources available to meet the demand. There is, therefore, a requirement for the identification of business objectives of the system to be developed from the start. In agreeing the business objectives, the activity seeks to elicit the business reasons for the systems investment which is being proposed and to agree them with the business user community. There are no specific rules and procedures for defining business objectives, but they must all obey the fundamental rule of relating to the operation of the business and to contribute to the profits of the organisation. In Chapter 3, we will show and discuss examples of business objectives from a live commercial information systems development project.

(iii) Conduct the Business Investigation

The investigation activities are referred to as preliminary survey and feasibility study. The main objectives of the investigation activity is to obtain basic understanding of the users' requirements including perceived problems and/or opportunities including likely savings or benefits that could be resolved through the development or acquisition of a new or revised information system. It is important to consider different ways of meeting the objectives and not to start with a solution seeking problem. The purpose of the investigation stage is, therefore, to justify the investment in the system on the basis of cost benefit analysis and avoid further significant investment (i.e. analysis, design, construction and implementation) in a project that will provide little or no benefit to the organisation. However, it is important to point out that at this early stage of the development process, precise costs and benefits cannot be obtained, as usually not enough is known about the possible system solutions. Costs and benefits can only really be accurately quantified once the analysis and design phases are reasonably complete. One major issue pertaining to the feasibility study is quantification of the benefits associated with each

proposed system solution. Benefits include such easily measurable savings such as displacement of personnel. However, system benefits should also include future cost avoidance and personnel productivity improvement. The feasibility of a system development project should not be a one-time up front decision, but is a matter that should be addressed throughout the system's development process.

The investigation activities can be described as an art aided by scientific approach to fact definition and recording. It provides information to facilitate an indepth analysis. It acts as a reconnaissance for the analysis phase. The main source of information gathered during the investigation stage is from the top and middle management. It is not necessary to interview the lower levels of staff at this stage. It is important at this stage to identify the decision makers and the decision making process with special emphasis on the expert knowledge used throughout the decision making process.

(iv) **Strategic Entity and Function Modelling**

The objective of this activity is to produce a high level strategic model of the organisation's business data and functions to the level appropriate for developing a system strategy. The models constructed are intended to be at a high level of definition. Further detail analysis will be required during the systems analysis phase prior to the design phase. Data and function analysis techniques will help us achieve the objective of this activity. The specific technique proposed is the CACI technique described in Section 2.1.1. However, there are no reasons for not using other techniques discussed in Chapter 1 to construct the models. When constructing the strategic model, we need to identify all key entities, only key attributes (those required to identify the entities) and the main relationships. When constructing the strategic function model, we need to develop all functions within the scope to the same level of detail. We need to decompose to a level necessary to help identify and scope system areas. We need to identify only significant events. It is important not to decompose to the elementary function level and not to spend a lot of time on functions that are unlikely to be computerised. The system areas identified allow us to define a system or group of systems. This is obtained by manipulating the rows and columns on the function, entity matrix to obtain clusters, boundaries drawn round clusters identify system areas and define their scope.

(v) **Idea Processing, Project Options and Recommendations**

This activity considers the different ways in which the business objectives might be attained. The facts gathered during the earlier activities are used to consider possible business solutions which meet the objectives. This may not necessarily be a consideration of a computer based solution. It might make good business sense to pursue a practical, manual solution. The activity is an exercise to formulate, structure, compare and manage ideas and issues gathered. It is an exercise in lateral thinking and may well benefit from a "brainstorming" session. The participants of the session are users and the team of analysts conducting the investigation. Each option considered needs to be related to the business objectives identified and it is likely that each option gives rise to a project of greater or lesser size with consequent differences in development and running costs. Whilst the options considered will often have a strong technical consequence, this activity is not itself concerned with the pursuit of technical options beyond the point of checking general feasibility and likely cost impact. Choosing the appropriate option to address all of the perceived problems or opportunities which were highlighted during the investigation activity is important and difficult. This is due to the fact that the recommended project scope may or may not embrace them all. However, it is important to take position with regard to each of the problems raised. Some of the problems identified may be totally outside the scope of the project concerned and need to be promoted as project opportunities in their own right. Other problems may be symptoms of more fundamental problems, which the project currently discussed may or may not address. There may be times when the selection of the project scope itself is difficult. Where this is the case, the project manager should take advice from senior management. The project scope gives rise to a rigorous consideration of the technical options open to the organisation to meet this requirement. There are three options available for recommendation. First in house development, the second system package acquisition and finally turnkey system development. The traditional in house systems development follows a clearly established process where the activities include what we are proposing within the new systems development life cycle model. System packages may be acquired from software systems vendors and implemented with little customisation in situations where there is a good match between user requirements and functions provided by the package. However, another option is available that is the package acquired from vendors is subjected to significant customisation. Existing systems may have to be altered significantly to

interface with the acquired system. This is called a complex package acquisition process and has to be handled with care. Experience at the author's own organisation shows that this type of process encountered serious problems in addressing all the problems identified during the investigation stage and the system development cost incurred to change the package in fact exceeded the benefits identified. An entity may contract the entire systems development process to an outside consultant or system house to develop and implement the required system. The organisation may or may not enforce a specific systems development standard on the developer. Other options which are capable of meeting the requirement include utilising an existing system from another corporate organisation and perhaps enchance the current system in operation to meet the new business requirements.

(vi) Knowledge based DSS consideration:

When examining the current business problems and potential solutions, it may be found that the problems identified are not amenable to solution by conventional methods. This in turns leads to the question of does the problem area lend itself to knowledge based techniques? To answer the question, we need to consider the guidelines described previously, the guidelines for building knowledge based systems. If the problem identified meets all the criteria discussed in the guidelines then during the investigation phase the knowledge engineer and the expert will determine the important features of the problem. These include type and scope of the problem, the requirements for additional experts, the aims of the system to built and the computing facilities required.

The Business Analysis Phase Deliverables

The project proposal document is intended to provide the information to enable senior management to make a decision on the future of the project to convince them of the need for a new system by describing its contribution to the organisation's business, to justify further investment in the project and to obtain commitment to support the development of the new system. The proposal document should contain the following: executive summary, business objectives, perceived problems/opportunities, business entities and function overview (supported by the entity and function model), project options considered, recommended project scope, systems options considered and recommended, business

issues, cost/benefits analysis and finally proposed project timing.

2.4.3 The Detail Analysis Phase

The analysis phase defines the business functions which span the project proposal, identifying which are addressed by the project, show how the business objectives are met, confirm the costs and benefits and user commitment to support the project. The focal point of this stage is the achievement of the cost effective business objectives. It is important to relate the problems and/or opportunities of the proposal to business functions and data, which must be fully explored. A complete understanding must be gained of the variety, complexity and interrelationships of existing procedures and practices. The proposed areas of change must demonstrate how the business objectives will be met, confirm what is included/excluded from the scope of the system, identify control requirement and highlight any key issues. User expectations must be soundly based and the size of the project clearly understood. Costs and benefits must be re-appraised and approval obtained to commit further resources to the project.

The core analysis team consists of one or more systems analyst, knowledge engineers and should also includeuser representative(s).

The analysis phase calls for a detailed understanding of the part of the business which was the subject of the investigation. This detailed understanding requires knowledge about the business functions and data, confirmation of current practices, identification of auditable control requirements, the confirmation of assumption and project scope, the area of change and finally the identification of key issues. We propose nine major activities and three deliverables for the analysis phase. We will now briefly describe each activity.

(i) **Analysis of Business Function and Data**

This activity calls for a detailed analysis of business functions and data using data and functional analysis techniques. We are proposing the use of CACI methodology described previously. However, there are no reasons for not using methodologies and techniques described in Chapter 1 to carry out this task. This activity can only be truly successful if there is an effective partnership between the user(s) and the system development team. It requires indepth exchanges with the business users who are responsible for performing the functions. Where there is a requirement to analyse the business functions of more than one location, it is most important to do this separately and not assume a similarity which may not exist. Any differences which do emerge must be checked out.

A full analysis of the data used to carry out business functions is important in achieving a complete understanding of these functions. The analysis of business data seeks to model the structure of the data itself as this will generally be much more stable than the mechanisms which currently use it. It is important that the model reflects the logical structure of the data, rather than the way it is organised.

At the end of the function and data analysis, the entity and function model will be constructed. In addition, the interaction between the models can be examined by the function and data matrix discussed previously.

The activities for knowledge based DSS development consists of knowledge engineer and expert working together to define concepts, entities, relationships and control mechanisms needed to describe the problem including all the details of how they are combined to solve a particular problem. The knowledge engineer needs to use the knowledge acquisition techniques we have described previously. For these type of systems the entity definitions and the relationships will contain much more semantic contents for knowledge based systems rather than traditional information systems.

(ii) **Confirming the Current Practices**

For each function within the project scope, the current mechanism by which it is carried out, whether manual or computer based, must be recorded. A full understanding of current practices is useful for checking the completeness of analysis and will be invaluable later in the project for constructing the training program necessary to support the migration to the new system.

(iii) **Auditable Control Requirements**

The auditable control ensures that the information systems selected meet the needs of the user requirements. In the context of the analysis phase, the objective of control is to ensure that the processing of data is complete, accurate, authorised, timely and verifiable. A method available for this control is the computer control guidelines by CICA (1986).

(iv) **Project Scope and Assumptions**

The project scope was identified during the business analysis phase and agreed as part of the project proposal. The proposal was prepared with an overview of the business function data available to the analyst. At this stage, the overview has now been expanded to detail function decomposition and a comprehensive entity model is constructed. The full implications of the proposed project scope can now be examined.

Furthermore, perceived problems and opportunities identified during the business analysis phase will now be better understood which helps to confirm the business issues and assumptions. The assumptions can be of several types - about stability, direction of development of the business environment outside the control of the project team, specific assumptions made about current business data and its suitability to support the business in the future.

(v) **Area of Change**

The area of change must describe all the current mechanisms which will be modified as a consequence of implementing the proposed project. All current practices within the business area investigated must be considered to see if they will be changed by the proposed system. Business functions not within the project scope may be changed by the system as information may be recorded or extracted in different ways. Describing the changes for the user community is generally insufficient. What is needed is a full presentation of the area of change supported by videos or a prototype of the system. This presentation must be planned and developed with the user community affected by the changes.

(vi) **Business Issues**

During the project issues which are crucial to the success of the project are identified and highlighted so that they are addressed. This includes any lack of specific resources or skills required to complete the project. Other areas include training requirements, the need for a change in business concepts or working practice and industrial relations exposure.

(vii) **Traffic Analysis**

The traffic analysis of the entity model provides the necessary information on the proposed use of data for the design of an efficient "in-house" database or the extension of an existing one. The traffic analysis is an extension of the analysis of the interaction of business data and function.

(viii) **Review Available System Application Packages**

A complete evaluation of available packages in the market can only be carried out on the completion of an analysis of the business, data and the design phase. However, it is a matter of judgement how far this evaluation should be taken at the analysis phase for a particular project. In reviewing packages at the analysis phase, it is important to investigate both the interfaces with the current systems and the implications for future

systems developments of adopting a particular package solution. Other activities include the investigation as to the compatibility of a package's data requirements with the company's database developments, the flexibility available for extracting that data for use in other areas, extensions which are available for extracting that data for use in other areas and the extensions which are available or planned for the package all have implications and may have limitations on future systems development.

For knowledge based DSS application we propose that serious consideration should be given to tools for building knowledge based DSS to run on personal computers (shells)

(ix) **Cost/Benefit Analysis Revisited**

As we said previously, the updating of cost/benefit information is a continuous process and should be carried out at each stage of the development.

(x) **Project Timing Revisited**

At the end of the analysis phase, the project timing and dates proposed are reviewed in the light of detailed information now available and new dates are published.

The Analysis Phase Deliverables

The detail Information Analysis Phase Deliverables can be seen as three separate requirements, the business systems specification, the technical systems requirements and the pre-design approval request.

The objective of the business system specification is to demonstrate to the end users that the business environment is well understood, define the scope of the analysis and identify the changes needed to meet the business objectives, to obtain user agreement to the boundaries and principles of the new system and finally to get commitment to support project development through design.

The objective of the technical systems requirements is to define in detail the Business functions and data which are within the scope of the system. This should describe, in sufficient detail for the systems design to be undertaken, the technical functions and data required to meet the business requirements.

The objective of the pre-design approval request is to re-assess the business justification for the system agreed at the business analysis phase. To take account of the agreed scope of the system proposed, project timing, the resources consumed to date, the updateed cost/benefit analysis and finally to make an investment recommendation.

2.4.4 The Design Phase

The objective of the design phase is to provide a cost effective system design which meets the business requirements and define a common framework for systems construction. This phase is concerned with designing the technical framework within which the new system will be developed and installed. It determines how the areas of change, agreed at the analysis phase, will be supported technically. Logical and physical data structures must be defined, the characteristics of all jobs and programs described, human-computer interfaces in form of screen and reports are identified and control requirements are confirmed. This phase must establish the overall technical operational requirements of the system, define explicitly the constraints inherent in the design and identify the testing resources required.

Before we discuss these activities in detail, it is important to discuss the key elements and the aims of the design process. Edmonds et al (1989) have argued that the attention paid to the different aspects of the system development varies according to the circumstances and the objectives of the power agents in any given project. Very often a key individual with enthusiasm and drive is to be found behind a successful scheme. If a system is to become more than someone's pet project and perform a productive commercial function, it goes without saying that it should be used. All too often, the importance of end users in the measurement of success or otherwise is notional. To ensure sufficient attention to user requirements analysis and specification and to place it correctly in relation to other activities, such as knowledge acquisition, it must be included explicitly in the project plan. It is not an easy task to ensure that users are taken account of, let alone be directly involved, in design. However, some activities lend themselves more readily than others to enabling the user voice to be heard. Candy and Edmonds (1988) have also argued that the approach to evaluation and prototyping as vehicles for increasing user involvement in the design is an important part of any satisfactory strategy. Further, Edmonds et al (1989) have argued that to clarify user requirements, a combination of user evaluation and the deployment of the user interface prototype can be used with practical mechanisms for assessing the appropriateness of the whole system design. It is important that the design process is based on the findings from the analysis phase. In particular, in giving consideration to the requirements of different classes of users. Further, facilities should be available to enable the users to articulate their requirements. As a design aim, the designer needs to consider all the design issues and guidelines discussed previously in Chapter 1 and this chapter. However, the designer needs to be aware of design considerations. We will discuss these considerations briefly now:

Design Considerations:

(i) **Practicality:** To provide practical solution to the requirements as defined in the business specification document the output from the analysis stage. The system should be capable of being operated over a long period by competent but average intelligence persons.

(ii) **Efficiency:** The best use of hardware, software and human resources including the accuracy, timeliness and comprehensiveness of the systems output.

(iii) **Least Cost:** The objective is to have minimum cost incurred provided the system meets the requirement defined in the business specification.

(iv) **Flexibility:** The system needs to be adaptable to the changes inevitably requested by the end users. Structured systems development methods discussed in Chapter 2 alleviate some or all of the problems created by the traditional methods including the flexibility problem.

(v) **Security:** the degree of importance of information security to an organisation should be reflected in the nature of the organisation structure and policies established to provide security of information and information systems.
As we said earlier "security" is used with many different meanings - availability of correct data for those with a right to access it. It can thus be split into three aspects, which we will discuss in turn.

Reliability: one of the prime requirements for a shared database is that it should be always be available and correct when required. This has always been important for computer held data, but becomes even more vital when the data is likely to be shared by many applications. Systems can fail. The application program itself may fail, either because of a program bug or because the user was unable to complete transaction, otherwise known as a success-unit. Computer and communications hardware can also fail. It is therefore important that we can recover the database to a complete and consistent state after any such failure. The first and most obvious way to recover is to use a copy of the database taken previously, and to reprocess any work completed since the time of the copy. Another way available is what is called "forward recovery" a log database of the changes is available to be re-processed. It is important to note that any form of provision for recovery costs money. Copies of files and databases takes time, and occupy disks or tapes, spare hardware or backup arrangements at another site also costs money, as does the effort of making recovery plans. It is therefore necessary to balance carefully the cost of recovery with the cost of various sorts of failure, so as to decide what level of recovery capability is appropriate for each system or database. A major factor here will be the cost to the business of being without the system, or the database, until recovery is complete.

Integrity: One of the attractions of systems based on personal computers is that the

designer need not consider the complications of shared access to data, and possibility that two users will want access to the same record at the same time. In a database context, though such sharing is one of the objectives, and it is to be expected in systems based on minicomputers or mainframe computers, as well as networked personal computers.

Furthermore, an often neglected aspect of DSS design is the security of DSS. Baskerville (1989) points out that such systems are intended to support decision making in what are often crucial areas of an organisation's activities. A decision based on misinformation may have catastrophic effects on the success of the organisation. Information systems, and equally DSS, are vulnerable to accidental or deliberately induced errors, errors in the data and operation. The designer of these systems needs to introduce fundamental design security principles. The first is the design principles that support the requirement to protect the hardware, software and data from destruction or abuse. The design principles that maintain the integrity of data through ensuring that the definition of the logical and physical data relationship is secure should include user identification and password protection facilities allowing the end user to interact with the full or partial facilities available. Another important design principle is to ensure adequate backup and recovery procedures. As for the security of DSS, Baskerville (1989) provides an analysis of the main risk factors and suggests an approach to DSS design which helps the designer in identifying threats and in providing controls which reduce the damage of decisions based on false information. We consider Baskerville's inversion model of information system risks is useful as a basis for design consideration for the security of DSS. The MSK inversion model suggests controls which should be strongly considered for DSS, such as the protection of data against integrity loss. This protection must be automated with the DSS, transparently imposed on the decision maker who is using the tool. Two major aspects need to be considered. The first is the data gathering phase. This is the most dangerously error-prone phase of the development. The second is the data integrity during analysis and construction of alternatives may not be properly implemented by end users. The guidelines proposed previously will address these two issues of security. However, we support Baskerville's view that "controls design should not be deferred to the decision maker". These security issues equally applies to knowledge based DSS.

Privacy: The third aspect of security is the control of access to the data. This is not always necessary for all data; when it is required, there are many possible levels at which

the control may be applied. The purpose is to ensure that confidential data cannot even be seen except by those explicitly authorised to do so, and that updating of data is only possible for authorised users. Such control always carries a cost, both in terms of implementation effort and in application performance, and control should therefore only be applied where necessary. The subject of processing controls are discussed in detail. As for accessing the data, there are many levels at which access may be controlled, ranging from access to the terminal or computer down to the individual fields of identified records. Broadly speaking the finer the level of control, the higher the cost. Some relational database management systems allow control of access to specific rows of tables, using selection expression. Access through a viewed table also limits access. As well as control by the database management system, the application by control access itself, perhaps for specific types of transaction. One danger to be aware of is that end user enquiry and updating facilities sometimes bypass such controls, and it may be necessary to limit their use.

(vi) **Distributed System Approach:** Several factors may indicate an opportunity to gain benefits by the distribution of data processing based on the requirement definition for distributed functions for multi national companies and other geographically dispersed organisations, such as the author's own organisation, Massey-Ferguson.

Sometimes the prime motivation is company policy. Each profit centre is to be autonomous and manage its own data. This is fine in principle, but may make it very difficult for Head Office to put together the necessary central information; here remote access to the local databases may solve the problem. It is also important to avoid creating a system structure which is only appropriate to the current organisation structure. In many companies the organisation structure changes more often than it is economic to develop new systems.

At the detail analysis phase, the designer needs to formulate the architecture for distributed database solution and predict the effect at each distributed node. The designer can then define data distribution, volume by node and database. Size by node. Having defined the processing nodes necessary to meet the computing requirements of different business units, the mapping between business functions and business unit may then be used to define the transaction types of the node, the frequency of transaction and

cross node transactions. On completion of these activities, the designer can then determine the impact of alternative distributed processing and/or database architecture on machine utilisation and performance. Three types of distributed computing approach are available for the designer: distributed processing, standalone computing and personal computing.

Distributed processing refers to the situation where processing is carried out at more than one location. Distributed processing involves hardware and software application systems running at the distributed locations for use on either a standalone basis or on a linked basis. In the latter case, computers are linked to central databases or to a series of centralised databases and data is entered or retrieved (uploaded or downloaded) at the distributed location. The distribution of processing needs across time zones is a factor which can work either for or against distribution. Sometimes a single processing centre can deal with processing needs in, for example, the USA and Europe, since the peak times do not overlap and total processing costs are lower. On the land it may happen that processing costs are lower if a system is operated during local office hours. In Chapter 4, we will describe a specific live commercial system using this type of distributed processing.

Standalone computing refers to the use of a computer in a manner that is separate from the computerised information systems environment of the organisation, but where the results obtained from its use could have an affect on a decision making process. In this type of distributed processing, data may have been extracted and downloaded from the central databases to a microcomputer for decision support type application. It is important to note that the data manipulated by the user will not be uploaded to update the central databases.

Personal computing is another form of distributed computing and it refers to the use of PC's in such a way that it is not integral to the computerised information system. Data is not downloaded to the PC or uploaded. However, this type of computing is widely used and it is used as decision support activities in the area of financial data analysis using Lotus 1-2-3 or graphics facilities for presentation purpose. Other examples include the use of wordprocessing packages on the PC for text manipulation.

In 1986, the Canadian Institute of Chartered Accounts published the major issues concerning distributed computing. It was argued that the importance of a distributed system could be assessed by a form of risk analysis based on the following questions:

- To what extent does management base important decisions on information produced by the distributed computer system?

- What would be the effect on the day-to-day activities of the entity if the distributed system was not available?

- Is a significant control function carried out at the distributed location?

- Could the user of a distributed computer inadvertently or deliberately alter or destroy data on the central system needed by other users?

- What exposure could result if unauthorised persons gained access to computer hardware and/or programs and data at the distributed location?

The designer needs to address all these issues highlighted before deciding on distributed approach as a solution. In Chapters 4 and 5, we will be discussing this subject again. The context is in the form of designing live commercial distributed computing systems.

(vii) **Database Technology:** Database designers, in conjunction with systems designers, take input from information analysis, data modelling, data use analysis and function modelling to design the physical database. By consolidating information on the requirements of all the transactions, they identify precisely what, within the logical database, has to be included in the physical design. Structuring rules for the particular database software package are applied. The loading pattern is reviewed and the physical design is revised until an efficient structure is obtained. The designer then prepares scheme and sub-scheme specifications and recovery, security, archiving, restructuring (reorganisation), monitoring and operations procedures. Database software environment available to the designer are the hierarchical and relational structures. Both need to be considered with care and applied as appropriate. Experience at the author's own organisation, Massey-Ferguson, shows operational systems designed for stable applications, which change very little over the years. These systems have high transaction rates, typically several thousand per day, fast response. In addition, these systems have complex pre-defined data structures to provide the required performance with current data updated in real time. These systems are needed for the day-to-day running of the business and highly related structures are needed. The "hierarchical" model of the data is appropriate, as we can build the relationships required into this model in advance, e.g. IMS/VS and DOS/VS DL/1 are IBM's database products for this environment used at the author's own organisation. However, there is no reason for not using equally appropriate hierarchical database product from other vendors. Experience also shows that operational systems are not designed to meet the increasing business needs for timely and accurate information. What are required are decision support systems (DSS) described previously. These systems have the characteristics of data requested and are business related. They have a short life - they are used to solve a specific project, simple and changing data requirement so considerable flexibility is required. These systems have the need for summarised and historical data. For these types of systems, a simple data model is required. The relational model of the data, where the data is viewed as tables, is most suitable for this environment. Examples of these types of technology are IBM's relational database product DB2 and SQL/DS which is used at the author's own organisation, Massey-Ferguson. In Chapter 4, we will describe two case study projects using the hierarchical and relational approach.

(viii) **System Prototyping:** Prototyping is based on the premise that users and management cannot initially grasp all the ramifications of any proposed system. With prototyping, improvements are dynamically incorporated. The perception of what the proposed system can achieve grows with experience, growth in user population and trial and error. The designer can assume that efficiency, controls and data security can be ignored in the prototype. He is left with input data, the output data, the file structures and the system logic. The ideal prototyping software would generate the input output screens and the internal files under the control of the prototype software itself. In addition, transactions to update the database management systems include output reports. Different fourth-generation languages and application generators may possess some or all of these features. However, fourth-generation software tools may not be available for the designer. At the most basic level, the designer can use "screen painting" facilities to construct input output screen and report layouts so that users view the proposed format and agree and, as an interative process, the final product is produced. Prototyping and system design techniques blend together successfully. We discussed this subject in Chapter 1. We equally feel that prototyping is an important activity within the design phase to construct the human-computer interface, if the designer aims to achieve the concept of user-driven rather than technology driven design. Furthermore, the recording of the overall experience gained during the prototyping will help to avoid mistakes in full implementation and act as training for the users. Clark et al (1985) have argued that "tracking the cultural impact of the prototype is a must for assessing the wider effects of a fully developed system". Prototyping have a major influence in the design of DSS and knowledge based DSS. Hayes-Roth et al,(1983) have argued that the espoused development strategy for expert systems is an evolutionary, prototyping approach where the system evolves in multiple stages through continual interaction between the knowledge engineer and the expert(s). Jenkins (1982) has stated that prototyping is desirable, even essential, in environments where system requirements are dynamic.

The Design Phase Activities

The major activities within the design phase include the following: planning the design phase, the determination of the logical and physical databases, the definition of screen and report layouts, and the preparation of system "demos". In addition, design constraints and description of job and program requirements including confirmation of auditable control requirements are completed.

These activities will result in three deliverables - the business system design, the technical system design and pre-construction approval request. The business system design defines the system from a business standpoint, covering both technical and procedural perspectives, to the level of user supervision. Further, the business system design ensures that user(s) agreement to the system content and a commitment to support the project development through the installation. The contents include the business objectives and the scope of business analysis, planned area of change, illustrative screen and report layouts including system demonstration, auditable control requirements, business issues, project control-containing resource requirements and proposed project timings.

The technical system design defines in detail the technical functions which the system must provide, identifies all the technical components required, including hardware, software conversion needs, testing facilities and operational characteristics/feasibility. The content of the technical design document includes the logical database structures and the physical database characteristics, screen and report layouts, system component identification, program synopsis/ characteristics/interrelationships and operational requirements. Finally, it contains auditable control facilities and project testing resources. The pre-construction approval request re-assesses the business justification for the system agreed at the analysis phase. It also takes account of the design of the system proposed, project timing, the resource consumed to date, the updated cost/benefit analysis and finally to make an investment recommendation. The pre- approval request contains an executive summary, business objects and issues, summary of changes, boundaries of the new system, project organisation and resource requirements including project timing and finally costs/ benefits including one-time costs incurred. Specific activities related to information systems design we propose that these activities and

requirements for deliverables be a part of the process of the design stage within the new development life cycle model. In addition when designing information and decision support systems the result of the analysis stage is incorporated into the design process. Two areas are addressed by the design stage, the external and the internal design. External design includes the production of on-line-screen, report layout and the creation of the relational diagram. During the design stage the screen layout is reviewed again. This may be necessary due to technical requirements, for example the acceptable response time limitation for an on-line transaction which may force the designer to break down the transaction further to smaller transactions. The internal design is concerned with the specification of the data base, program design and operational procedure. The program design consists of identifying all possible execution paths for each problem to be solved. We propose the use of dialogue networks to design the human-computer interface. Such technique will ease the design burden. As for designing knowledge based decision support systems, the step the knowledge engineer will take is the design of the knowledge base. We discussed a methodology for designing the knowledge base previously. During the design stage the key concepts, rules and relations identified within the analysis stage are expressed in a formal way. At the design stage the knowledge engineer will decide what method to use for representing the knowledge. We discussed various methods of knowledge representation previously. In this phase the knowledge engineer needs to consider the human-interface carefully. Three specific techniques may be adopted. Explanation, reasoning and voluntering techniques. We described these previously.

2.4.5 The Construction Phase

The objective of the Construction Stage is to develop and test a system in accordance with the agreed design and prepare all ancillary materials required to install the new facility. In this stage the computer system is developed in accordance with the approved technical design and is tested to the satisfaction of the Management Informations Services team. All the necessary information to enable the computer centre to operate the system to meet the business requirements is produced.

The activities within the construction phase include the following: scheduling the construction activities, establishing a project test, the creation of test data and database, the installation of special testing equipment/ software. The most intensive activity in this phase is the writing of the program specification and design and code of programs in the appropriate language. In addition, user training material and procedures are prepared.

The project deliverables within the construction phase consist of the five major deliverables - the user reference guide, the Data Centre reference guide, the installation plan, the application software documentation and finally pre-installation approval requests.

The objective of the user reference guide describes in detail how to use the facilities provided by the system and provide a continuely maintained and up-to-date point of reference for the users of the system. The deliverable contains business objectives, Help Desk contacts, user procedures, processing schedule, screen layouts, Help screens, report layouts and auditable control requirements.

The Data Centre reference guide defines the operational requirements in terms of jobs, operations, interdependencies, JCL, report distribution, security, scheduling and recovery, such that the Data Centre can operate the system to meet the business requirements.

The installation plan defines the steps to be followed in implementing the system, including the schedule for user system testing, system cut-over and the phasing out of superseded facilities. The plan identifies the training requirements of the users and how they are to be met and to

enable the Data Centre to install the system by making available its constituent parts. The contents of this deliverable include business issues, detailed activity schedule, system start up considerations, cut-out of superseded systems. In addition, it will contain resource requirements.

The application software documentation provides comprehensive and up-to-date information to enable the jobs and programs comprising a system to be maintained and/or amended. The documentation is required in a different form for systems which have been developed "in-house" than for systems which have been implemented using a package. The content of in-house developments include program specification, program structure/flowchart, program source and link listing, program test plan and test data, JCL and test results. The packaged based implementation consists of package technical guide and installed software components log. The technical guide contains package introduction, software communication medium of the vendor, supporting documentation supplied by vendor installation processes, physically changing the software, the testing environment and finally software transfer processes.
The pre-installation approval request re-confirms the business justification for the new system, updating the cost/benefit analysis with the one-time costs consumed and the latest prediction of ongoing costs and benefits, in addition to making a recommendation about the installation of the new system and its timing.

The core construction team consists of management systems (analyst, knowledge engineer, technical designers, Programmers and Data base analysts) and end users. The nature of the technical solution and the organisation of the systems development group influences the composition of the design team. The management systems staff are mainly involved in the construction of the system using appropriate software tools. and the coding of the program from the program specification created during the design stage using a specific computer language. The choice of the computer programming language must reflect both productivities and programming requirements. Many organisations recognise that the increase in productivity from the programming area must be enhanced if they are to meet the demands from the user community. This resulted in development of powerful languages which enable results to be obtained without the use of professional programmers. The languages vary greatly in their capabilities. Some are query languages, high level languages such as COBOL, others claimed to generate complete applications. We propose that these activities and requirements for deliverables be a part of the process of the construction stage within the new

system development life cycle model. we proposed the use of the dialogue networks for the construction of interactive systems.

The construction phase activities for knowledge based DSS is a continuation of the prototypingactivities carried out during the design phase. The prototype with additional revision in certain cases constructing the interface with the corporate systems, the prototype becomes ready for testing. Testing of the knowledge base should proceed rapidly to check the effectiveness of the design stage. During the testing of the knowledge base the expert will evaluate the system and, in conjuction with the knowledge engineer, will run the system on many examples before it is released for more testing by the user community.

Waterman (1985) stated that during the development of the knowledge base, the knowledge base may have reached an unmanageable size, with systems conclusions unorganized and presented at the right level. At this stage the knowledge engineer and the expert should attempt to re-examine the problem and the associated knowledge representation model which could lead to more appropriate design and construction.

2.4.6 The Installation Phase

This stage is concerned with the procedure of cutting over the system to production and the evaluation of the success of the project and the system in meeting their original objectives. To train users in the new system, confirm availability of user procedures, perform user systems acceptance test, implement any dependent organisational change, install the system and remove all mechanisms superceded by it. This stage is concerned with testing all aspects of the new system. It includes the installation of hardware/software which was approved as part of the project justification. The major aspect of this stage, however, is when users test the capabilities of the new system in acceptance trail, whatever form that take. Depending on user experience with testing the new system, so the implementation of the new facilities will be scheduled, parallel running requirements confirmed and the cut-out date of superseded facilities agreed. The deliverable from he installation phase is the installed system.
The activities within the this phase include the scheduling of the installation stage, the installation of approved hardware and or software, the completion of user training programs, the validation, conversion and load of start-up data. In addition, user system testing, system security are implemented. Other activities include the creation of production database, implementation of the new system and discontinuation of the superseded systems.

The core installation team consists of management system staff (analyst, knowledge engineers, technical designers, Programmers, DBA and computer centre operational and control) and end users. We propose that these activities and requirements for deliverables be a part of the process of the installation stage within this new development life cycle model.

Transfering knowledge based DSS from its development and test stage to production takes place at the installation phase. Standalone systems are easy to install and effectively the test system becomes the production version. However, where, the system interfaces with the other corporate systems-the interfaces are established reliably with the implementation of security, backup and recovery procedures.

2.4.7 The Audit and System Evaluation Phase

This stage is an important process which aims to make an ojective appraisal of both the development of the systems project, as well as the resulting system. The audit stage evaluates the project in terms of the quality of the system produced, the timing achieved and the cost incurred. To review the effectiveness of the system, its cost of operation and the benefits obtained. To improve, as necessary, the project development and approval processes. The business problems and business benefits are the yardstick for determining the success of the project in meeting its business objectives. The cost benefit analysis of the system implementation strategy is the yardstick for determining the success of the project in meeting its technical objectives. The assessment is based on the extent to which budget and target dates were met and the quality of development as measured by change requests and problem reports raised during the development life cycle.

The deliverable from the audit stage is the project audit report. This report verifies the effectiveness of the development process for the project, in terms of the one-time resources consumed, the timing achieved and the quality of the resulting system, to establish if the installed system meets its business objectives and review its costs and benefits. The report recommends any necessary changes and actions with regard to the development approval process and/or the system.

The core audit team consists of the auditors supported by analyst members of management systems group and end users

2.4.8 Systems Development Activities/Deliverables Assessment

The principal sections of the new systems development life cycle provided guidelines on the activities to be considered and deliverables required for the effective conduct of the system projects. The sections a head concentrate on procedures and techniques and cover in some detail how to address specific subjects of developing processing controls, walkthrough procedure for system quality and finally system change controls.

(i) Developing Processing Controls

The subject of auditable controls arises in several phases of the new systems development life cycle. We have argued previously that these controls are a vital part of the system's effectiveness and must be addressed at the correct time during the development process. The analysis phase identifies the control requirement - the need for and nature of the processing controls. The subject must be considered in its entirety, from the point of data origin to the multiple points of usage, and should include both manual and automated controls, as appropriate. The criticality of the controls must be assessed and due consideration given to the subject of cost effectiveness in relation to business functions identified for controls. The design phase specifies how the control requirements are met. There are various control techniques which can be used to meet the control objectives of processing of data - mainly completeness, accuracy, authorisation, timeliness and adequacy of management trails. The design of the new system will typically confirm the degree to which it can accommodate the automated control requirements in the context of the computer programs supporting the relevant business functions. The construction phase builds the controls, whether the controls are in the form of a computer program or a user procedure. The installation phase uses and tests the effectiveness of the controls which have been incorporated. The audit stage reviews the entire scope of the project and assesses how effectively the controls have actually operated to protect the company's business interests.

(ii) Walkthrough Procedure for System Quality

A "walkthrough" is a procedure for confirming the quality of a deliverable, through subjecting subjecting it to open scrutiny by a group of peers, in the presence of the author. It is a most

most effective way of ensuring that a deliverable incorporates collective brainpower.

The procedure was promoted in the mid-1970's by IBM as a method for improving the quality of programs - an area in which it has proved very successful and gained wide acceptance. Since then, the use of walkthroughs has expanded out of the programming area and is commonly used throughout the system development process. Indeed, the walkthrough procedure can be used to review any set of deliverables for any type of project.

The benefits to be obtained from using walkthroughs are:

- Systems project quality - discovery of errors and omissions.

- Investment control - earlier trapping of errors which, if undiscovered, would require costly effort to correct.

- Productivity - improved chances of "right first time" in meeting well defined, accurate requirements.

- Improved project control by highlighting the completion of deliverables.

- Familiarisation of other project team members with the deliverable - an insurance against the unexpected loss of a team member.

- A training vehicle for new team members and encouragement of a team approach.

The formality of a walkthrough is varied according to the type of deliverable concerned. A walkthrough of a program specification, for example, is normally an informal review between the author and the programmer. On the other hand, a walkthrough of a major deliverable, such as a business system specification, might involve several reviewers with varying interests and must be conducted in a more formal manner to achieve maximum effectiveness. This type of walkthrough is referred to as a "formal walkthrough" and these are identified as activities in the system development cycle and are shown in the checklist against each appropriate phase.

Formal walkthroughs should be highly visible milestones, established ahead of time and explicitly incorporated into the systems project plan and resource estimates.

The principles behind this procedure apply to all walkthroughs, but the degree of formality exercised will depend on circumstances. There are three distinct steps to be followed: Preparation, Review and Follow up. We will now discuss each step.

- **Preparation**

The author must allow the walkthrough participants sufficient time to absorb the message of the deliverables and to consider their reaction. Thus, the first step is to communicate its contents in advance of the review and to some extent this will depend on the nature of the deliverable itself. If it takes the form of a written document, then the preparation will take the form of distributing copies of it to the participants. If it takes the form of a demonstration, then much will depend on its duration and complexity, as well as the available demonstration facilities, in determining how best to give the necessary advance insight into the deliverable's contents.

The author is responsible for ensuring that the preparation step is completed, which will include advising participants of the time and venue for the review.

- **Review**

The purpose of the review is to achieve a full understanding of the contents of the deliverable and to subject it to constructive criticism, with a view to ensuring that its quality is either confirmed or enhanced by the collective involvement of the participants. It provides an excellent forum for discussing any omissions, disagreements, difficulties of concerns which the participants might have with the deliverable. Actions must be agreed for all outstanding points, which need to be addressed to produce the highest quality deliverable possible. The following must be included in the review:

. The author gives a brief overview of the deliverable.

. Any participant with points to raise must be given an opportunity to register them.

- The author "walks" the review group through the deliverable in sufficient detail so that any concerns are satisfactorily explained or brought into focus as shortcomings in the deliverable.

- Outstanding points are recorded and decisions taken on who will address these and by which date.

Normally, a review should not exceed two hours (half a day maximum) and it should be reconvened at a later date if this time is exceeded. The spirit to be achieved is one of "constructive criticism", aimed at the QUALITY of the deliverable. The author should not be put on the defensive or feel the object of undeserved criticism - it is the collective responsibility of the reviewers to ensure that this does not happen. If SIGNIFICANT revision to the deliverable is required, then it should be submitted for a further review. Sign-off responsibility for an outstanding concern is generally assigned to the participant(s) who raised it.

Notes:

- Begin formal reviews with a clear statement of the business objectives of the project producing the deliverable, as well as the purpose of the particular meeting taking place.

- Appoint a secretary to minute the points registered.

- If the walkthrough is expected to be controversial, appoint an independent chairman.

- A reviewer who is unable to attend personally should be represented by someone who is capable of active participation, with the authority to make decisions.

- **Follow Up**

The author must ensure that each outstanding point is successfully resolved and the deliverable amended accordingly. The amendment needs to be duly approved by the appropriate individual(s).

(iii) **Change Control Procedures**

One of the greatest causes of project slippage is the failure to manage requests for change which are received during the life cycle of the systems project.

It is undesirable to suppress change requests artificially by the imposition of a "freeze", since this will generate user dissatisfaction and may result in a final product which does not meet the business requirements. On the other hand, the unquestioning acceptance of requests for change can have a serious impact on the project schedule and budget, which will also give rise to customer dissatisfaction and may have a detrimental effect on the business. The objective of a change control procedure is to evaluate the impact of each change request, so that an informed business decision can be made on whether or not it should be accepted. It is essential that the procedure be operated with minimum overhead and it must be appear unduly bureaucratic.

The benefits of a change control procedure are:

- Improved decision-making in response to requests for systems change.

- Elimination of arbitrary and unpopular "freezes".

- Greater understanding of the implications of a change request by the originator.

- Ability to measure the cumulative impact of change requests on a systems project. A large number of small changes can have as great an overall impact as a few major ones.

- Identification of weaknesses in the systems development process, enabling appropriate action to be taken.

The change control procedure only applies to deliverables which have completed a formal walkthrough and been accepted. It operates as follows:

- The change request is submitted to, or raised by, the project manager.

- The project manager estimates and records the effort required to accommodate the change.

- Wherever possible, decisions to accept/reject requested changes are made at regular meetings, so that priorities can be better assessed, but there will always be exceptions!

- If a requested change involves additional resources from a support group (e.g. programming, where this is a separate group), this must be discussed with the appropriate manager, before making a decision.

- If the requested change does not jeopardise the project schedule or budget, the project manager is authorised to accept the request.

- If the requested change places the project schedule or budget in jeopardy, the project manager may seek guidance from a member or members of the project steering committee, as appropriate, before accepting the change.

- The actual effort involved in actioning the change is recorded.

The Initiators:

. Anybody can initiate a change request (user, analyst, database analyst, programmer, Data Centre staff, etc.).

. If a project manager intends to reject a requested change, the reasons for rejection should first be explained to the originator.

. A log, showing all the change requests received for a systems project, must be kept, irrespective of whether or not a request is accepted.

. If a request is rejected, the originator has the right of appeal to a member of the project

steering committee.

- A change request is raised if the estimated effort involved is greater than one day.

2.5 System Development Productivity Tools

Butler Cox Foundation (1990) have reported that suppliers of the system development tools in the market argue that their products will address all the problems encountered by the systems builders. They have argued that this does not mean, however, that the tools should be ignored. There are benefits to be gained if they are adopted selectively. There are several types of development tools currently available and in widespread use these include the following; third generation languages, object oriented programming systems, expert systems, end-user tools, non professional 4GLs, fourth generation languages, computer aided software engineering (CASE), data dictionaries and integrated project-support environments. But which tool? The advice hinges, primarily, on methodology on techniques employed to acquire and organise the information. Tools can assist in the planning, analysis, design, documentation, prototyping and construction of applications in accordance with one or more structured data and function analysis techniques discussed in Chapter 1 and this Chapter. Advocates of tools say that, used properly, tools help produce correct and maintainable software; detractors say that tools produce nothing and end up as shelfware. We support the view that tools can help define, design and build system and in the next section we will discuss the application of tools to the appropriate phases of the new Life Cycle model.

Let us consider the core phases of a system development life cycle i.e. analysis, design, construction (coding) and implementation. The other phases initial project definition and post implementation review are not really relevant to this discussion.

(i) **Analysis Phase:**

This is the logical modelling of the business system requirements. There are two distinct elements to this, with various techniques which may be used:

- Strategic Data Definition:
  - Entity - Relationship Diagram
  - Entity Description
  - Attribute within each entity
  - Attribute definitions
  - Data Access profiles - (keys)

- Function Definition:

Function Hierarchy
Data Flow Diagram
Data Flow to/from user
Data Flow to/from data store
Function logic definition
Horizontal Balancing
(checking completeness of function/data interaction)

Analysis tool kit should provide extensive facilities to support all of the above activities.

(ii) **Design Phase:**

Here again there are two aspects:

- Database/File Design

Sequential Files
ISAM Files
VSAM Files
DL1 Databases
DB2 Databases

- Program Design/Specification:

Program designation
Menu layout/user dialogue
Screen layout/user dialogue
Report layout/control blocks
Data manipulation logic
Program structure design

Design tool kit should provide extensive design support, including prototype screen, program module descriptions and inter-module parameter definitions.

(iv) **Construction phase:**

Once more, the two elements:

- Data Definition: File layouts
  Database schemas

- Program Generation: Screen formats
  Screen dialogue code
  Report formats
  Report control code
  Data manipulation code

Design tool should generate DB2 schemas, CICS BMS maps and IMS/VSMFS blocks.

The application generator should provide the application system. This eliminating the need for coding in COBOL/PL1.etc). The application can be created on a pc and then migrated to the mainframe environment.

2.6 Conclusion

The central concern of this chapter has been the systems development process. In particular, the application of methods, techniques and test which would support the following objectives. Better project management and control, enforcement of standards, improved communication between end users and development staff - better requirements definition, improved designed, improved productivity, fewer staff, better quality reduced maintenance reusable components and better documentation. To meet the above objectives we have considered three major areas. The activities and deliverable. Within the systems development process, structured data and function analysis techniques and the use of systems development tools within the development process. We envision the new model to be an architecture, philosophy or open framework having the objectives of managing the system project through the whole cycle of activities needed to develop information systems including decision support systems. Furthermore, producing an asset of high quality, which is capable of making a continuing contribution of the business. In constructing the new model we have enhanced the systems development standards Moubarak (1989) to include major important features mainly; the integration of corporate strategic planning, the inconsideration of distributed function technology and application package evaluation. In addition developing processing control procedures, walkthrough procedures and change control procedures are added.

As for structural analysis and design techniques we reviewed the need for methods of analysis which will allow us to focus on the real business needs. As we have seen, an important aspect of this analysis is the consideration of the data required to support the business. The entity model was proposed as a tool for showing how the various types of data of interest to the business are related. Another important aspect is function analysis, which is a companion to data analysis. Where data analysis concentrates on the things with which a business is concerned, function analysis deals with what the business has to do to be successful, independently both of the organisational structure and of how any existing systems, manual or automated, behave. It is important that function analysis is business orientated and should not be concerned with analysing how existing systems work, except in so far as it helps us to understand what the business needs to do. Nor should we be unduly influenced by the existing structure of the organisation and management structure change. The underlying needs of the business are much more stable. Entity and function models as a language for describing complex things:

The CACI techniques of entity and function modelling we have proposed provide a key to rapid understanding of the organisation's business information needs and the functions performed by the organisation. The greatest difficulty in designing a business information system is to overcome the poor communication between the designer and the end user community. The diagrammatic conventions proposed by the CACI techniques can be used as a powerful aid in the communication of complex subject matter. There are specific communication areas we feel the CACI techniques improve. First the communication between the analyst/modeller and the end users who know the business in detail, but lack the technique to articulate those aspects which form the essence of information systems. The second is the communication between users from different departments who find that diagrams often reveal their conflicting or ambiguous perceptions in a way which makes them easy to resolve. The third is the communication between the analyst/modeller and his/her previous conceptions so that his/her own understanding can advance steadily instead of going round in circles. Finally, communication between the analyst/modeller and the technical database designers whose skill and knowledge is about technical database design and efficiency of operation of these databases.

The data and function model gives a clear understanding of a user's business. The model gives them a clear and self-consistent definition of the things about which information is to be held, as well as the relationships between them which must be built into the database in one way or another.
In all the four cases, the benefit derives because the diagrams clarify the subject matter and provide a new and relevant specialised terminology with unambiguous definitions which is self-contained and self-consistent. This is done with the aid of entity relation diagram.

Other techniques described in Chapter 1 can equally represent the entities and their relationships - some diagrams are better than others. We discussed the advantages and disadvantages of these techniques. However, the test for any model is whether people other than the model authors can understand them. Two main questions need to be answered. First is the model self-consistent, does it hang together? The second is can people familiar with the subject matter understand what it is saying and correct it as necessary?
In Chapter 4, we will discuss the use of models in live commercial business information

systems and in Chapter 5, we will attempt to test the model to address the questions addressed above.

As for the use of systems development tools, we showed that tools can assist in the planning, analysis, design, documentation, prototyping and construction of information and decision support systems in accordance with one or more structured data and function analysis techniques described in Chapter 1 and in particular within the new cycle model. Development tools suffer from unrealistic expectations. Tools can help define, design and build systems, but there's no magic involved. Successful implementation requires dedication and skill. It must be remembered that the facilities offered by any tools make possible changes in working method which would be impractical without them. Therefore the implementation of the tool should be seen as an evaluation of a complete new way of working supported by appropriate tools within the phases of the systems development process. This will lead not only to a technical implementation of the tools themself but also the implementation of changes in organisation and working practices which should be considered. Furthermore, suppliers of the system development tools in the market place argue that their products will address all the problems encountered by the systems designer and builders. As in the past not all of these promises will be realised. Butler Cox Foundation Report (1990) have stated "Do not commit to integrated CASE". They have argued that at present, no complete integrated case environment exists. The author's own experience in evaluating various products during 1989-1990 supports this view. However, we feel that system development tools should not be ignored and as we showed previously there are benefits to be gained from using development tools.

CHAPTER 3

THE ROLE OF INFORMATION COMPUTER

SYSTEMS IN A MAJOR BUSINESS INITIATIVE

3.1 Place of Strategy in Shaping and Guiding a Company

Yavitz and Newman (1982) have described major characteristics of strategy, namely "the strategy focuses on basic longer-term direction, is primarily qualitative, provides guidance for preparation of short-term plans, integrates functional plans into an overall scheme for the company, is realistic and action orientated, and is understood throughout the top and middle levels of the organisation". The emphasis in strategy is on the quality and texture of the business. An actual strategy is a longer-term plan that sets the direction and tone of the shorter-range plans. A strategy must be feasible in terms of resources that will be mobilised and it must identify ways by which at least some form of superiority over competitors is to be achieved. Yavitz and Newman (1982) report that an essential feature of all strategic planning is a forecast of the world ahead - or at best, a forecast of those parts of the environment that will have significant impact on the company's successes and failures. This chapter presents strategy as a trajectory for managerial action. In the sections ahead, we will be discussing the need for strategic direction and how strategy is formulated at the author's own organisation, Massey-Ferguson.

3.1.1 The Need for Strategic Direction

The need for strategic management is increasing as managers are being confronted by a wider range of external pressures that must be taken account of in their major decisions, with a constantly changing business environment involving risks in declining or unprofitable market segments that require a fresh approach or withdrawal. The strategic mission of any organisation should address three major areas. The first is where and what the organisation would like to be in the future world. Secondly, what are the particular products or services the

organisation can provide and to what markets in a distinctive manner with the resources the organisation can mobilise. A desired position in a predicted future world, that is an organisation "mission", is to evolve a trajectory or flight path towards the target strategy Yavitz and Newman (1982). In addition, to achieve the target strategy, what an organisation needs to do is to bridge the gap between the selected strategy and short run action. The force of any strategy is closely tied to the consistency and reinforcement of other practices within the company. Clearly, strategy is not a separate device that can be merely implemented onto an existing organisation. If it is to be effective, it must be made an integral part of the total management process and system. Another important issue needing attention is organising the corporate headquarters to support the strategy. The strategic action usually takes place at the operating division and business units levels. This is due to the fact that these units are on the firing line, that is where sound organisation is crucial. Nevertheless, supporting organisation at corporate level is also necessary. The corporate headquarters can either and/or hinder the execution of corporate strategy. The organisation's view of strategic planning should not be based on the existence of a Strategic Planning Department. Rather, it is important that provision is made for several vital activities and also that work which should be performed in the operating divisions, not be usurped by headquarters staff.

3.1.2 Strategy for Competitive Advantage

Many organisations can achieve a meaningful competitive advantage by developing an effective strategy. This advantage can be translated into new market opportunity as well as sustaining the current business. In identifying and exploiting a sustainable competitive advantage, the strengths and weaknesses of actual competitors need to be assessed. A sustainable competitive advantage is derived from tangible opportunities to differentiate from competitors. It is generally accepted that a business must stay in tune with the needs of its customers to be successful or even to survive in today's competitive markets. Yet the notion that keeping up-to-date with changing competitive responses is increasingly a problem of

information management OASIS (1989). Skilled information management is the key to increasing an organisation's ability to act effectively, anticipating customer needs, competitor moves and other external simuli. It can also be a powerful catalyst for competitive innovation in products, customer services and organisational arrangements. Janulities (1984) has argued that information systems technology can play a very important role in a corporation's competitive strategy. These are the exploiters of information systems technology who change the way their industries do business, the competitors who use information technology to support their business and the participants who only view information technology as a necessary tool. The dilemma facing many organisations has been the integration of information system technology with a constantly changing business environment. We will be discussing this subject in the next section.

3.2 Information Systems for Competitive Advantage

Effective information management is the key to marketing success and, therefore, the success of the whole business. In the case of the author's own organisation, as with many other organisations, we have mainly directed systems development efforts to the provision of systems to support specific operational business processes. In contradiction to the strategic direction of the organisation, what is needed is an information systems strategy to support the strategic direction to achieve the objectives of the business strategic plan. The strategy needs to aim to develop systems to support operational processes. These are the fundamental systems that are needed before the organisation can be in business and which support the objectives of meeting service levels and minimising operating costs. Such systems make only a small contribution to providing the management information needed to measure and plan the business to meet its overall aims. Our response to resolve this issue is to develop corporate operational and decision support information systems using the new development life cycle model described in Chapter 2. Further, we aim to evolve the corporate information systems, maintaining key information required to run the business with flexible software tools to enable a broad range of end users to access and exploit the information. In Chapter 4, we will discuss

the development of strategic business information systems at the author's own organisation to support the business strategic driving force - the "business plan for the organisation". However, before proceeding to describe case study live commercial business information systems projects, it is worthwhile describing briefly the business environment in which these systems have been developed for.

3.3 The Corporate Business Environment

The Massey-Ferguson Parts Company, a division of Varity Corporation, provides an international procurement, storage and distribution service for Varity Corporation product divisions. The existing customers include: Perkins Engines, Massey-Ferguson, Pacoma, M-F Industrial, Massey Combines Corporation and Dayton Walther. Massey-Ferguson Parts Company is based in six countries, operating from five base distribution centres with four languages, in six currencies and serving approximately 5,000 plus outlets world-wide in 150 markets, each with their own unique language, currency and culture.

European business has a high export content and is affected through four distribution centres, Urmston Manchester (UK), Athis (France), Fabbrico (Italy) and Eschwege (Germany). Each distribution centre has its own management structure to control full range of functions, e.g. Order Processing, Warehousing management, Inventory management, Accounting, Specification and Procurement.

North American business is primarily domestic to US/Canada and is effected through a master distribution Centre in Racine, Wisconsin, and eleven smaller parts distribution centres. Here a centralised approach has been taken with the Parts distribution centres providing warehousing and order processing, but all other support being provided from Racine. Each year the Massey-Ferguson Parts Company handles over 6 million order lines of which 2.5 million are requested for same day despatch. Massey-Ferguson Parts Company recognises that in today's environment, it has to aim to be responsive to its customers in terms of the support and

service it provides. In order to do this, it will:

- pursue activities which are consistent with divisional strategies and objectives.

- provide a service which is competitive in terms of cost, quality and availability.

- develop flexibility in responding to its different customers' needs.

To this end, Massey-Ferguson Parts Company commissioned its business development group with the strategic mission to examine the organisational structures, current practices and service levels provided at each location. A full-time core team was formed in late 1985, consisting of two parts distribution specialists in the area of in-bound and out-bound logistics, and two computer professionals from the Management Information Systems group, with the author being a member of the team. The team was further supported by part-time members from divisional sales and marketing and other departmental representatives.

3.3.1 The Business Requirement

The team collated and used the information gained and gathered to create a working document containing key weaknesses and recommendations. The document (MF-Document, 1987) identified series deficiences in the area of out-bound logistics systems, specifically in the areas of customer order processing and marketing intelligence. The current interface between the Massey-Ferguson Parts Company distribution centres and the dealer/distributors in terms of Order Entry and availability of order status information is to varying degrees in need of drastic improvement as much is based on exchanges of paperwork, which is both slow and unreliable. The order processing system varies by location. In Europe we have old batch systems which are cumbersom, inefficient and unreliable (e.g. the average time from receipt of a stock order to printing picking tickets is 3 to 4 days). North America has an excellent business system, but it

is over 17 years old and utilises communication techniques developed early in teleprocessing. Applications were constructed in low level languages, where maintenance personnel are not readily available - nor affordable.

As for In-bound logistics, the document (MF-Document, 1987) identified serious deficiencies in the area of forecasting and material management. The key fundamental weaknesses identified were the following: poor material control and too many inadequate suppliers, distribution and transport, lack of in-house expertise to manage transport contracts in today's changing distribution environment, distribution centres state of technology and the ability to cope with fast moving spare parts, people - skill mix and customer awareness and accountability. technically obsolete systems which need updating to meet business needs and give the Parts Company a competitive edge.

The document proposed the formation of three strategic business projects supported by computer systems projects and outlined with the business objectives of each. Projects identified were the orders management, marketing intelligence and material management.

(i) The Worldwide Orders Management Strategy

The team identified an urgent need for a new world-wide orders management system which will be closely integrated with the material management and marketing intelligence systems. The new system to be developed will replace the current technically obsolete systems and will have the following business objectives:

The business objectives:

- Provide an effective mechanism for Dealers/Distributors to communicate parts requirements and obtain information thereby improving company image, developing loyalty and promoting product and aftermarket sales.

- Reduce the decision costs associated with processing orders and providing information to customers.

- Maximise utilisation of assets through computerised visibility of world-wide inventories.

- Provide the material management system with only one true customer sales demand, including reversals for returns, to enable more accurate forecasts. To work with material management to provide a cost effective method to control the internal distribution of inventory.

- Allow life cycle visibility of orders and provide the new marketing intelligence system with timely and accurate information.

- To reduce the management information system costs through the elimination of maintenance and equipment associated with five obsolete/independent order processing systems.

- To stop the excessive costs associated with patching the existing order processing systems and to denote these resources to the development of strategic solution.

(ii) The Worldwide Marketing Intelligence Strategy

The need for a system to provide the information that allows the sales and marketing end user to analyse their parts sales information in a flexible way. The business significance of this system is that it will become a major business tool to develop and monitor targetted sales initiatives and improve the performance of the business. The emphasis of the system is on

flexibility - in addition to agreed standard reports provided, the end user will be supplied with computer based tools to analyse and report information appropriate to the business needs. The business objectives of the marketing intelligence system identified to be:

The business objectives:

- monitor and analyse the volume, value and profitability of their parts sales at any level within their sales organisations from individual customer (Dealer/Distributor) account to complete market.

- establish budgets/targets at various levels within their sales organisations and to measure performance against them.

- create sales campaigns and incentive programmes and monitor their effectiveness.

- monitor and analyse the sales performance of categories of parts.

- accumulate historical sales performance information to assist in identifying trends, refining forecasts and formulating business development programmes.

(iii) Material Management

The team identified an urgent need for a new material management system which will be closely integrated with a new orders management system described previously and a purchasing system. This system will be flexible and enable the world-wide parts inventory to be managed as 'one inventory'. It will not impose undue complexity on one client in order to meet the needs of the other client divisions. It will provide a tool which allows management to directly influence demand forecasts and inventory levels, in other words to balance service

levels with inventories. The business objectives of the material management system are as follows:

The business objectives:

- allow world-wide parts inventory to be managed as 'one inventory' with stock allocation via the master stock control system against individual warehouses.

- provide if' capabilities to simulate different demand forecasting and inventory holding alternatives.

- integration with orders management system to simulate availability for proformas, tenders and abnormal demand.

allow scheduled orders to be loaded and included in future demands.

- identification of discrete customer demands at all stages to facilitate direct shipments and trans-shipments, particularly on urgent orders and specific fast lane (fast movers) parts.

- speed up the whole logistics cycle significantly reducing the lead time from identifying a requirement to placing an order on a supplier.

- segregate demands to reduce cycle time to customers and minimise distribution/handling costs.

- support simulation of inventory requirements for variable fill percentages by season by location/area by product.

The document containing the recommendations was then sent to the Varity Corporation's Corporate Headquarters for investment approval - initially to start the detail investigation for each project. This approval was obtained and three project teams were formed: the orders management, material management and marketing intelligence projects. The author was given the project management responsibility for the orders management and the marketing intelligence systems.

3.3.2 The Business Information Systems Strategy

The systems development strategy is driven by the strategic mission identified in the strategic business plan, which will result in integrated operational systems supporting a single parts company business. This will result in the replacement of a heritage of ageing systems originally designed to support individual locations. This strategy aims to support new business organisation restructure, new business practices and resulting cost reduction. In addition, the strategy will also focus on management information systems. Such systems make contribution to the processes of decision making, control and management of the business or its future planning. The strategic planning activities discussed in chapter 2 are used to formulate the information system strategy. The total systems development and operating strategy consists of five major initiatives. They are:

(i) Operational System Strategy:

The objective of this strategy is to replace all the operational systems with new major systems mainly in the areas of inventory management, orders management, Purchasing and Finance systems.

The Inventory management system is a package based solution which has resulted in single demand forecasting, inventory planning and material procurement mechanism for all locations. An additional on-line system is used to manage the expediting of material from suppliers with fast communications via a telex link. The Purchasing system administers the purchasing process and provides access to purchase prices across the Massey-Ferguson group. The orders management systems strategy results in the replacement of the current three individual order processing systems by a flexible system supporting one-stop order shopping throughout the company distribution network with a single customer communications system. This strategy will follow a modular approach with progressive implementation. We will discuss the development of the orders management strategy in detail using the new systems development life cycle model. The Finance system strategy includes the implementation of accounts payable and export accounts receivable packages.

(ii)Management Information Systems:

Three specific projects are pursued in this area. The Marketing intelligence system supporting a major database containing Sales and Marketing including Product Marketing definitions. The second is an executive information system (EIS). The senior management of the company will undertake a planning process which will culminate in the definition of the key information that is required to provide statements of performance, which can be used to actively support monitoring and control of the business. Such a system will provide for the integration of information from many sectors of the business (a mix of computer based and manually recorded data), a weakness of operational systems, the latter being purpose built design for processing - efficiency being a high priority. Systems support will ideally be provided through a package operating on a distributed computer, to which operational data can be transferred from the central computer as required. The third project is the end-user decision support systems on pcs in the areas of sales and marketing.

(iii) End User Decision Support Systems:

The process of placing pcs and mini-computer technology into the hands of professional end users who are running and planning the business. The process is actively encouraged with self control exercised through multi-disciplined groups - communities of end users with a common interest to share information, to put into place appropriate standards and to ensure that the company avoids the vulnerability of key knowledge being possessed by a single person.

(iv) The Hardware Computing Strategy:

The core operational systems are an integrated set of orders, material, product and financial systems which will continue to process on an IBM or compatible mainframe computer. Distributed computing is performed on IBM mini hardware (9370 and 4331) machines.

(v) Telecommunications:

All locations are connected via the "Varity" network, including electronic mail capability.

3.4 Conclusion

In the proceeding sections of this chapter, we stressed the selective nature of strategy.

Strategy concentrates on only crucial thrusts and centres on relatively few characteristics and assumes that other features of company activities can carry on satisfactorily. Yavitz and Newman (1982) have argued that "care must be exercised that neither the planning system nor the executives using it inject so many dimensions that this selectivity gets buried". They have also argued that "as an embryo strategy progresses through the cycle, although that progress may be halted to overcome a major uncertainty or to develop commitment, the eventual outcome can and should be a dominant guiding mission for the business". A well defined strategy will articulate commonly held corporate goals and objectives. It will also focus and stipulate the broad information system requirements of the company. The strategy will form a framework and points of reference for the development of strategic information and decision support systems. In developing these systems, opportunities should be conducted within the context of competitive strategy development. Only then the organisation will be sure that all the options have been explored. Furthermore, for the organisation to identify all strategic information systems opportunities and to defend against strategic information systems's threat from the organisation's competitors, then the organisation must allow for active participation of the systems professionals in the strategic planning process. Developing competitive weapons from information systems is not done without effort. In addition To achieve maximum benefit requires a joint effort on the part of user community and systems professionals. However, The dilemma facing the systems professional is that the demand for new systems from the business community to address tactical business problems, while the strategic mission continues, is clearly outstripping the ability of systems professionals to supply these systems. What is needed is that the information systems strategy should focus on two areas of systems development. Firstly, the strategic systems supporting the strategic direction of the organisation. Secondly, tactical systems developing strategy supporting one-off projects undertaken by the organisation and the support for day-to-day activity of the business.

In chapter 4 we will describe strategic live commercial system development projects with the aid of the new model.

CHAPTER 4

THE NEW SYSTEM DEVELOPMENT LIFE CYCLE MODEL IN USE

4.1 From Concept to Practice

The corporate/organisation's strategy is described in the preceding chapter as a powerful tool. From a business viewpoint, it pulls together the activities of the various business units into a balanced, synergistic program. In addition, it proposes a plan for the use of scarce resource and endorses missions for operating managers. Further, targets for overall results are set. This strategy will address long-term and tactical short-term issues. In turn, these decisions will have a direct bearing on the formation of the systems development strategy. The business strategy and the system strategy need to be tested for workability. No strategy is well conceived until its workability is evaluated. If the chances of its being carried out are remote or the cost of doing so is very high and benefits identified cannot be obtained, then the strategy itself should be at least reassessed. Following this approach, we are taking the information systems strategy described in Chapter 3 through the workability test. The systems strategy is implemented with the aid of the new systems development life cycle model described in Chapter 2. In turn, this implies that we are testing the new life cycle model for workability by developing life commercial international information and decision support computer systems and thus we will be using "action science" methodology to test our model. Action science is described by Argyris et al, (1985) as a science of practice whether the professional practice of administrators, educators and psychotherapists or the everyday practice of people as members of families and organisations. Action science calls for basic research and theory providing that they are intimately related.

The case studies described in this chapter are live commercial international systems development projects developed at the author's own organisation, Massey-Ferguson. The author being responsible for all aspects of activities within the systems development for these projects. These projects are based on the business information system strategy discussed in Chapter 3. In this chapter, we are interested in two specific strategies - the worldwide orders management strategy and the worldwide marketing intelligence. The worldwide orders managment strategy consists of operational and decision support strategies. The marketing intelligence being of decision support systems strategy. We will now describe how these projects were developed using the new systems development life cycle model. In addition, in section 4.3 we will discuss methodology for evaluating the new model. The discussion is in the context of these strategic projects.

4.2 Case studies-Portfolio of Examples Application

This section will desribe the two major projects to support the strategic corporate business direction at the authors own organisation Massey-Ferguson. The projects are live commercial worldwide information systems developed with the aid of the new model. The first project is the worldwide orders management and the second is management information system strategy which involves the development a worldwide sales and marketing intelligent information and decision support systems.

4.2.1 Worldwide Orders Management System Project

In this section We will describe the activities and deliverables within each phase as it was applied using the new systems life cycle model.

(i) **The preparatory phase**

- **Scope and Objectives of the Project:** This project was embarked upon in the early part of 1983 to support the business strategic direction to sustain the competitive advantage and to specifically provide facilities to achieve the objectives identified previously in Chapter 3. From the business standpoint, the intent of the orders management system was to provide a new improved universally acceptable vehicle for managing orders, not to simply duplicate functions as they are currently performed. Wherever practical, common worldwide customer interfaces. Customer interfaces consist of human-computer interfaces or simply common forms and documentation (e.g. order forms, picking lists, delivery bills). The advantage of single customer interface is that it would make it easier for the customer, who will be placing order on multiple locations and who will be receiving shipments from multiple locations and for those customers who have multiple franchises, to do business with us. The benefits and capabilities of the new orders management worldwide systems were set to be:

- Replace the current manual customer interface with a customer (dealer) electronic communication package with a full function parts order processing, tracking and information system. This interface, the human-computer interface, will be designed as a user friendly, improved response, expanded information, single business image with full security utilising modern data entry methods.

- Supporting the concept of "one stop shopping" which is providing each customer (dealer) with a single entry point for all orders and having the system scan the locations. This facility will provide the dealers with a higher perceived parts supply service as parts are automatically directed to alternative locations, rather than backordered and re-ordered later. It will also support processing stock orders from the base warehouse when economically justified and supported by the manufacturing base. In addition, the system allows orders to be automatically directed to any stocking location which could include factories, packagers and third party suppliers.

- The system will provide comprehensive workload and pipeline tracking from order receipt through to despatch. In conjunction with the marketing intelligence system, it will provide sales information by territory, market and customer levels.

- Ability to switch orders between warehouses to balance workloads or bypass industrial relation problems.

- Faster order processing and allocation of orders from stock and thereby faster order turnaround (saving two to three days in Europe).

- Less administrative effort to manage orders. A major area of reduction in Europe would be inter- location expediting due to improve overall service levels, thereby less urgent order placement, less chasing and less paperwork.

- The important facility is the ability to schedule warehouse picking of parts in line with orders available and warehouse capacity. In addition, the flexibility to quickly accommodate changes in warehouses and warehouse structure.

- Further reduce inventories of slow moving parts in Europe via centralisation of safety stocks into a single location. This is possible as the system will automatically allocate and redirect parts to customers independent of their point of orders.

In addition to the above benefits, the project aimed to eliminate the problems identified in the areas of pricing, discounting and invoicing. The worldwide order management strategy included the development of worldwide special pricing and invoicing system. The planned system is part of the total system strategy covering the areas of special pricing and discounting, warehouse order completion, invoice production and interfacing with appropriate marketing intelligence and accounting systems. The aims of the system were identified to be:

- Produce invoices in the most economic and efficient way.

- To provide for pricing and marketing management more flexibility to allow the promotion of various incentive and discounting facilities for the distributor network in order to permit sales development.

- To reduce the incidence of invoicing errors, improve data controls, on-line enquiries and update facilities.

- To improve the cash flow for sales revenue.

- To provide sufficient data to enable the introduction of management accounting routines designed to achieve sales analysis, marketing statistics and control.

- To improve customer service. This will be achieved by meeting customer requirements, thus reducing customer delays, contributing to reducing order turnround within the supply locations and standardising the layout of the export invoices produced.

- **The project Team:**

To start the project, a project manager with a broad knowledge of the business functions was confirmed. Reporting to the project manager a systems development manager and user project leaders were confirmed - user project leaders representing appropriate functions. To conduct the business analysis phase, three systems project leaders with indepth systems analysis experience were confirmed. Further, a steering committee was formed. The committee was made up of five top management representing the locations (countries), project manager and the systems development manager. Appendix (A) shows the project team organisational structure.

- **Establish Methods and Standards:**

The team were asked to attend a one day seminar. The objective was to present the new systems development life cycle model with specific emphasis on activities and deliverables within each phase within the life cycle model. In addition, an overview of and a background to structured development method were given. In particular, why a new approach is needed for systems development, to show how the data and function analysis techniques of the method fit into the systems development cycle. The CACI structured development method was confirmed as the data and function analysis methodology. Furthermore, an overview of project control facilities developed in-house were given. Appendix (B) shows the document, setting methods and standards for the project.

- **Planning the Project:**

The planning in detail of the business analysis phase was constructed with the identification of precise activities which are pertinent to the business analysis of the identified functions, together with the estimation of resources to complete each task within the business analysis phase for the user representatives and the system development team members. The plan also showed training, location visits and deliverable dates. Appendix (C) shows the project plan as it was constructed.

- **Training and Education:**

Two courses were identified as the necessary courses before the commencement of the business analysis phase. Firstly, the CACI function and data analysis course designed for the experienced systems analyst and secondly, a much simpler course giving an overview of CACI data and functional analysis methodology for the end user team members. Both courses were given by external consultants in-house. The content of both courses are shown in Appendix (D).

(ii) **The Business Analysis Phase**

The principle activity within this phase is the investigation activity. Before conducting the investigation, the team were invited to attend a review session. The objective of the session included the confirmation of the business and system strategies to support the overall strategic direction of the corporation.

- **The Business Investigation:**

Although a high level business investigation was conducted during the formulation of the strategy, the team visited all the locations as planned extracting information from top and middle management and identifying the decision makers and how decisions are made. During these visits, the team recorded the basic users requirements and discussed problems identified and validate benefits claimed or identify benefits that may be obtained by developing the system.

- **The Strategic Entity and Function Modelling:**

The team attended a two day session to construct the strategic and function model. The sessions were driven by the systems project leaders and took the form of identifying business entities, entity types, the relationship between them and information required for each entity. In addition, a strategic model for the high level functions was constructed. Appendix (E) shows the entity and function model.

- **Idea Processing, Project Options and Recommendations:**

The Investigation team were asked to attend a five day session to discuss the findings from conducting the investigation. The problems identified and the requirements were examined. In addition, possible options were considered. Three alternatives were considered in detail - the "do nothing", purchase a "software system package" or develop an orders management system "in-house". The "do nothing" option does not solve any business problems. It will not allow the organisation to change the way the business operates in order to eliminate the inefficient use of expensive manpower and prevent the perpetuation of high interest costs associated with an excessive level of inventory. It would not provide a means of replacing the aged and virtually non-supported data processing equipment being used. It would perpetuate the use of twenty year old systems without the skill levels required to support them, which makes the business extremely vulnerable, particularly if any changes to these systems are required. We will remain an "organisation of the part". As for the purchase of software package, the team concluded that there were few packages on the market that are structured to support the complex requirements of orders management. The team argued that packages are most advanced and general purpose in the "planning systems", areas of demand forecasting and inventory management. They are weak in the area of order processing, where vendors find that the requirements of organisations such as ours tend to be specific and packages undergo substantial modification. The "in- house" development option was recommended as the best option utilising extensive knowledge and experience available within the user representatives and the systems development team capable of meeting the projected needs of such systems. The "in-house" solution identified the requirements for three types of systems - operational mainframe systems, distributed systems and decision support systems approach. Based on experiences in developing information systems at the author's own organisation, Massey-Ferguson, the team concluded that developing one singular system to provide required complex facilities would result in a poor quality system and an impossible system implementation schedule within the multi-national and multi-location environment. A modular approach to the system development with progressive implementation was recommended. The orders management system project as it is known will now be segmented into sub projects with

early implementation into locations returning the largest benefits. It was also proposed that the sub projects with the largest benefits will be developed first. In Sections ahead, we will describe each sub project in detail.

- **The Business Analysis Phase Deliverables**

The project proposal document output was published and it contained a background to the project, current business problems, actions to date, recommendations, the strategic entity and function model, estimates of resource requirements, project timing and cost/benefit analysis. The document was sent to Varity Corporation Corporate Headquarters for investment approval. This approval was obtained and the next phase, the detail analysis phase, started using the scope and objectives of the project identified during the business analysis phase.

(iii) **The Detail Information Analysis Phase**

The analysis team was made up of three teams. Each team consisted of a project leader and two systems analysts. Each team having the responsibility for the specific sub project. The team attended a one day session to plan the detail analysis phase. The teams were reminded that the focal point of this phase is the achievement of the cost effective business objectives and that it is important to relate the problems and/or opportunities of the proposal to business data and function which must be fully explored.

- **The analysis of Business Function and Data:**

The team conducted a detailed analysis of the business functions and data using the CACI methodology. Indepth discussions during structured interviews with the business users resulted in constructing the entity and function models. These models were validated during review sessions with appropriate user departments. The strategic entity model was expanded during the detailed entity analysis to construct the detailed entity model. Equally, the strategic function model was fully decomposed during the detailed function analysis to construct the detailed function model. Appendix (E) shows the detailed entity model. To ensure that the teams were fully involved in the constructing of the entity model and function model, one of the project leaders was assigned the task of

co-ordinating changes to the entity and function model, in particular to discuss the changes with the system project manager before the changes are implemented. The resulting models were cross checked by testing the functions through the entity model (the inter-relationship of data and functions). At the end of these activities, the team documented the entity, attribute and function definition including entity relationships, function matrix, attribute function matrix, entity life cycle and function dependency charts. Appendix (F) shows samples of these documents.

- **Confirming the Current Practices:**

  During the detailed structured interviews, the team examined the functions performed by creating and decomposing the function model and they re-confirmed the business practices during the business analysis phase. These practices were reviewed with the user communities using the function model as a communication tool.

- **Auditable Control Requirement**

  The entity and function models were reviewed in detail with the Internal Audit team. The review sessions addressed the audit requirements of control and security. The check list from CICA (1986) was used by the Internal Audit to ensure the requirements satisfied the External Auditors.

- **Project Scope and Assumptions:**

  The teams attended a review session where the scope and objectives of each sub project were reviewed against the function model. The scope and functions to support each objective were confirmed. In addition, assumptions made by each team were presented and examined in detail to ensure all the team members and the representatives of the user community understood the full implication of these assumptions in achieving the objectives of each sub project.

- **Area of Change**

Representatives from the user communities were invited to present to the development team their perception of the way the systems were going to impact their day-to-day activities. These review sessions turned out to be very successful in that they formed a breakthrough for the project team, making the user community relate itself to the systems via the changes in their area (it was real now). Another important output of these reviews included a discussion on IR issues.

- **Business Issues:**

The objective of the orders management project was set to provide a new improved universally acceptable vehicle for managing orders, not to simply duplicate functions as they are currently performed. As a general positioning statement, this means all the locations with their own current procedures, systems and practices, including documentation and screen/report formats used, have to move now to common worldwide forms, visual displays, common definition of order type, etc. This means that all business issues needing resolution have to be highlighted and agreement obtained, some issues before the start of the design phase and other issues before the implementation of the sub project. Appendix (G) shows examples of business issues.

- **Traffic Analysis:**

During the structured interviews and from current systems in operation, data on volume of entities and traffic of functions and entity relationships are obtained and documented. One way of documenting this information is to present it on the entity and function diagrams.Appendix (F) shows traffic analysis document.

- **Review Available System Application Packages:**

During the business analysis phase, information available to the project team consists of the strategic entity and function model and a high level definition of what is required and problems. Examining the suitability of a package to fit the business requirement is difficult. However, it assisted the project team in deciding that "in-house" development to be recommended. While the project team were carrying out the detail analysis, a new package was introduced to the market and the project team were asked to review and examine the facilities provided by this package against the detailed entity and function models. The team concluded that the package cost of 0.7 million (USD) with customising for facilities proposed with multi languages and distributed approach at additional 0.8 million (USD) brings the total to 1.5 million over and above this expense. In addition, the development team would still have to write all interfaces to existing systems. Finally, based on broad estimates, the total cost to install a purchased package would be likely to exceed 2.0 million (USD) where the incremental costs of the "in-house" development option was 1.1 million (USD). There would also be an ongoing user maintenance charge of 0.2 million (USD) annually. Based on these facts, the project team recommended the rejection of the package solution.

- **Cost/Benefit Analysis Revisited:**

The detail analysis activities and deliverables give the project team a complete picture of what is required (hardware/software - systems development to support the agreed function model). This allows the project team to construct a revised and more accurate cost/benefit analysis. Appendix (H) shows the cost/benefit analysis constructed.

- **Project Timing:**

With the availability of detailed entity and function model, the development team estimated a more accurate number of programs required to support the agreed functions. A new project timing was published. Appendix (I) shows the project timing published.

The Detail Analysis Phase Deliverables

Three documents were produced containing deliverables as specified in Chapter 2. The detail entity and function models plus the traffic analysis and attribute definitions form the important base for the design of the logical and physical databases. The pre design approval was presented to the steering committee for approval. The approval was obtained and the design phase started.

(iv) The Design Phase

To start the design phase, the team were joined by two database analysts and two senior programmers. The team were invited to attend a session to discuss the activities within this phase. The objectives of the design phase were re-emphasised - that is to provide a cost effective system design which meets the business requirements and design consideration discussed in Chapter 2. The detail analysis phase re-confirmed the requirement for three types of systems - operational systems, operational distributed systems and decision support systems. The main activities included the design of the databases, the design of the human-computer interfaces and core system processing.

- **Database Design:**

  The database technology available to the project team consisted of two IBM database management system products - IMS DB/DC/DL1 supporting the hierarchical database design and DB2.DL1 supporting relational approach. DB2 was examined as a possible DBMS for the entities, but was rejected for fundamental reasons. A high proportion of data already exists as DL1 under IMS databases and hence the problem of maintaining data on different DBMS. High volume of transactions. Lack of confidence in DB2 by the internal development team. In addition, no inbuilt referential integrity on the DB2 version was available. However, DB2 has the advantages of being easy to change and implement in comparison to IMS. IMS DL1 DB/DC was, therefore, selected as the DBMS and the orders management databases will exist in the security cycle as other related databases. Links between these databases was provided in such a way as to cause

minimal changes to the existing databases. All the databases, apart from the secondary indices, have been implemented using bidirectional symbolic pointers. This enables logically linked databases to be accessed from either side of the relationship and greatly reduces the time required for database reorganisations. OSAM was chosen for the access method as this, at the time of the design, gave better access for on-line transactions. Other access methods considered included VSAM. This method was proposed, which gives better performance or as the databases grow in size (expanding over more than one pack). Appendix (J) shows the databases of orders management.

The design of the Human-Computer Interface:

The detail analysis phase identified two different types of users interacting with the system. The first group identified represented the external users (dealers/ distributors). The second group represented the internal users within the organisation. The external users communicating via a PC to the mainframe. With the dealer/distributor communication package, the dealer/ distributor interface was designed and named "VISCOM". The design consideration included the easy to learn and use interactivity with full security facilities. It was designed utilising windowing (superimposing one screen on another) and "help" screens to easily step the user to place part order entry, part price enquiry and reporting, order information and cancellation ability, stock order submission, electronic mail and invoicing information. The internal user-system interface again consisted of both interactive and batch facilities. When designing the interactive on-line user-system interfaces, the designers considered the efficiency aspects of these transactions. They addressed this by adopting the fundamental design principle - that is whenever a new on-line interactive transaction is implemented, it is vital that its effect on existing transactions are estimated prior to the actual implementation. If this is not done, then all the existing on-line transactions may be adversely affected by one rogue transaction which utilises more of the computer's resources than it should be allowed to. One of the more easily predictable and perhaps the most critical resource requirements of any transaction is its CPU time utilisation. The team used this estimate and an estimate of the daily volume of the transaction. It is possible to predict the impact of the new transaction on existing transactions. Appendix (K) shows the specific form used to list all of the transactions in the proposed system and the expected daily volumes of

each of them. CPU time utilisation was then estimated for the critical transactions within the system. This analysis was performed for the top 50 per cent (in terms of daily volume) of the transactions as well as for any transaction with a daily volume in excess of 100, even if this transaction is not in the top 50 per cent. The general guideline of CPU time of 200 msecs or less was used as the target for any one transaction. Once this analysis was complete, the results were then reviewed to improve the database design. The program specification was only commenced when the timing estimates had been accepted.

Security:

Another important feature incorporated in the design of the total system was the "security" at data and transaction levels. At the transaction level, the use of IBM's security facility "RACF" and the data level where access to the data is checked against the authorisation code using the user-database and terminal ID.

Prototyping:

The internal and external user interface was prototyped. Two categories of prototypes were used. Evolutionary prototyping (described in Chapter 1) was used to construct the external users' (dealers/distributors) interface to the system known as the dealer communication package "VISCOM". The prototyping system was used initially by the development team in conjunction with the users to evolve the design. It was also used to sell the product to the dealers and as a tool to train them. In addition, it was used to test the efficiency of the system in the area of network and system response time discussed previously. Furthermore, the development of the package was carried out by separate teams. The PC based system was developed by the North American development team and the mainframe based system by the central team in England. This approach to developing the package was possible with the aid of the evolutionary prototyping and significantly reduced the complex communication requirements between the two teams. Appendix (L) shows the dealer communication prototype. In chapter 5 we will discuss dealers view of the developed product.

The throw-it-away prototyping approach was used to design screen and report layout for the rest of the system using a software package. Once the user requirements were clarified and the layouts agreed, the development was started and the prototype was duly used to train new members of the development team.

Distributed Processing Requirements

We said earlier that the orders management system aims to integrate the order processing activities of locations in both North America and Europe. In North America, a central major warehouse supporting eleven strategically placed local warehouses covering both the US and Canada. In Europe, warehouses located in UK, France, Germany and Italy. Between them, these locations process some 6 million order lines per annum, varying from 120,000 to 1.3 million. These order lines are placed by over 4,000 customers worldwide from a selection of over 600,000 part numbers. Specific functions considered for distributed processors located in locations and linked to mainframe central processor included order entry and validation, modification/authorisation of invalid orders, management of allocated orders through warehouse picking and packing, with production of supporting documentation and bar code facilities. In addition, decision support type application.

Why Distribute?

In making the decision to go distributed or not, the design team have argued that whether the function is to be supported on one or more will be dependent upon such factors as data consideration (enquiries, creation and modification), whether it is of local or international interest and the volume of it. Further, the cost of performing the function on one computer as opposed to another. In particular, the cost of a PC or distributed processor plus data transmission charges plus incremental development costs. In addition, three major factors were considered. Firstly, the impact upon the business if a particular processor is down and not having the back up of an alternative such as the inability to enter orders, inability to produce warehouse documents for picking and packing parts and finally the inability to produce shipping documents to despatch goods. The second factor was the need for speedy and reliable processing not impaired by long response times, especially transactions

which are high usage and used for urgent enquiry. The third factor was the cost justification of specific hardware in each location. In smaller locations, it was argued that it was not possible to cost justify both a distributed processor and PC. Therefore, in those locations without PC's, it is necessary to provide order entry facilities for stock orders on the distributed processor.

Proposed Area for Distribution:

The functions required to be supported vary from location to location, dependent to a large extent on the size of the location, volume of order lines processed, types of order category processed and mix of domestic and export business. The team attempted to portray three possible scenarios, but within each location there will be variations at the individual location level. A small location processing low volumes of data which dictated that all required functions be developed for a single low cost processor such as a PC, printer and one or two visual display units (VDU's). The medium location, with higher volumes and greater complexity of business requires support for more functions and may justify a small to medium size single processor capable of supporting high speed printers and several VDU's. The large location, high volumes, large user community and greater complexity of business justifies use of both PC's and a distributed processor supporting high speed printers, approximately twelve to fifteen VDU's, laser printers and bar code readers.

In all the three cases, the functions proposed included the order entry, entry of customer complaints and returns, warehouse-bank , limited enquiries, production of export documentation for locations dealing with export business and local reporting for operational and decision support.

Communication Between Processors:

Consideration needed to be given to the required frequency of communication between processors (mainframe to distributed or distributed processor to the mainframe). This varied depending upon such factors as the function itself, requirements of individual locations and categories of orders being processed). In particular, consideration was given to three requirements. First for communication of data, where delays between processing needs to be minimised so that service levels can be achieved by the despatch of goods within what are typically short lead times. secondly, the availability of the data field on one processor to systems running on another processor (mainframe/distributed) including database management systems running on the mainframe-required for tracking information. Finally the requirements for transferring volumes of data on a scheduled basis, so that it is available for scheduled processing.

Communication Techniques between Processors:

Specific techniques considered included the "trickle- feeding" of data, the initiation of a task on one processor "pulling" the data it needs from the other, frequent transfer of batches of data - regular transfers of large volumes of data and the ability to meet schedules required by each location and finally the transactions and programs accessing data on more than one processor, especially with regard to response time and cost.

The Final Decision - Distribute or Not to Distribute?

Three options were presented. The first option proposed a fully centralised single processor providing all the orders management functions and supporting all the locations. The single processor will be supported by PC's or mini processors for data entry facilities on the basis of store and forward basis. The transmission of data between the distributed processor and the host (both ways) at times to be determined by the end user. The distributed processor would support printers and VDU's, including bar codes. The second option proposed included having major functions on the central processor, but functions related to local activities, e.g. the recording of picking and packing and simple domestic invoicing production, will be performed on local mini computers. The third option proposed argued that each continent

would have its own computer processing all the functions with minimal interaction and few centralised functions. Reviewing these options on the basis of costs, functionality and speed of development to meet the business objectives. Option 1 was accepted as appropriate for the environment currently supported, with the recommendation that wherever possible, the databases and functions will be designed so that they can be transported to single distributed processors if required at a later date.

- <u>The Role Of Dialogue Networks within the Design Phase:</u>

  During the design phase, the author, with a selected group of analysts, attempted to use dialogue networks to perform the appropriate activities within the design phase. The activities included the design of screen and report layout for the dealer communication package discussed previously. They found dialogue networks easy to use and improved productivity in constructing the screen in conjunction with the user. However, the problem encountered related to the fact that the system was only available on one work station and, therefore, they concluded that the availability on more than one work station would be required.

  The results of the design phase were presented to the steering committee and the end users for approval. A series of presentations and walkthroughs were conducted with the users at each location for approval. This approval was obtained and the construction phase started.

(v) <u>The Construction and Installation Phases</u>

The construction and installation team was formed. At the beginning of the construction phase, the design team was extended to include programmers and operational staff. We previously stated that we followed a modular approach to the construction and progressive implementation according to benefits obtained. The sequence in which the sub-systems were constructed and implemented was:

. Order tracking and invoicing system.

. Dealer/distribution communication package.

. Warehouse bank system.

Each system was identified as a project with a project leader co-ordinating and leading all the activities of construction and pre and post implementation support. A full project plan for the construction and implementation plan is given in Appendix (C). The team members were given specific responsibilities. Systems analysts were given the responsibilities for program specification, program unit testing, system testing, construction of user guide and help procedures in conjunction with the user project representatives. The programmers were given the responsibilities for coding the program in a specific language, in our case COBOL DL1 was used, the unit program testing, assisting the analyst in testing the total system, assisting the operational staff in setting up the operational scheduling procedures.

4.2.2 Management Information Strategy

The strategic business plan highlighted a significant business weakness - that is of lack of any formal computer or manually based sales and marketing related management information systems. There was a plethora of reports available across the business providing fragmented information at different level of detail and frequency. To address this major weakness and as a result of the strategic business direction, three major systems projects were pursued. The marketing intelligence known as PRISM, the Executive Information systems (EIS) and finally the end user DSS. From the start, the top and senior management were made aware of an important problem based on experience in developing systems for the sales and marketing user community. They were informed that experience has shown that we have not yet succeeded in obtaining the full active participation of senior and middle management to define the information required to manage and control the business. In turn, this has resulted in individual functions identifying their own requirements without relating it to other requirements from other functions in the sales and marketing areas and in some cases totally opposite to the strategic direction as seen by the senior management. Based on these experiences, the project team was formed with specific emphasis on the active participation

of senior management representatives.

The development of this project followed similar steps to the orders management project. All the stages within the new systems development life cycle model were performed as specified in chapter 2. The analysis phase proposed an "in-house" approach for the marketing intelligence system as the nature of the requirement was specific to the varity corporation type of business and therefore, no package could be found to support the identified requirements. In developing the syastem four major requirements were addressed: .

(i) Non-Computer specialist access:

The most important design consideration, that is the design of the databases in such way to allow ond users with no specific skill in programming languages to access these databases. The other consideration, the choice of a user friendly language for these users to construct "queries" to interrogate the database.

(ii) Business data capture:

Interface system to capture data from appropriate systems (e.g. orders management, material management and financial systems) for the creation of databases for the user access.

(iii) Standard reporting:

A system to produce regular reports for users. These types of reports defined to be of a complex nature and, therefore, needing special attention during the design phase.

(iv) Decision support ad-hoc reporting:

A series of "queries" type facilities having the ease of use as a major feature for end-user to request. Furtermore, the need for down-loading information to PC for DSS PC based applications.

The design phase proposed the use of IBM's DB2 relational database management system as a solution with the use of two languages provided by IBM (SQL and QMF) to be used. Appendix (E) shows the entity model and appendix (J) shows the relational database model for the marketing intelligence system.

(v) The Role of Dialogue Networks within the design Phase

During the design and construction of the system a small team was formed to investigate the possibility of using dialogue networks for the construction of DSS type applications. The team was made up of 2 users and the author. The requirement for the extraction of information from the mainframe databases to a workstation or a PC, so that interactive dialogue network is constructed was identified.

It was thought that dialogue network systems constructed with the aid of a software tool can play a major role for decision support type applications. This view is based on the ease of use by which interactive dialogue network is constructed with no requirements for the use of a specific computer language.

4.3 Evaluating the New Model - Evaluation Methodology

The extent of the success or otherwise of any systems development methodology usually relies on the skills of the practitioners and their experience and familiarity with the concepts and mechanics of the methodology. The actual benefits that accrue from using a methodology need to be established. This is done through the use of the methodology and the possibility of considering the views of the systems developer, data centre operational staff, user support (help desk) and the end users of the implemented systems should be considered. We attribute the success of the information systems projects described in this chapter to the proposed new systems development life cycle model. Success is a relative measure and hard to quantify. In order to validate our claim, we need to make an assessment of the model on four major assessment considerations.

The first consideration is the subject of project risk assessment. In this consideration, we will discuss a structured approach to assessing risk which can be carried out at any phase in the system development cycle. Its aim is to promote the objective recognition of risks so that the appropriate degree of timely management attention can be given to those areas which might jeopardise the success of the system. In addition, to ensure that the project meets the objectives we set for it, we need to carry out a risk analysis on the project - in this case the projects discussed previously. The assessment of risk was carried out by the user Project Manager and the Systems Project Manager (the author) in conjunction with the Steering Committee. The assessment technique employed consisted of a generalised checklist of risk factors developed by the IBM Corporation and are used at the authors own organisation, Massey-Ferguson. In Section 4.3.1 we will describe this technique.

The second consideration relates to the activities and the deliverables within the development process with special emphasis on developing processing controls. Processing controls describes the need to ensure that effective processing controls are properly identified, agreed and developed as an integral part of a system's development. Section 4.3.2 describes a specific technique for assessing auditable controls.

The third consideration relates to user involvement in the activities as proposed by the model. This is considered in the context of measure of system quality, measure of system usage and the measurement of perceived quality and user information satisfaction.

The fourth consideration is the systems developers view of the model. In this consideration, the model is considered as a contributor to the productivity and quality by improving communication among developers and users, increasing flexibility of approach, providing consistency across the development life cycle and enhancing the quality of the final application. In addition, the concept of usable systems is considered in the context of the model. In sections 4.3.3, 4.3.4 and 4.3.5 we will examine the management of user involvement, measure of system quality and acceptability and system success due to user involvement.

4.3.1 Project Risk Assessment

The objective of using the checklist is to guide people to think in a general way about the environment in which the project existed as well as the culture and attitudes of the people who will be affected by it. All the questions in the guideline should be answered and, under the heading "more concern" or "less concern", tick the column that corresponds to the assessment for the appropriate project. The inclusion of "yes" and "no" answers in the table should not imply that the exercise is perceived as being a precise one - this is clearly not in keeping with its nature. In reality, one will be concerned with "degrees" of concern and the table can be a useful trigger for registering these. If the person is in any doubt, it is suggested that the "more concern" column is ticked. The objective of the assessment is not to obtain a numerical score at the end of the exercise. Instead the areas that have the highest proportion of "more concern" answers, together with any individual issues that are considered of particular importance, need to be noted. These areas of concern and issues to evaluate the decision you make in developing/implementing the project should be used to ask the specific question - "does this decision help to reduce any of these risks?". At the end of this process, the assessor would have done something positive about every item of concern, or at least resolved to keep it under close review.

	Level of Concern	
	Less	More
1. Business Case		
1.1 Has the business case for this project been fully endorsed at board level?	Yes	No
1.2 Was the business case put together recently? If not, it may be out of date.	Yes	No
1.3 Are any further board level decisions needed on any aspects of the project?	No	Yes
1.4 Is the size of this investment (for instance, in terms of effort, cost, amount of upheaval) significantly bigger than the level which is "business as usual" for your organisation?	No	Yes
1.5 Is this a project that your organisation MUST do (for example, because failure to embark on it will put you at a serious competitive disadvantage or because you simply can't do business unless you have this system)?	No	Yes

	Level of Concern	
	Less	More
1.6 Is this project different in nature from other things that you have successfully done in the recent past (for example, much bigger, much more complex, or much more critical)?	No	Yes
1.7 Are there policy changes in prospect in your organisation that might cause decisions about the project to be delayed or fudged?	No	Yes
1.8 Are there other major changes in organisation, systems, or the way of doing business going on at the same time at this project, which might jeopardise its success?	No	Yes
1.9 Will any problems with this project be a matter of public comment (for example, in the press)?	No	Yes

	Level of Concern	
	Less	More

Technical Environment

2.1 Are you trying to do something which is considered leading edge in your part of the industry	No	Yes
2.2 If there is software to be developed, is it of a size or complexity that is greater than your organisation has tackled before?	No	Yes
2.3 Is there a source of expertise conveniently available to support any new components (hardware and software)?	No	Yes
2.4 Are any of the requirements for the performance, reliability, or availability of the system more severe than those you are used to achieving?	No	Yes
2.5 Have you planned adequate machine facilities for development and testing activity?	Yes	No

	Level of Concern	
	Less	More

3. User Impact

3.1 Are the system requirements well understood, clearly defined, and endorsed by users and their management?	Yes	No
3.2 Do the users know and agree the realistic dates by which they will really start to feel the benefits of the system?	Yes	No
3.3 Do the users understand the real costs of the system (for instance, in terms of extra work which they will have to do in order to provide data which will benefit someone else, not them; or work involved in collecting or cleaning up data)?	Yes	No
3.4 Will the user definitely agree any changes to his working practices required by the system?	Yes	No

	Level of Concern	
	Less	More
3.5 Does the system represent a significant culture change for the organisation in order to realise the project benefits (for example, improved standards of data accuracy, analytical capabilities of clerical staff, computer system at the heart of the business process)?	Yes	No
3.6 Will the formal structure of your organisation have to change significantly to use of take advantage of the system (for example, new departments set up, new lines of reporting or responsibility, or significant shifts in balance of power)?	Yes	No
3.7 Do the the users understand the extent of the user training needed for this type of system?	Yes	No

	Level of Concern	
	Less	More

4. People

4.1 Is the project heavily dependent on the drive, energy and personal standing of one individual in your organisation?	No	Yes
4.2 Are people with the necessary skills and experience committed to this project for the time required?	Yes	No
4.3 Do you have all the necessary project management skills and resources to support this project?	Yes	No
4.4 Are some of your people being spread too thinly - perhaps because there are too many projects going on at once?	No	Yes
4.5 Will you be self-sufficient technically once the system is completed?	Yes	No

	Level of Concern	
	Less	**More**

5. Project Structure/Management

5.1 Are clear project goals established - the things it is trying to achieve?	Yes	No
5.2 Does your organisation accept the importance of a formal project management system for this project?	Yes	No
5.3 Are you trying to implement the system in a shorter time than you would normally allow for this type of project?	No	Yes

Project Structure/Management

5.4 If several distinct departments, divisions, or groups must work together, drawn from inside or outside your organisation, is the Project Manager used to handling this degree of co-ordination?	Yes	No

	Level of Concern	
	Less	**More**
5.5 Does the project involve people in different locations working together as a team?	No	Yes
5.6 The management of this project may demand the setting up of a project organisation which cuts across normal reporting lines. Do your senior management understand the implications of this?	Yes	No
5.7 Is the success of this project dependent on the success of some other project?	No	Yes
5.8 Are there any critical deadlines to be met?	No	Yes
5.9 Is a director or an executive level manager clearly identified as project sponsor (the person who has most to gain, in business terms, when the project succeeds)?	Yes	No

	Level of Concern	
	Less	More
5.10 Is there a Project Manager who is in command of the project?	Yes	No
5.11 Is there a useful Project Review Board in place?	Yes	No
5.12 Is the sub-project structure and associated responsibilities proving workable and acceptable?	Yes	No
5.13 Are there milestones which are being achieved?	Yes	No
5.14 Is there a system in place for the management of changes and problems which is being used and proving useful?	Yes	No

4.3.2 Developing Processing Controls

The subject of auditable controls arises in several phases of the new systems development life cycle. We have argued previously that these controls are a vital part of the system's effectiveness and must be addressed at the correct time during the development process. The analysis phase identifies the control requirement - the need for and nature of the processing controls. The subject must be considered in its entirety, from the point of data origin to the multiple points of usage, and should include both manual and automated controls, as appropriate. The criticality of the controls must be assessed and due consideration given to the subject of cost effectiveness in relation to business functions identified for controls. The design phase specifies how the control requirements are met. There are various control techniques which can be used to meet the control objectives of processing of data - mainly completeness, accuracy, authorisation, timeliness and adequacy of management trails. The design of the new system will typically confirm the degree to which it can accommodate the automated control requirements in the context of the computer programs supporting the relevant business functions. The construction phase builds the controls, whether the controls are in the form of a computer program or a user procedure. The installation phase uses and tests the effectiveness of the controls which have been incorporated. The audit stage reviews the entire scope of the project and assesses how effectively the controls have actually operated to protect the company's business interests. The Internal Audit were invited to the walkthrough of the deliverables which referred to the processing controls. In the next section, we will discuss a specific control specification form to check for the processing control objectives.

(i) The Control Specification Form

The content of the form anchors the whole subject of processing controls to business functions. For each function, the requirement and technique(s) are completed for the following - completeness, accuracy, authorisation, timeliness and management trail. The following are examples of control specification forms completed for one function of the orders management

system.

(ii) Control Specification Assessment

The control techniques in support of the objectives we identified as completeness, accuracy and authorisation take the form of a matrix, which groups the techniques under two headings - "preventive" and "detective/ corrective". The matrix gives an indication of the likely effectiveness of each control technique in meeting these objectives. The effectiveness ranking guide has control significance as follows:

A = High.

B = Good, when combined with additional controls.

C = Useful, but not adequate alone.

We use this ranking as a guideline, since the real effectiveness will depend on a combination of factors. For example, a rigorously applied "B" technique may be more effective than a loosely applied "A" technique. Furthermore the number of "C" techniques being used is large. This by itself would generally not be considered adequate to meet a control objective. The matrix as it was completed for the orders management project is shown below. Control techniques in support of the objectives of timeliness and adequacy of management trails are shown as a straightforward checklist, following on from the matrix.

	Completeness	Accuracy	Authorisation
1. Preventive Controls			
1.1 Simultaneous recording of a transaction with the business activity itself.	B	B	C
1.2 Properly scheduled system activities (collection, conversion, transmission, processing, reporting and reconciliation).		B	
1.3 Data "prompting" through interactive systems.		B	
1.4 Formatted input screens.		B	
1.5 Input forms designed to match systems needs.		B	
1.6 Visual scrutiny of input.		B	
1.7 Data input key verification.			B

	Completeness	Accuracy	Authorisation
1.8 Authorise transactions prior to conversion.			B
1.9 Data ownership - password security.			B
1.10 Restrict access to authorised users.			B

	Completeness	Accuracy	Authorisation
2. Detective/Corrective Controls			
2.1 Maintain and investigate suspense record of outstanding items.	A	B	
2.2 Systems matching of expected transactions.	A	B	B
2.3 Verify transaction impact against major assets and liabilities.	B	B	
2.4 Analyse variances between actual and budget.	C	C	
2.5 Consecutive numbering of transactions and checking of continuity after processing.	A		
2.6 Reconciliation of the number of source document to the number of transactions processed	B		
2.7 Control totals.	A	B	

	Completeness	Accuracy	Authorisation
2.8 Reconcile changes in assets/liabilities to transactions processed during the period.		B	
2.9 Retention of source document until the transaction are processed. processed.	C		
2.10 Record changes to sensitive semi-permanent data and compare with system listing of accepted changes	B	B	B
2.11 Verification of output by users.		B	B
2.12 Input editing - checks for reasonableness, logical data relationships and self checking numbers		A	
2.13 Process checking - reasonability of results		B	

	Completeness	Accuracy	Authorisation
2.14 Periodic user review of semi-permanent data.	B	B	B
2.15 Detailed matching of processed results to source data.	A		B
2.16 Review transactions for authorisation.			B
2.17 Check "lack of activity" for regular business.	C	C	

3. Timeliness

3.1 Prepare documented procedures for production scheduling and control.

3.2 Use software, to the extent possible, to to reduce discretionary manual procedures and to monitor the performance of the system.

4. Management Trail Controls

4.1 Document and enforce record retention procedures.

4.2 Unique identification for each record.

4.3 Filing in significant and planned sequence.

4.4 System design to allow tracing data backward and forward through the system.

4.5 Logging of on-line and internally-generated transactions (including information such as User, ID and date of the transaction).

4.6 Periodically save semi-permanent data which is changed by overwriting the old contents in the same physical storage location.

4.7 Provide users with data scrutiny and analysis capabilities.

4.3.3. Managing the User Involvement

We have stated previously that the development of a quality system requires the harmonisation of a wide range of business and technical skills and the efficient planning and control of a large number of activities. A user project manager is the champion of the project, whose mission is to drive it to a successful conclusion. In the case of the orders management project we described in Chapter 4, which involved more than one department and location (multi-function, but also multi-location) and the functions provided were aimed at specific business objectives supported by a specific systems development. The question of the project management process of the user involvement was considered from the very inception of the project. This involved the decisions on who is the user project manager - the quality and experience, as well as the membership of the project steering committee and the representatives of the individual functions. The user project manager's problem was choosing the members with appropriate knowledge and skills to represent the user community. Most of the user input came from a few highly skilled, very experienced staff. Their knowledge was important in providing high-quality knowledge of the current business environment and problems, but they lacked in vision - identifying future requirements and seeing the system in achieving business advantage. This problem was difficult to overcome. From the beginning, it was difficult to move them away from their current day-to-day problems. As the project progressed, the user representatives became aware of what was required and gained confidence. This meant new ideas were generated. However, "blue sky" requirements that would entail an enormous systems development resource requested by the users had to be dropped by the user representative after detailed discussion with the developers. The process needed careful consideration as developers often tend to associate complex user requirements as "blue sky" requirements. Our experience shows that this process may cause lack of interest in the project by the end user community due to change in their attitude from **"will get what I want from the new system"** to **"will not get what I want from the new system after all, so why contribute"**. Managing this process is important and should be performed with care. In addition, the developers felt that using user representatives as go-betweens between themselves and end users will cause difficulty in defining and

obtaining the end user requirements. Thus it is difficult to generalise about the opinions of user population from the few users who are most heavily involved in development. They may or may not reflect general perceptions, not only because of their commitment to the project, but because they may not reflect the shifting composition of the population. The top management assumed that the heavy involvement of a few user representatives would give them the complacent belief that they are in touch with the end user community. Another important activity was the marketing of the project so that project supporters are identified and encouraged to market the project further. Roberts (1974) has argued that a sponsor and a champion provide necessary advocacy for any new idea. This view was developed further by Rothwell (1977) arguing that this support is no guarantee of success. In fact, failed projects can have highly visible champions. Leonard-Barton (1987) has argued that "champions and their projects are successful to the degree that they stimulate support in the right quarter". In our experience, when developing a large complex system for multi-locations bringing with it the symptoms of "not invented here", champions or supporters for the project are a mandatory requirement. In our case, the orders management was championed by the managing director of the Parts Company. He in turn marketed the project to the corporate headquarters and created an environment for the creation of network of supporters from each location making each location relating to the project and creating a feeling of "project ownership". The primary challenge facing the champions and sponsors was keeping the users' interest in the project. Regular reviews were conducted with objectives of informing and reminding the users of the benefits provided by the system. In the case of the orders management project, our experience shows that appropriate support that is visible, carefully timed and appropriately positioned was important to address issues and problems encountered. However, visibility at high level can also hinder projects. The main problem facing the project team included a constant justification of resources used, detail project status reporting at a low level to top management. In our experience, project visibility at a high level can be worse than none at all. Furthermore, support at the appropriate time is important. In our case, we received strong support during the analysis and design phases, but reduced during the construction and installation phases. These phases

were seen by the sponsors as technical in nature. Another important factor impacting the implementation of the orders management was the hourly workers' attitude to the system. It was seen as a threat to their job security and, therefore, resisted the implementation strongly. Part of the management's problem was identifying and working with union representatives to develop the needed social and physical support. Our experience shows that careful consideration should be given the social aspect of system implementation and that trade union involvement in the early phases of the project will result in a smooth implementation and acceptance of the system. We will discuss the subject further in chapter 5.

4.3.4 System Quality and Acceptability

The "right first time" principle is strongly supported as a key quality objective by the new cycle model, especially where well defined tasks are involved. However, in our experience, the development activities at the earliest phase of the project scope definition contained areas of uncertainty. In these instances, we felt some degree of iteration was essential to clarify issues, gain understanding and secure user agreement. Furthermore, because of the impact on the remainder of the phases of the project, the need to be "right" was paramount. The practice of the walkthrough described previously occupied a fundamental role in our life cycle as a major contributor to quality. In the context of user involvement, the walkthrough sessions were used as a process of subjecting the specific deliverable to open scrutiny by the user departments relating to the particular function in the presence of the user representatives and members of the development team. The atmosphere encouraged in these sessions was one of constructive criticism with the goal of the early detection of errors and omissions.

To investigate the relationship between user involvement and system quality and acceptance, we conducted a study at the author's own organisation, Massey-Ferguson, with special emphasis on extracting users' views. Before proceeding to the discussion on the study, it is worthwhile identifying who the users are. We classified our users as "owners" of data and as "access users",

who do not own data but who may have access to data. As an owner of data, they carry with it a commensurate responsibility for control over the data and/or specific functions provided by the system or the total system. Throughout this section, we will be using two terms - internal users and external users/ access users. Internal users refers to the users of the system employed by Massey-Ferguson and they are "owner" users and "access" users. The second term, the external users (dealers and distributors), refers to customers of Massey- Ferguson and they are "access" users. In chapter 5 we will be discussing the external users' view of the model.

4.3.5 User Involvement and System Success

The new system development life cycle model we have proposed is aimed at bringing the developers and users closer within the systems development process so that useful, usable, desirable and quality systems are developed. To examine system success due to the involvement of users, we will focus our investigation on two class of outcomes of user involvement-system quality and system acceptance. We proposed that the quality of an information system should be measured in terms of its satisfaction of business requirements. We have argued that the new life cycle model ensures that a clear and precisee understanding of these requirements is communicated accurately across the life cycle and refelected in the final system. This communication has resulted in the user community reporting that the quality of the systems developed with the aid of the model can be measured in higher satisfaction with these systems. To provide data to support a positive or negative relationship between user involvement and system quality, we carried out our study at the author's own organisation, Massey-Ferguson, investigating the role of users in designing and developing the information systems described previously. Involving the user community in the design and development process necessitates more planning and complex activities than it is recognised. It requires attention to the management of the user involvement which has a direct impact on system quality and acceptance.

4.3.6 Improved Productivity-Is the New Model the Answer?

We have argued that the ever increasing demand for the effective and flexible information systems, coupled with shortcomings of traditional techniques in the development of these systems, has highlighted the importance of deriving more powerful concepts, techniques and tools for improving software productivity and functionality. We believe that the proposed development phases are the correct way to view the various activities involved in the identification, specification, development, use and improvement of information technology systems. As for achieving higher productivity by applying the proposed model and a device which provides automated support for production of deliverables, we used specific productivity tools from McDonnell Douglas Corporation (ProKit*Workbench and PRO-IV) to support the analysis, design and construction phases within the development life cycle. The application of these tools were monitored and evaluated. Benchmark results obtained are discussed in chapter 5 in detail.

4.3.6 Improved Productivity-Is the New Model the Answer?

We have argued that the ever increasing demand for the effective and flexible information systems, coupled with shortcomings of traditional techniques in the development of these systems, has highlighted the importance of deriving more powerful concepts, techniques and tools for improving software productivity and functionality. We believe that the proposed development phases are the correct way to view the various activities involved in the identification, specification, development, use and improvement of information technology systems. As for achieving higher productivity by applying the proposed model and a device which provides automated support for production of deliverables, we used specific productivity tools from McDonnell Douglas Corporation (ProKit*WORKBENCH and PRO-IV) to support the analysis, design and construction phases within the development life cycle. The application of these tools were monitored and evaluated. Benchmark results obtained are discussed in chapter 5 in detail.

CHAPTER 5

RESULTS AND DISCUSSIONS ON THE USE OF THE NEW LIFE CYCLE MODEL

5.1 The New Life Cycle Model and Systems Success

The decision to install the systems described in Chapter 4 necessitates a choice of mechanisms to determine whether the systems described were needed and once implemented, whether it is functioning properly and that the users believe that the information available to them meets their requirements. These requirements identify the needs for evaluating measures of system success which can be viewed as measures of system quality, measures of perceived quality and information satisfaction. The overall project assessment is one way of assessing that these requirements have been addressed. Assessment activities consisting of project management and structure, the business case (business benefits/cost saving) the technical environment of the system (development and implementation) and people skills and personalities. The project plan should make generous use of the checklist of high level activities identified for each project stage and sub-divide them, as necessary, for effective control. This should go some way towards minimising the project risk which accompany the omission of essential activities. However the subject of risk is never far from the surface in the life of the project, especially large international projects, and it must be given regular attention to protect the company's investment opportunities. The assessment of risk carried out using the techniques described in chapter 4 showed a high proportion of "Less Concern" than "More Concern". In section 5.2 we will discuss the result in detail. Another important area contributing to system quality is the processing of data completely, accurately, timely and verifiably. Each of these requirements constitutes specific control objectives which need to be systematically considered. The technique proposed in Chapter 4 expresses the effectiveness ranking guide has control significance of high, good, when combined with additional controls and useful, but not adequate alone. The technique applied to system developed with the aid of the new model by the internal audit section at Massey-Ferguson showed "high" ranking controls. A third important factor contributing to the quality of deliverables within the development process is the "walkthrough" procedures proposed. It is a most effective way of ensuring that a deliverable incorporates, collective brainpower - subjecting the deliverables to open scrutiny by a group of peers, in the presence of the author. In addition, change control procedures we proposed addressed one of the greatest causes of project slippage - the failure to manage requests for system change which are received during the life cycle of the systems project. The procedures was seen by the developers and end users as having the objectives of evaluating the impact of each change request, so that an

informed business decision can be made on whether or not the change should be accepted. it was stated that the procedure be operated with minimum overhead and it must not appear unduly bureaucratic. Section 5.3 discusses the results of auditable processing control assessment in detail. Walkthrough and change control procedures used are discussed in Section 5.4 and 5.5.

As for the measurement of user satisfaction and it's relationship with system success, we propose that it is on perceptional or subjective measure of system success, it serves as a substitute for objective determinants of information system effectiveness which are frequently not available. The subject of user satisfaction is evaluated in the context of the new model being a contributor towards achieving information satisfaction and system success. A survey of internal and external users of the systems developed with the aid of the new model was conducted. The survey has shown that a good quality useable system can be developed with the aid of the model. The overall reaction of those who responded was very good. Section 5.4 discusses the internal and external user' view of the systems developed with the aid of the new model. As for the developer's view of the model, a survey was conducted within the development team at the authors own organisation. The discussion in the context of the model providing the developers a medium for communication with end users, flexibility, consistency, productivity, maintenance and usability. The survey resulted in the developers reporting that, whereas the system development standards described previously Moubarak (1989) ensures that system developed meets users's requirements at the level of individual applications, the new model guarantees that the entire corporate application conforms to the goals laid down by strategic information processing. Support is available for the developer throughout the entire Life Cycle from the original idea and it's specification to implementation of the application and it's subsequent replacement. The framework for all development based on data and function analysis. The heart of the architecture is the integration of the end-users within the development process. In Section 5.5 the developer's view is discussed in detail. Another important factor contributing to system success is for the system to be fully utilised, the point when the system will be accepted. The process of acceptance was described by the participants in our survey as an adaptive process. the adaptive process starting at the implementation phase with the system ownership by the users of the system.

The survey showed that system changes were recorded within the change control procedures described in Chapter 2 were due to various misalignments between the system functions through

which the solution was delivered to the users of the system. Business and organisational factors, systems and technical factors and political factors were identified as specific misalignments. Implementation as an agent of change including these misalignment factors are described in Section 5.7.

5.2 Project Risk Assessment-Result

The project risk assessment was carried out by the user and system project managers in conjunction with the steering committee. As stated in chapter 4 the objective of using the checklist is to address the areas that have the highest proportion of "more concern" answers, together with any individual issues that are considered important. The project manager addressed every item of concern and resolved issues highlighted during the assessment. Specific areas of concern highlighted included the following:

(i) The business Case:

Specific areas of concern included the implication of policy changes in relation of business direction for example of acquisition of third-party business, in particular what type of business. The issue was addressed by the top management stating "the systems need to support functions relating to core Massey-Ferguson distribution business".
A second area of concern related to the criticality and the size of the project. This concern was based on requirements for the commitment of all companies with the varity group. Re-assurance was obtained by the steering committee.

(ii) Technical Environment:

Specific areas identified were performance reliability and availability. The concern expressed in relation to the system being available 24 hours with over 2000 internal and external users accessing the system. This requirement resulted in the construction of specific standards for the design of on-line transactions discussed in chapter 2.

(iii) User-Impact:

Two related areas were identified. First the change to existing working practices and the culture

change in relation to a single site mentality to a common single point of contact for customers of Massey-Ferguson on a worldwide bases. The problem was addressed by forming a working group consisting of human resources, representative of each business unit and project personnel was formed resulting in the creation of a package, containing full training and education courses for the clerical and commercial staff.

5.3 Control Specification Assessment-Results

We stated previously that the audit phase reviews the entire scope of the project and accesses how effectively the control have actually operate to protect company's business interest. The control specification form described in chapter 4 was used by the internal audit department to perform the assessment. The assessment resulted in the preparation of the audit report for the senior management. The report started that all specified auditable controls were considered adequate to meet control objectives. Specific issues highlighted included on-line security and the use of password at individual user level rather than collective department/terminal-id level. This recommendation was accepted by the user management and implemented.

5.4 The End Users' View of the Model

The subjects chosen in our study represented two groups - internal and external users. The internal users group was made up of 27 users with a mixture of good formally educated experienced subjects and experienced with no formal education background subjects, but all have the experience of using interactive and batch systems developed using traditional systems development methodologies,their views are discussed in detail in section 5.4.1. The external users were represented by selected number of dealers from the Massey-Ferguson dealership community(180) dealers from the North America were chosen. in section 5.4.2 we will discusses the results of a questionnaire sent to these dealers. In addition, the role of the internal and external users during the development process are discussed in section 5.4.3.

5.4.1 The Internal Users' View

This group consisted of the following members:

- Three distribution managers, one from each location.

- Three warehouse managers, one from each location.

- Three distribution order processing supervisors.

- Three invoice processing supervisors.

- Three export documentation clerks.

- Three warehouse supervisors.

- Four pickers and packers.

- One pricing manager.

- Two Sales managers.

- Two Marketing managers.

To obtain the group's view in our study, we interviewed members of each group. The interviews were conducted by the author at the members' own environment over a period of six months. The discussions with the users were concerned with the following areas of interest.

(i) **New cycle model as a framework for communicating business requirements to systems developers:**

The discussion took place in the context that the new model ensures that a clear and precise understanding of users' business requirements are communicated accurately and reflected in the final system. The managers reported that the business requirements identified at the strategic planning phase have been supported by the system and that specific economic benefits have

been obtained in the area of decision costs, reduced inventory levels, increased revenue by introducing new marketing pricing campaigns and improved service levels. However, they all felt that phase implementation of the project prolonged the realisation of the economics benefits. The phase implementation was defended on the grounds that a major, complex and large system such as orders management has a major impact on the organisation's internal structure and users of the system. In addition, the logistics of supplying over 2,000 dealers/distributors customers of Massey-Ferguson with communication packages so that they can place orders and communicate with the new system all at the same time was seen as a task with serious risks and major business impact. This was accepted and they admitted that the internal user organisations could not absorb the changes that the new system would bring.

(ii) The new cycle model as an agent for reducing business complexity:

The idea processing and the brainstorming sessions taking place at various phases of the project allowed for complex business problems to be identified before rushing around looking for a solution. The managers reported that the brainstorming sessions elicited good ideas in a fairly short time. The ideas acted as catalysts provoking everyone to come up with more thoughts and new lines of thinking. However, one good way they felt to help along the sessions would have been for someone outside the project team and immediate organisation to sit in on the session and to have cross-fertilised the run of ideas by bringing knowledge on related subject. In our case, all the sessions were led by a user or systems project manager. The argument presented by the managers is that the complexity remains even with the introduction of the new systems and procedures. An example given by the managers included the pricing, invoicing and discounting rules - has the model failed in reducing business complexity? We felt that the model provides a framework for exchanging ideas and allowing the users to see the complexity and thus ensuring that the complexity is avoided. It could also be that these problems start further back in the organisation or the work process than first appeared. In the case of the example given, market forces forced the need for complex pricing and discounting structures.

(iii) Decision making performance:

A point made consistently by the managers during the analysis phase was that the information provided by their current systems was not sufficient and that they found too great a mismatch of information produced from different systems. As one manager said, "It is like comparing apples

with oranges". Furthermore, customer service level data and various data extracts for decision support applications originated from local systems, each with their own rules and algorithms. These problems identified the need for common and comprehensive decision support type reporting. The material thus gathered was used to construct a function called "measure orders management activities". The objective set for this function was set to be: Establish performance criteria, capture and evaluate measurement data and to provide performance reports. All the managers concluded that the four major objectives of this function allowed them to run their departments to improve customer service to their clients. They reported that setting performance criteria at each location for individual clients using facilities provided by the system allows for the analysis of performance of order turnaround by order type by location. In addition, pipeline analysis is supported by extracting data at selected control points. As for the analysis of data needed to develop management information, all the managers agreed that the approach taken to provide a selected standard report on an agreed schedule and with additional support by extracting information on a daily basis from the operational mainframe databases for downloads to their PC's was the most efficient way of using resources. They reported that the use of software tools such as Lotus 1-2-3 on this extracted data and other DSS type applications has improved their day-to-day decision making process. The main problem identified by the managers was the introduction of computing-literate personnel to their departments to carry out tasks in constructing "what if type" analysis. The study showed that the managers strongly supported the prototyping of the human-computer interfaces and felt time allocated to these activities during the early stages had resulted in producing quality and usable information. As one manager commented, "It is bound to be right, we designed it, you just programmed it." The discussions with the subordinates were concerned with discussing the essential elements of the system which they have contact with - the human-computer interface (on-line screens and reports). Although the participants had ideas about what the system would do for them in terms of information, decision support and usability aspects, since they played a major part in designing their interfaces, the extent of which varied according to the participants' education, career history and previous computing experience, especially their experience in using interactive computer systems. The interactive facilities provided consisted of PC assisted on-line transactions and on-line transactions on traditional VDU's for enquiries only. The essential elements of the discussion on the interactive human-computer interfaces included the following:

- Simplicity of commands.

- Consistency of commands.
- The availability of menu and help facilities.
- Presentation of information on displays in a meaningful and simplistic way.
- Adequate response time during the interaction with the system.

All the participants reported that PC assisted facilities proved to be more superior to the transactions on the VDU's and strongly requested the availability of extra PC's throughout the departments.

5.4.2 The External Users' View

On July 28th 1989 a survey was sent to selected North American dealers-180 dealers users of the dealer communication package developed within the orders management project described previously in chapter 4. At the time of writing the thesis we have received 29 responses. The level of responses was disappointing but it may have been partially due to the busy time of the year when many of the dealers may simply not wanting to take the time to answer the survey. While the number of responses were disppointing the overall reaction of those who responded was very good. A summary and result of the surveys will now follow:

(i) Survey Questionnaire

The following questions were asked:

1. How long have you been using the package?

Please tick one of the following:

30 days, 60 days, 90 days, 120 days, 120 plus days

Results:

30 days	3
60 days	6
90 days	3
120 days	4
120 plus days	13
	--
Total	29
	--

2. How were you trained and how do you rate your training?

Method of training (please tick):

Telephone, regional training centers and visit to dealer locations

Results:

	No.	Exc.	Good	Adq	Poor
Telephone:	20	3	7	9	1
Regional training centers:	8	2	3	3	0
Visit to dealer location:	1	0	0	0	1
Total	**29**	**05**	**10**	**12**	**02**

3. How often do you use the package functions?

	Daily	Occasionally	Never
Parts Ordering	28	1	0
Parts Inquiries	15	14	0
Electronic Mail	22	7	0
Wholegoods Info.	0	16	13
Retail Contract	1	10	18

4. How would you rate the package functions?

	Excellent	Good	Adequate	Poor
Parts Ordering	11	15	3	0
Parts Inquiries	6	18	4	1
Electronic Mail	8	15	6	1
Wholegoods Info.	0	11	6	1
Retail Contracts	2	7	4	0

5. Suggestions for improvement

Suggestions offered	-	24
No suggestions	-	5

A wide range of suggestions were offered. Each suggestion was reviewed to determine if changes were required and could be made.

6. List any additional features you feel would be beneficial.

1 Warranty Claim Submission

3 Wholegoods Order status

2 Parts back order list

2 Parts back order promises

2 Transmit credit application

1 Retail contract payment calc.

1 Parts return submission

7. How do you rate package response line Help Desk support?

Excellent	=	11
Good	=	14
Adequate	=	04
Poor	=	00
		—
Total		29
		—

8. How would you rate the overall performance of your package?

Excellent = 6 **Good** = 16 **Adequate** = 7 **Poor** = 0

9. If you have another manufacturer's communication system, how does the package compare?

No other system - 23

	Better Than	Same As	Worse Than
Ford	1	0	3
Chrysler	0	1	0
Komaster/ Dresser	0	1	0
Total	1	2	3

10. Do you feel you made the right decision when you purchased the package?

Yes - 29

No - 0

(ii) Results Summary:

The survey has shown that a good quality usable system can be developed with the aid of the model. The overall reaction of those who responded is very good. Some important statistics obtained are:

- 100% of the dealer's think they made the right decision when they purchased the package.

- The response line support(help-desk facilities) was rated very high:

. 86% Excellent or Good

. 14% Adequate or Good

. 0 Poor

- Overall package performance is very favorable:

. 76% Excellent

. 24% Good

. 0 Poor

5.4.3 The Role of Users within the Life Cycle

We have argued previously that effective user participation within the development process enhances the final product developed. This effectivenes depends on appropriate role played by the participants. The table below illustrates the role of user envolvement in the systems development life cycle activities during the projects.

A = Senior Management, B = Executive-Business-Strategist, C = Middle Management, D = User representative, E = Supervisors, F = Clerical Staff, G = User Project Manager

The Life Cycle Phases	High	Low
(1) The preparatory phase:		
. Definition of scope and objectives of the project	A,B,C,G	
. Organising the project team	G	
. Establish methods and standards	G	
. Planning the project	G,D	
. TRAINING and education	G,D	A
(2) Business Analysis phase		
. Strategic Planning activity	B,C,D,G	
. Agree business objectives	A,B,C,G	
. Conduct the business investigation	D,G	

The Life Cycle Phases	High	Low
. Strategic entity and function modelling	D,E,G	
. Idea processing, project options options and recommended project proposal.	B,C,D,E,G	

(3) Detail Analysis Phase

. Confirming the current practice	C,D,E,G	
. Auditable control requirements	D,G	
. Project scope and assumption	D,G	E
. Area of change	C,D,G	
. Business Issues	C,D,G	A
. Traffic analysis	D,E,F	G
. Review available system application packages	D,E,F,G	C,B
. Cost/benefit analysis revisited	B,C,D,G	A
. Project timing revisited	D,G	A,C
. Business system spec.	D,G	C,E

. Technical approval

The Life Cycle Phases	High	Low
(4) Design phase	D,E,F,G	C
(5) The construction phase	D,G	
(6) The installation phase	D,E,F,G	B,C
(7) The Audit and system evaluation	C,D,E,F	A,B

5.5 The Developer's View of the Model

In this section, we consider the experience of another group referred to collectively as developers. Mostly professionals from the management information services department at Massey-Ferguson all members of the project team developing projects described in chapter 4. The participants of our group consisted of:

. 3 system project leaders.

. 3 systems analysts.

. 1 senior programmer.

. 1 database analyst.

. 2 programmers.

To carry out our investigation, we interviewed members of the group. The interviews were concerned with discussing specific benefits that the new model would bring to reduce the risk associated with application system development in particular projects described in chapter 4 and the role of developers within the development process.In the next two sections we will discuss these benefits and the role of developers within the development process. Section 5.5.1 discusses these benefits. In Section 5.5.2 we will discuss the role of developers with the development process.

5.5.1 Benefits Identified

The following are some of the benefits identified:

- Better Communication

The model as a media for communicating ideas and concepts among the developers and the end users.

- Flexibility

The model as a framework that allows top-down approach for strategic systems development including tactical, pragmatic approach to address immediate needs (tactical solutions with strategic vision).

- Greater Standardisation

By providing a checklist of activities and deliverables for each phase using the same vocabulary, the end users and the developers can be sure the meaning and intent of their work is carried through to the implemented system. The model ensures that this consistency applies across all systems.

- Improved Productivity

The model as contributor to systems development productivity throughout the development life cycle - transformation from manual to automation. It is through integration that productivity and system quality improvements are made possible.

- Maintenance

The model as a tool controlling and co-ordinating maintenance alongside simultaneous development projects.

- Higher Quality Applications

The role of the model in designing reliable and responsive systems. Application that focuses on calculating errors early in the development process-before detailed design and implementation.

Achieved through better communication, increased productivity and greater standardisation so that higher quality applications are developed-has the right functions in it so that users can do their work better and like it.

As a starting point, the discussion centered around the role of developers within the systems development process and what they require from a methodology. They defined their goal as designers and developers of information systems reflecting user needs and to convince the users of the final product of the developed systems. Systems that support business functions that allow users to do things they want to do. But more significantly, these systems must evolve rapidly to support change in the business and environment it serves. To achieve this responsiveness requires a solution to the problems that traditionally fail them. Mainly user requirements not clear,development takes too long, it costs too much and the quality of the resulting system is poor. The sceptics of the group initially argued that the structured approach as proposed by the new life cycle model would lengthen the development process. They would be adding on more work and, therefore, more time and effort would be required. In addition, they argue that because the business requirements change faster than developers can respond, many application systems being developed will never be implemented. This view fails to recognise that the model is designed to address these fundamental problems. Another view expressed by the analyst members of the group is that of "de-skilling". In their view, they felt that the involvement of users as a driving force at the early phases of the analysis would make their role as a subordinate passive role to user needs, rather than the analyst defining what the user should have.

As a communication medium, the participants of the group agreed that the model provides a framework for the communication of ideas and concepts important to the systems development process. The framework is largely based on a combination of phases containing activities and deliverables representing a structured and consistent framework within which the system can be developed. The communication is enhanced by making the communication models pictorial. Verbal models, the traditional means to communicate systems analysis requirements used by our participants, have several defects. Mainly, unless a structured language is used, it is nearly impossible to get an unambiguous meaning for even a relatively small/simple subject. For a model of any reasonable size, it is difficult to check self-consistency. Easy to comprehend modelling makes user participation simple to enforce.

However, our experience shows that during the analysis phase of the orders management project, the analyst members of the group reported that entity modelling was more difficult to comprehend by the user community and that only graduate level educated users could understand the entity relationship. On the other hand, function modelling was reported as the most successful media for communicating business functions and that with the walkthrough sessions carried out to examine the entity and function models, everyone understands the requirements definition and can measure the system against it. The participants have all agreed that the model allows flexibility and precision in the range of problems that can be addressed and systems that can be developed. Systems described in Chapter 4 support this view. Furthermore, they argued that the separation of data and functions as proposed by CACI methodology simplifies documentation, analysis and enhancements due to change in function requirements. Separation of techniques makes the project control easier by ensuring definition of small analysis and design steps. The whole development has calculable and controllable timescales.

During the analysis phase, much quantitative information on data and business functions was collected, which for it to be of benefit in the design and construction phases, then it must be organised and recorded in a central information directory with graphical support for modelling. In our case, the tools available to the developers consisted of a separate non-integrated tool kit with no automatic interfaces. For example, a data dictionary was used as a means of recording the attributes and the entities, but no computing medium for a function model was available. Furthermore, no automatic Interface between dictionary and prototyping exists and thus no use of data dictionary for screen design. The participants all strongly agreed that there is a need for an integrated tool kit that provides graphics facilities to create data and function models and program structure charts with prototyping facilities to help analyse business requirements and model screen dialogues. Behind these facilities, linking them all together, is an active data dictionary which should document all aspects of the system from a high level business overview, through detailed process and data definitions, down to the physical system elements of programs and files. During the initial phases of the project, a study was carried out to evaluate analysis and programming tool. The study proposed that no one suitable affordable tool was available for the company to invest in. However, since then a number of tools have been introduced to the marketplace which can play a major part in addressing the problems identified by our participants.

The participants agreed that such tools can be introduced within the model to improve productivity during the analysis phase. As for the design phase, the participants reported that breaking the deliverables of the design phase into smaller areas is useful. The business system specification is the output of the business system design. Designers use the information gathered during the analysis phase to describe the information system that can satisfy the business' needs. User representatives found this extremely useful to understand the system better in the area of human-computer interfaces. The designer ignores the details of the operating environment at this point. The designers reported this has been the most difficult task for them - the border line between considering the operating environment for technical system design and not considering it for business systems design. The activities and considerations identified by the model in the area of security, distributed approach and the direction towards choosing the appropriate database management system was strongly supported by our designers. However, they all felt that the approach encourages integrated systems designed for centralised data sharing. This view failed to recognise that the activities within each phase can equally apply to de-centralised databases and systems.

The developers identified prototyping as the most important tool they have come across. With sample screens, reports and menu, users can not only validate, but uncover additional requirements previously overlooked. Simulated execution is an added benefit. A model of screen execution can be created to show how screens will behave in production as it tests both navigation and logical criteria. Various levels of complexity can be modelled - from function key screen transfers to complex data orientated navigation. Our investigation shows that the prototyped human-computer interfaces inform of on-line screens and reports layout were not changed during the system implementation and no changes were requested post implementation in two years.

The construction stage invited considerable comments from the programmers. They felt that the model has nothing to offer them as they still code programs in the appropriate language - no productivity gains. This view failed to see that this phase sees the program being updated within the level of information necessary to generate a high level program specification which in turn can be converted to program code using traditional 3rd and 4th generation languages. It is true, however, that a greater productivity is achieved by automatic generation of code. Again, no tools were available to the developers for automatic code generation.

In our conviction, we feel even with the use of a construction tool set available there will be complex requirements which needs the facilities supported by "3GL".

5.5.2 The Role of Developers within the Life Cycle Phases

We have described the new model to be an architecture or open framework facilitating the coordination of users and systems developers efforts towards a common goal. We asked the participants to identify roles of developers within phases of life cycle. The table below illustrates the roles as identified by our participants. In addition the role of senior MIS managers and developers are identified based on experiemce during the development of the projects described previously in chapter 4.

A = Senior MIS Management
B = Systems Development Manager
C = System Project Manager
D = System Project Leader
E = Systems Analyst
F = Data Analyst
G = Systems Designers
H = Database Designers
I = Knowledge Engineering
J = Programmer

Life Cycle Phases	High	Low
(i) The preparatory phase		
. Definition of scope and objectives of the project	B,C	A
. Organising the project team	B,C	
. Establish methods and standards	B,C	
. Planning the project	B,C,D	

Life Cycle Phases	High	Low
. Training and education	B,C,D	
(2) Business Analysis phase		
. Strategic Planning activity	B,C,D	
. Agree System objectives	B,C,D	
. Conduct the business investigation		B,C
. Strategic entity and function modelling	C,D,E,I	B,E,F
. Idea processing, project options and recommended project proposal proposal.	C,D,E,I	B
(3) Detail Analysis Phase		
. Confirming the current practice	D,E,I	C
. Auditable control requirements	D,E,I	C
. Project scope and assumption	B,C,d	
. Area of change	C,D,E,I	B
. Business Issues	B,C	D

. Traffic analysis	E,F		
Life Cycle Phases	**High**	**Low**	
. Review available system application packages	C,D,E,I	A	
. Cost/benefit analysis revisited	B,C	D	
. Project timing revisited	A,B,C		
. Business system spec.	B,C,D	A	
. Technical approval	B,C,D,H,I		
(4) Design phase	D,E,F,G,H,I	B,C	
(5) The construction phase	E,G,H,I,J	C,D,F	
(6) The installation phase	C,D,E,G,H,I,J	B	
(7) The Audit and system evaluation		C,D	B

5.6 Development Productivity Gains-Results

This section reports results of a study to evaluate a specific product from McDonnell Douglas ProKit*WORKBENCH to support analysis and design phase, and 4 GL PRO-IV to support the construction phase.

(i) Productivity Gains

Life Cycle Phases	Model with aid of productivity tools	Existing method
Analysis	2	1
Design	2	1
Construction	5	1
Testing	10	1
Implementation	2	1

The evaluation team was made up of 2 analyst/designers and two programmers. The objective set for the team was to develop various functions so that the products were monitored for performance and to assess its integration capability within the new model and technical environmnt. In addition, to carry out user acceptance trials for the dialogue structures. Database management and file organisation tested included the following; CICS DL1, VSAM and DB2. The study was carried out on three platforms: Mainfram, Midrange and PC computers.

The benefits obtained during the analysis phase was due to the use of structured techniques described previously in chapter 2 and automation facilities provided by analysis/design tool kit ProKit*WORKBENCH to support entity/data modelling and data dictionary. The savings identified were conservative and should increase as the knowledge of the product is gained. The design phase is supported by facilities for prototyping the user interface. As for database design, automated tools did not provide logical structures for hierarchical data bases. However, they did provide the designer with schemas for relational type data bases.

The application generation part of the construction phase was strongly supported by 4GL PRO-IV application generator capability. Thus, reducing the laborious coding requirements. Testing benefits were high due to walkthrough sessions throughout the phases and especially because, of the availability of standard routines within the application generator for screen handling and data base I/O calls.

Implementation benefits were achieved by the availability of interactive help and standard dialogue design facilities.

(ii) Response Time and CPU Usage Analysis

To determine the way the system's internal processing contributes to delays in terminal response time, on-line transactions were monitored. The following are examples of results obtained for CPU usage and response time:

Function Key Action		CPU Usage		Response Times	
3GL	4GL	3GL	4GL	3GL	4GL
Enquiry Transaction:					
Enter	Enter	0.0093	0.0129	1.698	0.206
Enter	Enter	0.0321	0.0318	1.306	1.416
PF8	PF19	0.0267	0.0221	0.457	0.397
PF7	PF20	0.0487	0.0605	1.188	1.572

Creation Transaction:

Enter	Enter	0.0089	0.0184	0.382	0.625
Enter	PF5	0.0103	0.0087	0.466	

The enquiry transaction showed that the average response time for the 4GL was 0.54 seconds compared with 0.49 seconds for 3GL. However, this was mainly due to design approaches of forward and backward paging. CPU usage for 4GL showed average .03 seconds compared with 0.0273 seconds for 3GL. The average response time for 4GL when adding data was 0.33 secs compared with 0.51 second for 3GL. The above exercise has indicated that some transactions (e.g. forward,backward and deletion transactions) will need to be designed differently and wherever possible functions should be kept single. This approach to interactive on-line transaction design was strongly recommended in the design of all systems.

(ii) Training and Skill Requirements:

After sending a selected number of staff on the methodology course (CACI) and the development productivity tools, we found that they had learned enough to be able to use these tools for design and development of information systems. We expected the development staff to be productive in 4GL in 4 weeks after initial training. The initial learning curve was short but it took (2-3) months before the person became totally proficient in the more complex multi-screen, multi-file access and update type transactions. Experience at Massey-Ferguson has shown that it takes (6-12) months to get full productivity from a trainee in 3GL environment and in the majority of cases 18 months. Furthermore, our study showed that Productivity tools were more appropriate to analyst/programmers rather than for the traditional programmer (coder). In addition, these products allowed the end user to define a sequence of screens to prototype a conversation, using appropriate function keys to control screen flow and write DSS type application using the 4GL PRO IV.

5.7 Implementation as an Agent of Change

We have previously stated that the area of change must describe all the current mechanisms which will be modified as a consequence of implementing the proposed system. Effectively, all current practices within the business area investigated must have been considered to see if they will be changed by the proposed system. The participants from our groups reported that the nature of these changes must be clearly identified to the end users of the system to ensure that they fully appreciate the implications for their future methods of working. Furthermore, they have argued that this is a demanding and important task. Demanding because people with considerable exposure to current working practices often found it difficult to appreciate the changes in methods of working that the new system required. This was true for the orders management project. Important because practical experience of the relevant area of business was invaluable in the validation of the new system to highlight potential problems and evaluate the opportunities which it makes possible.

For the new system to be fully utilised, it is necessary to determine a point when the system will be accepted. This process of acceptance was described by our participants as an adaptive process. In this environment, most of the participants including the user representatives in our study felt that even with the extensive use of prototyping, walk-through system testing procedures and parallel running throughout the project development, the final product remained "not perfect". This is mainly due to the fact that in their view the user community related to the system as "owners" of the system only at the later phase - the implementation phase. Indeed, at this stage, the users started asking whether the system is doing what is required and adapted their way of working to the system. However, system change requests were received for enhancement during the adaptation process. These changes were the results of various misalignments between the system functions through which the solution was delivered to the users of the system. The specific misalignments identified were due to business and organisational factors, systems and technical factors and finally political factors. We will now discuss the impact of these factors:

(i) **Business and Organisational Factors**

The participants reported that in order for the new system to be successful, many business issues had to be addressed.

These tended to be "things which needed to be changed" - changes to culture, attitudes, agreements, existing practices, procedures and or organisation - all outside the direct authority of the project to change. The business strategic direction itself was subjected to modification due mainly to expansion. As the company acquired and established new parts distribution and business units, two approaches were available. A tight or loose approach with respect to business and systems integration. The "tight" approach would have supported the original business strategic requirements that is to integrate new business to the core single parts business. This will result in integrated operational systems supporting the current and the additional business units. The "loose approach" supported a view to treat the businesses as a self contained entity with defined interfaces to the existing business. This approach was chosen for the additional new business, resulting in reduced benefit realisation for the orders management project.

Impact on the day-to-day business activities was another factor affecting the adaption process. Managers did not foresee the impact of the new specific changes and control procedures on the day-to-day activities of running the business. These changes and procedures were implemented in order to best exploit the new system functions. This error of judgement resulted in taking some of the control procedures off and re-implementing it at a later stage of the project.

Another detrimental factor was the education of the lower ranking staff in the organisation using the new system. Although considerable training was given in the use of the system, it was initially thought the lower ranking staff would not benefit from detailed educational sessions of why they would have to do the same task in another way. One reason given argued that at this level of skills, the staff had neither the skill nor the educational background to absorb complex business rules (e.g. allocation of inventory from different locations and shipping it to the customer using appropriate complex pricing and customs rules).

(ii) **Systems and Technical Factors**

During the adaptation process, the users using the new system have asked the fundamental question of "does the system meet the objectives set for it?" More accurately, detailed questions have been asked. Specific questions included the following:

- Does the system support all the business functions described in the function model? Are all the entities and attributes supported?

- Does the system support the new organisational structure? Most importantly, is it supporting the areas of change?

- Are the people using the system happy with the human-computer interface?

- Does the system work or are there errors in the program logic or program code?

The answers reported were in the form of "yes", "not quite", "maybe" and "yes, but". The answer "no" was not used, suggesting that "no" related to total rejection of the system or specific function within the system.

The adaptation process suggests that you cannot get it right the first time, no matter how experienced the project teams are and how efficient the methodology used is. However, our participants reported during the initial stage of the adaptation process that when users started to use the system, it was difficult to know whether a fault was a system malfunction, an operator error or poor documentation. Knowing what ought to be coming out of the system when processing real live commercial data helped the users to gain confidence in using the system and to recognise whether the system performs satisfactorily.

There were systems changes brought about due to business and organisational factors described previously. In addition, there were system changes due to systems and technical factors. Factors which covered the following areas of concern. First, from the human-computer interaction aspects, the users of the on-line facilities were asking whether the system is supporting all the interactions that are necessary for the users to perform the required business

function. Furthermore, are each of the transactions in a form that is suited to the particular style of the type of user involved? These two questions were posed during the prototyping of the human-computer interface at the design phase and during the system user testing. Other questions asked included the suitability of the style of the dialogue. Various degrees of unsuitability were considered. There were mainly four - user rejection, user irritation, slow operation and screen errors. Positive answers were given for both questions, yet during the adaptation process, changes to the on-line screen were requested. However, these were minor changes of moving fields. Most requests were related to control and security procedures. In particular in the area of control procedure, where applied to data belonging to separate locations.

The second factor relates to whether the new system and new organisational procedures and structure were actually solving the problems of the business. In certain cases, the answers suggested yes. However, the adaptation process in large projects such as orders management continued for six months and, in fact, in the complex areas continued for over a year (e.g. pricing, discounting and export businesses). During this period, various changes were requested and actioned. No structural changes were carried out on the databases or major changes to the programs.

The third factor relates to delivery mechanism misalignment. Inadequate computer hardware, slowed software performance or speed of error/problem recovery are examples of these types of factors. In the case of the marketing intelligence system, the system was rejected during the adaptation process due mainly because response time was so slow. Users associated the slowness with the computer hardware. In fact, the problem was in the area of the database management system DB2. The first release of this product proved to be totally inefficient for large volumes of data manipulation which users wanted for Sales and Marketing analysis. As a result, the adaptation process was terminated and parts of the system were rewritten using conventional files and languages. This had a serious impact on user confidence. It took over fourteen months to gain the confidence and have the system accepted by the user community.

(iii) <u>Political Factors</u>

These factors tended to be the most frustrating and counter-productive to the adaptation process. Lack of interest in the system was due to various political factors.

The first and perhaps the most important factor identified by the participants was the "not invented here" problem. This was presented by the users in the form of responses such as "not interested here", "not required here" or "over-complex here". All of these reasons given were due to the fact that the system potential or actual usefulness was not understood initially. However, considerable effort was made to educate the users so that full understanding of the benefits and potential opportunities of the system was achieved. This problem still remained and "not invented here" was identified as the cause. In order to support the business strategic direction, it was necessary to implement the system in all locations. Therefore, the top management decided to support the project team and forced the system implementation to begin. The adaptation process where this forced implementation took place caused the developers a considerable amount of problems. Small insignificant problems were reported as major and any system problem was blown out of proportion. The quality of the system and the richness of the functions supporting the strategic direction plus total commitment and professionalism from the developers continued throughout the adaptation process to the end.

Another political factor encountered consisted of user rejection in the form of industrial action. This was due to trade unions using new systems implementations to get better salary and conditions for their members. The lesson learned here and universally accepted throughout the organisation supported the notion that "trade unions should be part of the development process". This means that a full explanation of the system impact on their members should be given and the implementation plan agreed with the trade unions before the user testing and adaptation process takes place.

5.8 Conclusion

Our study was conducted in the field In relation to live commercial work and so it was not an experimental one in the tradition of laboratory studies. The main reason for choosing to test our model in the real commercial computing world was that we were concerned with testing the model in managing live information systems projects through the whole cycle of activities needed to develop a computer system. In addition, testing the model for producing an asset of high quality, which is capable of making continuing contribution to the business. In our study, we have met developers who were certain that the development process will be lengthened and more expensive if they followed the activities proposed by the model. We have also met developers who were certain that useful, usable and desirable systems can be developed with no user involvement. But few systems are designed without some form of user involvement. The tradition at the author's own organisation, Massey-Ferguson, was that a team of systems specialists took responsibility for the design and implementation of the system and invite user involvement to the degree and in the form that they deem appropriate. The basic premise of the new life cycle model is that the users of information must be thoroughly involved in the information systems development process. These users took responsibility for the nature of their systems by becoming directly involved in processes which enabled them to specify their own requirements.

The system developer's view of the model extracted from our participants highlighted important benefits for the model. They felt that undoubtedly the most tangible benefit of using the model is in the area of communication. The model not only aided communication between the various functional specialists of the systems development department (systems analyst, application programmers, database analyst, etc.), but it also helps the system development team to communicate with the user communities. This communication ability aided the developers to better understand complex business environments. The participants reported that techniques of entity and function modelling helped to ensure that systems were designed with the future in mind. Whereas the functions that the organisation performs frequently change over time, the data which an organisation handles is frequently more stable. Consequently, systems were designed around the data which the organisation handled which made the developed system more likely to be able to respond to future change requirements.
As for managing the project with the aid of the model and our participants, the model splits large projects into smaller, more manageable activities.

The identification of different phases kept the development staff motivated and users committed to the project as milestones are achieved. Moreover, the model provided a formal framework within which projects were frequently re-evaluated. There were several reasons why the projects described were frequently re-evaluated. As the project moved from phase to phase, additional information was uncovered which changed not only the basis on which the earlier resource estimates were made, but also the desirable scope of the project itself.

Another important benefit highlighted was the role of the model in aiding the continuity of the project in the face of high staff turnover rates. The use of the model ensured that a high level of system documentation was maintained and consequently helped familiarise new recruits with both existing operational systems and systems which were under development.

As for the use of productivity tools we have shown in this chapter that there are benefits to be gained from using analysis, design and programming tools if they are adopted selectively and with regard to the future. We have shown that the use of analysis and design tools including the use of 4GL within the new systems development life cycle will facilitate the rapid development of applications. The analysis and design toolkit can be used effectively to carry out the business analysis and produce the business specification that an essential starting point for the first prototype and design of databases. Our study showed on average 2:1 productivity gains can be achieved during the analysis and design phases. Our study also showed that with the help of 4GL on average 5:1 productivity gains can be obtained during the construction phase.

We believe the implementation of systems development methodology and application of productivity tools are the only way for the companiess to more rapidly develop quality system incorporating full user requirements, make better use of resources and technology and support current and future needs. Furthermore, the key point is that the commercial benefits of a user centred process, using the tools and methods that have been proposed, has been clearly demonstrated in practice.

CHAPTER 6

PROJECT REVIEW, FUTURE RESEARCH AND CONCLUSIONS

6.1 Summary of Proceeding Chapters

The following sections recall the main areas of interest, results and conclusions arising from this program research in relation to the development of strategic and tactical information systems.

6.1.1 Information Systems Development - the Theoretical Context

As we have seen from the earlier chapters of this thesis, a great many methodologies and techniques have been proposed for the development of information and decision support systems, including knowledge based decision support systems. Several methodologies exist, each having at least a published paper based on it. Many methodologies and techniques are little more than the combined expertise of a handful of experienced information system developers who have worked closely together and developed systems. A few have had wider use and published as books.

In this thesis, we have reviewed six methodologies and techniques for the development of information systems. We have looked for the underlying philosophy and essential features of each methodology. We discussed the advantages and disadvantages of each methodology. Specific methodologies and techniques reviewed included the following: information engineering techniques, Multiview - a framework for information systems definition, Jackson System Development (JSD), structured analysis and design, structured requirements definition and system prototyping.

The information engineering methodology developed by James Martin and refined by James Martin Associates is a structured framework within which techniques are employed leading to the development of high-quality, integrated information systems. This framework provides a basis for defining and managing projects through the various stages of development. The methodology consists of a set of building blocks, each reflecting the stages of the information

engineering methodology. The horizontal blocks represent the basic stages of the methodology to analyse the business and to specify and construct its systems. The vertical blocks provide strategies for shortening the development life cycle, given the information gathered beforehand by the basic techniques. The methodology is marketed as a graphics orientated discipline. During the analytical stage of the project, the business models are created. These are then refined during the design stage to create the technical models that will be used directly for translation into system code. The systems produced, therefore, have their origins in the business models and it has been argued by James Martin Associates that it is more likely to serve the business needs than systems produced by different route. They have identified two major benefits from the methodology. They are improved quality of systems and improved productivity in development. The first is achieved by focusing on the business requirements. The second is achieved by breaking free from the stages of development cycle and adopting strategies to shorten it and by automating wherever possible.

The second methodology we reviewed was the Multiview approach to information systems definition as proposed by Wood-Harper et al (1985). Multiview addresses problems associated with the analysis and design activities of information systems definition. It is a methodology to structure the tasks for the analysis and users during the analysis and design activities. The methodology exhibits five different views, all of which can be emphasised, reduced in scale or even omitted according to the circumstances. The five stages proposed include the analysis of human activity systems, the analysis of entities and functions, the analysis and design of socio-technical systems, the design of human-computer interface and the design of the technical sub-system. The authors of the methodology have argued that the order of the stages is important, moving from the general to the specific, from the conceptual to hard fact and from issues to task. Outputs of each stage either become inputs to following stages or major outputs of the methodology. The final outputs of the methodology are the social systems, the role set and people tasks, the human-computer interface and the technical specification and the necessary inputs and outputs to support the non-application system. The first stage of the analysis is to draw the picture of the organisation in order that the analyst,

client and problem owner agree on the starting point. The first stage in the design process is the analysis of the human activity system. A full understanding of the purpose of the organisation is proposed before designing the system. In particular, the people in the organisation including the interaction between them and the deliverable in the form of the definition of the "purpose" of the system before attempting system design. On the completion of the human activity system, the system designer creates a conceptual model. Specific tasks include listing all the activities associated with the root definition, grouping them and then drawing them together as a system to perform these activities. The system chart created shows all the relationships between various activities and the environment. The conceptual model is then compared with reality to see if any improvement can be made in the way that activities are organised. The next stage is information modelling - events, functions and information. Information modelling starts with the overall picture of the information flow in the conceptual model and develops the activities in this into the function hierarchy chart. Another area of interest in this stage is the business entities. The third stage known as the socio-technical design is the analysis and design where the problem solver lays out the alternatives and the problem owner decides which alternative to adopt. The fourth stage is the design of the human-computer interaction. In this stage, for each function to be performed, the contents of the dialogue, that is the information and instruction between the users and the computer, needs to be defined. The definition of a specific system which can implement the analysis and design requirements is the last stage of the methodology. In this stage, the technical sub-systems are developed.

The third methodology we reviewed was Jackson System Development methodology. Described by Michael Jackson (1983) as a "method for specifying and implementing computer systems". These activities include requirement specification, functional specification, logical system design, physical system design, program specification and design, program implementation, and system and program maintenance. Connor (1985) reported that JSD covers every aspect of system design and development.

However, JSD excludes activities such as project selection, project planning and management and cost benefit analysis. It excludes procedures for system acceptance, installation and cutover. It excludes the whole area of human engineering in such matters as dialogue design. JSD also excludes specialised application skills. JSD has been applied to information systems development for finance, administration, text handling, communications, real time process monitoring and simulation. The methodology supports any programing language, operating system, application support software, teleprocessing monitor and DBMS on any hardware. JSD consists of six development steps - entity action, entity structure, initial model, function, system timing and implementation. The entity action step describes the real world entities and the actions they perform or suffer are described. In the entity structure step, an abstract model of the real world is constructed. This abstract model will display the entities and the actions defined in the entity action steps. The tool used is "structure diagram" to illustrate this abstract. In the initial model step, the developer specifies the system to be built by specifying a simulation of the real world. In the function step, the developer specifies the system functions in terms of the model. The model is extended to provide required output. The system timing step is considered in JSD when the information is needed, which in turn affects when the processes are run. In the system implementation step, the task faced by the developers during this implementation step is to identify how many real or virtual processors are to be used. Other factors considered by the developer include the programming language to be used, database management system, enquiry language and TP monitor. In summary, the developers using Jackson System Development start from a rough statement of new requirement, develop its specification in two phases, modelling and function. In the third transformations that should preserve essential facilities. Jackson is emphatic that JSD is not a top-down method for developing systems. The basic idea of top-down is stepwise refinement. The object to be developed is regarded as a hierarchy. The developer begins by stating the highest level of the hierarchy and decomposes this level into a number of smaller objects. "What is JSD then?" argues Connors (1985). "JSD can be considered a network which takes a series of sequential real life and computer processes and through synthesis builds them into a system. Further, these processes are defined using structure text, even during the modelling of the real world

entities, and this ensures that the specification can be programmed".

As for concepts and terminology used by JSD, Maddison et al (1983) have reported that JSD uses different concepts and terminology from other methodologies (e.g. avoiding data analysis, global conceptual data model, normalisation, relation and relationships). Thus, developers sympathetic to JSD may be irritated by other methodologies.

The fourth methodology was structured systems analysis and design. Structured analysis and design is a set of process orientated techniques, rather than a methodology argues Yourdon. It covers the analysis and design of information systems, but not strategy and feasibility. The process orientated techniques include data flow diagrams, structure diagrams and structured english, with guidelines to show how to apply them. The data dictionary facility can be combined with these facilities. The analysis and design phases are broken down into well defined sub phases. The analysis accuracy and completeness can be checked by referring back to the users. The design phase produces a set of program specifications with supporting design documentation and operational procedures in the data dictionary. Maddison et al (1983) have reported that the techniques are intended for use on medium to large development projects where the problems of complexity and communication with the users are serious. The most important feature of the structured analysis and design is the step by step evolution of the analysis and design. The biggest disadvantage is the large amount of data that must be recorded and controlled. The design philosophy is to provide more maintainable systems which is most important and more efficient, the least important.

The fifth technique reviewed was structured requirements definition. Connors (1985) has reported that the structured requirements definition approach to system design provides mandatory outputs defined by a rigorous requirements definition process. The primary objectives of structured requirements definition is to produce systems which are a direct reflection of the outputs, since it is the output that provides the user with the data he needs in order to function. This emphasis, however, tends to result in systems with multiple files in

which considerable duplication of data could occur. The technique employs "Warner-Orr" diagrams as a tool for the analysis and design. Specific diagrams include data structure diagram, system and process diagrams and finally assembly line diagrams.

In comparing information engineering, structured analysis and design and structured requirements definition, we identified the common factor between these methodologies and techniques is the normalised data model. Connor (1985) has reported that this data model is developed before procedures are identified in information engineering. In structured analysis and structured requirement definition, the model is developed based on the logical files developed from the procedural flow. Another common factor between the techniques reviewed is that between JSD and information engineering. What they have in common is the entity, which has almost the same meaning in both techniques. The sixth and final technique reviewed was system prototyping. Prototyping is not a design technique by itself. Effective prototyping can only be done in conjunction with another system design technique. Prototyping reported Connor (1985) is defined in Webster's new collegiate dictionary as "an original or model on which something is patterned" and as "a first full scale and usually functional form of a new type or design of a construction (as an airplane)". Although prototyping is regarded by most systems developers as production of quick versions of the system, in particular in the area of human-computer interfaces. We have in fact reviewed three categories of prototyping - throw-it-away, , evolutionary and incremental. In reviewing these categories, we discussed the question of whether a prototype should become a final system. Other questions such as how it should be constructed and when it can be accepted as a final product were discussed. Hekmatpour and Ince (1986) have reported that "throw-it-away" prototyping corresponds to the most appropriate use of the term prototype. It is often used for the purpose of requirement identification and clarification. They stress that the relevance of this approach to requirements analysis and specification has often been referred to as specification prototyping. In this type of prototyping, quality factors such as efficiency, structure, maintainability, full error handling and documentation are ignored.

Evolutionary approach introduces the system into the organisation gradually while allowing it to adapt to changes that take place within the organisation. Hekmatpour and Ince (1986) have reported that evolutionary prototyping is the most powerful way of coping with change. This approach they argue requires the system to be designed in such a way that it can cope with change during and after development. In incremental prototyping, the system is built incrementally one section at a time. Hekmatpour and Ince (1986) have reported that incremental and evolutionary prototyping have been used as synonyms (Baldwin 1982). They have, however, identified significant differences between the incremental and evolutionary prototyping. Incremental prototyping is based on one overall software design (Floyd 1984). In evolutionary prototyping, the software design evolves continuously. In incremental prototyping, a full scale design is first conducted and then modules implemented and added in sequence. As with evolutionary prototyping, the system grows gradually, but in a considerably less dynamic way.

As for the prototyping and system design techniques we have reviewed, protyping software is used in information engineering for screen painting, procedures for maintaining the records in the data structure and procedures for producing output records. Connors (1985) has argued that information engineering and prototyping software to blend together. In structured analysis and design, prototyping is used as a programing aid after the structured design process is complete. In particular, the prototyping software is used to produce file maintenance and output programs. Prototyping can be used in conjunction with structured requirements definition argues Connors (1985), but only after mainline functional flow diagrams, the file definition and systems outputs are defined. As for JSD and prototyping, the time orientated systems suited to JSD may not lend themselves to prototyping Connors (1985) has argued. However, prototyping can be used in Step 6 of the JSD methodology. At this stage, the concept of file is introduced. The five preceding steps are defined in considerable detail in procedural language. Thus prototyping effort could be wasted.

Another area we discussed was the human-computer interaction. We reported that on average 59% of the program code accounts for the user interface. The user interface is the way in which individual users communicate with the system developed for them. They may be interacting with the system by entering data, enquiring or interpreting output produced by the system. In the other cases, the user may instruct the computer to run specific programs written for them or written by themselves. This all has to be done in a way that users find easy to understand. This will vary according to the background and experience of the users and the particular process required. Therefore, it is not surprising that a major part of the project effort may be expanded on the design and implementation of the interface. Ten Hagen (1980) has argued that proper design of the user interface is such an important step in system development that many authors believe that it should be the first part of the system to be designed. Other researchers in the field of user interface design have also argued that the design of human-computer interface is central to the design of an interactive system, Edmonds (1982). Models of the user interface have been developed by several researchers (e.g. Fitter (1981), Edmonds (1982) and Innocent (1982)). The definition of the interface varies slightly between authors, but it is generally agreed that it includes all points of contact which the user has with the system, Benyon (1985) has argued. We have also reported that considerable effort has been directed towards development of formal user interface analysis tools with the objectives of improving the design process for the user interface development. In Chapter 1, we discussed several topics which we felt were related to the development of user interfaces. The evaluation of user interface complexity was discussed by reporting notions presented by authors in the field of human-computer interface design, model of interaction and types of human-computer dialogues (e.g. Guedj et al (1980), Karat et al (1987) and Shneiderman (1983)). We reported that the basic style of interaction with the computer systems from interface dominated by command languages to "direct manipulation" interfaces. Other alternatives reported included dialogue networks and production rules to construct interactive human-computer dialogues. The interface system for the design of human-computer dialogues developed by the author were briefly described. One of the central dilemmas of interface design reported by researchers in the field is how to satisfy the conflicting demand of difficult

users. Sutcliffe (1988) proposed adaptive interfaces as a potential solution to the mixed user population dilemma, although the consistency of the interface has to be maintained as it adapts. Various notions on the subject of adaptive human-computer interface design were presented. Notions proposed by Damodaran and Eason (1981), Benyon (1985), Maguire (1981), Edmonds (1981) and Innocent (1982).

As for the prototyping approach and the human-computer interface, prototyping requires the design of the human-computer interface to be an iterative process involving a large degree of user participation. This approach allows the designer to derive a conceptual model that appeals to the majority of users, Hekmatpour and Ince (1986) have argued. User participation was considered not only for prototyping in the design stage, but throughout the systems development process. The need for user involvement in this process was discussed by reporting on notions proposed by several authors (e.g. Lucas (1975), Biggs, Birk and Atkins (1980), Glasson (1985), Damodaran and Eason (1982), Ives and Olson (1983)). Types of user involvement were discussed in relation to one way involvement, mainly systems development team driven, user representatives, participative design with the aims of bringing all users to the design process and finally user design participation by providing expert advice. In addition, measures of user involvement were discussed. This was done by reporting the three most common strategies for investigating user involvement as presented by Ives and Olson (1983). The case for user involvement and no user involvement was also considered by reporting views published by Ives and Olson (1983) on reviews they have carried out on studies carried out by the researchers in the field.

Another area closely linked to user involvement is that of user support. Damodaran and Eason (1982) have argued that user involvement and support strongly influence the effectiveness of any computer application. Methods of user support discussed included training, documentary support and human support mechanisms. These users are groups of individuals who are jointly responsible for performing a specific task or making decisions using information produced by the computer systems developed. The decision making process and providing

systems to support the process has been given much attention during the past decade by both in the commercial computing environment and the academic environment. Practitioners and scholars have suggested that a decision support system requires a unique development approach with a high level of user involvement. The level of user involvement can vary widely. A review of the decision support system (DSS) technology, the management activities and the nature of the task interdependence all interact to affect the user involvement. Despite the growing use of this technology, the older type of human-based manual systems continue to play a crucial role in the planning and decision making process. DSS are capable of providing refinements in the ability to provide relevant data to model and simulate, and to provide answers to "what if" questions. Land (1989) has argued that properly designed and used DSS can improve the process of planning and decision making. The challenge, he argues, is to blend the capability of the human decision support system with the facilities of the technology based systems to provide a support service neither type can provide on its own. Doukidis et al (1989) have argued that the technology based approach of the management support systems have to be integrated with the human activity systems they serve. They have based this notion on the notion proposed by Turban (1988) in which he suggests that knowledge based management systems can be created by integrating the expert system and decision support approaches. The developer's task, therefore, is to harness the synergy resulting from the blending of human and system capabilities. To this end, we discussed the design and development approaches for DSS, in particular the design of human-computer decision systems and prototyping for DSS. The design of the interface was discussed in the context of decision centered design methodology. There are four major stages - decision systems analysis and problem finding, new decision system specification, new decision system design, decision system development and evolution. As for prototyping for DSS, we reported that prototyping can be an effective approach for developing and implementing DSS. Design strategy is effective in establishing meaningful user involvement and highuser satisfaction. We have also reported that knowledge based techniques may facilitate decision making methods which are not possible with traditional mathematical approaches. Further, knowledge based techniques allow any number of decision making strategies to be combined. Thus, we

reviewed the area of expert systems for decision support. We reported that the decision support approach provides inspiration for the designers of expert systems and allows them to produce systems accessible to their users. In addition, for the benefits of DSS and knowledge based DSS to be fully realised, they have to be fully integrated with the operational and information systems of the organisation. We have argued that a conceptual integration of the corporate total business information system provides a significant step towards giving end users easy access to company information by using terms they understand. This integration is achieved with the help of a framework, incorporating the design and development of decision support within the systems development process - the subject for the next section.

6.1.2 A New Model for the Development of Information Systems

During the late 1970's, it was recognised at the author's own organisation, Massey-Ferguson, that 75% of all information system projects were cancelled before they were completed. The majority did not come up to the users' expectations. Projects usually arrived late and over budget and then these systems tied up development staff forever on maintenance of these systems. Furthermore, with the introduction of decision support system technology including knowledge based decision support systems, the management, users and systems developers found that they had no control over systems development. They identified the need for a framework containing guidelines which allows the developers and the user to work together within the systems development process so that the final product is implemented with the strategic business objectives in mind, ensuring that the eventual users of the system get exactly what they want. The realisation that the successful creation and introduction of information systems requires the co-ordination of many different skills and techniques, set another requirement for the framework. To support this requirement, the framework needed to help the project team defining the precise part that each member of the team will be playing in the development process. They will know their responsibilities and will have the right skills, so they will come up to expectations. Another important feature required for the framework was the introduction of the best current practices in software development (e.g. data and function

modelling, prototyping and productivity tools) that will make the framework effective.

Based on these requirements and notions discussed in the previous section, we create the new model for the development of strategic and tactical information systems including decision support systems. One of the principles behind the new model is the idea that each phase in the project life cycle progressively refines the project definition and reduces or eliminates uncertainty about a whole range of factors, including system content, technical design, resources and timing, etc. Although this refinement is progressive, it is not continuous and mainly revolves around the effective management of a phase. Each phase includes an activity to plan the following one in detail, in order to provide substance to the project approval request process.

We set the aim for our model to incorporate decision support, including knowledge based decision support systems development. To this end, we discussed a methodology for the DSS development. Arinize's (1969) six stages of an outline methodology were reviewed. The stages proposed by Arnize included the following - entry where the problem is introduced, description where the organisational, information and decision elements of the decision making process is described, modelling design where thė full description of the data and processes that are implied and derived from decision making. We reported that an experimental approach in the design of DSS is in establishing the conditions for learning and communication about decision process. To understand the role of experimental approach to information systems in the design of DSS, we discussed the issues involved in the experimental approach to information system development, the relationship between this approach and different problem types and the role of experimental approach in the context of DSS requirements analysis and design.

As for the knowledge based decision support system, we discussed knowledge acquisition and knowledge representation in DSS. In addition, major features of DSS were described. Finally, guidelines for developing knowledge based DSS were described. The guidelines proposed included design strategies, properties of knowledge based DSS and stages of knowledge

based DSS development. These guidelines and notions discussed were incorporated into the new systems development life cycle, including dialogue networks.

The systems development life cycle model we proposed in Chapter 2 consists of a combination of phases and deliverables, representing a structured and consistent framework, within which systems projects can be developed. The schematic shown in Chapter 2 illustrated the phases in the new systems development life cycle model proposed. Seven phases were identified - the preparatory phase, the business analysis phase, the detail analysis phase, the design phase, the construction phase, the installation phase and finally the audit and system evaluation phase. The objectives of the preparatory phase is to prepare the project environment necessary to ensure project success. We identified eight major tasks for this phase - the definition of scope and objectives of the project, organising the project team, establishing methods and standards, planning the project, training and education of the project team members.

The business analysis phase concerns itself with strategic planning activities, agreeing business objectives, conducting the business investigation, the construction of the strategic entity and the funciton model, and the evaluation of project options and recommendations. The deliverable from this phase is the project proposal. The method recommended for the creation of the entity and function model is the CACI structured data and functional analysis. Six important principles were identified for this methodology. The separation of data and function from strategy to construction phases The top down method of analysis, the whole process of successive narrowing and detailing of the area being analysed, ensures that the detailed development work fits together in the framework that the top management requires. The third principle is making each phase the use of pictorial models developed for previous phase. The final principle proposed is the checking out of the results with the bottom up method.

The detail analysis phase defines the business functions which span the project proposal. It calls for a detailed understanding of the part of the business which was the subject of the investigation. Data and function analysis techniques proposed by the CACI methodology

was proposed for detailed analysis of the business entities and functions. The analysis of business data seeks to model the structure of the data itself, as this will generally be much more stable than the mechanisms which currently use it. The model should reflect the logical structure of the data, rather than the way it is organised. Specific activities proposed with this phase included the confirmation of current practices, auditable control requirements, project scope and assumptions, identifying areas of change due to the implementation of the new system, any business issues needing attention, volumes of entities and traffic analysis. Furthermore, evaluation of application packages and a revision of the cost benefit analysis. The detail analysis phase deliverables consists of three major deliverables. The business system specification, technical system specification and a pre-design approval request.

The design phase aims to provide a cost effective system design which meets the business requirements and defines a common framework for system construction. A framework within which the new system will be developed and installed. Several design considerations were discussed - practicality, efficiency, least cost, flexibility, security and the distributed system approach to distributed functions. Furthermore, database technology options and prototyping were discussed.

As for the activities and deliverables of the design phase, the major activities identified included the design of the physical and logical databases, the definition of screen and report layout. Three deliverables were proposed - the business system design, defining the system from business standpoint and the technical system document covering both technical functions and components required, including hardware and software, logical database structures and physical database characteristics, screen and report layouts. Finally, the construction approval request.

The objective of the construction phase is to produce program specification, program coding, the creation of the test data, the plan for cost testing and the creation of user training and support documentation.

The installation phase is concerned with the procedure of cutting over the system to production and the evaluation of the success of the system in meeting user needs. This is followed by the audit and system evaluation phase, where an objective appraisal of both the development of the system project as well as the resulting system is conducted.

Making effective use of modern development tools within the systems development process was another important issue we addressed. We reported that the benefits of modern development tools are not being fully exploited , because developers have failed to understand that the selection and use of such tools in the development environment is a very different, and much more complex process than the introduction of programming languages such as COBOL or PL1. There are various development tools on the market but each has different types of application. We have however, argued that development methods need to be supported by appropriate tools. Further, we showed that analysis and design tools to support data modelling techniques, dataflow diagramming, a designer facility that can be used to document system and program structures including logical structure for relational type structures, and a well regarded prototyping facility that enables developers to create prototype scrrens, menu, forms and reports. Further, we showed that true productivity can be obtained from using "powertool" 4th generation language technology during the construction phase. We identified productivity gains of 2:1 during the analysis/design phase and a 5:1 gains for the construction and testing phases.

6.1.3 From Concept to Practice - How the model was used

We described the Corporate/organisation's strategy as a tool pulling together the activities of the various business units into a balanced, synergistic program with a plan for the use of resources. The strategy addresses tactical and long-term projects. Although the potential competitive benefits of business systems strategy, that is information systems strategy supporting the corporate/organisation's business strategic direction are generally recognised by the business, there is a great gap between the recognition of such value and the ability of many firms to apply the information systems development effectively. In the case of the author's own organisation, as with many other organisations, systems development effort was directed to the provision of systems to support specific operational business processes. However, it was recognised that such systems make only a small contribution to providing the management information needed to measure and plan the business to meet its overall aims. To this end, two major projects were commissioned to support the Strategic business direction of the organisation. These two projects were developed with the aid of the new systems development lfie cycle model we have proposed in this thesis. In chapter 4 we described how these projects were developed in detail. In chapter 5 we argued that the extent of the success or otherwise of our model relies on the skills of the practioners and their experience and familiarity with the concept and mechanics of the model. To establish the actual benefits of what occurs from using the model, we tested the model on five major assessment factors. The model was first assessed on the total project with the objectives of recognising the risk, so that appropriate degree of timely management is given to the areas which might jeopardise the success of the system. In this assessment we considered the business case, the technical environment, user impact, people and project structure management.

These areas were assessed on the level of concern (less or more). The model was then assessed in terms of the activities and deliverables proposed within the model. The assessment was in the context of auditable processing controls, preventative controls, detective/ corrective controls, timeliness and management trail controls. The first three were

assessed on the three factors - completeness, accuracy and authorisation. The activities and deliverables were also assessed by using the walkthrough procedure for system quality and change control procedure. The walkthrough procedure confirmed the benefits as a contributor to quality and improved project communications. The change control procedure identified the need to operate an effective change control procedure during the development of a system in order to control a project's schedule and cost. The third area discussed was user involvement and system success. The discussion in the context of first managing the user involvement within the development life cycle activities. In this discussion the question of the project management process of the user involvement was considered from the very inception of the project. The user project manager, the quality and experience and the membership of the project team including the steering committee. Other topics discussed included the advantages and disadvantages of project supporters and champions. We argued that project champions and sponsors provide necessary advocacy for any new idea and this support is no guarantee of success. However, we reported that in our experience, project visibility at high level can be worse than none at all, and that support at the appropriate time is important. As for the system quality and acceptability, the "right first time" principle is strongly supported as a key quality objective. Further, to investigate the relationship between user involvement and system quality and acceptance, we conducted a study at the author's own organisation extracting the internal users' view in the organisation and external views - that is the dealers and distributors of the Massey-Ferguson. The internal users group was made up of 27 users with a mixture of good formally educated experienced subjects and experienced with no formal education background subjects.

The discussions with the internal users were in the context of; the new model as a framework for communicating business requirements to the systems developers, as an agent of change for reducing business complexity, and the decision making performance. The group felt that the model provides a framework so that a clear and precise understanding of users' business requirements are communicated accurately and reflected in the final system. The idea processing and brain-storming sessions at various stages of the project allowed for complex

business problems to be identified before rushing roun looking for solutions. The support for the decision making process was highlighted as an important consideration during the investigation and analysis phase. As a result of these considerations, good decision support application systems were constructed to support their day-to-day decision making activities. The overall reaction of the external user groups was very good to the dealer communication package (the user-interface) system developed with the aid of the model. Reaction of those who responded was that they thought they had made the right decision in purchasing the product, the user support facilities were rated as very high.

As for the developers' view of the model. The participants were mostly professionals from the management information services department at the author's own organisation, all members of a project team developing the projects described previously. Members of the group were interviewed and the discussions were concerned with covering the model again as a communication media between the users' and developers' flexibility; the top-down approach for strategic systems development, including tactical development consistency; providing a checklist of activities and deliverables for each phase using the same vocabulary; Productivity: the model as contributor to systems development productivity throughout the life cycle. Maintenance: the model as a tool controlling and co-ordinating maintenance alongside simultaneous development projects. Usability: the role of the model in designing reliable and responsive systems. The important criterion for the developers was that the new model would encourage the involvement of users as early as possible in the development cycle. In particular, when the users are more committed and involved in the process of system design, the system more quickly fits their needs. The end result is better overall productivity throughout the development process. Prototyping was identified as the most important tool; the developers argued that the users believed that it would help to overcome the communication gap between end users, describing their needs and designers whose job is to develop the system. Prototyping helps because it allows the users, who are not too sure what their own needs are, to experiment with different options.

Another important area discussed was the full system utilisation - the point when the system is accepted. Our participants have argued that this process is an adaptive process. The adaptive process starts for the users from the prototyping, training, user system acceptance and implementation stages. However, we reported that this adaptive processing continues through the post implementation phase depending on the complexity of functions supported and various factors. We identified three major factors - Business and organisational factors, these tend to be things which needed to be changed (changes to culture, attitudes etc.). In addition, day-to-day business activities was another factor affecting the adaption process, and education level of the lower ranking factors.

The second factor related to systems and technical issues. Does the system and its technical components operate in an efficient way and support all the business functions described in the function model. The human-computer interaction plays a major part. Our experience shows that this was where the adaption mostly takes place. Hardware and system software can also play a major role in this misalignment (e.g. database management systems and over-worked hardware).

The third major factor discussed was the political factor. We argued that this tends to be the most frustrating and counter-productive to the adaption process. The "not-invented here" and industrial relations factors impact trade unions bargaining using the system as bargaining power to obtain better conditions and salary for their members.

6.2 Suggestions for Further Research Work

This program of research concerned itself with proposing a new model for the development of information systems including decision support systems. Within this area, however, remains quite a number of opportunities for further work.

The following path of research may be followed :-

6.2.1 Factors influencing productivity

There are many factors influencing productivity. We will identify nine factors which we feel are important. Standards, project management, staff development, training, quality, hardware and services, software, work content and automation.

The new model provided a process of procedures and standards, geared to producing a set of prescribed activities and deliverables supporting a skill based procedure for capturing knowledge, analysing problems and representing knowledge. Although, in recent years, there has been significant progress towards the automation of deliverables within the life cycle, for example graphical support for entity and function modelling. However, we feel that further investigation should be explored to provide automated support for producing deliverables applying graphical support as appropriate. The proven business orientated disciplines we have proposed within our model, have provided the consistency, co-ordination and control required for rapid development. However, what we feel is needed is a fully integrated computer-aided software engineering (CASE) tools.

To address problems associated with staff development, training, work content and hardware and services, we need to expand our framework to include guidelines to address these issues.

6.2.2. Knowledge based technique and productivity

We believe that real breakthroughs are achieved, not by merely mechanising and automating the activities and deliverables within our model, but by using knowledge based techniques to acquire knowledge from the user expert, as a systems analyst would do, and then perform data and function analysis as the data analyst expert would do. Further, this process would continue to the completion of database design as the database designer expert would perform.

6.2.3 The Model and the emerging industry standards

The commercial computing industry analysts have predicted that the IBM corporation will deliver the AD/Cycle in mid-1990. IBM anticipates that AD/cycle will eventually improve productivity and quality during the development and maintenance of business systems. Although industry analysts predict that the management and enforcement capabilities of the AD/cycle will probably take five years to evolve. However, we feel that the introduction of AD/cycle will have a major impact on the commercial computing world, and it will be necessary to re-examine the proposed model again in relation to activities and deliverables of AD/cycle.

6.3 Conclusions

The great potential of the proposed model is for dramatically increasing the effectiveness of information systems, including decision support systems. This thesis has argued that the primary problems in realising this potential have been methodological, technological and political. This is due to the fact that designing, developing, constructing and implementing computerised business information systems is not an easy task, as Connor (1985) has argued that information systems development requires the equivalent knowledge and effort that could be required in any major engineering project. Experiences at the author's own organisation have shown that one cannot expect to develop an effective information system using traditional methods and tools. Accordingly, the thesis has proposed a framework for the development of information and decision support, including knowledge based decision support systems. A framework in the absence of theory is helpful in organising a complex subject such as the methodological, technological and political, identifying the relationships between the individual parts and revealing the areas in which further development will be required. The frame we have proposed in this thesis has evolved over the last 8 years. During the evolution of the model we have seen the introduction of computer-aided software engineering tools. The advocates of these tools have argued that, if used properly, these tools can assist in the planning, analysis, design, documentation, prototyping and construction of business information systems, in accordance with one or more popular software engineering methodologies. The detractors say that these tools produce nothing and end up as shelfware. The proponents argue that correct definition of system requirements and correct database design with prototyping support are the key components of quality systems development. Many software engineering methodologies, including the model proposed in this thesis, address the appropriate phases of the systems development process, from the requirement analysis to implementation. The tools in general provide the facility and framework for producing specification for databases, as in the case of our model.

It would follow then that the tools will become available to construct complex information business systems with major productivity gains. What the tools are not going to eliminate is the need to identify and define the user's requirements into logical information system design. In other words, the guidelines we have proposed in our systems development framework will never become obsolete. In creating this framework, we have viewed systems project life cycle by constraining the definition of system for the purposes of this thesis. A system is defined as the method of processing information to support specific business function or specific decision making process. It may consist of a mixture of human procedures and computer systems and must operate within the organisational structure of the company.

The proposed model sees a systems project as one-time investment activity which is necessary to develop and install a new system. The model proposes that there should be a beginning and an end to a systems project, and that the process of developing a system can be broken down into a series of phases, containing sets of activities, which are essential for effective control. Key business objectives are defined for each phase, and these apply, irrespective of the technology proposed for the source of the software. Completion of a particular phase must be registered and approval given to continue to invest in the system project. A phase may require the creation of one or more deliverables, which are designed to help achieve the phase objectives. A deliverable then is not an end product in itself. Its role is to make a specific contribution towards the objectives set for a particular phase. The combination of phases and deliverables proposed represent a structured and consistent framework, within which systems projects are developed. This framework is known as the new model for developing strategic and tactical information and decision support systems with the quality objectives of the "right first time" principle, especially where well-defined tasks are involved.

The conclusions of the argument offered in this thesis is that user involvement at an early stage and throughout the systems development process is very important for the design and development of useful, usable, desirable and quality information systems. It is clear from our study that users have an important role to play, and that user related issues are significant

when the developed system is introduced during training, user acceptance testing and final installation. This process we have argued is an adaptive process where the user adapts to the system functions and as the system becomes accepted by users, it stimulates the development of new requirements for system enhancement. Further, as the system becomes successful, further business opportunities are explored with the aid of the system.

Finally, we believe we have provided a useful framework for the development of information systems in which a true synergy and symbiosis between systems developers and end user community are achieved.

REFERENCES

Alexander, Ian (1986). Expert Explanation. Workshop Conference Report.
Expert Systems User, Vol. 2, No. 1, pp. 25-27.

Aleksander, Igor (1986). Designing Intelligent Systems an - Introduction
Kogan Page Ltd.

Alter, S.L. (1978). Development Patterns for Decision Support System
MIS Quarterly, Vol.2, No 3, PP. 33-42.

Alty J. L. (1981). The "CONNECT" User Manual. Man-Machine Interaction Group.
Computer Science Department, Strathclyde University.

Alty J. L. (1982). Use of Path Algebras in an Interactive Adaptive Dialogue System.
Department of Computer Science, University of Strathclyde, Glasgow G1 1XH.

Alty J. L. (1984). The Application of Path Algebras to Interactive Dialogue Design.
Behaviour and Information Technology (1984), Vol. 3, No. 2, PP. 119-132.

Argyris, et al, (1985). Action Science For Concept, Methods and Skills for
Research and Intervention. Jossey-Bass Inc, Publishers, San Francisco, California.

Arinze, B. (1989). Developing Decision Support Systems from a Model of the
DSS-User Interface. In knowledge Based Management Support Systems.
Doukidis, G.I, Land, F. and Miller, G. (eds). Ellis Horwood Ltd, Publishers.

Baldwin, R.R. (1982). Reportage On Spring 1982 IEEE Compcon Conference.
ACM SIGSOFT Software Engineering Notes 7 (2), 13-20.

Bally et al, (1977). A Prototype Approach to Information System Design and
Development. Information and Management I (1). 21-26.

Baskerville R. (1989). Security of Decision Support Systems. In Knowledge Based
Management Support Systems:Doukidis, G., Land, F and Miller, G. (eds).
Ellis Horwood Ltd. Publishers, Chichester.

Bell et al, (1977). An Extendable Approach to Computer-Aided Software
Requirements Engineering. IEEE, Transactions on Software Engineering,
3 (1), 49-60.

Benchimol et al, (1987). Developing Expert Systems for Business. North Oxford
Academic Publishers ltd., Kogan Page ltd., London.

Benyon,D. (1985). "Monitor" A Self-Adaptive User Interface. In Shackel. B. (ed).
Proc INTERACT 84. North Holland (a) IFIP, PP. 335-340.

Biggs, C.L. et al, (1980). Managing the System Development Process,
Prentice-Hall Inc., Englewood-Cliffs N.J.

Boehm, B.W. (1974). Some Step Towards Formal and Automated Aids to Software
Requirements Analysis and Design. Proceeding IFIP74, PP. 192-197.

Boehm, B.W. (1976). Software Engineering.IEEE Transactions, Computers,
PP. 1226-12411.

Boehm, B.W. (1980). Developing Small-Scale Application Software Product:
Some Experimental Results. Proceedings, IFIP,8th World Computers Congress,
PP. 321-326.

Boehm, B.W. (1981). Software Engineering Ecnomics. Prentice-Hall, Englewood Cliffs, N.J.

Boehm, et al, (1984). Structuring Into Sub-Systems. The experience of a Prototyping Approach. ACM SIGSOFT Software Engineering Notes, 9(5), PP. 23-27.

Boland, R.J. (1978). The Process and Product of System Design. Management Sci., 24,9(1978), PP. 887-898.

Bonczek, R.H, Holsapple, C.W. and Whinston, A.B. (1980). The Evolving Roles of Models in Decision Support Systems. Decision Sciences, Vol.11, No.2, PP. 339-356.

Bonczek, R.H. et al, (1981). The Evolution from MIS to DSS: Extension of Data Management to Model Management. Proceedings of the NYUS Symposium on Decision Support Systems. New-York, PP. 21-22.

Bonczek R.H.,, Holsapple C.W. and Whinston A. B. (1981). Foundations of Decision Support Systems, Academic Press, New York 1981.

Bonet, R. and King. A. (1984). Structure into subsystem. Experience of a Prototyping Approach. ACM Sigsoft Software Engineering Notes. 9(5), PP. 23-27.

Bowen K. A. and Kowalski (1981). Amalgamating Language and Metalanguage in Logic Programming. Report 4/81, School of Computer and Information Science. Syracuse University, Newyork.

Budde, R. and Sylla (1984). Modeling to Target System. In Approaches to Prototyping. PP. 31-48, Springer-Verlag, Berlin.

Butler Cox Foundation (1990). Systems Development Productivity Tools. Butler Cox and Partners ltd,. Butler Cox House, 12, Bloomsbury Square, London WC1A 2LL, England.

CACI, (1983). The CACI Structured Systems Delevopment Method. CACI INC. International.

Candy, L. and Edmonds, E.A. (1988). Expert System Development for an Office Environment. Users, Evaluation and the Design Process Proc.IFAC Conference on Man-Machine Systems. Oulu, Finland.

Candy, L. and Lunn. S. (1988). Design Strategies for Expert System A Case Study, Human and Organisational Issues of Expert Systems Joint ICL and Ergonomics Society Conference, Strafford,

Carlson, E.D, (1979). An Approach for Designing Decision Support Systems Data Base, PP. 3-15.

Carter et al, (1975). A Sudy of Critical Factors in Management Information Systems for U.S. Air Force, Colorado State University, Fort Collins, Colorado.

Checkland, P. B. and Griffin R. (1970). Management Information Systems: a System View Journal of Systems Engineering,1, No.2.

Checkland, P.B. (1972). Towards a Systems-Based Methodology for Real-World Problem Solving. Journal of System Engineering, Volume 3, No. 2.

Checkland, P.B. and Wilson, B. (1980). Primary Tasks and Issue-Based Root Definitions in Systems Studies. Journal of Applied Systems Analysis,Vol.7,PP. 51-54

Checkland, P.B. (1981). Systems Thinking, Systems Practice,
PP. 165-166, 229-233, Willey.

Checkland, P.B. (1984). System Theory and Information Systems. In Beyond productivity. Information Systems Development for Organisational Effectiveness. TH. M.A. Bemelmans (ed). Elsevier Science Publishers (North-Holland), IFIP.

Christensen N. and Kreplin K (1984). Prototyping of User Interface.
In Approachs to Prototyping. PP. 58-67. Springer-Verlag, Berlin.

CICA, (1986). Computer Control Guidelines. The Canadian Institute
of Chartered Accounts. 1SBN 0-88800-135-5.

Clark I.A (1981). Software Simulation as a Tool for Usable Product Design.
IBM System Journal 20(3), PP. 272-93.

Clark P. et al, (1985). User Acceptance of Information Technology Through Prototyping. Human-Computer Interaction. INTERACT 84. B. Shackel (ed). Elsevier Science Publishers B.V., North-Holland, IFIP.

Clausen, H. (1978). Concepts and Experiences with Participative Design Approaches. Proceedings of the Conference on Design and Implementation of Computer Based Planning Systems, Cologne.

Clency W. J. (1983). The Epistemology of a Rule-Based Expert System - a Framework for Explanation. Artificial Intelligence, Vol 20(1983), PP. 215-251.

Connor, Denis. (1985). Information System Specification and Design Road Map.
Prentice/Hall International, Inc, Englewood Cliffs, New Jersey.

Cyert and March (1983). The User Information Satisfaction.
Communications of the ACM, 6, PP. 126.

Damodaran (1976). The Role of User Support. NATO ASI on Man-Computer Interaction Proceedings. Shackel B., (ed), Loughborough University, PP. 261-272.

Damodaran, L, Simpson A & Wilson P (1980). Designing Systems for People.
ISBN 0-85012-242-2, NCC Publications, National Computing Centre,
Oxford Rd, Manchester.

Damodaran, L and Eason K.D. (1981). Design Procedures for User Involvement and User Support. In Computing Skills and the User Interface.
M.J. Coombs and J.L. Alty (eds), PP. 373-388.

Dearnley, P.A. and Mayhew P. (1981). Experiments in Generating System Prototypes.
Processing 1st European Workshop on Information Systems Teaching.

Dearnley, P.A. and Mayhew P. (1984). On the Use of Software Development Tools
in the Construction of Data Processing System Prototypes.
In Approach to Prototyping, PP. 68-79, Springer-Verlag, Berlin.

DeBrabander, B. and Edstrom A. (1977). Successful Information Systems Development Projects. Management Sci., 24,2(1977), PP. 191-199.

Demarco T. (1978,1979). Structured Analysis and System Specification,
Prentice/Hall, Inc, Englewood Cliffs, NJ,1979, PP. 16,47 and 215-226.

Denert E. (1977). Specification and Design of Dialogue Systems With State Diagrams. Proceedings of the International Computer Symposium, Lieg.

Dixon, F.J. (1985). Simplifying Screen Specifications the "Full Screen Manager" Interface and "Screen Form" Generating Routines.
The Computer Journal 28(2), 117-127.

Doukidis, G.I, L and F. and Miller G. (1989).Knowledge-Based Management Support Systems.
Ellis Horwood Ltd, Publishers. Market Cross House, Cooper Street, Chichester, England.

Downes, P.M. (1989). Practical Data Analysis.Blenheim Online Publications,
Blenheim House, Ash Hill Drive, Pinner, Middlesex, HAS 2AE, UK.

Draper, S.W. and Norman D.A (1985). User Interface. IEEE Transactions
on Software Engineering 11(3), PP. 252-258.

Eason, K.D. (1977). Human Relationships and User Involvement in System Design.
Computer Management, 19, 10-12.

Eason, K.D and Damodaran (1981). The Need of the Commerical User.
In Coombs M.J, and Alty J.L. (eds), Computing Skills and the User Interface,
PP. 389-426, London: Academic Press.

Eason, K.D. Harker, S.D.P, Raven, P.F, Brailsford, J.R. and Cross, A.D. (1987).
A User-Centred Approach to the Design of a Knowledge-Based System.
Proc INTERACT 87. Bullinger H.J. and Shackel B. (eds),North Holland, IFIP.

Edmonds E. A. (1978). Adaptable Man-Machine Interfaces for Complex Dialogues.
Proceedings of the European Computing Congress, London, PP. 639-646.

Edmonds E. A. & Guest S. p. (1978a). "SYNICS"- a Fortran Subroutine Package for
Translation Man-Computer Interface Research Group Report No. 6., Leicester Polytechnic.

Edmonds E. A. (1981). Adaptive Man-Computer Interfaces. In Coombs M. J., and
Alty J.L. (eds), Computing Skills and the User Interface, PP. 389-426. London: Academic Press.

Edmonds E. A. (1982). The Man-Computer Interface: a Note on Concept and Design.
Int. J. Man-Machine Studies (1982), 16, 231-236.

Edmonds E. A. and Guest S. P. (1984). The SYNICS2 User Interface Manager.
Proc. INTERACT 84, 1st IFIP Conference on Human-Computer Interaction, IEE.

Edmonds E.A. (1987). Good Software Design: What Does it Mean?. In Bullinger, H.J and
Schackel, B (eds), Proc INTERACT 87. North Holland c IFIP, PP. 333-335.

Edmonds E.A. et al, (1989). Issues in the Design of Expert Systems for Business.
In Forbes Gibb (eds). Expert Systems for Information Management. Publ. Taylor Graham.

Edstrom, A. (1977). User Influence and the Success of MIS Projects: A Contingency
Approach, Human Relations, Volume 30, No. 7, PP. 589-607.

Ference,T.P. and Uretsky,M. (1976). Computers in Mmanagement: Some Insight into
the State of Revolution, Management Datamatics,5,2(1976), 55-63.

Fitter, M. (1981). Toward More Natural Interface Systems.
Int. J. Man-Machine Studies (1982),11

Floyd, C. (1984). A Systematic Look at Prototyping. In: Budde, R, Kuhlen Kamp, K., Mathiassen, L. Zullighoven, H., (eds), Approaches to Prototyping (Springer Verlag, 1984), PP. 1-18.

Floyd, C. (1986). A Comparative Evaluation of System Development Methods. In Olle, T.W., Sol, H.G., and Verrijn-Stuart, A.A. (eds), Information Systems Design Methodologies. IFIP WG 8.1 Working Conference on Comparative Review of Information Systems Design Methodologies: Improving the Practice, North Holland, pp 19-37.

Foley J. D. (1980). The Structure of Interactive Command language. IFIP Workshop on Methodology of Interaction, Guedj et al, (eds), North-Holland, PP. 227-234.

Foley J. D. (1980). Model of Interaction. IFIP Workshop on Methodology of Interaction, Guedj et al, (eds), North-Holland, PP. 73.

Fox J., Alvey p. and Myers C. (1983). Decision Technology and Man-Machine Interaction. Expert Systems 83, Churchill College, Cambridge, PP.114.

Franz, C.R and Robey, D. (1978). An Investigation of User-Id System Design: Rational and Polhilal Perspectives, MIS Quarterly 2(4), 62-82.

Franz,C.R. (1979). Contingency Factors Affecting the User Involvement Role in the Design of Successful Information Systems. Ph.D. Dissertation, University of Nebraska, 1979.

Fuerst, W.T. (1979). Characteristics Affecting Decision Support System Usage. National AIDS Conference Proceedings,1979,172.

Gallier, R.D. (1984). An Approach to Information Needs Analysis. Human-Computer Interaction. INTERACT'81. Shackel B. (ed). Elsevier Science Publishers B.V. North-Holland.

Gallagher, C.A. (1974). Perceptions of the Value of a Management Information System Acad. Management J.,17,1(1974), PP. 46-55.

Gallier, R.D. and Lyons, L.W., (1984). Attitudes of Western Australian Managers to Computerised Information Systems. Human-Computer Interaction. INTERACT 84. Shackel, B. (ed), Elsevier Science Publishers B.V., North-Holland, @ IFIP 1984.

Galliers, R.D. (1985): An Approach to Information Needs Analysis. Human-Computer Interaction. INTERACT'84. Shackel, B. (ed), Elsevier Science Publishers B.V, North-Holland, @ IFIP, 1985.

Gaschnig (1982), Duda et al, (1979) and Reboh (1981). "PROSPECTOR". In Johnson L. and Keravnou E. T. (1985): "Expert System Technology - A Guide". ABACUS Press 1985. Gearing P. E. et al (1982). The Interviewer-Reasoner Model:An Approach to Improving System Responsiveness in Interactive AI Systems, the AI Magazine,24-27(Fall 1982).

Gehani, N. (1982b). A Study in Prototyping. ACM SIGSOFT Software Engineering Notes. 7(5),PP. 71-75.

Gerrity, T.P. (1971). Design of Man-Machine Decision Systems: An Application to Portfolio Management. Sloan Management Review, Vol.12, No.2, PP. 59-72.

Gibson, H.L. (1978). Determining User Involvement. System Management, PP. 20-22.

Gilb, T. (1981). Evolutionary Development. ACM SIGSOFT Software Engineering Notes 6(2), 17.

Ginzberg, M.J. (1979). Re-design of Managerial Tasks: a Requisite for Successful Decision Support Systems. MIS Quarterly, Vol.2, No.7, PP. 39-52.

Ginzberg, M.J. (1980). An Organisational Contingencies View of Accounting and Information Systems Implementation. Accounting,Organisations and Society, Vol.5, No.4, PP. 369-382.

Ginzberg, M.J. and Stohr, E.A. (1982). Decision Support Systems: Issues and Perspectives. In Decision Support Systems. Proceeding of the NYU Symposium on Decision Support Systems New York, 21-22.

Glasson, B.C. (1984). EDP System Development Guidelines, Butterworths, London.

Glasson, B.C (1985). Guidelines for User Participation in the System Development Process. Human-Computer Interaction, INTERACT'84. Shackel. B (ed).
Elsevier Science Publishers B.V. North-Holland @ IFIP 1985.

Good, D.J., Whiteside, A. Wiston, D.R. and Jones, S.J. (1984). Building a User-Derived Interface. Communications of the ACM 27 (10), PP. 1032-43.

Gorry, G.A. and Scott Morton, M.S. (1971). A Framework for Management Information Systems. Sloan Management Review, Vol.13, No.1. PP. 55-70.

Gould, J.D. (1987). How to Design Usable Systems. Human-Computer Interaction.
INTERACT 87 Bullinger, H.J. and Schackel, B. (eds).
Elsevier Science Publishers B.V. North-Holland @ IFIP 1987.

Green R. H. and Wood S. (1982). Expert Knowledge Elicitation or How to Grow Trees.
Expert Systems 83, Churchill College, PP. 272-276.

Groholt, P. (1980). Social Development and Accountability, Professionalism and the Future of Systems Designers. In N.Bjorn-Andersen (ed). The Human Side of Information Processing. Amsterdam, North-Holland.

Grubber, W.H., Synnott, W.R. (1981). Information Resource Management. John Wiley and Sons Inc.New-York.

Guedj et al, eds (1980). Definition of Interaction. IFIP Workshop on Methodology of Interaction, North-Holland, PP. 69-111.

Guenthner F., Lehmann H. and Schonfeld W. (1986). A Theory of the Representation Of Knowledge. IBM Journal of Research and Development, Vol. 30, No.1, PP. 39.

Guest S. P. (1982). The Use of Software Tools for Dialogue Design. Int. J. Man-Machine Studies (1982), 16,263-285.

Guilfoyle C. (1986). What the User Want. Expert System User,
Vol. 2, No.1. PP. 16.

Guitierrez, O. (1989). The Role of an Experimental Approach in the Decision Support Systems. In knowledge-Based Management Support Systems. Doukidis, et al, (eds).

Guthrie, A. (1974). Attitudes of the User Managers Towards Management Information Systems. Management Information, 3,5.

Guthrie, A. (1972). A Survey of Canadian Middle Managers- Attitudes Toward Management Information System. Carleton University Press, Ottawa, Ontario.

Guthrie, A. (1974). Attitudes of User-Managers Towards MIS.
Management Informatics, 3, 5(1974),221-232.

Hansen W. J. (1971). User Engineering Principles for Interactive Systems.
Americam Federation of Information Processing Societies Conference Proceedings,
39,523-532.

Hayes-Roth et al, (1983). Building Expert Systems. Addison-Wesley, MA,USA.

Haywood, S. (1987). The KADS Methodology: Analysis and Design for Knowledge Based
Systems: Esprit Report Y1. P1098, STC February.

Hawgood, J. (1985). Information Morphology in Bureaucratic Sub-Systems. In :
R.H. Sprague et al, (eds), Knowledge Representation for Decision Support Systems,
North Holland, Amsterdam,1985.

Hedberg, B. (1975). Computer Systems to Support Industrial Democray. In Human Choice
and Computers, E. Mumford and H. Sackman (eds). American Elsevier, New York.

Hekmatpour, S. and Ince, D.C. (1986). Rapid Software Prototyping Human-Computer
Human-Computer Interaction, INTERACT'84. Shackel B. (ed). Elsevier Science Publishers B.V.
North-Holland.

Hekmatpour, S. and Ince, D.C. (1987). Evolutionary Prototyping and the
Human-Computer Interface. Proc. INTERACT'87. Bullinger, M.J. and Shackel B. (eds),
North-Holland. C. IFIP.

Hendeson J.C. and Ingraham, R.S. (1982). Prototyping for Decision Support System:
A Critical Appraisal. In Decision Support Systems (ed E.A. Stohr), PP. 79-96. North-Holland,
Amsterdam.

Hopgood F. R. A. and Duce D. A. (1980). A Production System Approach to Interactive
Program Design. In Methodology of Interaction, Guedi, P.J.W. et al,(eds),
Amsterdam North-Holland, PP.247-264.

Hold E. D., Butler C. W. and Richardson G. L. (1986). Knowledge Based Systems in
Commercial Environment. IBM Systems Journal, Vol. 25, No. 2, PP. 147-158.

Hughes S. (1986). Expert Explanation. Workshop Conference Report.
Expert Systems User, Vol. 2, No. 1, PP. 25-27.

Igersheim, R.H (1976). Management Response to an Information System.
AFIPS Conference Proceedings 1976,877-882.

Innocent P. R. (1982). Towards Self-Adaptive Interface Systems. Int. J. Man-Machine
Studies (1982), 16, 287-299.

Ives, B., Olsen, M.H. and Bavoudi, F.J. (1983). The Measurement of User Satisfaction
Communications of the ACM 26,10.

Ives, B. and Olson, M.H. (1983). User Involvement and MIS Success: A Review of Research.
Management Science 30(5), (1984), PP. 586-603.

Jackson A. h. (1983). The Design and Development of an Expert System to Assist in the
Analysis of Crash Dumps. Expert Systems 83, Churchill College Cambridge, PP. 256-261.

Jackson, M.A. (1983). System Development. Prentice Hall, International, Ince, Series in
Computer Science, C.A.R. Hoare, Series Editor.

Jenkins, A.M. (1982). Prototyping: the New Pardigm for Systems Development.
MIS Quart. Vol 6 No. 3 (September 1982).

Johnson L. & Keravnou, E. T. (1985). Expert System Technology - A Guide.
ABACUS Press 1985.

Jones, C.B. (1980a). The Role of Formal Specification in Software Development Life Cycle Management, Infotech State of the Art Report 8(7),117-33.

Jones, C.B. (1980b). Software Development: a Rigorous Approach. Prentice Hall.

Kaiser, K.M. and Srinivason A.(1980). The Relationship of User Attitude toward Design Criteria and Information System Success. National AIDS Conference Proceedings, 1980, 201-203.

Kay, A. (1980). Model of Interaction. IFIP Workshop on Methodology of Interaction.
Guedj et al, (eds), North-Holland, PP. 73.

Karat, J. et al, (1987). Evaluating User Interface Complexity. In Bullinger, H.J and Shackel, B. (eds). Proc INTERACT 87. North Holland IFIP, PP. 490-495.

Keen, P. G. W. and Hackathorn, R. D. (1979). Decision Support Systems and Personal Computing. Sloan School of Management Working Paper 1088-79,
Cambridge, Massachusetts.

Keravnou E. T. & Johnson L. (1987). Intelligent Handling of Data by Integration of Commonsense Reasoning. Knowledge-Based Systems. Butterworth & Co (Publishers) Ltd., Vol. 1, No. 1, PP.32.

Keus, H.E. (1982). Prototyping: A More Reasonable Approach to System Developoment.
ACM SIGSOFT Software Engineering Notes 7(5), PP. 94-95.

Kidd A. L. and Cooper M. B. (1983). Man-Machine Interface for An Expert Systems 83,
Churchill College, Cambridge, pp. 8-14.

Kidd A. L. (1986). What do Users Ask? Some Thoughts on Diagnostic Advice.
In "All the Answers - to the wrong questions". Mainfeature, Expert System User, Vol. 2, No.1, pp. 16-17.

Kieras, D.E. and Polson, P.G. (1985). An approach to the formal analysis of user complexity. International Journal of man-machine studies,22,365-394.

King, W.R. and Rodriques J.I.(1978). Evaluating MIS, MIS Quart 2,3(1978),43-52.

King, W.R. and Rodriques J.I (1981). Participative Design of Strategic Decision Support Systems: An Empirical Assessment, Management Sci, 27,6(1981),717-726.

Kraushaar, J.M. and Shirland, L.E. (1985). A Prototyping Method for Applications Development by End Users and Information Systems Specialists. MIS Quarterly 3, 189-197.

Land,F. (1989). Decision Support Systems. In Knowledge-Based Management Support Systems.
Doukidis, G., Land,F. and Miller, G.(eds). Ellis Horwood ltd., Publishers, Chichester.

Lehman, J.D. and Yaveneh N. (1985). The Total Life Cycle Model.
Proceedings 3rd International Workshop on Software Specification and Design, PP. 135-137.

Leonard-Barton, D. (1987). The Case for Integrative Innovation:
An Expert System at Digital. Sloan Management Review (1987), PP. 7-19.
Harvard Business School, Cambridge, MA 02138 U.S.A.

Leonard-Barton, D. (1988). Implementation as Mutual Adaption of Technology and Organisation. Research Policy 17 (1988), 251-267. North-Holland.

Leibrandt, U. and Schnupp, P. (1984). An Evaluation of Prolog as Prototyping System. In Approach to Prototyping, PP. 424-433. Springer-Verlag, Berlin.

Levine, J. (1980). Why a Lisp-Based Command Language SIGPLAN Notices 15(5),49-53.

Lock, E. A. and Schweiger, D.M. (1979). Participation in Decision Making: One More Look, Organisational Behavior, 1(1979),265-339.

Long, J. (1986). People and Computers: Designing for Usability: An Introduction to HCI'86, Harrison and Monk (eds) Cambridge, (1986), pp. 7-23.

Lucas, II. C. JR. (1973). A Descriptive Model of Information Systems in the Context of the Data Base, 5,2 (Winter 1973), 27-36.

Lucas, II. C. JR. (1974). System Quality, User Reactions, and the Use of Information Systems, Management Information 3,4(1974a), 207-212.

Lucas, II. C.JR. (1974b). Toward Creative System Design, Columbia University Press, New York.

Lucas, II. C. JR.(1975). Why Information Systems Fail, Columbia University Press, New York.

Lucas, II. C. JR. (1976). The Implementation of Computer Based Models, National Association of Accountant, New York.

Lucas, II. C. JR. (1978). The Evolution of An Information System: From Key-Man to Every Person. Sloan Management Rev. Winter (1978).

Macdonald, I.G. (1986). An Improved, Automatable Methodology for the Design of Data Sharing Systems. IFIP WG.8.1 Working Conference on CRIS.86: Improving the Practice. Olle, T.W., Sol, H.G. and Verrijn-Stuart, A.A. (eds), North-Holland, Amsterdam.

Maddison et al, (1983). Information System Methodologies. Wiley Heyden Ltd., the British Computer Society.

Maguire M. C. (1981). An Evaluation of Published Recommendations on the Design of on the Design of Man-Computer Dialogues. Int. J.Man-Machine Studies (1982), 16, 237-261.

Maish, A.S. (1979). A User Behavior Toward His MIS. MIS Quart, 3,1(1979), 39-52.

Martin James (1973). Design of Man-Machine Dialogues. Englewood Cliffs, New Jersey, Prentice-Hall Inc.

Martin. J. (1989). Information Eengineering Ffacility. James Martin Associates. I.E. Product Group, Texas Instruments Incorporated.

Mason, R.E. and Carey, T.T. (1983). Prototyping Interactive Information Systems. Communications of the ACM 26(5), 347-54.

Mattern, R.D. (1981). The Integration of Business Information Systems for Decision Support Systems in APL. In Decision Support Systems: Issues and Challenges. Ginzberg, M.J., Reitman, W. and Stohr, E.A. (eds). Proc NYU Symposium on Decision Support Systems. North-Holland, Amsterdam.

McNielle, A.T. (1986). Jackson System Development (JSD). IFIP WG8.1 Working Conference on CRIS86: Improving the Practice. Olle, T.W., Sol, H.G and Verrijn-Stuart, A.A. (eds), North-Holland, Amsterdam.

Mckeen, J.D. (1983). Successful Development Strategies for Business Application Systems. MIS Quarterly, PP. 47-65.

Michalski R. S. and Chilausky R. L. (1980). Learning by Being Told and Learning from Examples: An Experimental Comparison of the two Methods of Knowledge Acquisition in the Context of Developing an Expert System for Soybeam Disease Diagnosis. Policy Analysis and Information Systems 4, No. 2.

Milliken K. R., Cruise A. V., Ennis R. L., Finkel A. J., Hellerstein J. L., Loeb D. J., Klein D. A., Masullo M. J., Vanwoerkom H. M. and Waite N. B. (1986). "YES/MVS" and the Automation of Operation for Large Complexes. IBM Systems Journal, Vol. 25, No. 2 (1986), PP. 159.

Moore, J.H., and Chang, M.G. (1980). Design of Decision Support Systems. Data Base, Vol.12, Nos. 1 and 2 (Ffall 1980), PP. 8-14.

Moran T. P. (1981). The Command Language Grammer - A Representation of the User Interface of Interactive Computer Systems. Int. J. Man-Machine Studies (1981), 15, 3-50.

Moran T. P. (1980). A Framework for Studying Human-Computer Interaction. IFIP Workshop on Methodology of Interaction, North-Holland, PP. 293-301.

Moran T. P, Card s.K, Newell A. (1983). The Psychology of Human-Computer Interaction. Hillsdale NJ, Erlbaum.

Moubarak, S. (1987). The Orders Management Systems Project Proposal. Massey-Ferguson, Coventry, England.

Moubarak, S. (1989). Incorporating Dialogue Design Support into the Systems Development Process. M. Phil Thesis. Leicester Polytechnic,Leicester.

Moubarak, S. (1990). Methods, Techniques and Tools. Internal Document. European Systems Development.TNT European Regional Computing Center, Atherstone, England.

Muford, E. and Banks, O (1967). The Computer and the Clerk. London: Routledge and Kegan Paul.

Muford, E. (1980). Consensus Systems Design: an Evaluation of this Approach, Design and Implementation of Computer Based Information Systems, N. Szyperski an E. grochla (eds), Sijthoff and Noordhoff Gromingen.

Munson, D.R. (1979). Software Maintainability: A Practical Concern for Life Cycle Costs. Computer 2, 103-109.

Musser, D.R. (1979). Abstract Data Type Specification In the AFFIRM System. Proc Conference on the Specification of Reliable Software, PP. 47-57.

Nauman, J.D. and Jenkins, A.M. (1982). Prototyping: the New Paradigm for Systems Development. MIS Quarterly 3, PP. 29-44.

Newell, A. and Card, S.K. (1985). The Prospects for Psychological Science. In Human-Computer Interaction,1,209-242.

Nosek, J.T. (1984). Organisation Design Choices to Facilitate Evolutionary Development of Prototype Information Systems. In Approaches to Prototyping, PP. 341-355, Springer-Verlag, Berlin.

OASIS (1989). The Management of Marketing Information. Organisation and System Inovations ltd., Tectonic Place, Holport Rd, Maidenhead, Berkshire,

Orr, K.T. (1977). Structured Systems Development, Yourdon Press, Nework.

Orr, K.T. (1981). Structured Requirements Definition. Ken Orr and Associates, Inc, Topeka, KS.

Palme, J. (1976). Interactive Software for Human. NATO ASI on Man-Computer Interaction Proceedings, Shackel B., (ed), North-Holland, PP. 167-221.

Parnas, P. (1969). The Use of Transition Diagrams in the Design of User Interface for an Interactive System. Proceedings National Association for Computing Machinery Conference, PP. 379-389. Man-Machine Studies (1982), 16, 263-285.

Parnas, D.L. (1972).Encomposing Systems into Modules, Communications of the ACM 15(12),1053-8.

Parnas, D.L. (1979). Designing Software for Ease of Extension and Contraction IEEE Transaction Software Engineering 5(2),128-37.

Patton, B. (1983). Prototyping- A Momenclature Problem. ACM SIGSOFT Software Engineering Notes 8(2), 14-16.

Powers, R.F. and Dickson, G.W. (1973). MIS Project Management: Myth, Opinions, and Reality, California Management Rev, 15, 3(1973), 147-156.

Reitman, W. (1982). Applying Artificial Intelligence to Decision Support. In Decision Support Systems, M.J. Ginzberg, W. Reitman and E. Stohr (eds). North-Holland, Publishing Co., Amsterdam, Holland.

Robey, D. and Farrow, D. (1979). Information System Development: Some Dynamics of User Involvement, National AIDS Conference Proceedings, November 1979, 149-151.

Robey, D. and Farrow, D. (1982). User Involvement in Information System Development: A Conflict Model and Impirical Test, Management Sci, 28,1(1982),73-85.

Roberts, E.B. (1977). Generating Effective Corporate Innovation, Technology Review, October-November 1977, PP. 25-33.

Rosen et al, (1986). Computer Control Guidelines. The Canadian Institute of Chartered Accountants (CICA).

Rosenburg V. (1973). A Technique for Monitoring user Behaviour at the Computer Terminal Interface. Journal of the American Society of Information Science 24(1),71.

Ross, D.T. and Schoman, K.E. Jnr (1977). Structured Analysis for Requirements Definition. IEEE Transactions on Software Engineering 3(1),6-15.

Rothwell, R. (1977). Some Problems of Technology Transfer into Industry: Examples from the Textile Machinery Sector, IEEE Transactions on Engineering Management, EM 25(1), February 1977, PP.15-20.

Rzevski, G (1984). Prototypes Versus Pilot Systems: Strategies for Evolutionary Information System Development. In Approaches to Prototyping, PP. 356-367, Springer-Verlag,Berlin.

Sale, A. E. (1985). The Codasyl Proposal for a Screen Management Facility. Computer Bulletin 1(1), 24-25.

Sartore, A. (1976).Implementating a Management Information System: The Relationship of Participation, Knowledge, Performance, and Satisfaction in An Academic Environment, Ph.D. Dissertation, University of California at Irvine 1976).

Schewe, C.D. (1976). The MIS User: An Exploratory Behavioral Analysis. Acad Management J., 19, 4, 577-590.

Schultz, R.Z. and Slevin D.P. (1975). Implementation and Organisational Validity: An Emprical Investigation. In Implementing Operations Research Management Science, R.L. Schultz and D P Slevin (eds), American Elsevier, New York.

Shackel, B. (1969). Man-Computer Interaction-The Contribution of Human Science. Ergonomics,12,485-489.

Shackel, B. (1976). Man-Computer Interaction (MCI) - Reality and Problems. NATO ASI on Man-Computer Interaction Proceedings, Loughborough University, pp. 4.

Shackel, B. (1981a). Man-Computer Interaction. Alphen-aan-den Rijn, Netherlands, Sijthaff and Noordhoff.

Shackel, B. (1981b). The Concept of Usability. Proc IBM Software & Information Usability Symposium, Poughkeepsie NY, 15-18 September, PP. 1-29.

Shackel, B. (1984). Designing for People in the Age of Information, Human-Computer Interaction. INTERACT 84/, B.Shackel (ed),Elevier Science Publishers B.V., North-Holland,IFIP,1985.

Shackel, B. (1986). Ergonomics in Design for Usability. People and Computers; Designing for Usability. Proc 2nd Conference BCS HCT. Specialist Group September 1986. Harrison, M.D. and Mouk, A.F. (eds).

Shneiderman, B. (1979).Human Factors Experiments in Designing Interactive Systems, Computer 12, PP. 9-19.

Shneiderman, B. (1983). Direct Manipulation: a Step Beyond Programming Languages. IEEE Computer 16,18 (Agust 1983), 57-69.

Shwa, L. G. and Gaines, B. R. (1983). Computer Aid to Knowledge Engineering. Expert Systems 83, Churchill College, Cambridge, PP. 263-271.

Silverberg, B.A (1981). An Overview of the SRI Hierarchical Development Methodology. In Software Engineering Environments, (ed R.Bunke),PP. 235-252, North-Holland, Amsterdam.

Spence, J. O. (1978). A Case Study Analysis of Organisational Effectiveness Between User-Manager and Information Service Department Personnel. Ph.D. Dissertation, Texas Tech University.

Sprague, R.H. (1980). A Framework for Research on Decision Support System. In Decision Support Systems: Issues and Challenges, Fick, G. and Sprague, R.H. (eds), Pergamon press.

Sprague, R.H. (1986). Decision Support Systems in Context: in E.R. Mclean and H.G. Sol (eds) Decision Support Systems: a Decade in Perspective, IFIP, 1986.

Stewart, T. F. M. (1976). The Specialist User. NATO ASI on Man-Computer Interaction Proceedings, Shackel B., (ed), Loghborough University, PP. 457-483.

Sutcliffe, A.G. (1988). Human-Computer Interface design, MacMillian Education Ltd. Publishers Houndmills, Basingstoke, Hampshire UK.

Sutton, J.A. and Sprague, R.H. (1978). A Study of Display Generation and Management in Interactive Business Applications.IBM Research Report RJ2392.

Swanson, E.B. (1975). Management Information Systems: Appreciation and Involvement, Management Scl, 21 (1974), 178-188.

Symonds, A. J. (1986). Introduction to IBM's Knowledge-Systems Product. IBM Systems Journal, Vol. 25, No. 2, PP. 2-13.

Taggart, (JR), W.M. and Tharp, M.D. (1977). A Survey of Information Requirements Analysis Techniques. Computing surveys, Vol.9, No.4.

Teichroew, D. and Hershey 3, E.A. (1977). PSL PSA: A Computer-Aided Technique for Structured Documentation and Analysis of Information Processing Systems. IEEE Transactions on Software Engineering 3(1), 41-48.

Ten-Hagen, P.J.W (1980). A Conceptual Basis for Graphical Input and Interaction. In Guedi, R.A., Ten Hagen, P.J.W, Hopgood, F.R.A, Tucker, H.A and Duce, D.A. eds, Methodology of Interaction, PP. 239-246 Amsterdam, North-Holland.

Thomas, R. C. & Burns A. (1982). The Case for Distributed Decision Making Systems The Computer Journal, Vol. 25, No. 1, PP. 148-152.

Thurston, P.H. (1959). Systems and Procedures Responsibility, Harvard University Press, Cambridge, 1959.

Turban, E. (1988). Decision Support and Expert System: Managerial Perspectives, London:Macmillan.

Vanlommel, E. and DeBrabarder, B. (1975). The Organisation of Electronic Data Processing. J Business, 48,2, (1975), 391-410.

Walker A. (1986). Knowledge Systems Principles and Practice. IBM Journal of Research and Development, Vol. 30, No. 1, PP. 2-13.

Wasserman, A.I. Pircher, P.A., Shewmake, D.T. and Kersten, M.L. (1986). Developing Interactive Information Systems with the User Software Engineering Methodology. IEEE Transactions on Software Engineering, 12, 2, 326-345.

Wasserman, A.I. (1983). Characteristics of Software Development Methodologies, in Olie, T.W. Sole, H.G. and Tully, C.J. (eds). Information Systems Design Methodologies: Feature Analysis. North-Holland, Amsterdam.

Waterman, D.A. (1985). A Guide to Expert Systems. Addison-Wesley, Teknowledge Series in Knowledge Engineering.

Weyuker, E.J. (1982). On Testing Non-Testable Programs. The Computer Journal 25(4), 465-70.

Wilson, B. (1984). Systems: Concepts, Methodologies and Applications. John Willey & Sons Ltd, UK.

Wood-Harper, A.T., Antill, L. and Avison, D.E. (1985). Information Systems Definition: The Multview Approach. Blackwell Scientific Publications, Computer Science Texts.

Xauhz, B. and Newman, W.H. (1980). Strategy in Action. The Free Press, Collier MacMillian Publishers, London.

Yavitz and Newman (1982). Strategy in Action. The Execution, Politics and Payoff of Business Planning. Collier Macmillan Publishers, London.

Yourdon, E. and Constantine, L. (1979). Structured Design, Prentice/Hall, Inc., Englewood Cliffs, NJ.

Zand, D. E and Sorenson, R.E. (1975). Theory of Change and the Effective Use of Management Science, Admin. Sci. Quart., 20, 532-545.

Zmud, R.W. (1979). Individual Differences and MIS Success: A Review of the Empirical Literature. Management science., 25,10 (1979), 966-979.

Zmud, R.W (1981). An Exploratory Study of User Involvement Role Sets. Working Paper, School of Business Administration, University of North Carolina.

Zorkoczy, P. (1985). Information Technology: An Introduction, Pitman Publishing, 128, Long Acre, London WC2E GAN.

APPENDIX

SAMPLE DOCUMENTATION

CONTENT

INTRODUCTION - *2*

APPENDIX A - *Project Team, 3*

APPENDIX B - *Positioning Statement, 6*

APPENDIX C - *Project Management, 10*

APPENDIX D - *Training & Education Courses, 22*

APPENDIX E - *Entity & Function Model, 27*

APPENDIX F - *Data and Function Analysis Sample Documentation, 38*

APPENDIX G - *Business Issues and Recommendations, 68*

APPENDIX H - *Cost/Benefit Analysis, 70*

APPENDIX I - *Project Timing, 73*

APPENDIX J - *Database Structures, 75*

APPENDIX K - *Transaction Volumes, 79*

APPENDIX L - *Dealer Communication Prototype, 81*

INTRODUCTION

This appendix includes some sample documentation which illustrates the typical output of using the new systems development life cycle model-the subject for this thesis.

Certain points should be noted:

- The documents are illustrative based on the real life projects described in chapter 4.

- The examples are samples only. A full set of outputs is not provided as would be too bulky. However, the samples is consistent enough to allow the reader to follow it through.

APPENDIX A

PROJECT TEAM

- PROJECT TEAM ORGANISATIONAL STRUCTURE
- PROJECT TEAM'S OBJECTIVES

PROJECT TEAM'S ORGANISATIONAL STRUCTURE

STEERING COMMITTEE

PROJECT MANAGER

EXTERNAL USER(s) TEAM

DEALERS CLIENT ACC. MANAGERS

INTERNAL USER(s)

UK EUROPE NA

SYSTEMS DEVELOPMENT

PROJECT LEADRS:

COMMERCIAL SYSTEMS

FINANCIAL SYSTEMS

OPERATIONAL SYSTEMS

PROJECT TEAM'S OBJECTIVES

- To ensure all users requirements are considered throughout the development of the orders management system proposal.

- To expose and resolve major business issues early in the project.

- To keep all users involved and informed on the progress of the project

- To provide a local point for inquiries concerning the poroject.

APPENDIX B

POSITIONING STATEMENTS

- DEVELOPMENT STANDARDS

- DEVELOPMENT METHODOLOGY

- GENERAL STATEMENT

DEVELOPMENT STANDARDS

The new systems will be developed according to the new system development standards constructed by Massey-Ferguson MIS.

The principle sections of the system development standards provide guidelines on the activities to be considered and the deliverable required for the effective conduct of the project.

DEVELOPMENT METHODOLOGY

The CACI complete structured method of data, function analysis and system design will be adopted through the analysis and design phases of the project life cycle. The objectives are as follows:

- To provide a complete set of techniques for systems development.

- To ensure that the system developed address real requirements and do not just address the symptoms of a problem.

- To enable the results of analysis to be tested and checked before starting design.

- To enforce user participation with all development stages, reducing the need for system changes.

- To produce systems and documentation that are readily modified and maintained when system requirements occasionally change.

- To provide a detailed basis for project estimating and design.

GENERAL POSITIONING STATEMENTS

From the company standpoint the intent of the new systems are to provide a new improved universally acceptable vehical for managing orders not to simply duplicate functions as they are currently performed. Flexibility will be provided where necessary but must be tempered to reduce costs and develop time.

APPENDIX C

PROJECT MANAGEMENT

- PROJECT MILSTONES BY PHASE
- PROJECT RESOURCE REQUIREMENTS
- CONSTRUCTION PHASE STATUS REPORT
- IMPLEMENTATION PLAN

DEVELOPMENT ACTIVITIES

1. COMMON EUROPEAN OPERATIONS SYSTEMS ACTIVITIES

Task	Responsibility	Planned Completion Date	Actual Completion Date
a. Business Analysis Phase			
* Req Defn Phase	UK. AS NA DL Europe LG	End Aug. End Aug. End Aug.	End Sept. End Sept. End Sept.
* Walkthrough Req. Defn with each company	UK NA Europe	End Dec.	
* Consolidate req. Def to form Common Strategic Approach	DL	End Dec.	
* Walkthrough Consolidated Req. Defn.	UK NA Europe	End Dec.	
b. The Detail Analysis Phase			
* Data and Function Analysis	DL/GM	End Jan.	
* Entity/Data Model * Function Model * Function/Data * Interaction * Entity Defn/Volumes * Entity Life Cycle * Identify Attributes * Identify Key Issues * Business Tech * Define Area of Change * Identify Auditable Control			

Task	Responsibility	Planned Completion Date	Actual Completion Date
c. Design Phase			
* Business System Design Flow & Interfaces	DL/SM	Mid Feb.	
* Define Distributed/ Mainframe function			
* Define System Interfaces - Detail			
* Define Screen Layout			
* Define Report Layout			
* Prepare System Demo.			
* Complete Business System Specification 11 & 111			
* Walkthrough Business System Specification			
* Technical System Design			
* Determine Logical DB Structures			
* Determine Physical DB Structures			
* Describe Job Requirements			
* Describe Program Requirements			
* Define Design Constraints			
* Identify Operation (DCO) Requirements			

Task	Responsibility	Planned Completion Date	Actual Completion Date
* Prepare Technical System Requirement Documents			
* Walkthrough Technical System Design with:			
. Technical Support Team			
. Operational Team			
d. <u>Construction Phase</u>	DL/SM	Jan.	Ongoing
* Prepare Processing Rules		Jan.	Ongoing
* Prepare System Specification		Jan.	Ongoing
* Start Programming		Mid Feb.	

2. COMMON EUROPEAN INVOICING AND ORDER TRACKING SYSTEMS

Task		Responsibility	Planned Completion Date	Actual Completion Date
a. Business Analysis Phase				
Req. Defn.				
Invoicing	UK	DK	End Sept.	Complete
	NA	GS	End Sept.	Complete
	Europe	GW	End Sept.	Complete
S & M	UK	DK	End Oct.	Complete
	NA	GS	End Oct.	Complete
	Europe	GW	End Oct.	Complete
Cost Allocation				
	UK	DK	End Jan.	
	NA	JH	End Jan.	
	Europe	GW	End Jan.	
Consolidate Req. Invoicing		LJ	End Jan.	
Walkthrough Req. Invoicing all companies S & M walkthrough		LJ	End Jan.	
Cost Alloc. Walkthrough		LJ	End Feb.	
b. Detail Analysis Phase		LJ	End Feb.	

Data & Function Analysis

* Entity data model

  . Invoicing and tariff management

  . S & M

  . Review for cost allocation

* Function Model

  . Invoicing
  . S & M
  . Cost allocation

Task	Responsibility	Planned Completion Date	Actual Completion Date
* Function/Data interaction			
* Entity Defn/volumes			
* Entity Life Cycle			
* Identify Attributes			
* Identify Key Issues			
. Business . Technical			
* Area of change			
c. <u>Design Phase</u>	AU/LJ	End March	
* Define System Interface - Detail			
* Define Screen layout			
* Define Report layout			
* Prepare System Demo.			
* Complete Business System specification			
* Walkthrough business system specification			
* Technical system design			
* Determine logical DB structures			
* Determine physical DB structures			
* Describe Job Requirements			
* Describe Program Requirements			

Task	Responsibility	Planned Completion Date	Actual Completion Date
* Define Design Constraints			
* Identify Operational Requirements			

d. Construction Phase

* Prepare processing rules	AU/LJ	March ongoing	
* Prepare system specification	AU/LJ	March ongoing	
* Start program	AU/LJ	Mid March	

ORDERS MANAGEMENT JUNE	JUN	JUL	AUG	SEP	OCT	NOV	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	JAN	MONTHS TOTALS	WEEKS TOTALS	DAYS TOTALS
ORDER ENTRY/VALN																							
ANALYST UK							1	2	2	1	1	2	1								10	42	210
PROGRAMMIMG								1	2	3	3	1	1								11	47	235
																					0	0	0
MODIFY ORDER																							
ANALYST UK							1	1	0.5		0.5										3	13.5	67.5
PROGRAMMIMG									1	2	0.5										3.5	14.5	72.5
																					0	0	0
CORE ALLOCATION																							
ANALYST UK							1	2	2	2	1.5	2	2								12.5	52.5	262.5
PROGRAMMIMG								1	2	3	3	3	1.5								13.5	57	285
																					0	0	0
WAREHOUSE BANK																							
ANALYST UK							3.5	3.5	3.5	3	2	1.5	1.5								18.5	79.5	397.5
PROGRAMMIMG							5.5	4.5	2	1.5	1.5	1.5	1								17.5	77	385
																					0	0	0
MAINT. REF. DATA																							
ANALYST UK									0.5	0.5	1	0.5									2.5	11	55
PROGRAMMIMG								1	1	1	1	1									5	21	105
																					0	0	0
ENQUIRIES																							
ANALYST UK								1	0.5	1	1	0.5									4	17	85
PROGRAMMIMG									0.5	0.5	1	1									3	13	65
																					0	0	0
ONE OFF LOADS																							
ANALYST UK									0.5	1	1										2.5	11	55
PROGRAMMIMG								1	1	0.5	0.5										3	12.5	62.5
																					0	0	0
REPORTING																							
ANALYST UK										1	1.5	1	1								4.5	19.5	97.5
PROGRAMMIMG											1.5	2	1								4.5	19.5	97.5
																					0	0	0
PARTSCO PRICING																							
ANALYST UK																					0	0	0
PROGRAMMIMG																					0	0	0
																					0	0	0
EPI CHANGES																							
ANALYST UK																					0	0	0
PROGRAMMIMG																					0	0	0
																					0	0	0
GRAND TOTALS																							
ANALYST UK	0	0	0	0	0	0	6.5	9.5	9.5	9.5	9.5	7.5	5.5	0	0	0	0	0	0	0			
PROGRAMMIMG	0	0	0	0	0	0	5.5	8.5	9.5	11.	12	9.5	4.5	0	0	0	0	0	0	0			

VOR URMSTON/ATHIS																									DAYS
	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	TOTALS
R Hickman																5	10	10	15	15		15	15	15		100
P Davies												11		11	21	21	13	13	21			10	10	10		141
A Uden															13	13	9	9	13	13		5	5	5		85
J Court												12	3	12	17	17	10	10	17	2						100
K Tomlin															17	22	17	17	22	22						117
C Wilkinson												11	5	16	22	22	17	17	22	6						138
C Jones												3		3	9	9	6	6	9	6		9	9	9		78
K Holmes															11	11	17	17	22	12						90
S Collins														11	22	22	17	17	22	11		11	11	11		155
TBA														22	22	22	17	17	22	22						144
VOR TOTAL			0	0	0	0	0	0	0	0	0	37	8	75	154	164	133	133	185	109	0	50	50	50	0	1148

SIMULATION	1989		1990												1991										DAYS
	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	TOTALS
A Uden																						5	5	5		15
K Tomlin																						22	22	22		66
K Holmes																						22	22	22		66
TBA																						22	22	22		66
SIMULATION TOTAL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	71	71	71	0	213

NORTH AMERICA	1989		1990												1991										DAYS
	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	TOTALS
P Davies																						11	11	11		33
A Uden																						3	3	3		9
J Court																						17	17	17		51
C Wilkinson																						22	22	22		66
S Collins																						11	11	11		33
NORTH AMERICA TOTAL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	64	64	64	0	192

ORDERS MANAGEMENT NOVEMBER	JUN	JUL	AUG	SEP	OCT	NOV	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	JAN	MONTHS TOTALS	WEEKS TOTALS	DAYS TOTALS
ONLINE ENTRY	ALLC																						
ANALYST UK												2	1	2	1	1	1	1			9	39	195
PROGRAMMIMG													2	1	2	1	1				7	30	150
																					0	0	0
STOCK ALLOCATION																							
ANALYST UK													1	1.5	0.5						3	13.5	67.5
PROGRAMMIMG													0.5	2	1						3.5	16	80
																					0	0	0
MODIFY ORDER																							
ANALYST UK													0.5		0.5						1	4	20
PROGRAMMIMG														1							1	5	25
																					0	0	0
ALBUQUERQUE BANK																							
ANALYST UK														2	2	2	2	2	2	2	14	62	310
PROGRAMMIMG														1	2	3	2	2	2	2	14	61	305
																					0	0	0
AINT. REF. DATA																							
ANALYST UK													1	1	1	0.5	0.5				4	17.5	87.5
PROGRAMMIMG													0.5	1	1	1	0.5				4	17.5	87.5
																					0	0	0
ENQUIRIES																							
ANALYST UK													0.5	1	1	1	0.5				4	17.5	87.5
PROGRAMMIMG														1	1	1	0.5				3.5	15.5	77.5
																					0	0	0
DEPLOYMENT																							
ANALYST UK														1	1	0.5	0.5	0.5			3.5	15.5	77.5
PROGRAMMIMG															1	1	1	0.5			3.5	15	75
																					0	0	0
REPORTING																							
ANALYST UK														1	1	0.5	0.5	0.5			3.5	15.5	77.5
PROGRAMMIMG														0.5	1	1	0.5	0.5			3.5	15	75
																					0	0	0
ORDMAN JUNE																							
ANALYST UK							6.5	9.5	9.5	9.5	9.5	7.5	5.5								57.5	246	1230
PROGRAMMIMG							5.5	8.5	9.5	11.	12	9.5	4.5								61	261.5	1307.5
																					0	0	0
SIMULATION																							
ANALYST UK															1.5	3	2	2			8.5	36	180
PROGRAMMIMG															0.5	1.5	4	2			8	36	180
																					0	0	0
PROFORMA																							
ANALYST UK																	0.5	0.5	1.5	1.5	4	18	90
PROGRAMMIMG																		2	2	1	5	22	110
																					0	0	0
WAREHOUSE BANK	NA																						
ANALYST UK																1	1	2	3	3	10	44	220
PROGRAMMIMG																		1.5	3.5	4.5	9.5	41.5	207.5
																					0	0	0
EPI DOMESTIC	NA																						
ANALYST UK																	1	1	3	3	8	36	180
PROGRAMMIMG																		1	2	2	5	22	110
																					0	0	0
GRAND TOTALS																							
ANALYST UK	0	0	0	0	0	0	6.5	9.5	9.5	9.5	9.5	9.5	9.5	9.5	9.5	9.5	9.5	9.5	9.5	9.5			
PROGRAMMIMG	0	0	0	0	0	0	5.5	8.5	9.5	11.	12	9.5	7.5	7.5	9.5	9.5	9.5	9.5	9.5	9.5			

OVERALL RESOURCES																								DAYS
	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC	TOTALS
R Hickman	0	0	11	11	8	8	11	11	0	11	15	15	9.5	15	15	15	10	10	15	15	0	15	15	15	0	220.5
P Davies	0	0	21	21	13	13	21	21	0	21	21	22	11	22	21	21	13	13	21	0	0	21	21	21	0	304
A Uden	0	0	11	11	8	8	11	11	0	11	11	11	6	11	13	13	9	9	13	13	0	13	13	13	0	189
J Court	0	0	17	17	10	10	17	17	0	17	17	17	8	17	17	17	10	10	17	2	0	17	17	17	0	244
K Tomlin	0	0	0	13	9	9	13	13	0	13	22	22	11	22	22	22	17	17	22	22	0	22	22	22	0	313
C Wilkinson	0	0	22	22	17	17	22	22	0	22	22	22	11	22	22	22	17	17	22	6	0	22	22	22	0	332
C Jones	0	0	9	9	6	6	9	9	0	9	9	13	3	9	9	9	6	6	9	6	0	9	9	9	0	139
K Holmes	0	0	22	22	17	17	22	22	0	22	22	22	11	22	22	22	17	17	22	12	0	22	22	22	0	338
S Collins	0	0	22	22	17	17	22	22	0	22	22	22	11	22	22	22	17	17	22	11	0	22	22	22	0	337
TBA	0	0	0	0	8	8	11	18	0	22	22	22	11	22	22	22	17	17	22	22	0	22	22	22	0	324
OVERALL TOTAL	0	0	135	148	113	113	159	166	0	170	183	188	92.5	184	185	185	133	133	185	109	0	185	185	185	0	2740.5

INVENTORY MANAGEMENT SUPPORT FOR OM - EUROPEAN IMPLEMENTATION.

CONSTRUCTION PHASE. 17/10/89

PROJECTS		PLANNED TOTAL		CUMM ACTUAL TO END SEPT				
				PROGRAMS				EFFORT
		PGS	M/DAYS	SPEC	CODE	PGED TEST	SYS TEST	M/DAYS
			A					A
			P					P
SEPPL-1	A	34	206	33	29	28	12	140.5
	P		305					194.5
REC/WHOUSING INTER-FACES	A	32	100	31	9	0	0	30.5
	P		140					9.5
CPMS/PPS INTER-FACES	A	50	300	3	0	0	0	8.5
	P		300					
INV. DATA BASE MAINTENANCE	A	19	180	3	3	0	0	6.0
	P		240					0.5
LOCAL FILE INTER-FACES	A	16	60	1	0	0	0	4.5
	P		120					
DELIV. PROMISES ON-ORDER	A	~ 20	250					
	P		250					
MATL MANGMT ASI INTER-FACES	A	5	30					
	P		60					
FULL SYSTEM // RUN AND IMPL IN EUROPE	A		420					
	P		360					
INVENTORY DB ENQUIRIES	A	18	180					
	P		180					
TOTALS	A	194	1726	71	41	28	12	190.0
	P		1955					204.0

APPENDIX D

TRAINING AND EDUCATION COURSES

- *DATA AND FUNCTION ANALYSIS - AN OVERVIEW FOR END USER*

- *CACI STRUCTURED DATA AND FUNCTION ANALYSIS TECHNIQUES FOR SYSTEMS DEVELOPERS*

DATA AND FUNCTION ANALYSIS - AN OVERVIEW FOR END-USER

Main Topics

1. *Introduction*
2. *Overview of data analysis*
3. *Introduction to Data Analysis*
4. *Workshop*
5. *Entity Modelling*
6. *Workshop*
7. *Attributes and the Entity Model*
8. *Entity Subtype*
9. *Overview of Function Analysis*
10. *Introduction to Function Analysis*
11. *Workshop*

1 Planning a Data Analysis Project

CACI STRUCTURED DATA AND FUNCTION ANALYSIS

Course Code: *ANDATF*

Course intended for: *Business and System Analysts, Systems and Database Designers, Database Administrators, DP and User Managers*

Course Duration: *5 days*

Course Purpose: *To give a full understanding of structured analysis techniques and to enable the participants to carry out data and business analysis using the CACI methods*

Main Topics:

. Scope of data and function analysis

. Entity identification

. Attributes and the entity model

. Normalisation

. Improving the entity model

. Function analysis

. Function dependency

. Entity life cycle analysis

. Function logic analysis

. How to run data analysis project

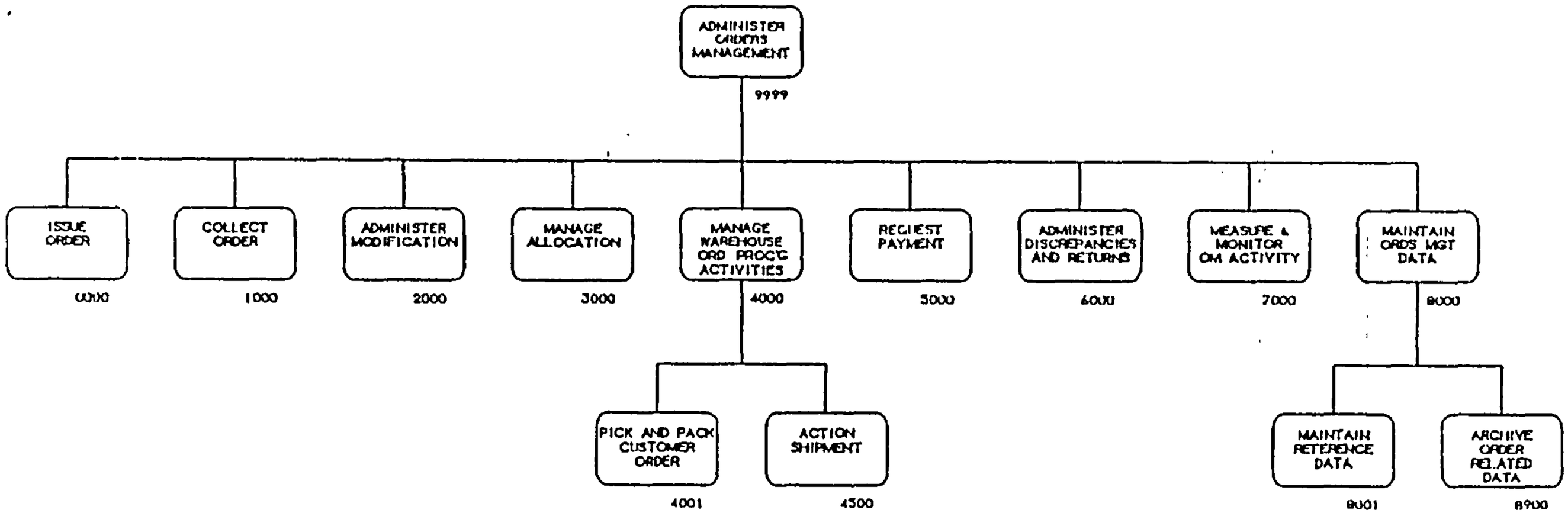
. Case study and Workshops

MISSING
PAGE/
PAGES
HAS NO
CONTENT

APPENDIX E

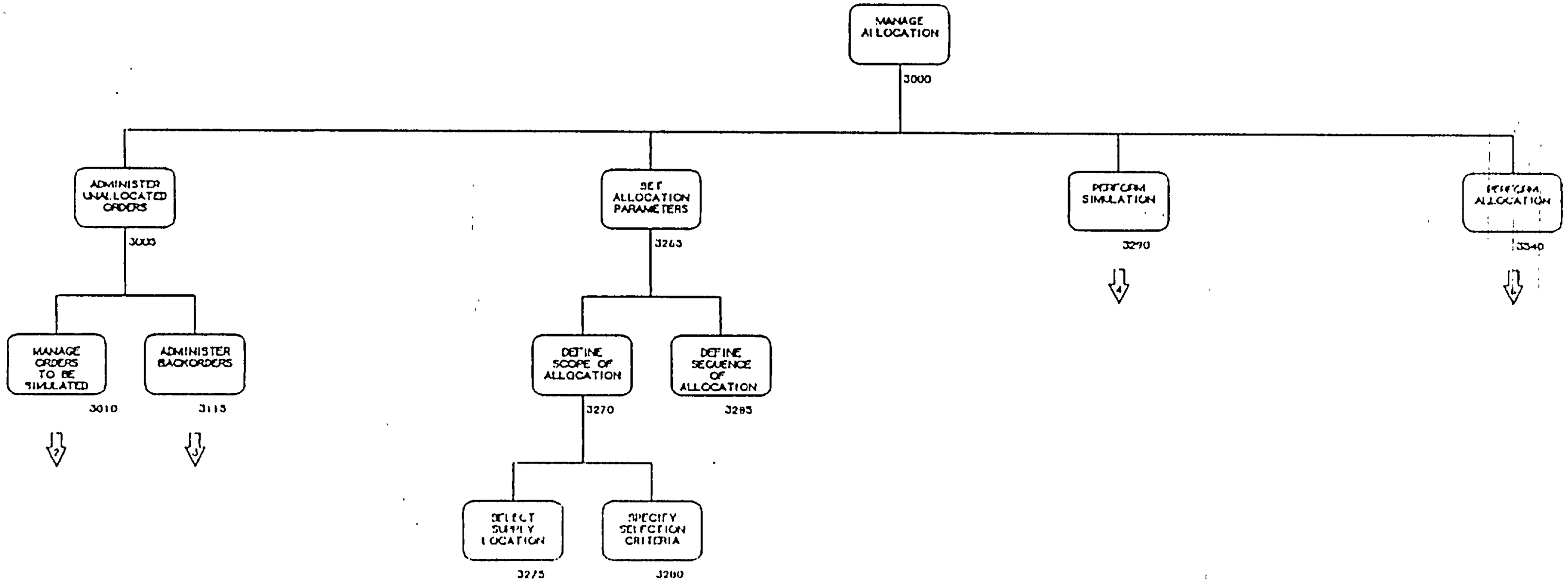
- *STRATEGIC ENTITY/DATA AND FUNCTION MODELS*

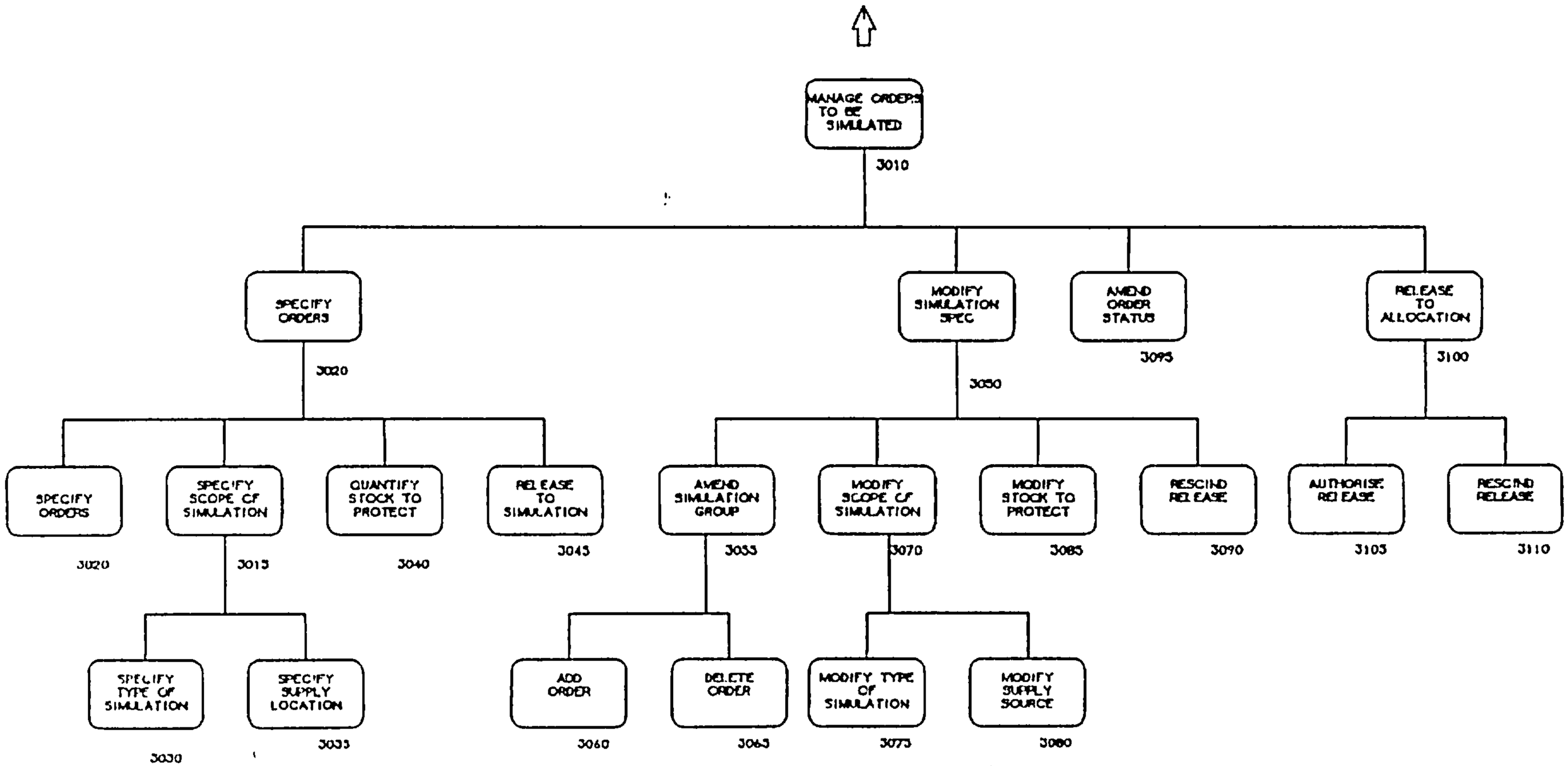
- *FUNCTION DECOMPOSITION*

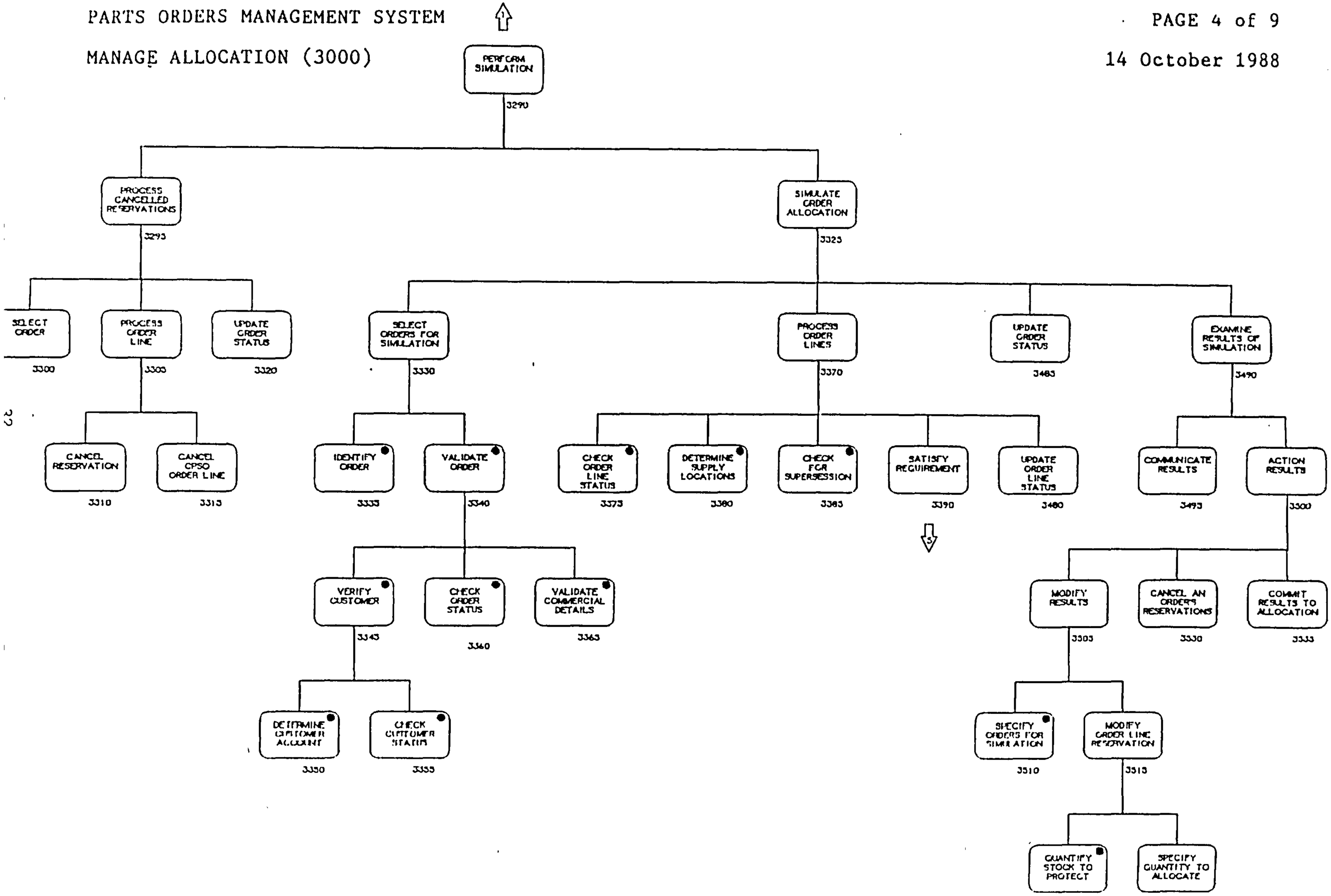

ADMINISTER ORDERS MANAGEMENT
9999
ISSUE ORDER
0000
COLLECT ORDER
1000
ADMINISTER MODIFICATION
2000
MANAGE ALLOCATION
3000
MANAGE WAREHOUSE ORD PROCG ACTIVITIES
4000
REQUEST PAYMENT
5000
ADMINISTER DISCREPANCIES AND RETURNS
6000
MEASURE & MONITOR OM ACTIVITY
7000
MAINTAIN ORDS MGT DATA
8000
PICK AND PACK CUSTOMER ORDER
4001
ACTION SHIPMENT
4500
MAINTAIN REFERENCE DATA
8001
ARCHIVE ORDER RELATED DATA
8900

MANAGE ALLOCATION (3000)

14 October 1988



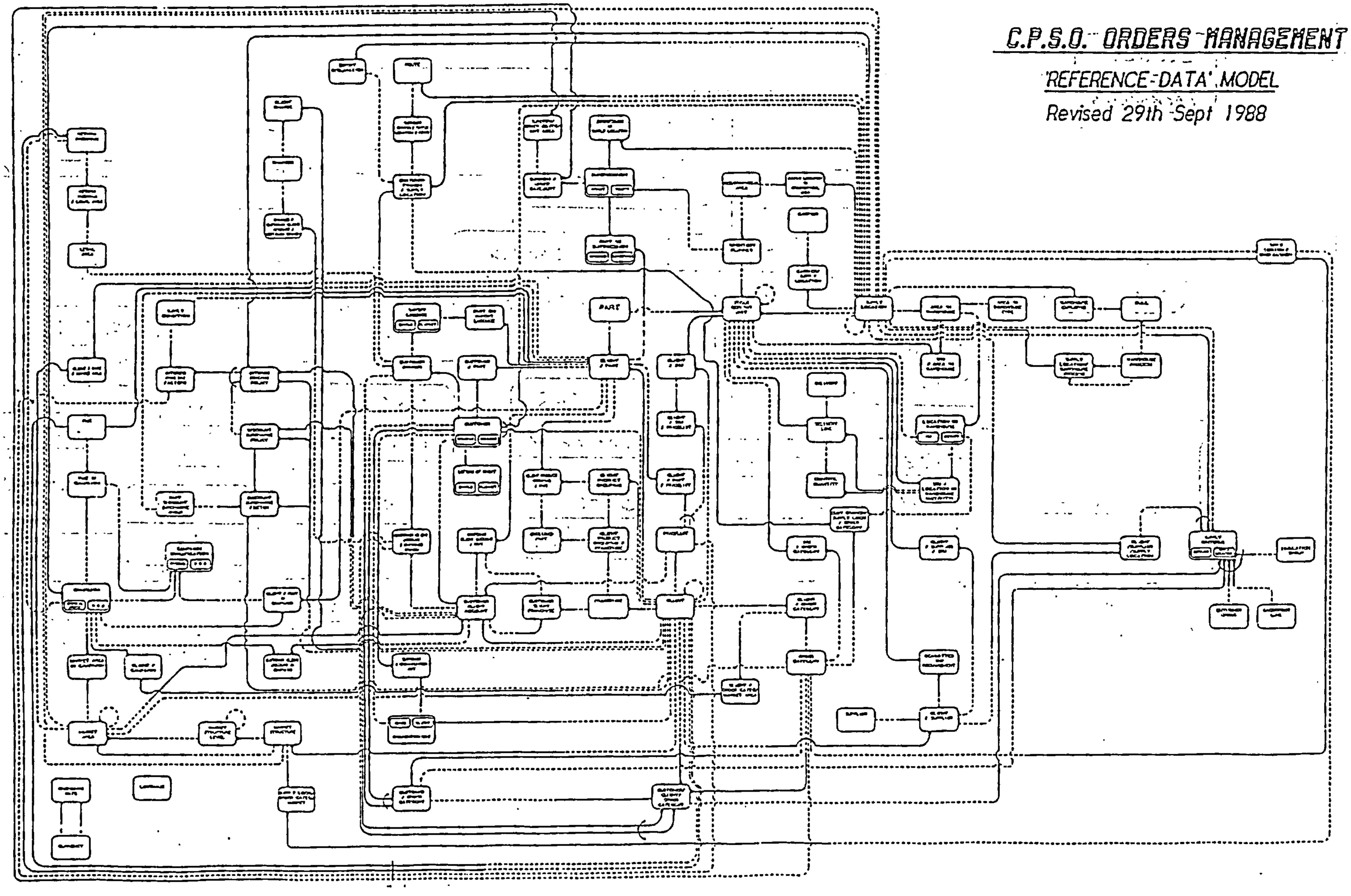

MANAGE ORDERS TO BE SIMULATED
3010
SPECIFY ORDERS
3020
MODIFY SIMULATION SPEC
3050
AMEND ORDER STATUS
3095
RELEASE TO ALLOCATION
3100
SPECIFY ORDERS
3020
SPECIFY SCOPE OF SIMULATION
3015
QUANTIFY STOCK TO PROTECT
3040
RELEASE TO SIMULATION
3045
AMEND SIMULATION GROUP
3055
MODIFY SCOPE OF SIMULATION
3070
MODIFY STOCK TO PROTECT
3085
RESCIND RELEASE
3090
AUTHORISE RELEASE
3105
RESCIND RELEASE
3110
SPECIFY TYPE OF SIMULATION
3030
SPECIFY SUPPLY LOCATION
3033
ADD ORDER
3060
DELETE ORDER
3065
MODIFY TYPE OF SIMULATION
3075
MODIFY SUPPLY SOURCE
3080

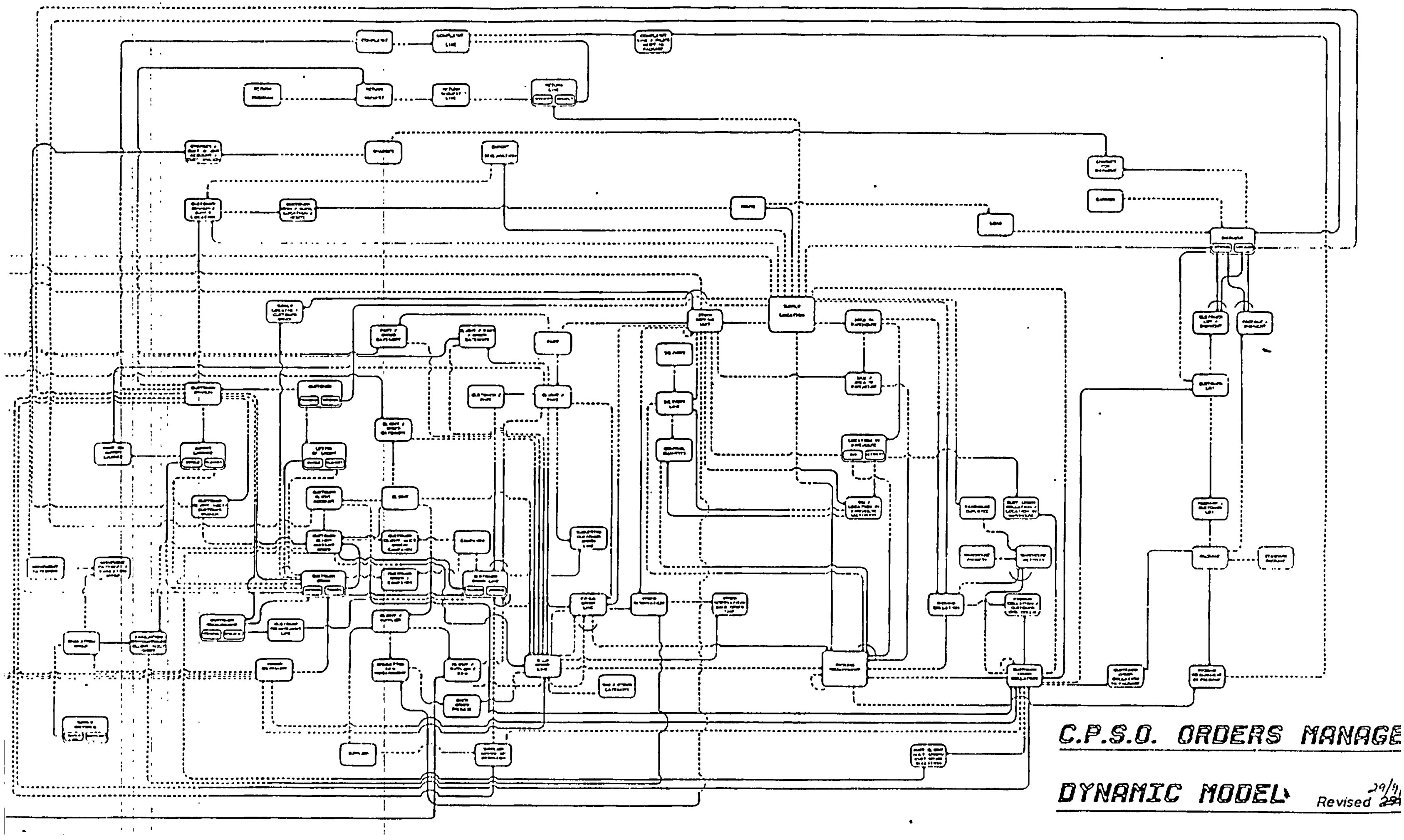

PERFORM SIMULATION
3290
PROCESS CANCELLED RESERVATIONS
3293
SIMULATE ORDER ALLOCATION
3323
SELECT ORDER
3300
PROCESS ORDER LINE
3305
UPDATE ORDER STATUS
3320
SELECT ORDERS FOR SIMULATION
3330
PROCESS ORDER LINES
3370
UPDATE ORDER STATUS
3483
EXAMINE RESULTS OF SIMULATION
3490
CANCEL RESERVATION
3310
CANCEL CPSO ORDER LINE
3315
IDENTIFY ORDER
3335
VALIDATE ORDER
3340
CHECK ORDER LINE STATUS
3375
DETERMINE SUPPLY LOCATIONS
3380
CHECK FOR SUPERSESSION
3385
SATISFY REQUIREMENT
3390
UPDATE ORDER LINE STATUS
3480
COMMUNICATE RESULTS
3493
ACTION RESULTS
3500
VERIFY CUSTOMER
3343
CHECK ORDER STATUS
3360
VALIDATE COMMERCIAL DETAILS
3363
DETERMINE CUSTOMER ACCOUNT
3350
CHECK CUSTOMER STATUS
3355
MODIFY RESULTS
3503
CANCEL AN ORDER'S RESERVATIONS
3530
COMMIT RESULTS TO ALLOCATION
3533
SPECIFY ORDERS FOR SIMULATION
3510
MODIFY ORDER LINE RESERVATION
3513
QUANTIFY STOCK TO PROTECT
SPECIFY QUANTITY TO ALLOCATE

C.P.S.O. ORDERS MANAGEMENT

'REFERENCE-DATA' MODEL

Revised 29th Sept 1988

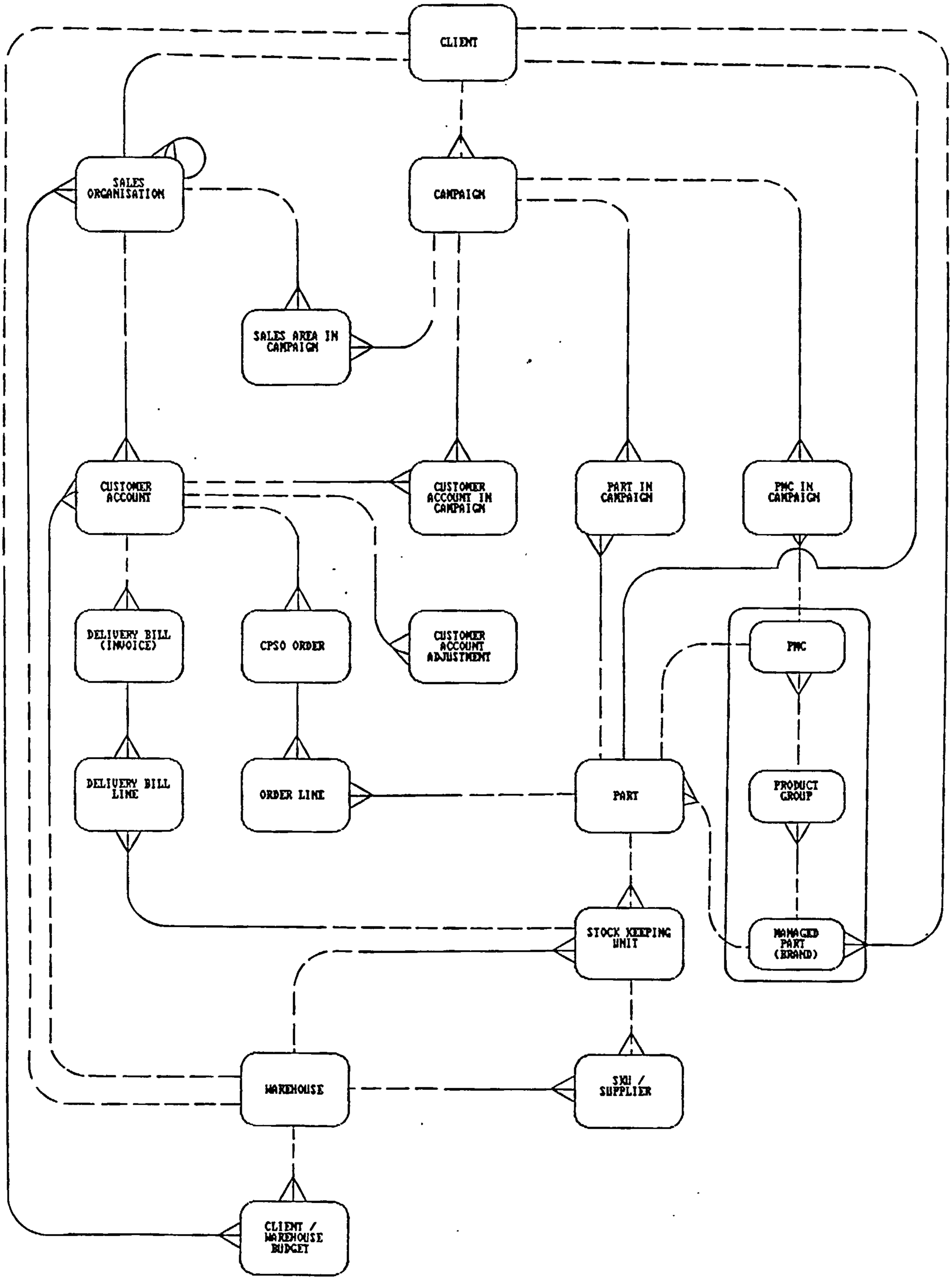


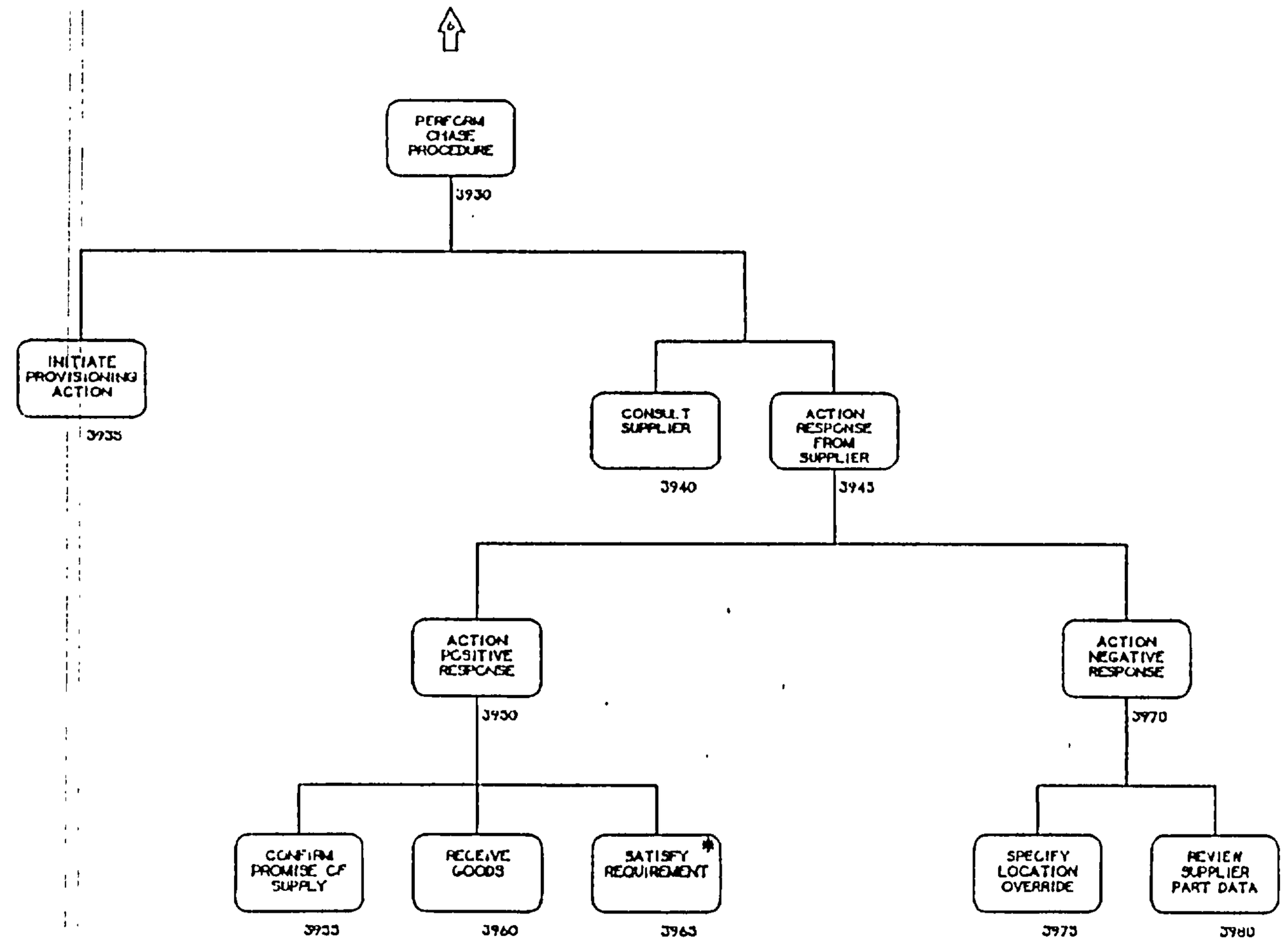


C.P.S.O. ORDERS MANAGE

DYNAMIC MODEL Revised 29/9

PRISM - PHASE ONE.




6
PERFORM CHASE PROCEDURE
3930
INITIATE PROVISIONING ACTION
3935
CONSULT SUPPLIER
3940
ACTION RESPONSE FROM SUPPLIER
3945
ACTION POSITIVE RESPONSE
3950
ACTION NEGATIVE RESPONSE
3970
CONFIRM PROMISE OF SUPPLY
3955
RECEIVE GOODS
3960
SATISFY REQUIREMENT *
3965
SPECIFY LOCATION OVERRIDE
3975
REVIEW SUPPLIER PART DATA
3980

APPENDIX F

DATA AND FUNCTION ANALYSIS - SAMPLE DOCUMENTATION

BUSINESS-ENTITY	"CUSTOMER"

ADMINISTRATIVE-DATA

L O C A T I O N	D A T E	NORMAL VOLUME	MAXIMUM PERIOD	RATE OF GROWTH % OR VOLUME	RATE OF GROWTH PERIOD
URMSTON		1672			
ATHIS		500			
RACINE		2500			
WORLDWIDE (DIST COS & SLS)		30			

DATE	ENTITY DATA HELD ON EXISTING SYSTEMS.
01-10-88	EPI - HOLDS CUSTOMER CLIENT ACCOUNT/CUSTOMER BRANCH, NOT CUSTOMER !

DATE	PHYSICAL DESIGN NOTES
01/03/89	ALLOW FOR SUPPLY LOCATION IN KEY. - RELATIONSHIP AS ATTRIBUTE THE SUPPLY LOCATION BY WHICH THIS CUSTOMER IS RECOGNISED IF INTER CO CUSTOMER

ALIAS ENTITY "CO035"

IDENTIFYING ATTRIBUTE(S).	DATA FORMAT	ALIAS
"CUSTOMER NUMBER"	X(08)	"NR-CUST (0J010 1)"

NORMALISED ATTRIBUTE(S).

"AGENT FLAG"	X(1)	
"C OF O REQUIRED - MIN VALUE"	S9(7)V99	
"CO OF MANUFACTURE REQ FLAG"	X(1)	
"CONFIRMING HOUSE FLAG"	X(1)	
"COUNTRY OF ORIGIN REQ FLAG"	X(1)	
"CUST B/O COLLATION CODE"	X(2)	
"CUST ECONOMICAL SHPMNT VALUE"	S9(7)	
"CUST LOC REQUIRED CODE"	X(1)	
"CUST M OF O - PAYMENT CODE"	XX	
"CUST MAX B/O HOLD TIME"	S9(3)	
"CUST METHOD OF ORDER ENTRY"	X(2)	OCCURS N
"CUST ORDER COLLATION CODE"	X(2)	
"CUST SPECIAL INSTRUCTIONS-INV"	X(25)	OCCURS 3
"CUST SPECIAL INSTRUCTIONS-SHIP"	X(25)	OCCURS 3
"CUST SPECIAL INSTRUCTIONS-WHSE"	X(25)	OCCURS 3
"CUST WHSE BANK HOLD B/OS FLAG"	X(1)	
"CUST WHSE BANK HOLD ORDER FLAG"	X(1)	
"CUST 100% DELIVER REQ FLAG"	X(1)	
"CUST 100% SHIP FLAG"	X(1)	
"CUSTOMER ABNORMAL DEMAND FLAG"	X(1)	

"CUSTOMER ADDRESS"	X(25) OCCURS 5	
"CUSTOMER ALLOCATIC HOLD FLAG"	X(1)	
"CUSTOMER B/O RELE- E DAY"	X(1)	
"CUSTOMER B/OS ALLOWED FLAG"	X(1)	
"CUSTOMER CATEGORY CODE"	X(2)	
"CUSTOMER CURRENCY NUMBER"	X(3)	
"CUSTOMER DATE CREATED"	S9(7)	YYYYDDDC
"CUSTOMER FAX NUMBER"	X(20)	
"CUSTOMER LANGUAGE NUMBER"	X(1)	
"CUSTOMER MAJOR CLIENT NUMBER"	X(2)	
"CUSTOMER NAME"	X(25)	
"CUSTOMER OBSOLETE FLAG"	X(1)	
"CUSTOMER ORDER DAY"	X(1)	
"CUSTOMER PART NUMBER FLAG"	X(1)	
"CUSTOMER POST CODE"	X(10)	
"CUSTOMER SUPERSESSION O/R FLAG"	X(1)	
"CUSTOMER TELEPHONE NUMBER"	X(20)	
"CUSTOMER TELEX NUMBER"	X(15)	
"CUSTOMER TYPE CODE"	X(1)	
"CUSTOMER VALIDATION HOLD TIME"	S9(5)	OHHMMC
"CUSTOMER VAT NUMBER"	X(10)	
"TRANSHIPMENT ALLOWED FLAG"	X(1)	
"TYPE OF SPECIAL INSTRUCTION"	X(1)	

LAST UPDATE BY CDUCW1 AT 17.47.52 ON 15 NOV 1988
DATE PRINTED 27 SEP 1989

ENTITY RELATIONSHIP

DESCRIPTION

FROM A TO B: BACK ORDER LINE MUST BE FOR A CUSTOMER ORDER LINE

FROM B TO A: CUSTOMER ORDER LINE MAY HAVE A BACK ORDER LINE

QUANTITATIVE INFORMATION

LOCATION	DATE	DEGREE	OCCURRENCES OF 'M' AVERAGE	MAXIMUM

DEGREE IMPLICIT IN RELATIONSHIP (A TO B) 1:1

OPTIONALITY

FROM A TO B: OPTIONAL

FROM B TO A: MANDATORY

EXCLUSIVITY IF YES, LIST 'OTHER' RELATIONSHIPS

FROM A TO B: YES BACK ORDER LINE MUST BE FOR A CPSC ORDER LINE

FROM B TO A: NO

BUSINESS-ATTRIBUTE	"CCA ECONOMICAL SHPMT WEIGHT"

DESCRIPTION

-
SPECIFIES THE MINIMUM WEIGHT BELOW WHICH THE CUSTOMER CLIENT ACCOUNT
CONSIDERS IT UNECONOMICAL TO MAKE A SHIPMENT OF BACK ORDER LINES.

ALIAS : QT-MIN-SHIP-WGHT (SK07001)
-

DATA-FORMAT

S9(8)V9(3)

CATALOGUED AS

"AA"
"BASIC"
"QUANT"

ADMINISTRATIVE-DATA

DATA PROTECTION - NOT APPLICABLE
VALID VALUES - GREATER OR EQUAL TO ZERO
SCREEN/REPORT ABBREVIATIONS - AAAAAAAAA
BBBBBB
CCC

DERIVATION

NOT APPLICABLE

LAST UPDATE BY CDUNXJ AT 11.24.58 ON 23 OCT 1989
DATE PRINTED 23 JAN 1990

BUSINESS-ATTRIBUTE	"CCA FREIGHT CHARGE CODE"

DESCRIPTION

-
IDENTIFIES HOW FREIGHT IS PAID ON SHIPMENTS TO A CUSTOMER.
-

DATA-FORMAT

X(1)

CATALOGUED AS

"AA"
"BASIC"
"CLASS"

ADMINISTRATIVE-DATA

DATA PROTECTION - NOT APPLICABLE
VALID VALUES - 0 = FREIGHT PREPAID
- 1 = FREIGHT COLLECT
- 2 = FREE OF CHARGE
SCREEN/REPORT ABBREVIATIONS - AAAAAAAAA
BBBBBB
CCC

DERIVATION

NOT APPLICABLE

LAST UPDATE BY CDUCW1 AT 08.34.25 ON 08 MAY 1989
DATE PRINTED 23 JAN 1990

BUSINESS-ATTRIBUTE	"B/O PROMISE REQUIRED CODE"

DESCRIPTION

-
SPECIFIES THAT B/O PROMISES ARE REQUIRED FOR ANY BACK ORDERS ON
CUSTOMER CLIENT ACCOUNT ORDERS OF THE SPECIFIED ORDER CATEGORY FOR THE
SPECIFIED CLIENT.
-

DATA-FORMAT

X

CATALOGUED AS

"AA"
"BASIC"
"CLASS"

ADMINISTRATIVE-DATA

DATA PROTECTION - NOT APPLICABLE
VALID VALUES - 0 = B/O PROMISE NOT REQUIRED
VALID VALUES - 1 = B/O PROMISE REQUIRED
SCREEN/REPORT ABBREVIATIONS - AAAAAAAAA
BBBBBB
CCC

DERIVATION

NOT APPLICABLE

LAST UPDATE BY CDUMXN AT 12.33.09 ON 09 JAN 1989
DATE PRINTED 23 JAN 1990

BUSINESS-ATTRIBUTE	"B/O PROMISE REFERENCE NR"

DESCRIPTION

-
A NUMBER WHICH UNIQUELY IDENTIFIES A BACK ORDER PROMISE.

SYNONYMS: INVENTORY CONTROL REF NO.
-

DATA-FORMAT

X(4)

CATALOGUED AS

"AA"
"BASIC"
"IDENT"

ADMINISTRATIVE-DATA

DATA PROTECTION - NOT APPLICABLE
VALID VALUES
SCREEN/REPORT ABBREVIATIONS - AAAAAAAAA
BBBBBB
CCC

DERIVATION

NOT APPLICABLE

LAST UPDATE BY CDUCW1 AT 11.15.10 ON 17 APR 1989
DATE PRINTED 23 JAN 1990

BUSINESS-FUNCTION | "BF-OJ-3000"

TITLE

MANAGE ALLOCATION

DESCRIPTION

THE INTRICATE PROCESS OF INTERROGATING AVAILABLE INVENTORY AT ONE OR MORE SUPPLY LOCATIONS, BASED ON PROXIMITY RULES FIXED BY CLIENTS AT CUSTOMER/ORDER CATEGORY LEVEL, TO SATISFY CUSTOMERS' DEMAND.
THIS INTERROGATION WILL EITHER RESULT IN THE ALLOCATION OF INVENTORY TO A RELEASED UNALLOCATED CUSTOMER REQUIREMENT, CREATING BACKORDERS WHERE NECESSARY, OR BE PART OF AN ALLOCATION SIMULATION PROCESS AGAINST IDENTIFIED LARGE ORDERS.
RESULTING ACTIONS WILL BE COMMUNICATED INTERNALLY AND TO THE CUSTOMER WHERE APPROPRIATE.

ADMINISTRATIVE-DATA

FREQUENCY INFORMATION

LOCATION	DATE	NORMAL		MAXIMUM	
		VOLUME	PERIOD	VOLUME	PERIOD

CONTAINS

BUSINESS-FUNCTION "BF-OJ-3005"
BUSINESS-FUNCTION "BF-OJ-3265"
BUSINESS-FUNCTION "BF-OJ-3290"
BUSINESS-FUNCTION "BF-OJ-3540"

LAST UPDATE BY MASTER AT 11.53.15 ON 28 JUL 1988
DATE PRINTED 29 NOV 1988

BUSINESS-FUNCTION	"BF-OJ-3005"

TITLE

ADMINISTER UNALLOCATED ORDERS

DESCRIPTION

THE PROCESS OF IDENTIFYING ORDERS THAT HAVE NOT YET HAD STOCK ALLOCATED TO THEM FOR THE PURPOSE OF:

A) VIEWING THE BACK ORDER SITUATION AND TAKING ACTION TO FACILITATE THEIR ALLOCATION
B) SPECIFYING ORDERS FOR SIMULATION

ADMINISTRATIVE-DATA

FREQUENCY INFORMATION

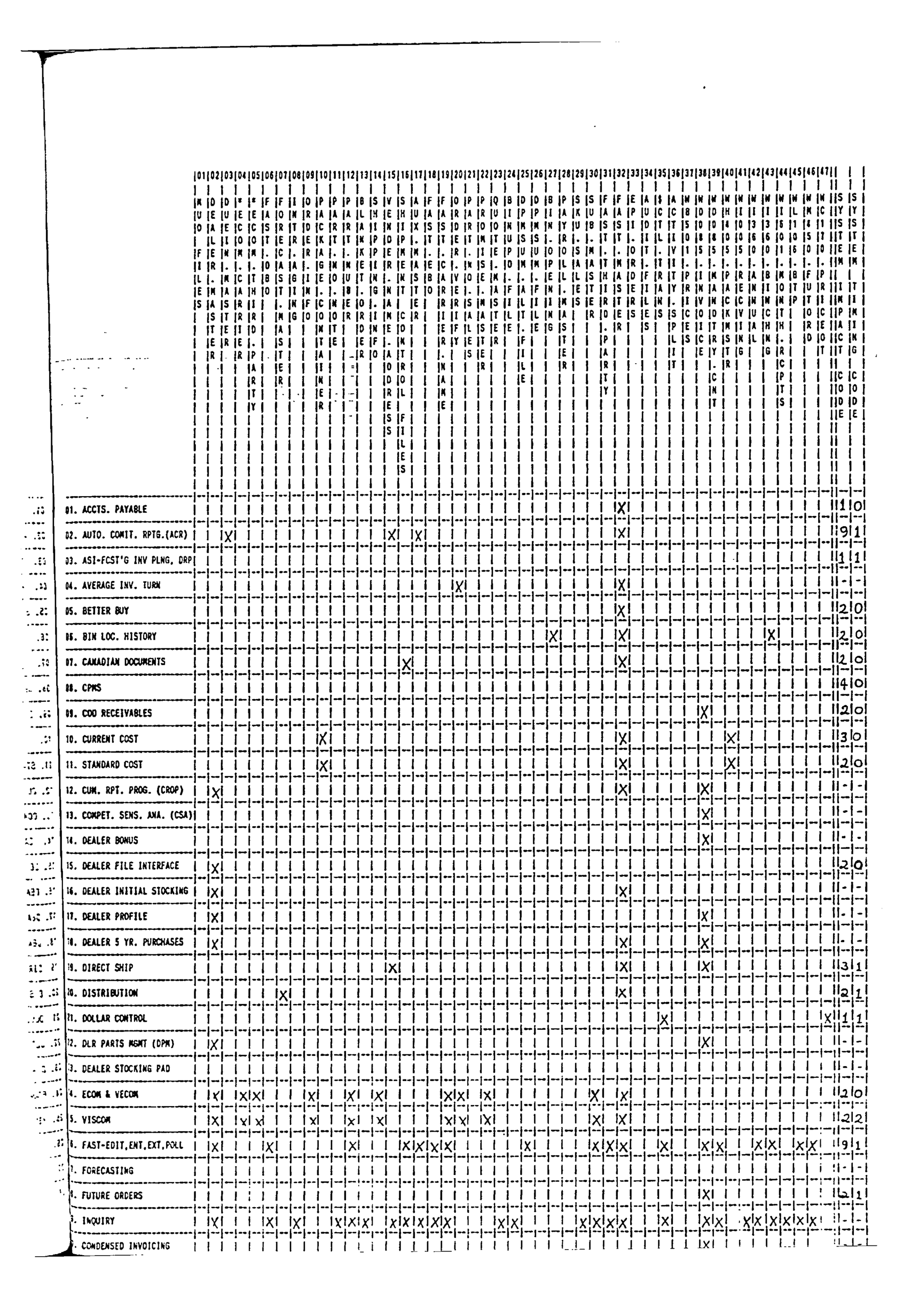
LOCATION	DATE	NORMAL		MAXIMUM	
		VOLUME	PERIOD	VOLUME	PERIOD

CONTAINS

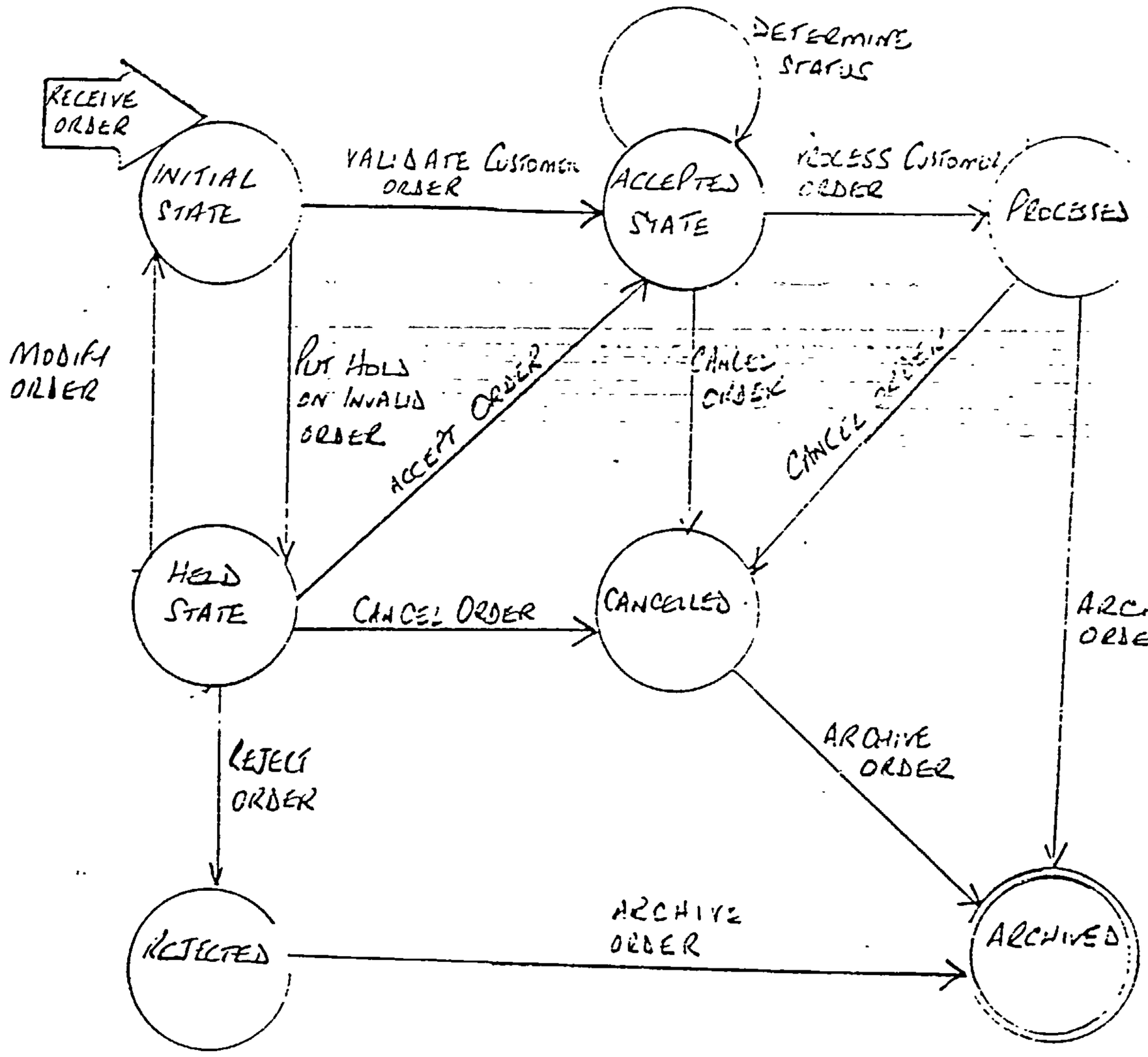
BUSINESS-FUNCTION "BF-OJ-3010"
BUSINESS-FUNCTION "BF-OJ-3115"

LAST UPDATE BY MASTER AT 11.53.16 ON 28 JUL 1988
DATE PRINTED 29 NOV 1988

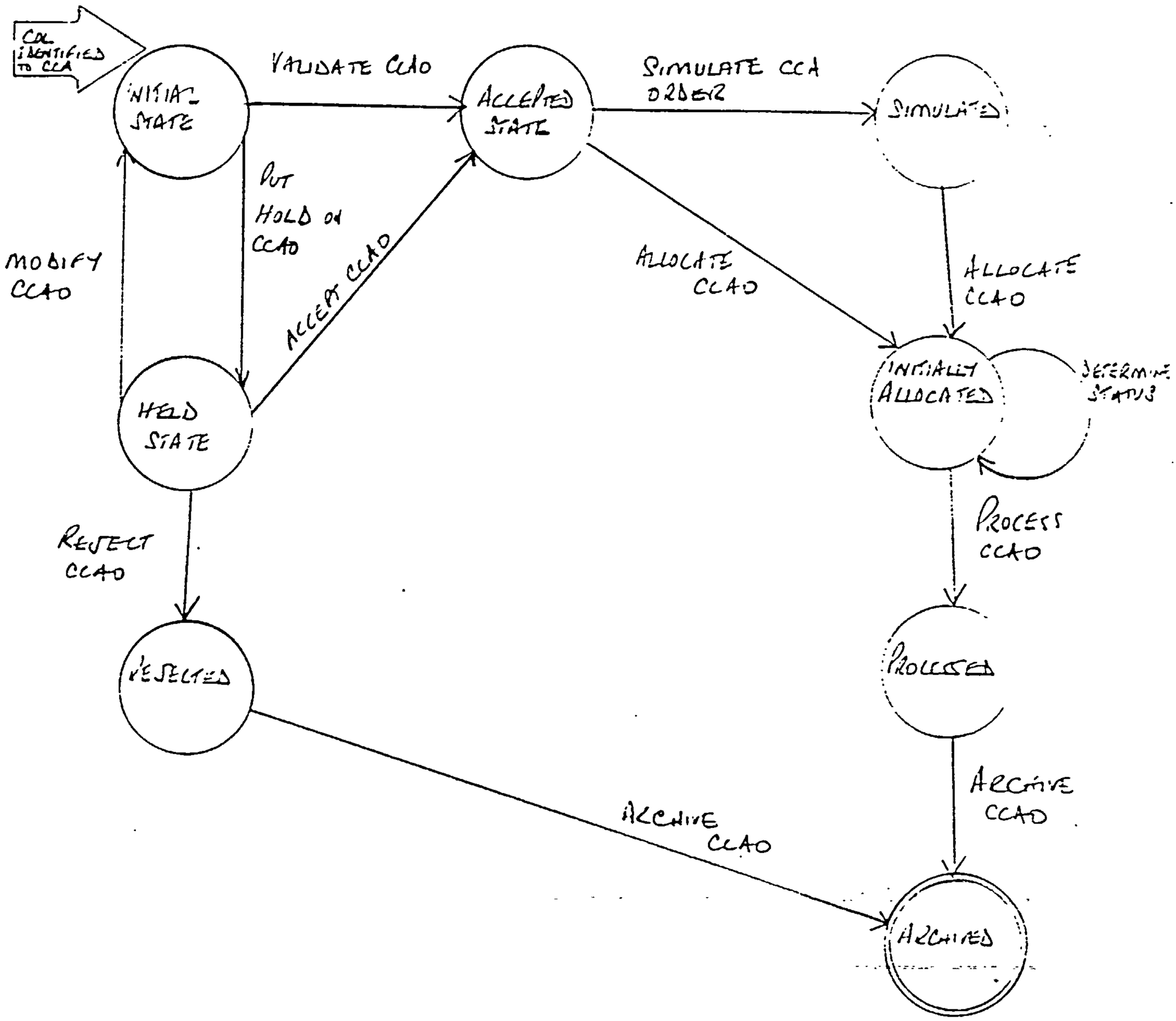
31. EXPORT INVOICING (EPI)																																	X															1	1
32. MA INVOICING		X																														X						X										2	0
33. KIT SYSTEM						X																								X		X																2	1
34. LABOR DISTRIBUTION																																													X			1	0
35. MANDATORY COUNT																																X						X	X									2	1
36. MICRO FICHE-MUO	X																																															-	-
37. MICRO FICHE-PICK PACKS																																																2	0
38. MICRO FICHE-PURCHASING																																																-	-
39. MICRO FICHE-RECEIPTS																																																2	0
40. MICRO FICHE-SUBS																														X																		-	-
41. MACHINE USED ON (MUO)	X																																															1	0
42. OPEN ACCTS REC. (OARS)		X																																				X										1	0
43. OBSOLESCENCE-SCRAP							X																									X																-	-
44. PACKAGERS INVENTORY										X																														X								2	1
45. PACKAGING MASTER										X																														X								1	1
46. PARTS BOOK	X																													X		X																-	-
47. PARTS MGR RECOGNITION																																						X										-	-
48. PAYROLL																																																-	-
49. PERFORMANCE REPORTING																																						X										2	1
50. PHYSICAL INVENTORY																								X			X					X																1	1
51. PICK LIST																																																2	0
52. PRICK BOOK																																X					X											1	0
53. COMPETITIVE PRICING																																X																-	-
54. FORMULA PRICING																																																-	-
55. PROFORMA INVOICING																																X																2	1
56. PARTS STATUS RPTG (PSR)			X			X	X										X	X			X									X		X																2	1
57. PURCHASING-LOCAL			X										X		X		X						X																									1	1
58. PURCHASING-PPS																																																-	-
59. PDC COMMUNICATIONS(5280)																																																2	0
60. RECEIVING			X																													X									X							2	0
61. ANNUAL RETURNS		X																																								X						-	-
62. RETURNS SCREENING		X																																			X											-	-
63. 30 DAY RETURNS		X																																														2	1
64. 30 DAY RETURNS EXCEEDED																																						X										-	-
65. PARTS CONTRI. ANALY(PCA)																																X			X			X						X				1	0
66. SHIPMENT CONTROL								X								X																																2	0
67. SOURCE ORDERING																																																-	-
68. SUBSTITUTION			X																											X		X																2	0
69. SUPPLIER CROSS REF.																																X																-	-
70. VOLUME VALUE ANALYSIS							X																									X																-	-
71. WRITTEN ORDERS																																																-	-
FILE IMPACT CODE	-	1	2	2	2	2	-	2	2	-	2	2	-	2	1	2	1	2	2	2	-	2	-	2	-	-	-	-	-	1	2	1	2	2	2	2	-	1	2	-	2	-	2	1	-	2	2	/	/
FILE TIMING CODE	-	1	1	1	1	1	-	0	1	-	1	0	-	1	0	0	1	1	1	0	-	1	-	0	-		-	-	-	0	1	1	1	1	1	1	-	0	1	-	0	-	0	0	-	1	0	/	/


01. ACCTS. PAYABLE
02. AUTO. COMIT. RPTG.(ACR)
03. ASI-FCST'G INV PLNG, DRP
04. AVERAGE INV. TURN
05. BETTER BUY
06. BIN LOC. HISTORY
07. CANADIAN DOCUMENTS
08. CPMS
09. CDO RECEIVABLES
10. CURRENT COST
11. STANDARD COST
12. CUM. RPT. PROG. (CROP)
13. COMPET. SENS. ANA. (CSA)
14. DEALER BONUS
15. DEALER FILE INTERFACE
16. DEALER INITIAL STOCKING
17. DEALER PROFILE
18. DEALER 5 YR. PURCHASES
19. DIRECT SHIP
20. DISTRIBUTION
21. DOLLAR CONTROL
22. DLR PARTS MGMT (DPM)
23. DEALER STOCKING PAD
24. ECOM & VECOM
25. VISCOM
26. FAST-EDIT,ENT,EXT,POLL
27. FORECASTING
28. FUTURE ORDERS
29. INQUIRY
30. CONDENSED INVOICING

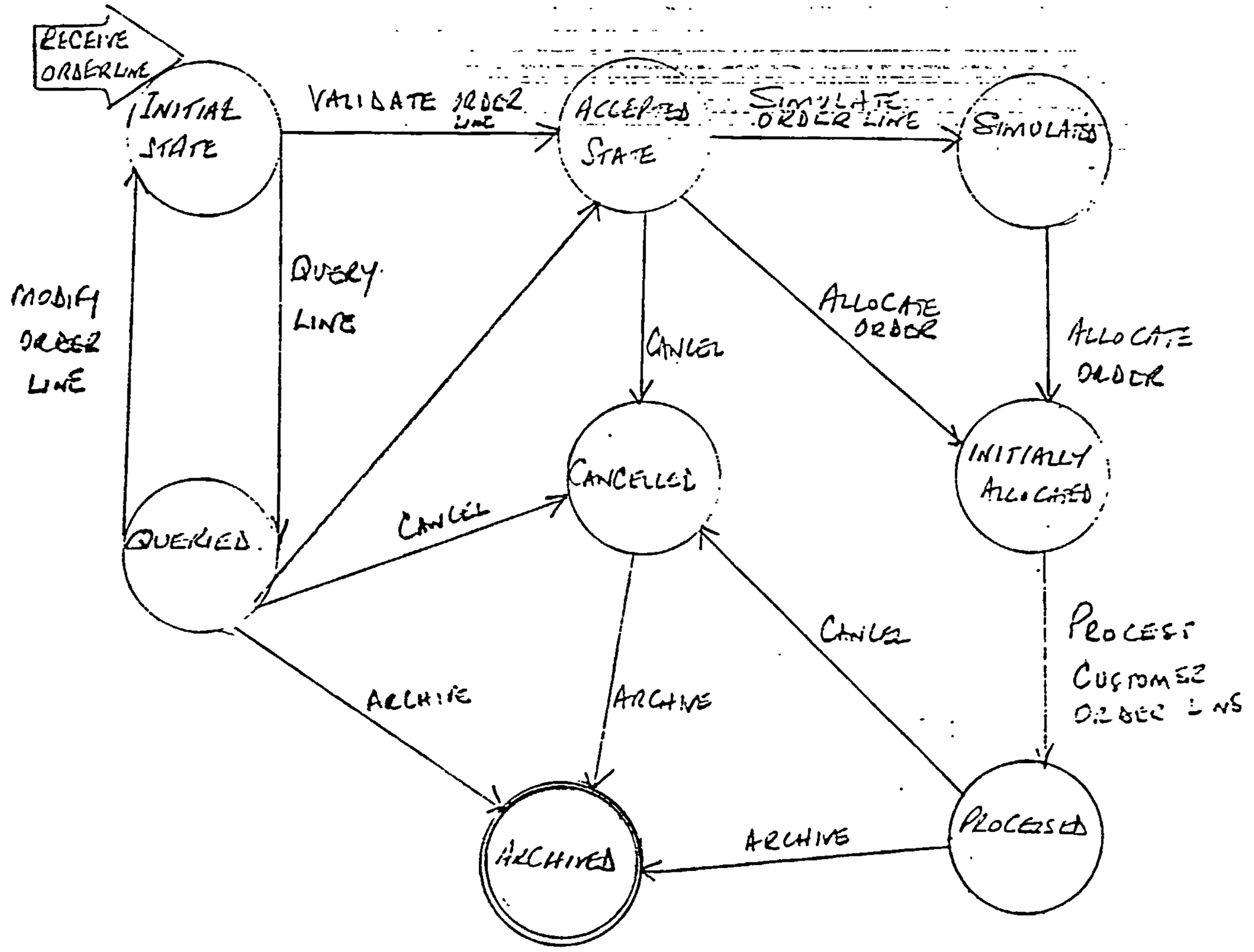
CUSTOMER ORDER



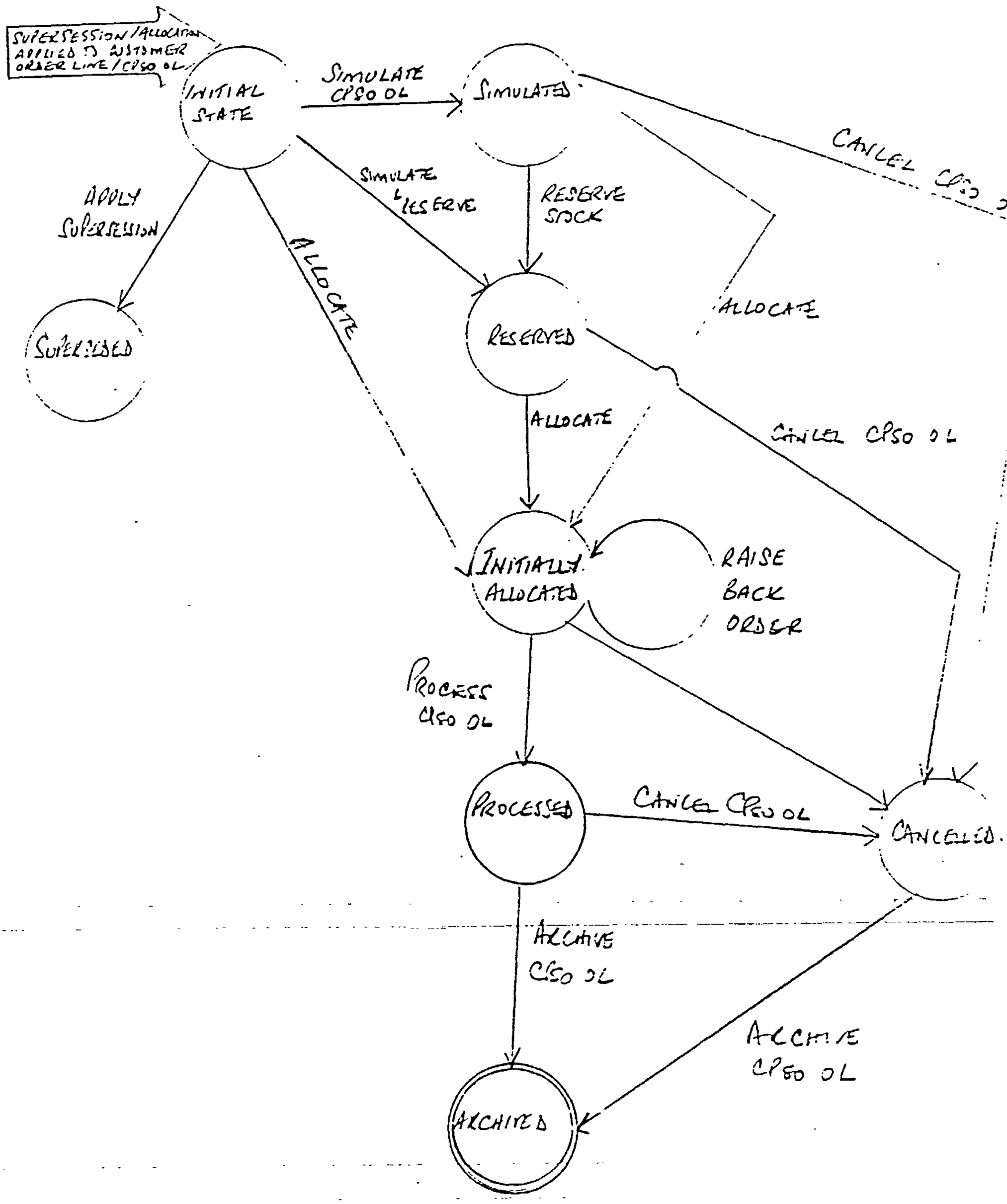
CUSTOMER CLEAR ACCOUNT ORDER



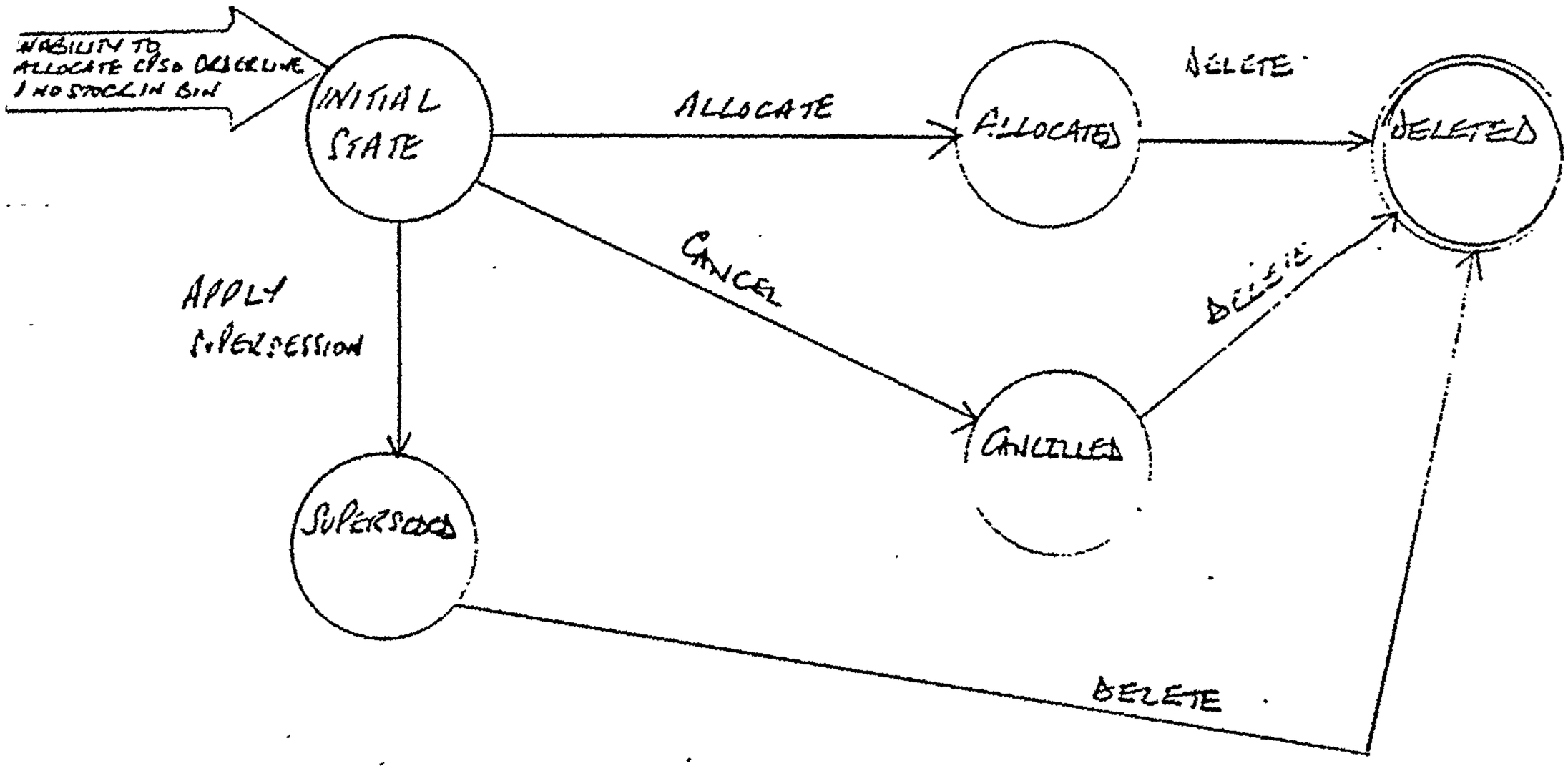
CUSTOMER ORDER LINE



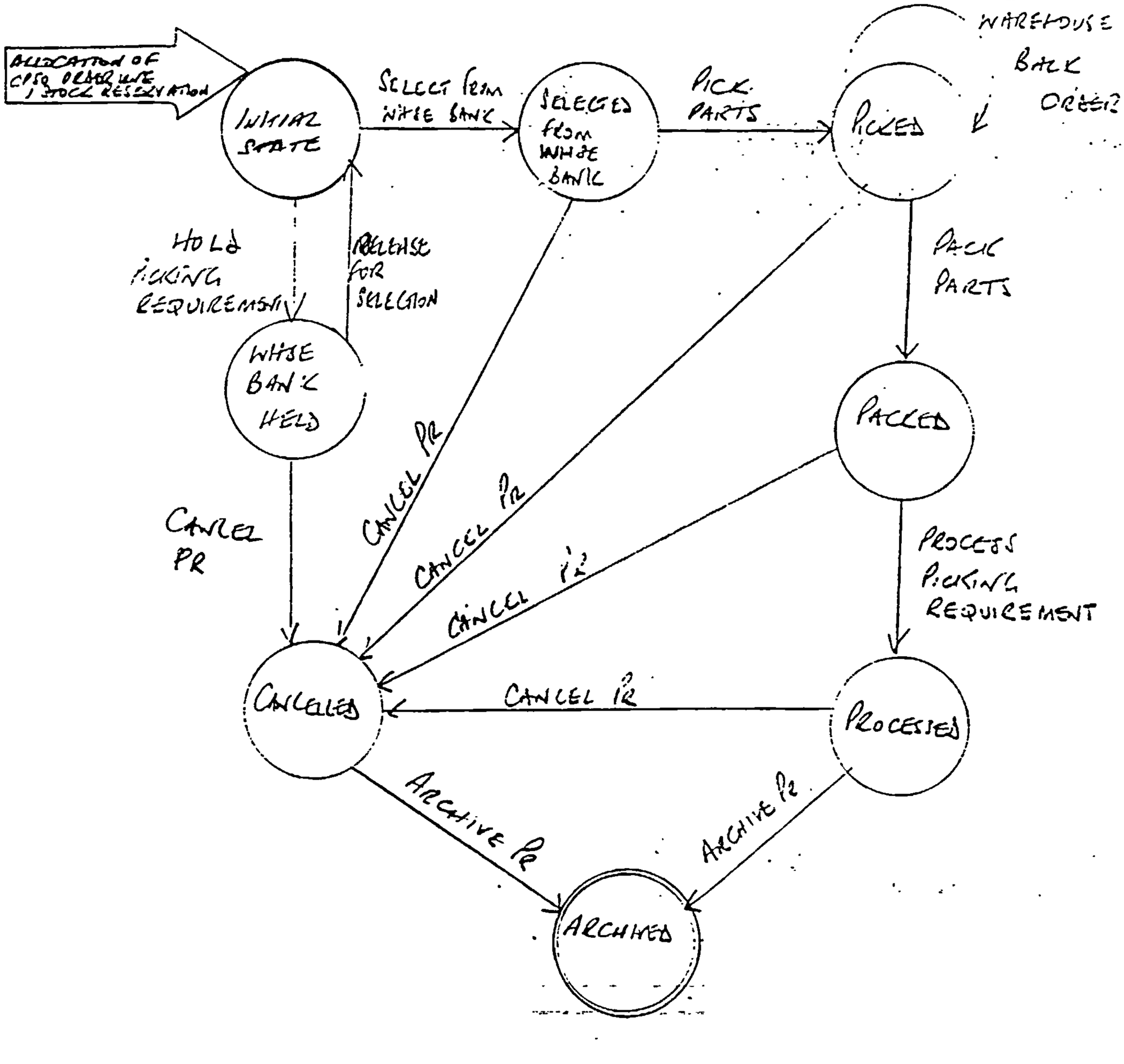
CPSO ORDER LINE



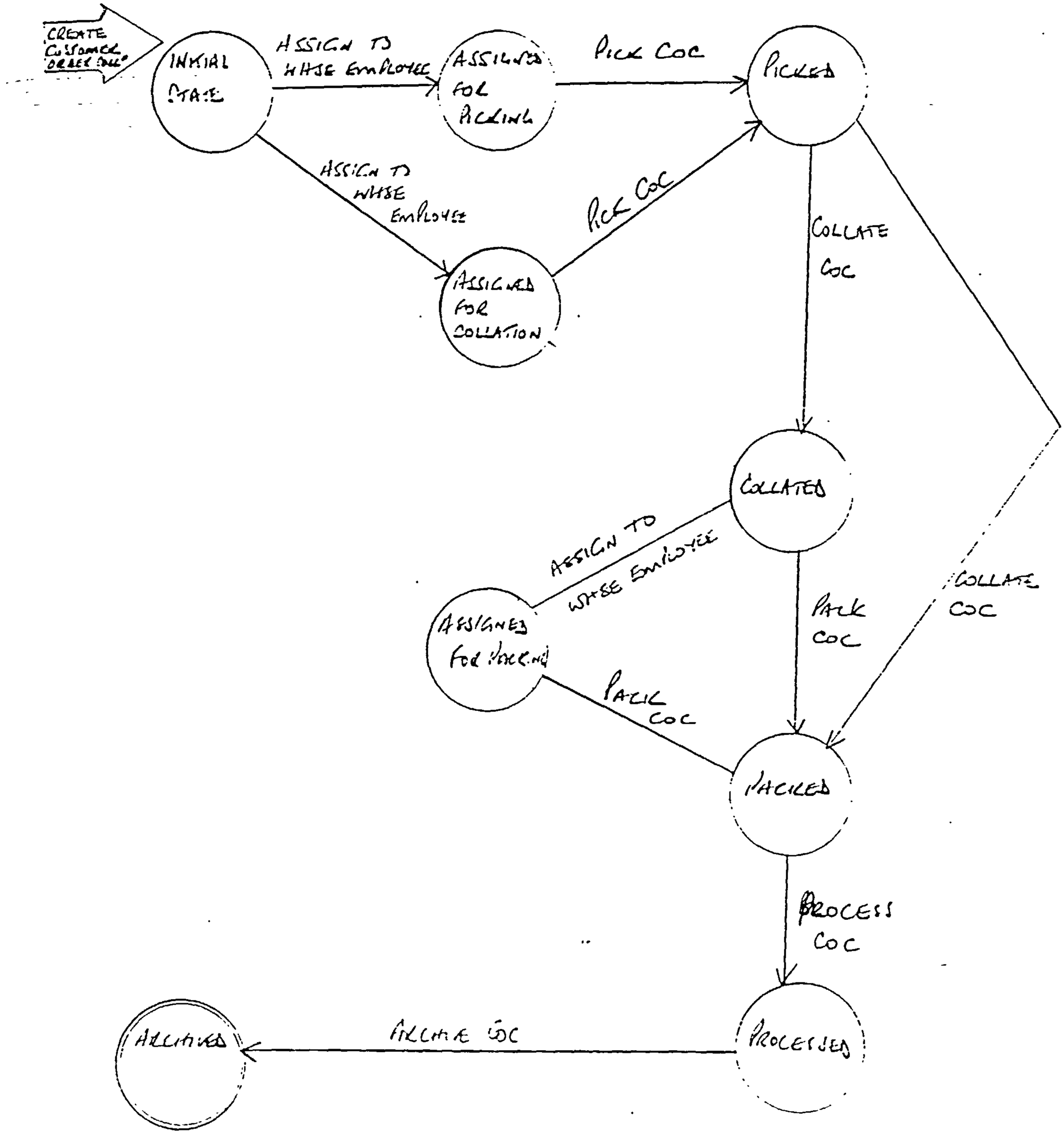
BACK ORDER LINE



PICKING REQUIREMENT



CUSTOMER ORDER COLLATION



ENTITY A NO.	ENTITY A NAME	ENTITY B NO.	ENTITY B NAME		FUNCTION	DAILY FREQUENCY A-B	DAILY FREQUENCY B
		C0001	AREA IN WAREHOUSE TYPE		R	0.0000	8.0000
				TOTAL C0001		0.0000	8.0000
		C0002	BACK ORDER LINE		C	0.0000	1.4400
					R	0.0000	487348.4400
					U	0.0000	577.9400
				TOTAL C0002		0.0000	487927.8200
		C0017	CLIENT/PART		R	0.0000	110178.4800
				TOTAL C0017		0.0000	110178.4800
		C0026	CLIENT/SUPPLIER/S.K.U		R	0.0000	17197.0000
				TOTAL C0026		0.0000	17197.0000
		C0041	CUSTOMER CLIENT ACCOUNT		R	0.0000	136510.0000
				TOTAL C0041		0.0000	136510.0000
		C0045	CUSTOMER CLIENT ACCOUNT ORDER		R	0.0000	90084.5200
					U	0.0000	1.8400
				TOTAL C0045		0.0000	90086.3600

ENTITY A NO.	ENTITY A NAME	ENTITY B NO.	ENTITY B NAME	FUNCTION	DAILY FREQUENCY A-B	DAILY FREQUENCY B
		C0051	CUSTOMER/CLIENT/ORDER CATEGORY	R	0.0000	111512.5340
			TOTAL C0051		0.0000	111512.5340
		C0060	CUSTOMER/ORDER CATEGORY	R	0.0000	39616.2462
			TOTAL C0060		0.0000	39616.2462
		C0064	CUSTOMER ORDER LINE	R	0.0000	398.8800
			TOTAL C0064		0.0000	398.8800
		C0071	DELIVERY LINE	R	0.0000	625.0000
			TOTAL C0071		0.0000	625.0000
		C0096	ORDER CATEGORY	R	0.0000	0.2740
			TOTAL C0096		0.0000	0.2740
		C0131	STOCK KEEPING UNIT	R	0.0000	428901.6400
				U	0.0000	2.8800
			TOTAL C0131		0.0000	428904.5200
		C0132	S.K.U/ORDER CATEGORY	C	0.0000	1.4400
				R	0.0000	220104.0000
			TOTAL C0132		0.0000	220105.4400

ENTITY A NO. NAME

ENTITY B NO. NAME

C0002 BACK ORDER LINE

FUNCTION USED IN	DAILY FREQUENCY	FUNCTION	FREQUENCY A-B	FREQUENCY B	DAILY FREQUENCY A-B	DAILY FREQUENCY B
BF-0J-3105	1.8000	U	0.0000	20.0000	0.0000	36.0000
BF-0J-3315	2.8800	U	0.0000	0.5000	0.0000	1.4400
BF-0J-3260	175.0000	U	0.0000	1.0000	0.0000	175.0000
BF-0J-3110	0.0200	U	0.0000	20.0000	0.0000	0.4000
BF-0J-3140	0.1000	U	0.0000	1.0000	0.0000	0.1000
BF-0J-3230	350.0000	U	0.0000	1.0000	0.0000	350.0000
BF-0J-3235	15.0000	U	0.0000	1.0000	0.0000	15.0000
				FUNCTION TOTAL	0.0000	577.9400
				TOTAL	0.0000	487927.8200

ENTITY A
NO. NAME

ENTITY B
NO. NAME

C0017 CLIENT/PART

FUNCTION USED IN	DAILY FREQUENCY	FUNCTION	FREQUENCY A-B	FREQUENCY B	DAILY FREQUENCY A-B	DAILY FREQUENCY B
BF-OJ-3410	89576.0000	R	0.0000	1.2300	0.0000	110178.4800
				FUNCTION TOTAL	0.0000	110178.4800
				TOTAL	0.0000	110178.4800

DATE 24/10/82 PAGE 4 OF 42

TASK SPECIFICATION

TASK NO. 3315 PROJECT ORDERS MANAGEMENT

DAILY ~~MONTHLY~~ FREQUENCY 2.87 DESCRIPTION CANCEL RESERVATION ~~MPT~~ / BMP / ~~BATCH~~

ENTRY ENTITY	ACCESS PATH (relationship/entity or) (entity,relationship,entity)	FUNCTION	FREQUENCY	DAILY FREQUENCY	SEQ/RANDOM PROCESSING	SEQUENCE FIELDS
	BACKORDER LINE → CLIENT/PART	LINK	0.50			
	Retrieve primary SKU to link backorder to SKU/order category					
	CPSO ORDER LINE → STOCK KEEPING UNIT	READ	0.50			
	SKU/ORDER CATEGORY	CREATE	0.50		RANDOM	KEY
	BACKORDER LINE → SKU/ORDER	LINK	0.50			
	CATEGORY					

DATE 24/10/83

TASK SPECIFICATION

TASK NO. 3315

PROJECT ORDERS MANAGEMENT

DAILY ~~WEEKLY~~ ~~MONTHLY~~ FREQUENCY 2 88 DESCRIPTION CANCEL RESERVATION ~~BATCH~~ / DMP / ~~BATCH~~

ENTRY ENTITY	ACCESS PATH (relationship/entity or) (entity,relationship,entity)	FUNCTION	FREQUENCY	DAILY FREQUENCY	SEQ/RANDOM PROCESSING	SEQUENCE FIELDS
	BACKORDER LINE → CLIENT	LINK	0.50			
	CUSTOMER CLIENT ACCOUNT ORDER →	READ	0.50			
	CUSTOMER ORDER					
	CUSTOMER ORDER → ORDER CATEGORY	READ	0.50			
	BACKORDER LINE → CLIENT/ORDER	LINK	0.50			
	CATEGORY					
	CPSO ORDER LINE → CLIENT/PART	READ	0.50			
	BACKORDER LINE → PART/ORDER	LINK	0.50			
	CATEGORY					
	BACKORDER LINE → CLIENT/PART/	LINK	0.50			
	ORDER CATEGORY					

DATE 24/10/88

DDT

PAGE 2 OF 4

TASK SPECIFICATION

TASK NO. 3315

PROJECT ORDERS MANAGEMENT.

DAILY ~~QUANTITY~~ FREQUENCY 2.88 DESCRIPTION CANCEL RESERVATION ~~MPP~~ / DMP / ~~BATCH~~

ENTRY ENTITY	ACCESS PATH (relationship/entity or) (entity,relationship,entity)	FUNCTION	FREQUENCY	DAILY FREQUENCY	SEQ/RANDOM PROCESSING	SEQUENCE FIELDS
	Create or amend backorder					
	CPSO ORDER LINE → BACKORDER	READ	1.00			
	LINE					
	BACKORDER LINE	UPDATE	0.50			
BACKORDER LINE	BACKORDER LINE	CREATE	0.50		RANDOM	KEY
	CPSO ORDER LINE → CUSTOMER ORDER	READ	0.50			
	LINE					
	CUSTOMER ORDER LINE → CUSTOMER	READ	0.50			
	CLIENT ACCOUNT ORDER					
	BACKORDER LINE → CUSTOMER CLIENT	LINK	0.50			
	ACCOUNT ORDER					

DATE 24/10/83

PAGE 1 OF 4

TASK SPECIFICATION

TASK NO. 3365

PROJECT ORDERS MANAGEMENT

DAILY ~~WEEKLY~~ ~~MONTHLY~~ FREQUENCY 2.38 DESCRIPTION CANCEL RESERVATION ~~MPT~~ / BMP / ~~BATCH~~

LIBRARY ENTITY	ACCESS PATH (relationship/entity or) (entity,relationship,entity)	FUNCTION	FREQUENCY	DAILY FREQUENCY	SEQ/RANDOM PROCESSING	SEQUENCE FIELDS
STOCK RESERVATION	STOCK RESERVATION	READ	1.00		RANDOM	KEY
	STOCK RESERVATION -> STOCK KEEPING	READ	1.00			
	UNIT					
	STOCK KEEPING UNIT	UPDATE	1.00			
	STOCK RESERVATION -> STOCK KEEPING	DELREL	1.00			
	UNIT					
	STOCK RESERVATION -> CPSO ORDER LINE	DELREL	1.00			
	STOCK RESERVATION -> SIMULATION	DELREL	1.00			
	GROUP / CUSTOMER CLIENT ACCOUNT ORDER					
	STOCK RESERVATION	DELETE	1.00			

BUSINESS PROCESSING RULES	REF. NO. 3.4.3.3.3

TITLE

SATISFY REQUIREMENT (CONT'D PAGE 2)

DESCRIPTION

WHERE SUPERSESSION RULES ARE APPLICABLE TO THE PART, AT THIS STAGE IT WILL BE EITHER SIMPLE OR COMPOUND, THE FOLLOWING RULES WILL APPLY:-

1. SELECT THE FIRST SUPPLY LOCATION FROM THE POTENTIAL ONES.

2. IF THE SUPERSESSION IS PENDING AT THE INITIAL POTENTIAL SUPPLY LOCATION THE PARTNUMBER BEING PROCESSED SHOULD BE USED, OTHERWISE THE SUPERCEDING OR SUPERCEDED PARTNUMBER(S) SHOULD BE USED DEPENDANT UPON WHETHER THE SUPERSESSION IS REVERSIBLE AND WHETHER SUPERSESSIONS CAN BE APPLIED TO THE CUSTOMER ORDER.

3. DETERMINE WHETHER THE RELEVANT PART(S) CAN BE SUPPLIED TO THE CUSTOMER FROM THE SUPPLY LOCATION.

4. ATTEMPT TO ALLOCATE THE PART(S) FROM THE SUPPLY LOCATION AS FOLLOWS

   - VOR'S TO HAVE ALLOCATED WHATEVER FREE STOCK IS AVAILABLE
   - OTHER ORDER CATEGORIES TO HAVE ALLOCATED WHATEVER FREE STOCK IS AVAILABLE AT THE PRIMARY SUPPLY LOCATIONS BUT ONLY THE TOTAL OUTSTANDING QUANTITY FROM ANY OTHER LOCATION

   TAKING INTO ACCOUNT ANY QUANTITY CHANGES REQUIRED BECAUSE OF THE SUPERSESSION.

5. IF THE FULL REQUIREMENT HAS NOT BEEN ALLOCATED, THE SUPERSSESION IS PENDING AND THE LINE BEING PROCESSED IS FROM A VOR ORDER THAT ALLOWS SUPERSESSIONS TO BE APPLIED THE SUPERCEDING OR SUPERCEDED PARTNUMBER(S) SHOULD BE USED IN AN ATTEMPT TO ALLOCATE STOCK BY REPEATING STEPS 3 AND 4.

6. IF THE FULL REQUIREMENT HAS NOT BEEN ALLOCATED, THE SUPERSSESION IS PENDING AND THE LINE BEING PROCESSED IS FROM A NON VOR ORDER THAT ALLOWS SUPERSESSIONS TO BE APPLIED THE SUPERCEDING OR SUPERCEDED PARTNUMBER(S) SHOULD BE USED IN AN ATTEMPT TO ALLOCATE STOCK FROM THE APPLICABLE SUPPLY LOCATIONS.

NB:- THE LEADING FACTOR IN THE ALLOCATION OF STOCK TO VOR ORDERS IS CONSIDERED TO BE LEAD TIME.

THE LEADING FACTOR IN THE ALLOCATION OF STOCK TO NON VOR ORDERS IS CONSIDERED TO BE THE USING UP OF OLD STOCK WITHIN LEAD TIME.

THE CRITERIA THAT WILL NOT ALLOW A SUPERSESSION TO BE APPLIED IS AS FOLLOWS:-

- CUSTOMER HAS SPECIFIED THAT SUPERSESSION IS NOT ALLOWED, EITHER AGAINST ALL ORDERS OR A PARTICULAR ORDER
- SPECIAL PRICE ORDER
- IMPORT LICENCE APPLICABLE
- LETTER OF CREDIT APPLICABLE

BUSINESS PROCESSING RULES	REF. NO. 3.4.3.3.3

TITLE

SATISFY REQUIREMENT (CONT'D PAGE 3)

DESCRIPTION

A PROXIMITY GROUP WILL HAVE THE CAPABILITY TO HOLD A SUPPLIER AS A SUPPLY LOCATION. WHEN SUCH A SUPPLIER IS REACHED IN THE PROCESSING IT IS REQUIRED TO DETERMINE WHETHER THE PART BEING PROCESSED IS KNOWN TO THAT SUPPLIER. IF THE PART IS KNOWN THEN ITS ALLOCATION WILL BE TERMINATED AND THE NECESSARY DOCUMENTATION PRODUCED TO ENABLE THE SUPPLIER TO BE CONTACTED TO DETERMINE WHETHER THE STOCK REQUIRED IS AVAILABLE. INITIALLY THIS FACILITY WILL BE REQUIRED TO TO ENABLE THE PROVISION OF A PART FROM A CLIENT'S FACTORY, EG FOR URGENT ORDERS IT IS REQUIRED, IN CERTAIN INSTANCES, THAT THE STOCK AT BEAUVAIS IS CHECKED FOR AN ATHIS ORDER BEFORE ONE-STOP SHOPPING FROM ANOTHER SUPPLY LOCATION.

IN THE INSTANCE OF THE STOCK BEING AVAILABLE AT THE SUPPLIER THERE IS A REQUIREMENT TO PRODUCE WAREHOUSE DOCUMENTATION PRIOR TO THE RECEIPT OF THE PART. STOCK WILL BE ALLOWED TO GO NEGATIVE, IT BEING PUT BACK POSITIVE BY THE RECEIPT OF THE PART.

WHERE THE SUPPLIER CANNOT SUPPLY THE PART REQUIRED THE CUSTOMER ORDER LINE WILL REQUIRE TO BE SELECTED ON THE NEXT ALLOCATION RUN SO THAT ITS ALLOCATION CAN BE CONTINUED THROUGH ANY REMAINING PROXIMITY LOCATIONS.

02/03/88

USER REVIEW	DATE	REVIEWED BY	COMMENTS

BUSINESS PROCESSING RULES	REF. NO. 3.4.3.3.2

TITLE

CHECK FOR SUPERSESSION

DESCRIPTION

TYPES OF SUPERSESSION ARE AS FOLLOWS:

- SIMPLE (A => B)
- COMPOUND (A => B + C)
- SIMPLE SELECTIVE (A => B OR C)
- COMPOUND SELECTIVE (A => B + C OR D + E)

FOR EACH OF THESE TYPES, THE SUPERSESSION CAN BE OF THE SUBTYPE:

- USE OLD STOCK
- DO NOT USE OLD STOCK
- PARTIAL (I.E. THERE WILL STILL BE DEMAND FOR THE SUPERSEDED PART FOR SPECIFIC RANGES OR MODELS OF PRODUCT)

SUPERSESSION OF THE 'USE OLD STOCK - SIMPLE' TYPE CAN BE REVERSIBLE, I.E. IF THE SUPERSESSION IS A => B, IF B IS ORDERED, THERE IS NO STOCK OF B BUT THERE IS STOCK OF A, A WILL BE ALLOCATED.

SUPERSESSION WILL BE CONTROLLED WORLDWIDE BY THE SPECIFICATIONS GROUP. THE DECISION ON MAKING A SUPERSESSION CONDITION ACTIVE WILL BE CONTROLLED BY INVENTORY CONTROL, WHO WILL BE OPERATING ON A WORLDWIDE BASIS, AT A SUPPLY LOCATION LEVEL, IE DIFFERENT CONDITIONS MAY APPLY FROM LOCATION TO LOCATION.

THE FOLLOWING SUPERSESSION CONDITIONS WILL RESULT IN AN ORDER LINE BEING QUERIED AND REPORTED, ALLOCATION NOT TAKING PLACE AND NO BACKORDER BEING RAISED:-

- THE PART BEING NO LONGER SUPPLIED UNLESS ON A SCRAPPING ORDER FOR INSTANCE
- THE SUPERSESSION BEING EITHER SIMPLE SELECTIVE OR COMPOUND SELECTIVE WITH THE SUBTYPE 'DO NOT USE OLD STOCK' WORLDWIDE.
- THE SUPERSESSION BEING SELECTIVE WITH NO AUTHORISATION AGAINST THE ORDER LINE TO SUPPLY THE ORDERED PART.

WHERE THE LINE BEING PROCESSED HAS A SPECIAL PRICE OR IS ON AN ORDER THAT HAS AN IMPORT LICENCE OR A LETTER OF CREDIT AND THE PART ORDERED HAS A SIMPLE OR COMPOUND SUPERSESSION WITH THE SUBTYPE 'DO NOT USE OLD STOCK' WORLDWIDE, ALLOCATION WILL NOT TAKE PLACE. A BACKORDER WILL BE CREATED FOR THE PART ORDERED WITH A STATUS THAT WILL REQUIRE USER INTERVENTION TO :-

- CANCEL THE BACKORDER IF A DIFFERENT PART TO THAT ORDERED CANNOT BE SUPPLIED

OR - AUTHORISE SUPERSESSION TO TAKE PLACE

NB:- NORTH AMERICAN PDC'S REFER TO RACINE'S DATA WITH REFERENCE TO SUPERSESSIONS.

02/03/88

	DATE	REVIEWED BY	COMMENTS
USER REVIEW			

APPENDIX G

BUSINESS ISSUES AND RECOMMENDATIONS

BUSINESS ISSUES AND RECOMMENDATIONS

1. *Common Worldwide forms*

2. *Visual Displays for Parts Applications*

3. *Single Point of Contact/Single Customer Price*

4. *Positioning Existing dealer communication packages*

5. *Single Customer Code*

6. *Common Definition of Order Types*

7. *Franchise Checking*

8. *Pricing Methods/Marketing Campaigns*

9. *One-Stop-Shopping*

10. *Inventory Push*

APPENDIX H

COST/BENEFIT ANALYSIS

1. Cash Flow

Benefits	<———Year (USD X 000)———>				
	1986	1987	1988	1989	1990
Inventory Reduction	*0000*	*2520*	*2145*	*0755*	*0000*
Headcount Reduction	*0000*	*0716*	*1036*	*1560*	*1560*
Costs	*(410)*	*(974)*	*(1154)*	*(610)*	*(610)*
Cash Flow	*(410)*	*2262*	*2027*	*1705*	*950*

2. Gross Benefits

Benefits	<———Year (USD X 000)———>				
	1986	1987	1988	1989	1990
Inventory Reduction @ 20% Holding	*0000*	*0504*	*0429*	*0151*	
Headcount	*0716*	*1036*	*1560*	*1560*	
Total	*1220*	*1465*	*1711*	*1560*	

3. Head Count Savings

Benefits	<——Year Number of Heads——> 1986	1987	1988	1989	1990
	1	*18*	*15*	*10*	*10*

4. Development Costs

	<——Years——> 1986	1987	1988	1989	1990
Manpower Depreciated	*35*	*305*	*663*	*861*	*548*
CDC Recharge (cpu)	*000*	*60*	*60*	*50*	*00*
Ongoing Costs	*50*	*300*	*455*	*610*	*610*
One-time Cost	*14*	*125*	*182*	*133*	*0000*
Total	*99*	*790*	*1360*	*1654*	*1158*

APPENDIX 1

PROJECT TIMING

PROJECT TIMING

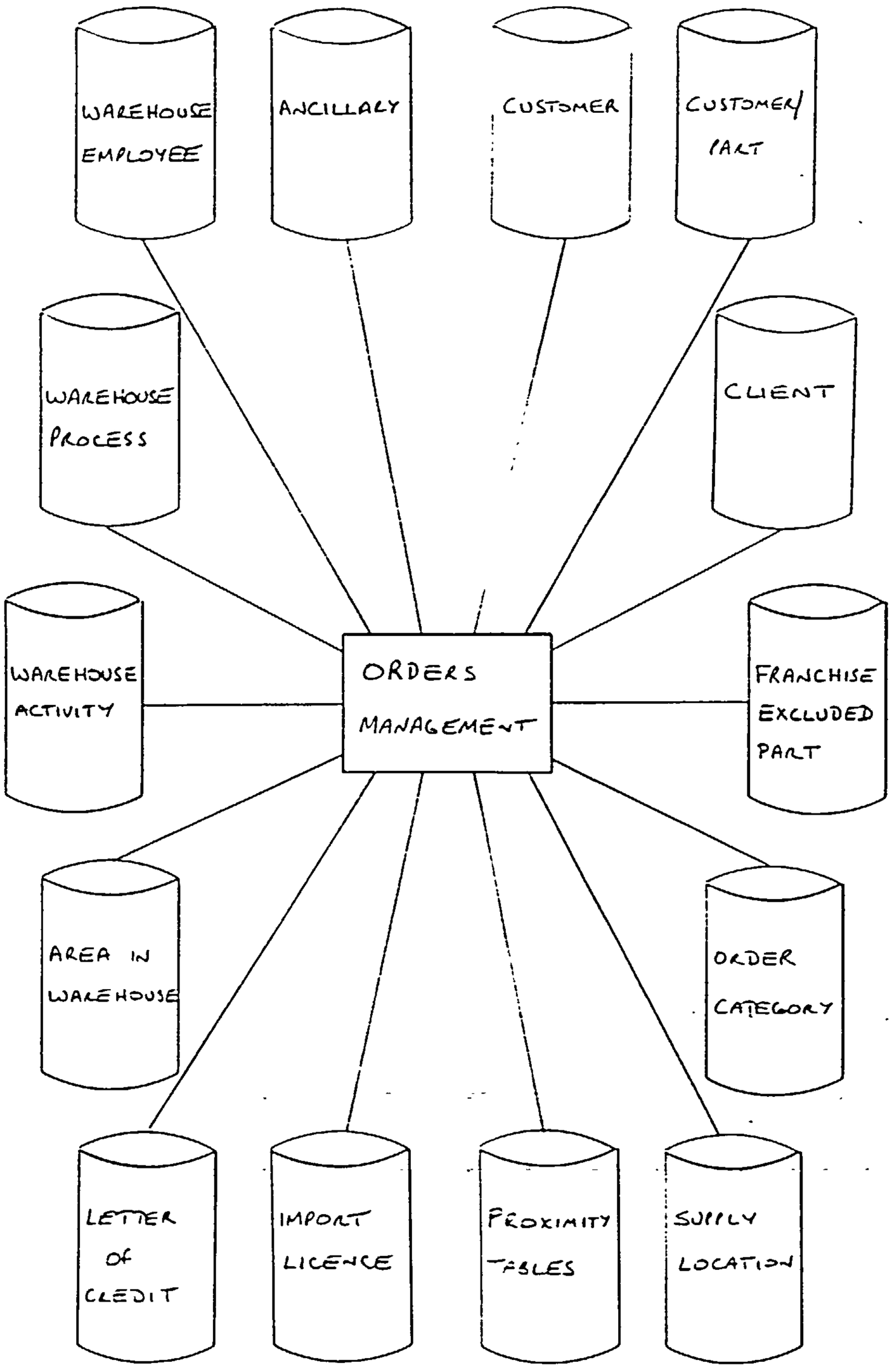
Systems	*Target Date*
1. Order Tracking and Invoicing	*UK Feb 1985*
	France May 1986
	NA May 1988
2. Dealer Communication package	*all Location 1988 ongoing*
3. Core Orders Management System	*UK Q3 1988*
	France Q4 1988
	NA Q1 1989
	Germany Q1 1990

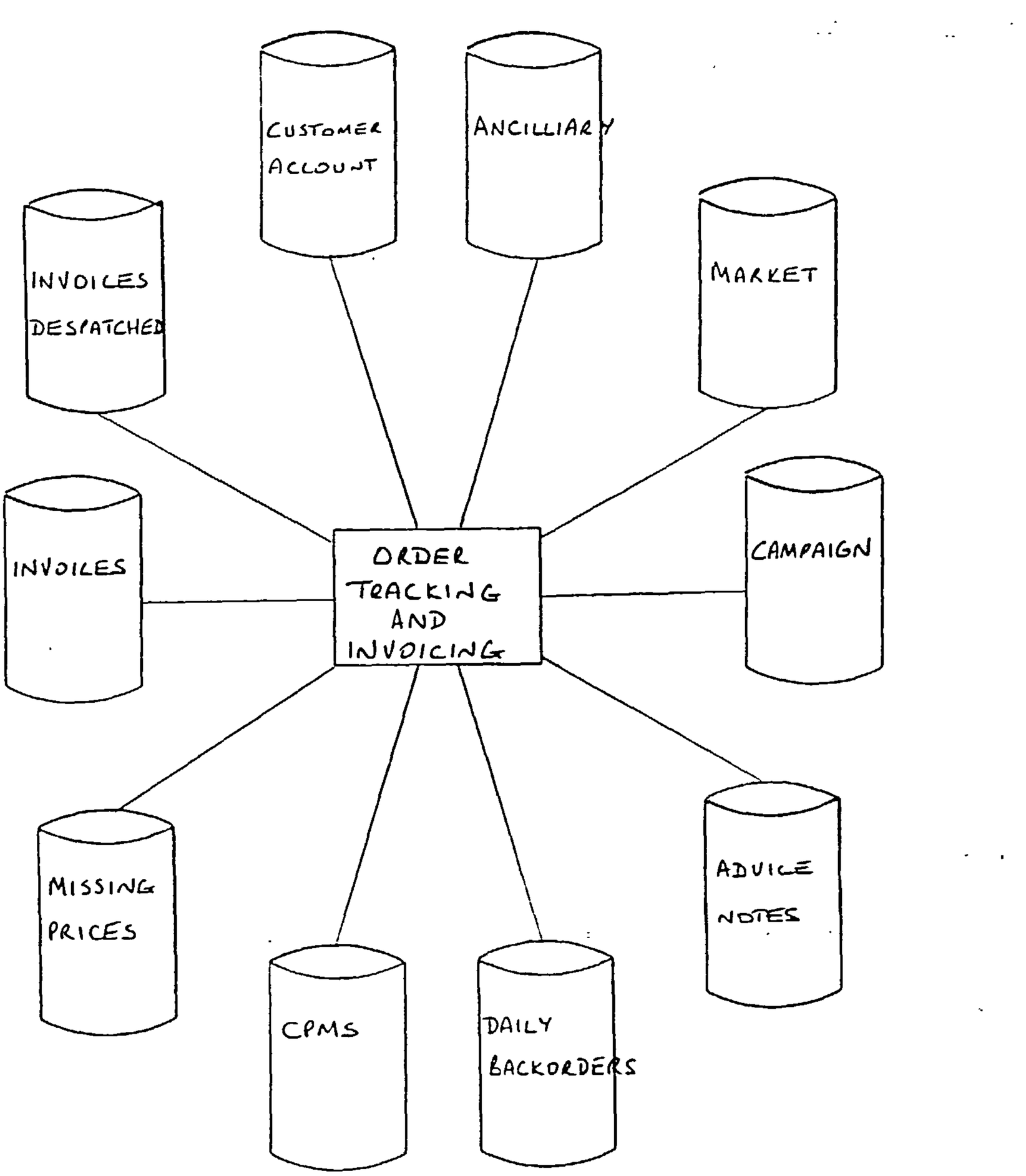
APPENDIX J

DATABASE STRUCTURES

- ORDERS MANAGEMENT
- MARKETING INTELLIGENCE

ORDERS MANAGEMENT SUPPORT DATABASES

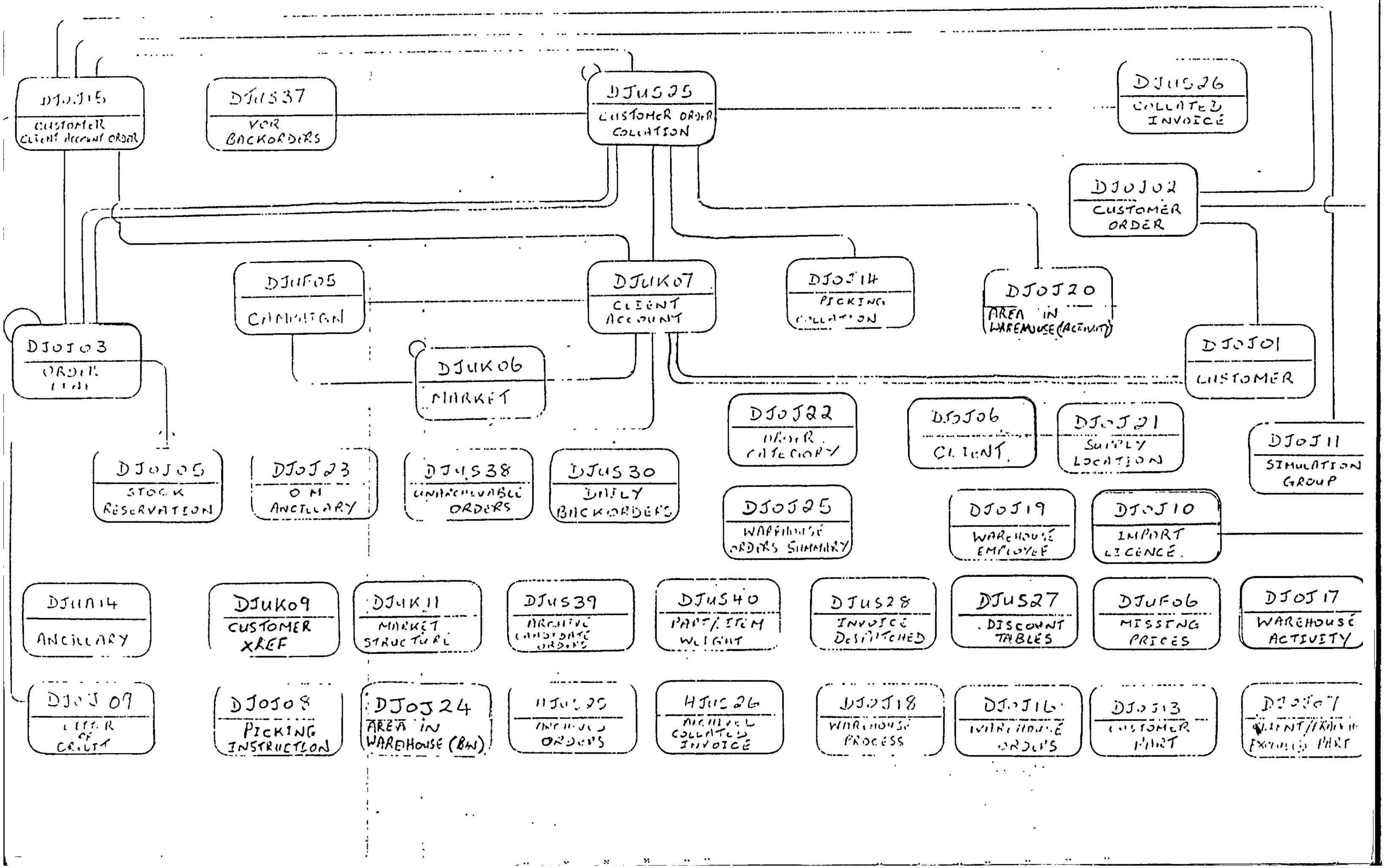



CUSTOMER ACCOUNT
ANCILLIARY
INVOICES DESPATCHED
MARKET
INVOICES
ORDER TRACKING AND INVOICING
CAMPAIGN
MISSING PRICES
CPMS
DAILY BACKORDERS
ADVICE NOTES

INC PARTS INVOICING - VOR II

AOCS GROUP MOJAVE

DATE 1-8-89



APPENDIX K

TRANSACTION VOLUME

PROJECT [illegible] DATE 1/2/1988

TRANSACTION CODE	DAILY VOLUME	MPP/ BMP	TRANSACTION DESCRIPTION	DATABASES ACCESSED	TASKS PERFORMED
GM001	10	MPP	Customer Order Enquiry	DB00101 DB00102	1
				DB00103	
GM002	10	MPP	Released Order Enquiry	DB00101 DB00102	2
				DB00103	
GM003	5	MPP	Stock Transfer Enquiry	DB00102 DB00104	3
GM010	5	MPP	Distributor Characteristics Enquiry	DB00103	5
GM01N	5	MPP	Distributor's Orders Enquiry	DB00103 DB00101	4
				DB00102	
GM02N	10	MPP	Product Characteristics Enquiries	DB00102	6
GM02N	5	MPP	Product/Order Enquiries	DB00102 DB00101	7
GM02N	5	MPP	Product/Stock Transfer Enquiries	DB00102 DB00104	8

APPENDIX L

DEALER COMMUNICATION PROTOTYPE

Function Menu

Press ESC to exit

Offline Functions :
1 Create / Update Part Orders and Enquiries
2 Customer Address File Maintenance
3 Electronic Mail Preparation/Reading
4 Print / Erase Reports
5 System Configuration

Online Functions :
A Parts Order Entry and Enquiry
B Electronic Mail - Send
C Electronic Mail - Receive

Programs for preparing parts order and parts pricing files.

MENU0001

22 Awaiting user input.

Parts Order Selection

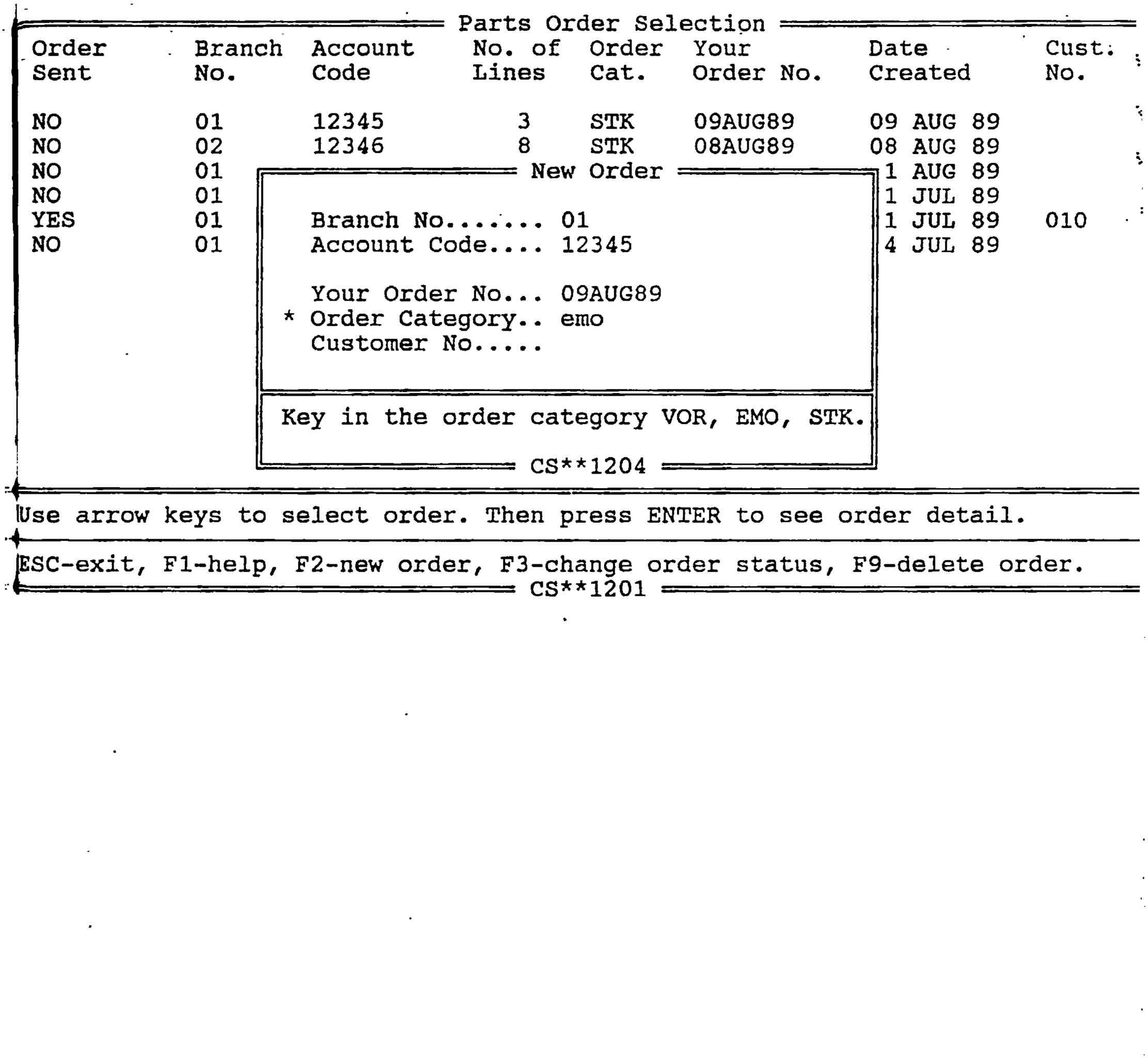
Order Sent	Branch No.	Account Code	No. of Lines	Order Cat.	Your Order No.	Date Created	Cust. No.
NO	01	12345	3	STK	09AUG89	09 AUG 89	
NO	02	12346	8	STK	08AUG89	08 AUG 89	
NO	01	12345	8	VOR	653217	01 AUG 89	
NO	01	12345	6	VOR	725731	31 JUL 89	
YES	01	12345	3	EMO	31JUL89	31 JUL 89	010
NO	01	12345	10	STK	14JUL89	14 JUL 89	

Use arrow keys to select order. Then press ENTER to see order detail.

ESC-exit, F1-help, F2-new order, F3-change order status, F9-delete order.

CS**1201

```
═══════════════ Parts Order Selection ═══════════════
Order      Branch   Account     No. of   Order   Your          Date          Cust.
Sent       No.      Code        Lines    Cat.    Order No.     Created       No.

NO         01       12345         3      STK     09AUG89       09 AUG 89
NO         02       12346         8      STK     08AUG89       08 AUG 89
NO         01                                                  1 AUG 89
NO         01                                                  1 JUL 89
YES        01                                                  1 JUL 89    010
NO         01                                                  4 JUL 89

                  ═══ New Order ═══
                    Branch No..... 01
                    Account Code.... 12345

                    Your Order No...
                  * Order Category..
                    Customer No.....

                  Your order number will appear on the
                  order picking document.
                        CS**1204 ═══

Use arrow keys to select order. Then press ENTER to see order detail.

ESC-exit, F1-help, F2-new order, F3-change order status, F9-delete order.
═══ CS**1201 ═══
```


Parts Order Selection
Order Sent
Branch No.
Account Code
No. of Lines
Order Cat.
Your Order No.
Date Created
Cust. No.
NO 01 12345 3 STK 09AUG89 09 AUG 89
NO 02 12346 8 STK 08AUG89 08 AUG 89
NO 01 1 AUG 89
NO 01 1 JUL 89
YES 01 1 JUL 89 010
NO 01 4 JUL 89
New Order
Branch No..... 01
Account Code.... 12345
Your Order No... 09AUG89
* Order Category.. emo
Customer No.....
Key in the order category VOR, EMO, STK.
CS**1204
Use arrow keys to select order. Then press ENTER to see order detail.
ESC-exit, F1-help, F2-new order, F3-change order status, F9-delete order.
CS**1201

Parts Order Selection

Order Sent	Branch No.	Account Code	No. of Lines	Order Cat.	Your Order No.	Date Created	Cust. No.
NO	01	12345	3	STK	09AUG89	09 AUG 89	
NO	02	12346	8	STK	08AUG89	08 AUG 89	
NO	01					1 AUG 89	
NO	01					1 JUL 89	
YES	01					1 JUL 89	010
NO	01					4 JUL 89	

New Order

Branch No..... 01
Account Code.... 12345

Your Order No... 09AUG89
* Order Category.. EMO
Customer No..... 005

Customer number is needed on orders to sent directly to customer.

CS**1204

Use arrow keys to select order. Then press ENTER to see order detail.

ESC-exit, F1-help, F2-new order, F3-change order status, F9-delete order.

CS**1201

Parts Order Detail

Supplier	Part Number	Quantity	Reference 1	Reference 2
MAS	145254M91	1	JIM LANG	T0905
MAS	264002M1	1	DON JOHNSTON	

Scroll up or down to select item. Press ENTER to ammend detail.

ESC-exit with no save, F1-help, F2-add to order, F5-save-order, F9-delete.

CS**1202

Parts Order Selection

Order Sent	Branch No.	Account Code	No. of Lines	Order Cat.	Your Order No.	Date Created	Cust. No.
NO	01	12345	2	EMO	09AUG89	09 AUG 89	005
NO	01	12345	3	STK	09AUG89	09 AUG 89	
NO	02	12346	8	STK	08AUG89	08 AUG 89	
NO	01	12345	8	VOR	653217	01 AUG 89	
NO	01	12345	6	VOR	725731	31 JUL 89	
YES	01	12345	3	EMO	31JUL89	31 JUL 89	010
NO	01	12345	10	STK	14JUL89	14 JUL 89	

se arrow keys to select order. Then press ENTER to see order detail.

SC-exit, F1-help, F2-new order, F3-change order status, F9-delete order.

CS**1201

Parts Order Detail

Ven	Part Number	Order Qty	UOI	Description	Status
MAS	376537X1	1			COMPONENTS TO BE ORDEF
MAS	365334X1	1		PLUG	SHIP FROM URMSTON
MAS	520124M1	1		HEAT SHIELD	SHIP FROM URMSTON
MAS	391021X1	1		RING	SHIP FROM URMSTON
MFI	441588M1	2	2	CAP	SHIP FROM ATHIS
MFI	442154M1	1		SPRING	SHIP FROM URMSTON
MAS	443010M1	1		PLATE	IMM BACKORDERED
MFI	5209488M91	1			SHIP FROM ATHIS

Order Status : Validation completed

ENTER-change or delete line, F2-add to order, F5-sends order for processi
F8-Totals order value. F10-toggles display fields. Hilighted items are sub

CS**0702

ok Awaiting user input

User Defaults

Import screens required..... Y (Y/N)

Order number to default to date.... Y (Y/N)

Mode of shipment / Carrier code.... N

Does screen display interference... N

Mode of Shipment / Carrier Code

Order Cat.	Mode	Carrier
VOR	01	0022
Emergency	02	0350
Stock	03	0527

Key in Mode and Carrier code for each category and press ENTER.

CS**0807

Key in a default mode of shipment and carrier code for each order category.

ESC-exit no save, F1-help, F5-save entries.

CS**0802