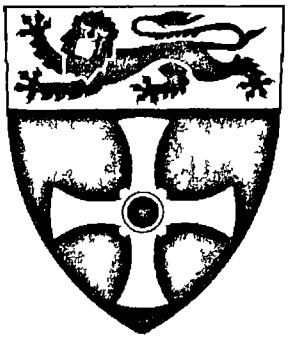
University
of Newcastle



A Strategy for Modelling the Design-Development Phase of a Product

A Thesis submitted for the Degree of
Doctor of Philosophy

May 1999

Author: Jaime A. Scott
The Newcastle EPSRC Engineering Design Centre

Supervisor: Professor Pratyush Sen
Department of Marine Technology

Abstract

This thesis describes a *strategy for modelling the design-development phase of a product.* Specifically, the aim is to provide product development organisations with a strategy for modelling and optimising sequences and schedules of design-development activities such that this phase of a product's life cycle can be managed and controlled in a more effective manner than before. This helps to ensure that product cost can be minimised, product quality can be maximised and the product's time to market can be reduced.

The proposed strategy involves carrying out five strategic functions, namely; (1) create a product design-work breakdown structure of design-development activities; (2) model the activities and their data-dependencies; (3) derive a near optimal sequence of activities; (4) derive an activity network diagram; and, (5) derive a resource-constrained schedule of activities.

The five strategic functions involve the use of a number of modelling and optimisation techniques. In particular, the thesis describes; (*i*) an enhanced version of a matrix-based modelling technique, namely the design structure matrix (DSM), which is used to model design-development activities and their data-dependencies; (*ii*) a newly created optimisation search procedure which combines a genetic algorithm with a heuristic-based local search to derive a near optimal sequence of activities; (*iii*) a newly created procedure which, based on the resolution of a matrix-model of activities linked by their mutual dependence on one another for data, is used to derive an activity network diagram of activities and precedence relationships; and, (*iv*) the development of a multiple-criteria genetic algorithm which is used to derive a near optimal resource-constrained schedule of activities.

Near optimal sequences are derived using objectives such as minimising iteration and maximising concurrency whilst near optimal schedules are derived using objectives such as minimising the time taken to complete all activities and maximising the utilisation of scarce resources. At the same time, throughout the thesis, a number of related concepts are discussed and developed. In particular, the thesis addresses concurrent engineering, a systems approach to business processes and design reuse.

In order to demonstrate how the modelling strategy can be applied, an industrial case study based on the design-development of a warship has been included.

Acknowledgements

I would like to thank the following people for the help that they provided over the three years that it took to create this thesis.

Firstly, to my friend and mentor, former Visiting Professor George Snaith, words simply cannot express the debt of gratitude that I owe. To my girlfriend, Debbie, I intend to fulfil all of the promises that I made in order to keep you happy when I needed to work.

Many thanks are due to Dr. David Todd of the Engineering Design Centre for his invaluable help, and also to the PDO members of the companies approached as part of the research, especially Dr Malcolm Courts of Vosper Thornycroft for his avid interest and support.

I also wish to thank my supervisor Professor Pratyush Sen for giving me the opportunity of working in this exciting area of applied research. Finally, I would like to thank the EPSRC and the Newcastle Engineering Design Centre for funding this research.

JS.

Contents

1. Introduction

1.1 Studying a Product's Design-Development Phase

The design-development of engineering products, although having been performed for many years, was not widely studied in a systematic manner until just after the middle of this century. However, the study of a product's design-development phase has now been a widespread preoccupation of engineers, designers and researchers for the past five decades.

During this time various schools of thought have emerged with respect to how the design-development phase of a product should be carried out. Broadbent [Broadbent 80] describes three such schools of thought; (*i*) the *semantics*; (*ii*) the *syntax*; and, (*iii*) the *past experience* schools.

The semantics school, attributed to Rodenacker [Rodenacker 70], believes that the design-development of a product should be chaotic and creative. In this respect, Jones [Jones 80] views an engineer who subscribes to this school of thought as a *black box* containing a set of mysteriously creative processes.

On the other hand, the syntax school believes that the design-development of a product should be systematic and scientific. In this respect, Jones views an engineer who subscribes to this school of thought as a *glass box* containing a set of completely explicable and rational processes.

Finally, the past experience school, in it's beliefs, lies somewhere between the semantics and the syntax schools and believes that design-development skills can only be acquired efficiently through experience.

In their survey of design philosophies, models, methods and systems Evbuomwan *et al* [Evbuomwan 96] summarise that the three schools of thought are all relevant in one way or another. However, in today's highly competitive business environment, it is becoming increasingly evident that approaches to studying design-development based on the syntax school of thought offer the best opportunities for improving a company's competitive advantage. In agreement, Wallace [Wallace 89] points out that the design-development phase of a product cannot be carried out efficiently if it is left entirely to chance. In this respect "the

aim of a systematic approach is to make a product's design-development phase more visible and comprehensible so that all those providing inputs can appreciate where their contributions fit in".

In Chapter 3 it is concluded that in today's business environment it is *the elapsed time it takes a company to get a new product to market* that has emerged as the primary focus of contemporary strategies aimed at improving competitive advantage. Furthermore, it is now becoming clear that the best opportunities for achieving this lie in the improvement of design-development performance.

To this end, engineering companies may need to develop strategies for reducing time to market through improved design-development performance. Increasingly, the adoption of *concurrent engineering (CE)* principles tends to be at the centre of such strategies. However, in addition, it is proposed that a systematic approach to studying and modelling a product's design-development phase also offers opportunities for improving a company's competitive advantage through improved design-development performance. This reflects the sentiments expressed by Smith and Eppinger when they state that "one approach to improve design-development is to recognise that it is often procedural and repeatable; therefore it can be modelled as one might model a manufacturing process that needs to be improved" [Smith 97].

1.2 Strategies

As part of the research summarised in this thesis, time was spent discussing and studying the design-development of products with project-managers, designers, planners and researchers at a range of engineering companies. These companies are involved in the design-development and production of a range of different types of engineered products including; DIY consumer products; air-filters; electronic jacquards for weaving-looms; offshore production platforms; surface warships; merchant ships; and, naval submarines.

The results of these informal discussions have helped to provide a guiding influence throughout the research. Most specifically, at the beginning of the research project, the companies involved were asked to discuss the notion that a systematic approach to studying and modelling a product's design-development phase offers opportunities for improving a company's competitive advantage through improved design-development performance.

Feedback from these discussions was mixed. Some companies argued against the need for modelling the design-development phase of their products (*See* Section 3.2). Some companies accepted that there was a need but believed that their approach to modelling, planning and organising *product design-development*, based in large on their approach to modelling, planning and organising *product production*, was more than adequate. Two companies, on the other hand, accepted that there was a need for modelling the design-development phase of their products, and also accepted that adapted approaches previously focused on modelling production-work were inappropriate (*See* Section 3.4). These companies were interested in finding a means of modelling and optimising sequences and schedules of activities such that the design-development phase of their products could be planned and organised in a more effective manner.

In broad terms, a *strategy* is simply a *means* of achieving an *objective*. In this respect, the research contained in this thesis aims to provide the product development organisation (PDO) of an engineering company with a strategy for achieving the above-mentioned objective. That is, *modelling and optimising sequences and schedules of activities such that the design-development phase of a product can be planned and organised in an effective manner.*

1.3 An Outline of the Thesis

The thesis contains eight chapters and four appendices and describes a *strategy for modelling the design-development phase of a product.* Following this introduction, Chapter 2 sets the scene by introducing the notion that, in order to gain competitive advantage, an engineering company should perhaps focus attention on the design-development phase of it's products without losing focus on marketing and manufacturing performance. By doing so, companies will tend to have a much greater chance of reducing the time it takes to get their products to market, a business objective which, in the majority of cases, should now be viewed as a priority.

In this respect, Chapter 2 highlights the potential need for engineering companies to develop strategies that incorporate the principles of *concurrent engineering (CE).* However, before some of the most relevant notions, techniques, and enabling technologies associated with concurrent engineering are elaborated, attention is drawn to the potential benefits as well as the costs of adopting reduced time to market as an objective for gaining competitive advantage. Chapter 2 also summarises a *systems approach* to the management of design-

development and other business processes and, in particular, highlights some of the *invisibles* that are crucial to achieving the improved performance of any business process. Such invisibles relate specifically to the creation of *a set of stable process whose output is subject to contained variability.*

Chapter 3 begins by making *a case for modelling* the activities of a product's design-development phase. In summary, it is proposed that, in addition to the adoption of CE principles and a systems approach, it may be that a systematic approach to studying and modelling a product's design-development phase also offers opportunities for improving a company's competitive advantage through improved design-development performance.

Following this proposal, a *traditional strategy* that engineering companies tend to follow when faced with the task of modelling a product's production or design-development phase is introduced, and a number of drawbacks that may be encountered when traditional modelling strategies are specifically applied to a product's design-development phase are detailed. Based in part on these drawbacks and feedback from industry, a set of requirements is listed for a modelling strategy specifically focused on design-development. The proposed strategy for modelling a product's design-development phase is then introduced and prior research on the subject is summarised. Ultimately, the contribution of this latest research at the strategic level is summarised by describing how the research behind the proposed strategy differs from that undertaken previously.

Throughout the thesis, the proposed strategy is described with reference to a case study that is based on the pre-contract design-development of a warship. (*See* Appendix I). The proposed strategy consists of five strategic functions (*See* Figure 3.3). In Chapter 3, the first and second of these functions, namely "create a product design-work breakdown structure" and "model the activities and their data-dependencies" are demonstrated with the aid of the warship case study. However, because the relevant techniques are yet to be described, the third, fourth and fifth functions are demonstrated, with the aid of the warship case study, in later chapters of the thesis.

Chapter 4 focuses on the third function of the proposed modelling strategy and, in particular, describes and tests the principal mechanism for solving the Activity Resequencing (ActRes) Problem of deriving a near optimal sequence of activities based on pre-defined objectives.

The mechanism, namely, the *Genetic Algorithm (GA)-Local Search Procedure,* is based on the application of two search techniques. The first, the *Optimal Sequencer Genetic Algorithm,* prunes the large, noisy and discontinuous search space to return an interim solution set of activity sequences which is then improved upon by the second technique, the *Heuristic Local Search.*

Resequencing activities based on a number of objectives derives new sequences. Chapter 5, using the GA-Local Search Procedure addresses the ActRes Problem of deriving a near optimal sequence of activities based on the specific objective of *minimising the iteration of design-development activities.* To begin, Chapter 5 considers the role iteration plays in the design-development phase of a product and introduces a classification scheme to describe iteration.

Chapter 5 then goes on to report on prior published research which, based on the objective of minimising iteration, uses a matrix-based modelling technique known as the *design structure matrix (DSM)* in conjunction with various search techniques to derive a near optimal sequence of activities. Finally, a new solution approach to the ActRes Problem of deriving a near optimal sequence of activities based on the specific objective of minimising iteration is introduced. Using five test cases, this new solution approach is then tested and the results compared against prior solutions published in the literature.

Chapter 6, using the GA-Local Search Procedure addresses the ActRes Problem of deriving a near optimal sequence of activities based on the newly considered objective of *maximising the concurrent processing of design-development activities.* Minimising iteration is always important, however in Chapter 6 it is demonstrated that, under an appropriate set of conditions, maximising the concurrent processing of activities is also desirable since it implies reduced lead-times.

Thus, Chapter 6 demonstrates that, supported by a PDO where the *reuse of design-data* is facilitated, the best approach to solving the ActRes Problem of deriving a near optimal sequence of activities is to use a *multiple criteria approach* based on both iteration and concurrency. Such an approach helps to ensure faster design-development whilst ensuring that the added cost due to iteration is still as small as possible. Finally, Chapter 6 brings together the research detailed in Chapters 4, 5 and the first half of Chapter 6. In this respect, Chapter 6

demonstrates, using the warship case study, the third and fourth strategic functions of the proposed modelling strategy, namely "derive a near optimal sequence of activities" and "derive an activity network diagram".

Chapter 7 focuses on the fifth and final function of the proposed modelling strategy, namely, "derive a resource-constrained schedule". Because the *Resource-Constrained Project Scheduling (RCPS) Problem* can be viewed as a generalisation of the *Machine Scheduling Problem*, Chapter 7 begins with a summary of the latter problem. Using a classification scheme, Chapter 7 also details various aspects of the RCPS Problem and reviews previously published solution-approaches to the problem before introducing a new solution approach to the resource-constrained scheduling of design-development activities.

Having explained the concepts behind this new solution approach, the *Multiple Criteria Genetic Algorithm (MCGA) Project Scheduler*, which acts as the principal mechanism of the fifth strategic function, is described. Using an example based on the warship case study, a number of tests are then summarised which have been used to demonstrate the functionality of the MCGA Project Scheduler.

Finally, Chapter 8 gathers the conclusions and contribution of the research contained in this thesis and suggests further work that could be undertaken.

2. Reducing Time to Market

2.1 Introduction

Both prior to and during the first half of the 1980s most *engineering companies,* that is those engaged in the design-development and production of products, were agreed that *product performance and pricing* should be the focus of strategies aimed at gaining competitive advantage. However, by the middle of the decade companies had begun to focus more on *product quality* [Roy 85].

More recently, as a result of rapid technological change, increased market segmentation, and reduced product life cycles, there has been a further shift of strategic focus. Specifically, in the 1990s, it is *the elapsed time it takes a company to get a new product to market* that has emerged as the primary focus of contemporary strategies aimed at gaining competitive advantage [Clark 91].

However, whether the focus of such strategies is a product's eventual performance, cost/price, quality, or time to market, in the majority of cases the predominant part of any one of these parameters is determined by decisions made during a product's *design-development phase.* As a result, whilst most successful engineering companies once tended to focus strategies for gaining competitive advantage solely on the improvement of their *marketing capabilities* and *manufacturing performance*, it is now becoming clear that the best opportunities for achieving such gains lie in the improvement of *design-development performance* [Kusiak 92].

In this respect, engineering companies may need to develop strategies for reducing time to market through improved design-development performance. Increasingly, the adoption of *concurrent engineering (CE)* principles tends to be at the centre of such strategies.

In Section 2.2, a comparison is made between the perceived benefits and the associated costs of adopting reduced time to market as an objective for gaining competitive advantage. Section 2.3 then highlights a number of notions, techniques, and enabling technologies associated with concurrent engineering.

In addition, it is postulated here that, without a rational framework based on a *systems approach* to the management of design-development and other business processes, the many

benefits listed by the proponents of concurrent engineering will be elusive. Factors associated with such an approach to managing business processes remain largely invisible, but they are crucial to achieving the improved performance of any business process. Therefore, in conclusion to the chapter, Section 2.4 describes some of these *invisibles* in the context of a systems approach.

2.2 The Benefits Versus the Costs of Reducing Time to Market

The two major benefits associated with reduced time to market are *increased sales* and *higher profits* [Smith 91]. By getting new products to market faster, increased sales can result simply from an extension to the product's sales life. Increased sales can lead to enhanced customer loyalty which, in itself can lead to yet more sales. Furthermore, companies that get their products to market first can benefit from the higher profits that can result when, in the absence of competitors, the first company to release a product has greater pricing freedom.

Finally, a company with a fast product design-development capability can increase product sales and profits through the ability to incorporate the very latest *component technology* into it's new products. If a company is faster at the design-development of new products than it's competitors, then it may decide to delay the start of the design-development of certain new products and await the availability of more efficient and more cost effective component technology.

Because of it's fast design-development capability the company can still deliver *delayed-start products* to the market at the same time as it's competitors. In such situations, increased sales may result from the stronger customer appeal associated with these delayed-start products that are perceived to encapsulate the very latest component technology. At the same time, higher profits may result from reduced product costs which, in turn, result from delayed-start products that encapsulate more cost-effective component technology that is cheaper to buy and/or produce.

Whilst the benefits associated with reduced time to market may be clear, increasingly the need for caution is being voiced [Crawford 92, Handfield 94, Lamb 97]. In this respect, the potential costs of adopting such an objective are becoming clearer. For example, one way of reducing time to market lies in the *concurrent processing* of *product design-development*

activities, *production-process design-development activities*, and *production activities*. Concurrency frequently calls for decisions to be made using partial data, information and knowledge. Frequently, decisions based on such are more likely to be subject to error. Errors of decision tend to result in wasted effort and the need for rework. In turn, rework increases product costs. Therefore, in addition to the readily quantifiable cost of the required investment in new techniques and enabling technology, unanticipated costs may arise from the rework that results when initiatives, such as concurrency, are *inappropriate* and/or are *poorly implemented*.

In deciding whether or not a proposed objective or initiative is appropriate, a clear understanding of the market is required. For example, whilst reduced time to market may be an overriding objective in the electronic consumer products market, other objectives such as cost reduction may be dominant in industries such as shipbuilding.

Proponents of strategies for reducing time to market often cite the popular McKinsey Report as summarised by Dumaine [Dumaine 89]. The report indicates that six months of delay can reduce a product's life-cycle profits by 33 percent. However, this observation about lateness costing more in lost profits than in being over budget, only applies to high-growth markets associated with products that have short life cycles. One such example is the electronic consumer product market which includes videocassette recorders and personal computers. Conversely, and according to the same report, for slow-growth markets associated with long-life products such as ships, a 9 percent total product cost overrun can result in anything up to a 45 percent decrease in life-cycle profits.

Therefore, before adopting reduced time to market, or any other objective as the focus of a strategy aimed at gaining competitive advantage, an engineering company should have a clear understanding of the drivers of competitive advantage and the market forces that apply to their market sector. That said, the research summarised in this thesis assumes that an engineering company has a clear understanding of the objectives that need to be satisfied in order to gain competitive advantage.

It is considered that the achievement of one or more of the following objectives can help an engineering company gain competitive advantage. These principal objectives include; (1) reduce time to market; (2) reduce product costs; and, (3) improve product quality. Ultimately,

the prevailing market forces and the market sector in question determine which of these objectives are appropriate. Based on discussions with a range of engineering companies and assuming that all three objectives are desirable, four lower level objectives that support the achievement of these three principal objectives have been identified and are addressed within the proposed modelling strategy. These are:

(i) Eliminate unnecessary iteration and rework from the product's design-development phase. By focusing on this objective, both time to market and product costs can be reduced (*See* Chapter 5).

(ii) Increase the potential for concurrency. If implemented effectively, the concurrent processing of activities can lead to reduced time to market (*See* Chapter 6).

(iii) Improve the utilisation of scarce resources. Management initiatives based on improving the utilisation of scarce resources can lead to reduced product costs and an improvement in product quality (*See* Chapter 7).

(iv) Improve the level of skills retained by members of the product development organisation. Management initiatives based on improving skill levels and maximising learning can lead to an improvement in product quality. (*See* Chapter 7).

2.3 Concurrent Engineering

The focus of this thesis relates to a systematic approach to studying and modelling a product's design-development phase as a means of improving a company's competitive advantage through improved design-development performance. In this respect, four lower-level objectives that support the achievement of the three principal objectives have been identified and are addressed within the proposed modelling strategy.

However, proponents of *concurrent engineering* (CE) [Hyeon 93, Shina 94, and Syan 94] propose that the adoption of this management philosophy can also help engineering companies to improve design-development performance. In particular, it is proposed that the adoption of certain techniques and enabling technology can also help engineering companies to achieve the three principal objectives of reducing time to market, reducing product costs, and improving product quality.

Consequently, for the sake of completeness, the aim of this section is to detail how the achievement of the three principal objectives can be aided through the application of concurrent engineering.

Based on prior research [Hyeon 93, Shina 94, and Syan 94] and discussions with practitioners, it is considered that the three principal objectives can be facilitated through (*i*) *the improved integration and communication of data, information, and knowledge* within the company; and, (*ii*) *the concurrent and faster processing of design-development and production activities*. The following two sub-sections elaborate on each of these facilitators in turn.

2.3.1 Improving the Integration and Communication of Data, Information and Knowledge

Techniques and enabling technology for achieving the improved integration and communication of shareable data, information, and knowledge include the following:

(1) The integration of product data and information

The integration of product data and information can be achieved through the application of computer-based management information systems such as a *Product Model* which forms part of a *3-D CAD/CAE/CAM System.*

A Product Model can be used to create and store a vast amount of data and information relating to a *product's design and production definitions* (*See* Appendix I). Such data and information includes 3-D accurate and full-scale geometry of the product "as it will be" and "as it will be built". Also included are the myriad of associated non-graphical attributes such as product specifications, the results of simulation predictions and performance analyses, schedules, as well as information required for production and maintenance.

Whilst the integration of product data and information is essential and can be facilitated through management information systems such as the Product Model, it may also be necessary to focus on how the data and information can be exchanged. Standardisation of data format, using compatible hardware and software represents one approach. However, if this is not possible, then the neutral file format is a valid alternative whereby differently configured systems communicate with each other via a standard format.

(2) The integration of the product development organisation's in-house structured knowledge and experience with that of other in-house functions and departments

This integration is achieved primarily through the formation of a *multi-disciplinary product development organisation (PDO)* and the application of methods that facilitate the formalised recording and communication of in-house design-development, production and marketing knowledge and experience.

Where appropriate the improved integration of knowledge between the PDO and those personnel involved in the marketing function can help improve product quality. By ensuring that the knowledge gained through the marketing function is passed onto the PDO, the *voice of the customer* is never lost and the product can be designed-developed so that it reflects the customer's perception of quality in the most cost effective manner. A formal technique for facilitating this objective *is quality function deployment (QFD)* [Akao 90].

Perhaps the best opportunities for reducing a product's cost as well as improving product quality lie in the initiative of design-developing a product for economic production. By adopting the principles of *design for economic production,* cost reductions can be made by avoiding the processing of unnecessary design-development and production activities, and by avoiding the rework that results from the PDO's poor understanding of the company's production system. These avoidable costs are incurred when the PDO fails to properly reflect production-related knowledge into the product's design-development phase.

At the same time, improved integration ensures that the production cost savings associated with new production facilities such as automation are fully realised. By developing a *product's production definition* that takes full account of the company's latest production techniques, the cost savings offered by new facilities can be achieved. A PDO's and production team's knowledge and experience can be defined and stored in various specialised *User Libraries of Standards.* Those types of libraries relevant to improving the integration of the PDO's and production team's knowledge and experience include;

- Documentation of the company's production system in terms of it's facilities and processes. Special attention should be given to those facility and process constraints which have the strongest impact upon decisions made by the PDO.

- Guidelines to be used by the PDO that document preferred production methods and working procedures. *Methods engineering* is becoming increasingly popular. For example, in the case of ships, the Product Model can be used to evaluate assembly sequences to ensure that the assemblies, which constitute the product's production definition, can in fact be assembled and accessed for welding/joining.

(3) Communication between members of the product development organisation (PDO)

On the basis of the above, there exists a range of situations where the integration of shareable data, information, and knowledge is desirable. Once such situations have been identified, it may be necessary to develop methods for improving the recording and communication of shareable data, information and knowledge such that all members of the PDO are fully aware of past, current and standard viewpoints and decisions. Where appropriate, both the *co-location* of teams, and the application of information technology (IT) systems can facilitate improved communication. The latter includes *Intranets, e-mail, The Internet,* and proprietary software such as *Lotus Notes*™.

2.3.2 The Concurrent and Faster Processing of Design-Development and Production Activities

There are also a number of techniques and enabling technologies which, in addition to improving the integration and communication of shareable data, information and knowledge, can also facilitate the concurrent and faster processing of design-development and production activities. These include;

- Tools that allow the PDO, during the initial stages of a product's design-development phase, to design-develop a wide range of varying *outline product solution schema* (*See* Appendix I) that can be evaluated quickly using heuristic-based performance analyses.

- Computer aided engineering (CAE) analysis tools such as computational fluid dynamics (CFD) and finite element (FE) software can help the PDO to identify performance failure earlier. This can help to speed-up and reduce the number of iterations and physical models that are necessary in order to converge upon an acceptable solution schema.

- Appendix I, using a warship as an example, describes a *staged approach* to the creation of a product's design definition. Such an approach, supported by the application of a Product Model, facilitates a number of time saving initiatives:

  - By effectively managing the interfaces between the sub-systems which constitute the product's design definition and by utilising the multiple user capability of the Product Model, different members of the PDO are able to work on the design-development of inter-related parts of the product's design definition concurrently rather than sequentially.

  - By adopting a staged approach to the creation of a product's design and production definitions, information is added to the Product Model as each design decision is made. Using the Product Model, any information yet to be made available can be date-tagged in order to ensure that any decision delayed by a lack of information will be returned to at a pre-defined future date. An effective procedure for managing information and uncertainty can help to avoid the unnecessary delays that are sometimes experienced during design-development and production when critical-path decisions cannot be made because of a lack of information.

  - Through a conscious effort on the part of the PDO to clearly define the *form*, *content* and *timing* of product data and information necessary as input to each design-development activity, unnecessary delays, that result from members awaiting *complete packages* of engineering information, can be avoided.

2.4 A Systems Approach to the Management of Business Processes

Everybody at some point in their life wishes that they could predict the future. This is because large benefits inevitably result from the ability to *forecast* and *plan for the future effectively and accurately.*

Everywhere there are examples to be found of people trying to forecast and predict future outcomes. One example is weather forecasting. Large benefits can result from an accurate weather prediction and, conversely, large drawbacks can result when those predictions are inaccurate. For example, based on a forecast of severe sea storms, a deep-sea fisherman may decide not to go to sea. If the forecast is accurate, the fisherman has avoided risking his life, but obviously if it is inaccurate then he has lost a day's fishing he can ill afford to lose.

It tends to be accepted that the first step towards accurately forecasting the behaviour of a *system* is to develop *an understanding of the system* that is being studied. In tying to understand a system, an observer will tend to build up a picture or, more precisely, a model. In the first instance, this may take the form of a mental model in the mind of the observer. However, for large and complex systems such as the earth's weather system, a more formalised model is required in order to communicate an understanding of the system to others. Consequently, it is proposed that the most effective way of understanding a system and communicating this understanding to others may be to use a formalised and structured model.

In addition to gaining an overall understanding of the system using a model, it may be necessary to gain a more specific knowledge of those system elements that have the strongest influence on the outcome of future events related to the system and more specifically how those system elements interact.

Based on these observations, the first requirement for forecasting system behaviour and planning for the future effectively and accurately has been identified as

The creation of a formalised and structured model
of the constituent elements of the system and an understanding of their inter-relationships.

As will be defined in the following sub-section, at the lowest level of abstraction, all systems consist of processes. It is the output from these processes that together determine the output, and hence the overall behaviour, of the system.

Certain systems and their constituent processes, such as a weather system and it's processes, are *chaotic*. Consequently, it is very difficult to predict their behaviour. However, man-made systems and processes such as an engineering company's production system and processes are more constrained in nature and are therefore less chaotic. In the absence of chaos, the potential for the accurate prediction of process and system outputs increases.

As will be explained later in Sub-Section 2.4.3, a process is considered to be *stable* if the statistical patterns of the values of the parameters that define the outputs of the process are Gaussian. If the output of a process can be measured in statistical terms, then it can be predicted in the same way. If this is the case for all of the system's processes then the output of the system can be predicted in statistical terms. In other words, "A process is stable if its

future performance can be statistically predicted" [Deming 82]. Furthermore, when the variability of these statistical patterns are engineered to be small, the values of the parameters that define the output tend to deterministic point-values and, as such, process and system outputs can be predicted more accurately and with more certainty.

Based on these observations, the second requirement for forecasting system behaviour and planning for the future effectively and accurately has been identified as

A set of stable processes with contained variability.

2.4.1 Definition of a System

Combining the observations of a number of authors, a *system* can be defined as *a set of resources* that are applied to *a hierarchy of sub-systems* (systems at a lower level in the hierarchy) linked to each other and to the *operational environment* by *a set of inter-relationships*, in order to produce *a set of outputs*. [Churchman 68, Kline 72, Hall 89]. Therefore, as summarised in Figure 2.1, *the elements of a system* include a hierarchy of sub-systems, resources, outputs and operational environments, all linked by a set of inter-relationships.

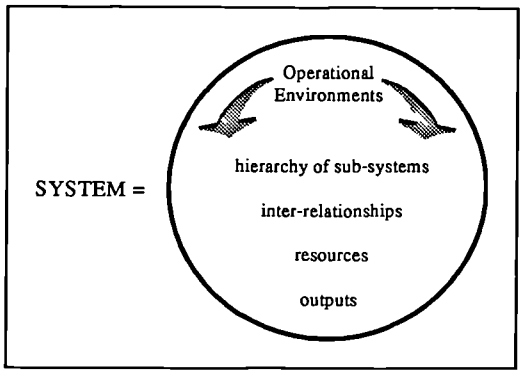


Figure 2.1: A System

Using the systems approach, everything that we encounter is a system, whether it be physical (e.g. an engineering company), or non-physical (e.g. the economy). Using the notion of a hierarchy, all systems belong to higher level systems and contain sub-systems of their own. For example, consider an engineering company as a system. The elements of the *engineering company system* consists of:

- *Sub-Systems:* Lower-level systems which, at the lowest level, consist of individual processes. For example, the engineering company system belongs to a country's industrial system but contains sub-systems of it's own including, amongst others, a design-development sub-system which consists of design-development processes.

- *Resources:* A collection of resources, including techniques, enabling technologies and human process operators.

- *Operational Environment:* Influences that are outside the control of the system operators (e.g. government rules and regulations) yet impact upon system performance.

- *Outputs:* The intended outputs of an engineering company are engineered products that are required to be superior to those of competitors in terms of price, quality and time to market. Using the notion of a hierarchy, each sub-system has it's own outputs that should go some way towards contributing to the achievement of the encompassing higher-level system outputs.

- *Inter-Relationships:* A set of inter-relationships which link all system elements that comprise sub-systems, processes, resources, the operational environment and outputs.

In addition to this principal definition, a system should be considered on the basis of a time dimension. *The life cycle of a system* defines the various phases of the system from origin through to disposal. Having considered the engineering company system, now consider a *product system* such as a warship.

In terms of a product life cycle, the warship originates with the formulation of a set of functional/operational requirements as part of it's *planning phase.* During *the design-development phase* of the warship's life cycle, the engineering company's design-development sub-system translates these requirements into a definition that meets all of the requirements. This definition needs to be of sufficient detail such that, during *the production phase,* the engineering company's production sub-system can manufacture and assemble the warship. The warship then moves on to it's *operational phase* where it performs it's intended purpose. Ultimately, when it can no longer perform it's intended purpose, or circumstances

change in such a way that the original intended purpose is no longer appropriate, the warship moves on to it's *disposal phase.*

The life-cycle of a product system, such as the warship, should be considered because, although the warship's hierarchy of constituent sub-systems may remain unchanged throughout it's life-cycle, other (warship) system elements (resources, operational environment, outputs, and their inter-relationships) vary with each phase.

In Section 2.3 the notion of improving the integration and communication of shareable data, information, and knowledge was first introduced. Here it is stated that, by considering the (warship) system's life cycle, knowledge relating to all phases of the life cycle can be considered and communicated to the PDO. As previously stated, doing so can help to reduce a product's time to market, reduce product costs, and improve product quality.

Attention to such considerations is covered by *designing for X (DFX).* One of the so-called DFXs is the previously introduced *design for economic production,* where knowledge associated with the product's production phase is integrated into it's design-development phase. Other DFXs include *design for operation* and *design for maintenance,* where knowledge associated with the operational phase of a product is integrated into it's design-development phase. For products such as nuclear-powered submarines and offshore oil-production platforms, there is also a need to consider *design for disposal,* where knowledge associated with the safe and environmentally friendly decommissioning and disposal of such products is considered during design-development.

Attention has been drawn to the notion of a product system's life-cycle in order to highlight the link, not only between the phases of the product's life-cycle, but also the link between the product system and the company systems that are used to design-develop, produce, operate, and dispose of the product. During the life cycle of the warship, the *warship product system* passes through a range of *company systems,* including the company's design-development system and production system, each of which has it's own distinct life cycle.

2.4.2 The Importance of Understanding the Inter-Relationships between System Elements

Based on an understanding of system inter-relationships, a systems approach to the management of business processes can be created that promotes a logical and concise framework within which all participants subscribe to a set of common objectives. In this respect, the goal of rationality can be achieved if (*i*) we *understand* the system about which we are making decisions; and, (*ii*) we can accurately *estimate the outcome* that results from those decisions.

More specifically, Pugh emphasises the importance of understanding design-development inter-relationships in his concept of *Total Design* [Pugh 90]. Within this concept, *engineering rigour* is the use of discipline-dependent techniques and the specific application of such to design-develop a product is *partial design*. Total Design, on the other hand, recognises the inter-relationships between discipline dependent processes and activities, such that members of the PDO can see how their differing partial design contributions fit into the whole.

Whilst not specifically a systems approach, a number of researchers argue the need for a *systematic approach* to the design-development phase of a product. Pahl and Beitz, perhaps the strongest proponents of *systematic design-development* [Pahl 96], explain that systematic procedures merely try to steer the efforts of the PDO from unconscious into conscious and more purposeful paths. At first glance it would seem that a systematic approach would detract from the importance of experience, inventiveness and intuition. However, the opposite is true: "..by imposing a systematic approach, talented members of the PDO can focus their experience and intuition on the product itself, rather than on the processes associated with the product's design-development" [Pahl 96].

Clark and Fujimoto [Clark 91], based on their extensive research into the automobile industry summarise that, what seems to set apart outstanding engineering companies from the rest is the overall pattern of consistency in their total design-development system including organisational structure, technical skills, problem solving procedures, culture and strategy. This consistency and coherence lie not only in the broad principles and architecture of the system, but also in it's working level details. It is considered here that this consistency and coherence are the result of using a rational, systems approach.

Towards the end of Sub-Section 2.4.1, the notion of integrating the knowledge associated with all phases of the system's life cycle was introduced. Here it is stated that system integration begins with a systems approach and, in particular, a clear understanding of the inter-relationships between system elements.

Perhaps the strongest argument for the adoption of a systems approach to the management of business processes lies in the requirement to plan accurately the detailed make-up of the total elapsed time and resources required to complete design-development and production activities such that the intended outputs of each can be achieved on time and within budget.

For those engineering companies whose *Goal* [Goldratt 84] is to make a profit, in line with the observations of researchers such as Clark and Fujimoto [Clark 91], one component of a strategy for achieving this could be based on satisfying the two requirements detailed at the beginning of this section. That is, (1) The creation of a formalised and structured model of the constituent elements of a system and an understanding of their inter-relationships; and, (2) a set of stable processes with contained variability.

The research summarised in this thesis is based on the development of *a strategy for modelling the design-development phase of a product.* In this respect, the strategy as a whole goes a long way towards satisfying the previously stated first requirement.

In terms of the second requirement, the remainder of this chapter describes the notion of process stability and, in particular the achievement of process stability as well as the improvement of such by reducing process variability. These notions have the strongest impact on the fifth function of the proposed strategy. In order for a schedule of activities to be truly representative, the estimated durations of each activity need to be accurate. To this end, it is the aim of the next sub-section to explain that the accuracy of predicting process (activity) outputs such as process durations is higher when the process in question is stable and subject to contained variability.

2.4.3 The Importance of Stable Processes

A process uses *inputs, adds value* by processing these inputs, and produces *outputs.* The *parameters* chosen to describe the *outputs of a process* are those which can be used to (*i*)

assess the quality of the outputs; and, (*ii*) assess the productivity of the process in terms of it's efficiency in the use of resources.

Once chosen, the values of each output parameter can be measured and recorded, and their *statistical pattern* made clear through the plotting of histograms. Typical measured output parameters for production processes include critical dimensions (e.g. length, squareness, and circularity) as well as resource expenditure aligned to parametric measures of the output (e.g. man-hours per metre). Examples of measured output parameters for design-development processes include the number of design change orders as well as man-hours per drawing.

A process is *stable* and *under control* when the statistical pattern of the measured values of each output parameter is essentially Gaussian (*See* Figure 2.2) In such cases, probability distributions of the output parameter values can be used for predicting the output of the process.

The range of this statistical pattern can be large or small. If the range is large, measured output parameter values may occasionally fall outside the pre-set range of acceptable variability, as expressed by a positive and/or negative tolerance. In such cases, depending on the measured parameter, either rework is necessary, or an improvement in productivity is required. (*See* Figure 2.3).

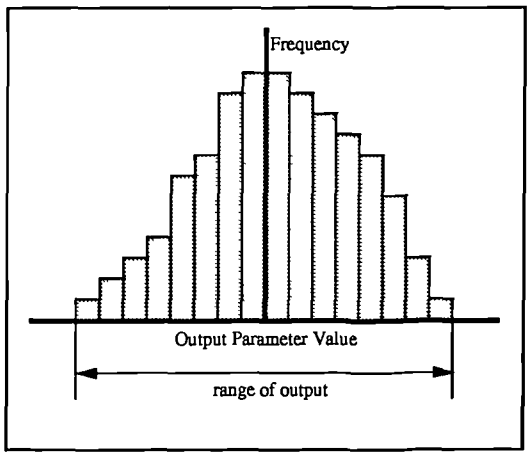


Figure 2.2: A stable process exhibits a Gaussian--type statistical pattern of output parameter values

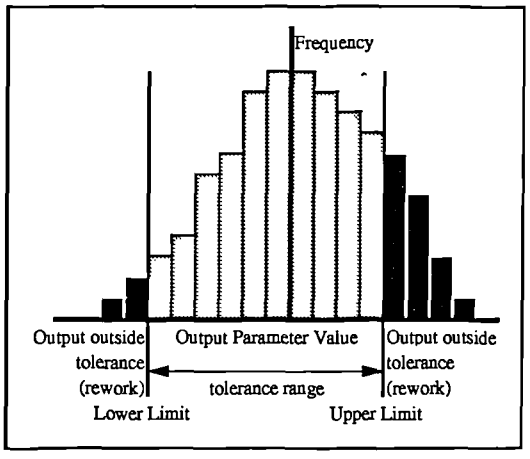


Figure 2.3: A stable process with some of it's output outside tolerance.

Rework is avoided and/or the process is efficient when the measured output parameter values consistently lie between the extremes of tolerance (*See* Figure 2.4). Furthermore, when the

range of output is very small, a mean value for each output parameter can be used for predicting the output of the process (*See* Figure 2.5).

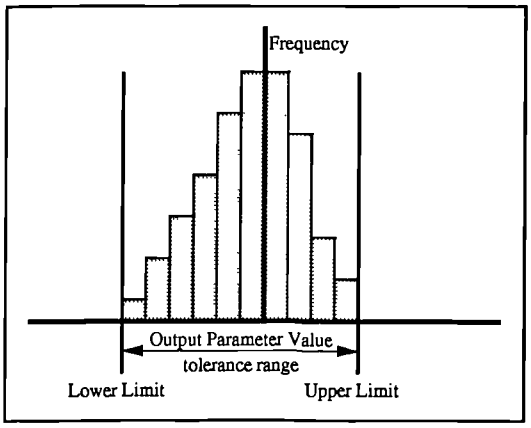


Figure 2.4: A stable process with output inside tolerance

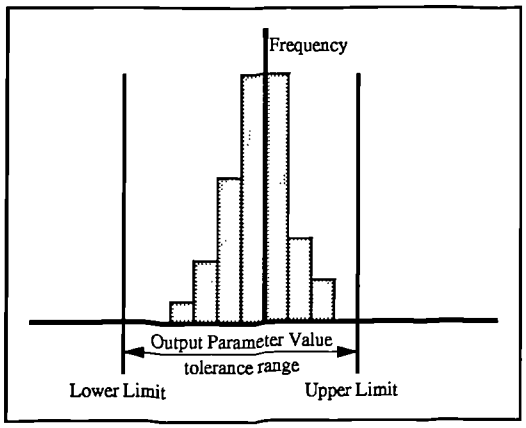


Figure 2.5: A stable process with minimal variability

A *process* is *unstable* and *out of control* when the statistical pattern of the measured values of each output parameter is unpatterned (*See* Figure 2.6). In such cases, output parameter values cannot be predicted with any confidence.

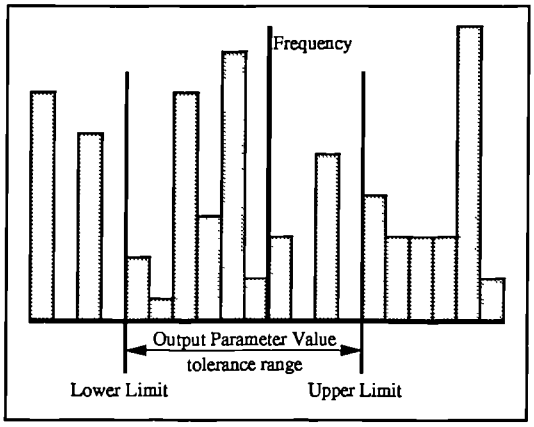


Figure 2.6: An unstable process - statistically un-patterned output parameter values

It may be observed from Figures 2.2-2.6 that, in effect, the process "speaks" to it's operators through the statistical pattern of it's output parameter values.

Planning is a key to the successful management of all business processes. In turn, effective planning is based on the capability of predicting, with confidence, the elapsed time and resources (man-time, consumables, etc.) required for a given process. Concentrating on processes and based on the discussion presented at the beginning of Section 2.4 as well as the

foregoing text, it is proposed that this capability is achieved by addressing two challenges, in turn;

Challenge 1: Achieve process stability. The benefit of meeting such a challenge relates to the fact that the output parameter values of stable processes such as process durations can be predicted using probability distributions. This helps to ensure that a schedule of activities based on such is truly representative and accurate.

Challenge 2: Reduce process variability so that output parameter values consistently lie within the pre-set range of acceptable tolerance. The benefit of meeting such a challenge relates to the fact that rework is avoided because, within such limits, the output is always deemed to be right first time. In an ideal world it would be possible to reduce variability down to zero such that a mean value for each output parameter could be used for the purposes of prediction. However, in the real world, reducing process variability down to zero is rarely practical or cost effective.

2.4.4 Achieving Process Stability and Reducing Process Variability

The output from a process varies due to *variation in the inputs to the process* as well as *variation within the process itself.* Therefore, in order to achieve process stability and reduce the variability of the measured output parameter values, it may be necessary to reduce the variability associated with the process inputs and that associated with the process itself.

The first step towards achieving process stability and reducing process variability is to identify those parameters associated with the inputs to the process and those associated with the process itself which, when subject to variability, have the strongest impact upon the variability of the measured output parameters. The identification of such parameters results from experience and experimentation.

The second step is to reduce the variability associated with these key input and process parameters. The reduction of such variability may be achieved by reducing the *variety* that is manifest in both the process inputs and within the process itself. Sub-Section 2.4.5 details a number of strategies for *reducing variety.*

Once a process is deemed to be stable, any tendency to become unstable can be identified by sampling values of the output parameters, and comparing the resulting measured statistical pattern of the output parameter values with the intended pattern. This notion of *process control*, as described by the *feedback loop* in Figure 2.7, can be extended to incorporate the notion of *continuous process improvement.*

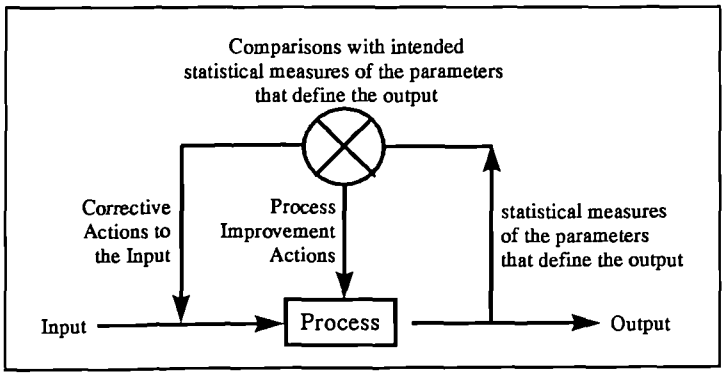


Figure 2.7: The control and improvement of a process

In this context, *Quality Circles* can be used to help process operators identify, evaluate and recommend *corrective actions to the input* and/or *process improvement actions.* This practice of continuous improvement focuses on the identification and reduction of errors and should be an integral part of each process.

2.4.5 Reducing Variety

Snaith [Snaith 95] recommends two strategies for reducing variety. The first is an *attenuating strategy*, which is used specifically to reduce the amount of variety that is inherent both within the inputs to a process and within the process itself. The second is an *amplifying strategy* used to manage the variety that remains. Typical attenuating strategies include:

- The standardisation of process inputs and process procedures. (*See* Sub-Section 2.4.6)

- Training management teams and process operators to enable them to understand and apply the standard process procedures effectively and consistently.

- Ensuring that changes to the standard process procedures are only permitted if they have been fully evaluated and then approved by the appropriate management teams.

- The maintenance of all equipment to the specification defined by the manufacturer. If equipment is not adequately maintained then, as a result of equipment wear and the associated increase in process variability, measured values of the output parameters may begin to fall outside the pre-set range of acceptable tolerance and, as a result, rework may be necessary. In such cases, the increased variability associated with the output is due to the increased variability of certain process parameters. If left unchecked, the statistical pattern of the output parameters will become unpatterned and the process will become unstable. In addition to regular maintenance, any *special causes* which arise from time to time, and which affect process stability, must be identified and eliminated immediately.

- Reducing the number of suppliers to a small number of carefully selected companies and establishing mutual supportive practices. In this way the best practices of a company can be promoted in the supply chain such that suppliers improve the stability and reduce the variability of their own processes.

Typical amplifying strategies include;

- Investment in the most appropriate facilities, equipment, techniques and enabling technology, as well as the recruitment of process operators with specific technical skills based on a clearly defined set of requirements derived from a clear understanding of the process.

- Joint ventures or strategic partnerships with carefully chosen companies.

2.4.6 Standardisation

Standardisation has strong implications for the performance of all business processes. A reduction in variety through a balanced policy of standardisation will help to reduce costs and improve product quality as a result of increased reliability and consistency [Flurscheim 77].

When standardisation is poorly managed, it becomes increasingly difficult to introduce innovation in the face of established practice. However, standards are not, in general, intended to be immutable. In this respect, standards provide the basis for normal operation and, if there is a case for using some other method, after full evaluation and approval by the appropriate management team, the new method can be incorporated as a new standard. In this way, the

best of both worlds is achieved: that is, arbitrary changes are not permitted, yet innovation is not stifled.

Ideally, standards are included in a company's technical quality assurance (QA) system, such as ISO 9000. This type of QA system documents *standard process procedures* and their *application.* ISO 9000 is aimed solely at preventing a loss in the quality of process outputs through the consistent application of standard process procedures. As such, ISO 9000 acts to maintain a status quo.

In order to improve product quality, proactive initiatives such as the application of *Total Quality Management (TQM) principles* may be necessary. For example, as previously stated, the use of *Quality Circles* can help teams of process operators to identify, evaluate, and recommend new and improved standards into the process as part of a programme of continuous improvement. Using ISO 9000, these new and improved standards can then be documented. In this respect, there should be fluidity to the creation, documentation, and application of standards.

Typical standardisation strategies include;

- The consistent application of *group technology (GT).* Processes can be organised on the basis of GT, whereby the outputs of each process are classified by the common problems associated with their processing. In group technology terms, similar, but not necessarily identical outputs should be produced in the same fashion. Such an approach encourages the development of a limited set of standardised process procedures and output measures. Kusiak and Park [Kusiak 90b] describe how the principles of GT can be applied to the outputs of design-development processes.

- Cluster the almost infinite number (*continuous function*) of inputs to a process into a limited number of logical groupings (*step function*) (*See* Figure 2.8).

- The development and application of *a checklist* can help reduce the variety of process inputs. Inputs that fail the check are simply ignored. Such checklists, as part of *a formalised decision support system,* can help management teams make consistent and logical decisions relating to specific scenarios. Examples include decision support systems

which help companies decide which enquiries to bid for as well as systems which help management teams decide whether to incorporate design change into the current product underway or wait until the next product update.

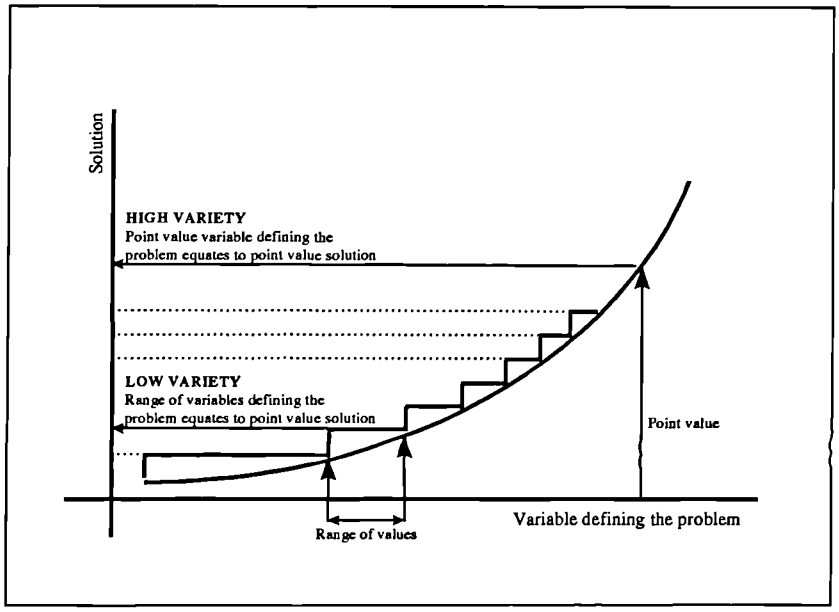


Figure 2.8: Limiting variety using a solution step function to replace a continuous function [Snaith 95]

- Full documentation and application of preferred process procedures and working practices ensuring a consistent approach. The use of such process procedures and working practices may avoid errors and the large variety that arises from the arbitrary use of unapproved procedures and working practices. As well as standardising what *should* constitute standard process procedures, it may be appropriate to create standards on what *should not* constitute standard procedures. These standards can be derived from past troubles and should be recorded for efficient retrieval in order to avoid their repetition.

- Recording of all decisions relating to the processing of inputs. Together, with reference to all relevant documents and drawings, the justification and logic behind all process-related decisions can be documented and, when required, communicated. This aims to avoid the application of arbitrary decisions being made in isolation.

- Develop a portfolio of fully evaluated modularised standard process outputs, covering a wide range of customer requirements, which can be re-used.

The decisions encountered within a company system are far reaching and relate to many issues. In order to make the most effective decisions, the information generated should be the *most appropriate* and the *most accurate* for the decision at hand.

Based on a systems approach, by understanding the inter-relationships between system elements, there is a much greater chance of generating the most appropriate information. Furthermore, if, at the decision point, this information is unknown, then it must be predicted. The more accurate the prediction, the more likely will be a successful outcome and, as stated previously, accurate prediction is greatly enhanced by the existence of stable processes whose output is subject to contained variability.

2.5 Chapter Summary

Chapter 2 introduces the notion that in order to gain competitive advantage, engineering companies should perhaps focus attention on the design-development phase of their products without losing focus on marketing and manufacturing performance. By doing so, companies have a much greater chance of reducing the time it takes to get their products to market, an objective which, in the majority of cases, is now viewed as a priority.

In this respect, the need for engineering companies to develop strategies that incorporate the principles of concurrent engineering (CE) is highlighted. However, before some of the most relevant notions, techniques, and enabling technologies associated with concurrent engineering are elaborated, attention is drawn to the potential benefits as well as the costs of adopting reduced time to market as an objective for gaining competitive advantage.

Chapter 2 also summarises a systems approach to the management of design-development and other business processes and, in particular, highlights some of the *invisibles* that are considered to be crucial to achieving the improved performance of any business process. Such invisibles relate specifically to the creation of a set of stable processes with contained variability.

In summary, it is considered that in addition to a systematic approach to studying and modelling a product's design-development phase, the application of CE supported by a systems approach to the management of design-development processes which are stable and whose output is subject to contained variability, offers some of the best opportunities for improving design-development performance.

3. Modelling the Design-Development Phase of a Product

3.1 Introduction

In Chapter 2, it was highlighted that the benefits associated with initiatives such as concurrent engineering will tend to be elusive in the absence of a systems approach to the management of design-development processes that are engineered to be stable and predictable in their use of resources. In this respect, because a model can be used to gain a better understanding of a system's behaviour, an effective modelling strategy should be at the very heart of a systems approach.

Previously in Sub-Section 2.4.1, it was described how a product, as a system, consists of life-cycle phases and how, during these phases it passes through a range of operational systems including an engineering company's design-development sub-system. In this context, the research contained in the remainder of this thesis is focussed on products, and specifically on the development of a strategy for modelling the design-development phase of a product.

By contesting some of the most common reasons quoted for the case against such modelling, Section 3.2 first presents a case for modelling a product's design-development phase. Section 3.3 then describes a *traditional strategy* that engineering companies tend to follow when faced with the task of modelling the production or design-development phase of a product.

Section 3.4 summarises some of shortcomings that may be encountered when traditional modelling strategies are applied to the design-development phase. Based in part on these shortcomings, Section 3.5 proposes a set of requirements for a modelling strategy specifically focussed on design-development. Section 3.6 introduces the proposed strategy for modelling a product's design-development phase whilst Section 3.7 summarises prior related research. Finally, Section 3.8 describes how the research behind the proposed strategy differs from prior research and also how the research contained in the thesis contributes to making significant practical advances to the subject.

3.2 A Case for Modelling the Design-Development Phase of a Product

Consider a number of statements that represent the significance of product design-development. "The Ford Motor Company estimate that even though product design-development accounts for only 5% of total product cost, 70% of this cost is influenced by design-development" [Boothroyd 88]. "It is believed that 40% of all product quality problems can be traced to poor design-development" [Dixon 88]. "As much as 80% of manufacturing productivity can be determined during design-development" [Suh 90].

Based on these observations, it is not unreasonable to assume that, since product design-development has such a significant impact on an engineering company's capability to compete then perhaps time and effort should be allocated towards it's improvement. As highlighted in the introduction to this thesis, one approach towards improving design-development is to recognise that it is often procedural and repeatable; therefore it can be modelled as one might model a manufacturing process that needs to be improved" [Smith 97].

To this end, the principal aim of the research contained in this thesis is to provide product development organisations (PDOs) with a *strategy* for *modelling* and *optimising sequences* and *schedules of design-development activities* so that they can be *planned* and *organised* in a more *effective manner* than before.

However, it is first necessary to examine a common misconception shared by some product development organisations (PDOs) that such modelling is of little value and is therefore unnecessary. The following text outlines four arguments that have been raised in discussions between the author and PDO members from a range of engineering companies (*See* Appendix I). Alongside each of these arguments is the author's counter-argument.

(1) An experienced manager does not need to create a formalised model

Sometimes, managers responsible for all or part of the design-development of a product believe that, because they understand design-development extremely well, or because the product in question is simple, it's design-development phase can be modelled mentally. This approach tends to circumvent the need for a formalised model that is documented and thus readily accessible to others.

For some products, managers may be justified in this belief and, as such, the proposed modelling strategy would not be appropriate. However, the design-development of complex engineered products such as cars, aircraft, ships or computers can involve co-ordinating hundreds or even thousands of individuals making a large number of design decisions over months or years. For such products, it is impossible to model the design-development phase mentally and a formalised model is perhaps the only way of ensuring that all of the required design activities are completed in the most efficient and effective manner [Eppinger 90].

(2) There is not enough time to create a model

According to Barclay *et al* [Barclay 93], lead-times associated with the design-development phase of new products reduced on average by 40% between 1987 and 1992. Whilst this rate of reduction has slowed over recent years, many engineering companies continue to invest in programmes and initiatives aimed at squeezing yet more lead-time from the design-development phase. As a result, PDOs are increasingly under pressure to begin design-work as early as possible.

Whilst few PDOs may be prepared to allocate time to create a planning-model, time and effort spent in such an exercise could save time and effort downstream. To clarify this notion, consider four major benefits provided by models: (*i*) they permit analysis and experimentation with complex situations to a degree that is impossible with the actual system; (*ii*) they provide economy in representation and inquiry; (*iii*) the amount of time spent on problem analysis can be significantly reduced; and, (*iv*) they focus attention on the most essential characteristics of the problem [Tersine 76]. Based on these benefits it would appear that saving time and effort is a central incentive behind the use of models and modelling.

In addition, in terms of saving time it may be possible for engineering companies involved in the design-development of *stereotype products*, such as certain types of ships, to develop a portfolio of *generic models* that can be easily modified for the design-development of each specific instance of such products.

(3) Because of iteration, design-development work cannot be modelled accurately

The design-development of products involves iteration. Iteration of certain activities arises because of their *mutual inter-dependence* on each other for input data. Initially, a group of activities have to be processed using *guesstimates* of the data they need as input. Once all of the activities have been completed, based on a comparison of the actual derived data against the initial guesstimate, if the initial guesstimate is not deemed to have been sufficiently accurate, then iteration will be necessary and certain activities will have to be re-processed.

Whilst traditional models such as *activity network diagrams* [Wiest 77] represent activities and their precedence relationships, they do not explicitly represent the dependence of activities on one another for data (*See* Section 3.4 for an explanation of the difference between data-dependencies and precedence relationships). Without the ability to represent even this basic characteristic of a product's design-development phase, such models certainly cannot represent that which implies iteration, that is, the mutual inter-dependence of activities on one another for data.

Consequently, since traditional models cannot represent that which implies iteration, this important and influential characteristic tends to be ignored whenever a product's design-development phase has been modelled. When a model omits significant and influential characteristics, then important system behaviour remains unexplained. As a result, an over-simplified model tends to represent a notional situation that does not reflect the real state of affairs. In addition, individual decisions are increasingly at cross-purposes simply because system operators, for example members of a PDO, are not fully aware of the implications that their actions have on specific elements that have not been modelled adequately.

It would appear that the inadequacies of these types of traditional models and planning techniques (*See* Appendix II) have unwittingly strengthened the argument in some quarters that modelling a product's design-development phase is a pointless, unnecessary and perhaps even impossible exercise. It is suggested that this state of affairs can only be countered by models that can accurately represent data-dependencies between activities and, more specifically, can accurately represent that which implies iteration, that is, the mutual inter-dependence of activities on one another for data.

(4) Innovation is stifled when formality is introduced

The question commonly asked is, "How can a product's design-development phase be modelled when it is so utterly dependent on *creativity*, *inspiration* and *old-fashioned luck*?" [Drucker 85]. Drucker argues that the paradox implied by this question is apparent only and therefore not real. "Most of what happens in successful innovations is not the happy occurrence of a blinding flash of insight but, rather, the careful implementation of an unspectacular yet systematic management discipline. At the heart of that discipline lies a knowledge of where to look for innovation opportunities, how to identify them, and finally, how to implement them".

In this respect, innovation is a strategic issue and perhaps should be addressed continuously by a research group within the PDO. Opportunities for incorporating innovation into new products should be sought, and a careful analysis of the various kinds of knowledge needed to make an innovation possible should be carried out. Ideally, such research should be carried out "off-line" and should only be integrated into new product design-development once the knowledge has been tested and validated fully.

Based on the view of a product as "a unique aggregation of multilevel standard physical objects" [Snaith 99], innovation can be incorporated into a product at any level in the hierarchy of sub-systems that make up the product. For example, at the highest level of a warship's hierarchy of sub-systems, innovation may be reflected in an outline product schema based on a hull sub-system which is made up of three hulls rather than a single hull. Innovation at a lower level may be reflected in the specific choice of an on-board sub-system. For example, an azipod rather than a propeller may be selected as the means of achieving the functional / operational requirements of the "propulsor sub-system". At the lowest level, innovation can be incorporated into equipment that is usually developed by equipment suppliers. An example of innovation at this level includes Rushton Diesel's "pipeless engine".

Drucker considers that innovation is most effective when it's impact on the overall product is small and it is focussed [Drucker 85]. In the same vein, Milner, based on his experiences with the Polaris submarine programme, advises against "across-the-board" innovation in every area of a product [Milner 71]. Perhaps because of the risk involved, few products are innovative at every level in the hierarchy of sub-systems that make up

the product. Therefore, for the vast majority of new products, whilst innovation may be incorporated into one or a small number of a product's sub-systems, the majority of sub-systems will tend to be based on a large number of standard physical objects.

As part of the process of testing and validating innovative product technology, the PDO should perhaps focus on how such technology will be incorporated into a new product. Most specifically, the PDO should define how the design-development work associated with the inclusion of innovative product technology is to be integrated with the work that defines the design-development of the rest of the product. Based on such an approach, a generic model of the design-development phase of a product can be modified to reflect any changes that result due to the inclusion of innovation.

The case for modelling is strengthened when the product is complex. Based on the definition of a system quoted previously in Sub-Section 2.4.1, *complexity* is the difficulty involved in using the inter-relationships between the elements of a system to infer the behaviour of the whole system. Phrased another way, complexity is how much more the whole is than just the sum of the elements.

The functional / operational requirements of a product are satisfied by a product's functional sub-systems. In order to satisfy more functional and operational requirements, more sub-systems may be required. As the number of sub-systems increase so does the number of sub-system inter-relationships. In turn, as the number of inter-relationships increases, complexity increases almost exponentially and, consequently, the effort required to optimise and manage each phase in the life cycle of the system tends to increase.

In view of this, it is perhaps necessary to improve the capability of analysing increasingly complex product systems and their life-cycle phases. Computers have made a vital contribution to improving this capability, but without the rational and logical understanding afforded by a formalised system model, system operators are unlikely to understand the implications of the results of computer calculations well enough to use them effectively.

In summary, the case for modelling the activities of the design-development phase of a product system begins with the notion that by creating a model it is possible to gain a better understanding of this important and influential phase of a product's lifecycle. With a better

understanding, there is an increased probability that initiatives, aimed at improving the efficiency and effectiveness of design-development will be focussed where they have the most constructive impact. Once PDOs accept this notion, it is necessary for them to have a strategy for developing an *accurate*, *flexible* and *practical* model that can be used as the basis for developing near optimal sequences and schedules of design-development activities.

3.3 How Engineering Companies Model the Lifecycle Phases of a Product

A number of proven modelling and planning techniques may come to mind when it is recalled that part of the principal aim of this research is to provide PDOs with a strategy for modelling and optimising sequences and schedules of activities. Such techniques include *network planning*, *critical path analysis (CPA)* and the *programme evaluation and review technique (PERT)*.

However, whilst the techniques and basic concepts of planning may be necessary, they are not in themselves sufficient for the effective planning, organisation, and indeed the management of a product's life-cycle phases. It is in fact the notion of an encompassing *strategy* that is actually more important than the specific group of techniques used [Harrison 92].

Based on discussions with designers, planners and project-managers, it has been observed by the author that, whilst techniques may differ between companies, when faced with the task of planning and organising a product's life-cycle phases, most companies tend to adopt variations of the *traditional three-function modelling strategy* (*See* Figure 3.1).

Whilst the strategy illustrated in Figure 3.1 may be used more frequently to model a product's production phase, in the absence of an alternative, engineering companies tend to adopt the same strategy when faced with the task of planning and organising a product's design-development phase.

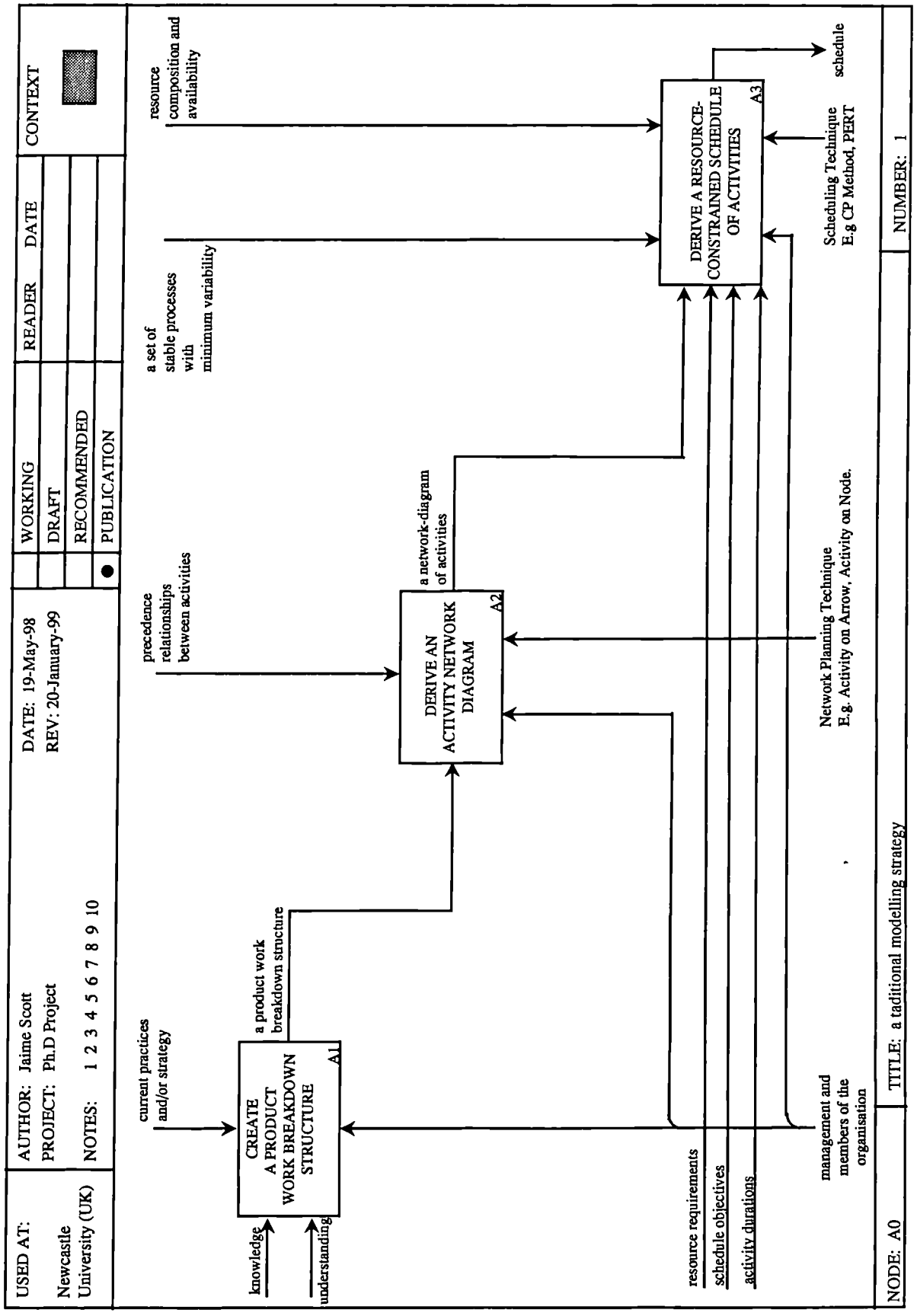


Figure 3.1: The traditional modelling strategy for planning and organising a product's life-cycle phases

Before summarising this traditional three-function strategy, it should be highlighted that, as part of a *total project management solution,* there are additional actions to be undertaken. According to Weiss and Wysocki [Weiss 92] such actions include "state the problem", "identify project goals", "define management style", etc. However, based on the specific aim detailed at the beginning of this section, it is considered that these actions are outside the focus of the research contained in this thesis. That said, the traditional three-function strategy of Figure 3.1 may be viewed as forming the core of Weiss and Wysocki's project management strategy.

As the name implies, the traditional three-function modelling strategy used to plan and organise a product's life-cycle phases consists of three functions.

Function 1: Create a Work Breakdown Structure (WBS)

The term *work breakdown structure (WBS)* is simply a name for *a sub-division* of an *engineering project* based on *end items.* Here, an engineering project encompasses all of the activities that make up a product's life cycle phase, whether that is the product's design-development or it's production phase. The WBS is constructed by progressively dividing the project into it's main identifiable elements in a logical manner. This process is repeated until the project has been divided into intermediate and final outputs aligned with the corresponding work assignments for individuals and/or groups.

Function 2: Create an Activity Network Diagram

Based on the activities represented in the WBS, it is possible to construct a sequence in which the activities will be processed. An *activity network diagram* is used to represent this sequence. The first step in constructing this network diagram is to determine, for each activity, those other activities that must be completed before the activity in question may begin. In effect, this step is used to determine each activity's *preceding activities* or, in other words, the step is used to determine each activity's *precedence relationships.*

There are a number of ways of presenting network diagrams, the two most popular being the *activity-on-arrow (AOA) method* and the *activity-on-node (AON)* method. The two methods represent activities and their precedence relationships in slightly different ways and it would appear that the choice of method depends largely on user preference.

In addition, there are a number of different types of precedence relationships including *finish-to-start, start-to-start, start-to-finish* and *finish-to-finish* (*See* Figure 3.2). However, irrespective of which method is used, or which types are chosen to represent each activity's precedence relationships, a network diagram can be used as the basis for time-based analysis and the preparation of a schedule of activities. The most basic of such analyses is to derive the *start time* and *finish time* of each activity based on knowledge of each activity's duration.

Function 3: Derive a Resource-Constrained Schedule

By assigning a duration to each activity, it's start time and finish time can be derived. By constraining the overall end-date of the project, critical path analysis (CPA) can be applied to derive the *earliest* and *latest start times*, the *earliest* and *latest finish times* of each activity as well as each activity's *float*. For each activity, float indicates the time that an activity *could be* delayed without causing the overall end-date of the project to be extended.

Resource-constrained scheduling can be defined as the planning of work where the resources *required* to process the work are matched with the resources that are *available*. When resource availability is limited, the scheduling of work, that is the *actual start time* and *finish time* of each activity is constrained to time periods during which resource requirements are equal to, or less than those available.

In the first instance CPA assumes infinite resource capacity and, as a result, based on knowledge of each activity's resource requirements and overall resource capacity, CPA-derived start times sometimes have to adjusted to avoid resource requirements exceeding those available.

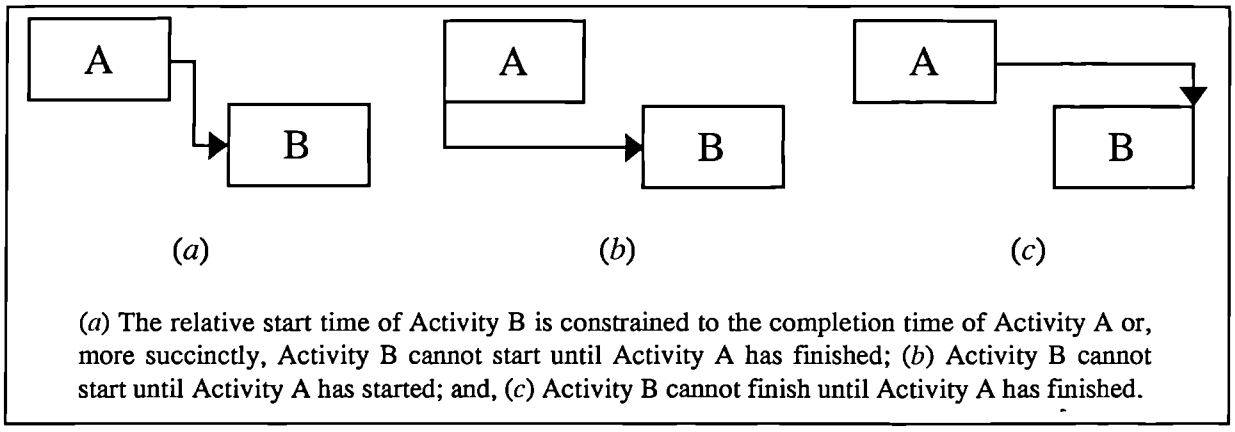
3.4 Drawbacks of the Traditional Modelling Strategy when Applied to the Design-Development Phase of a Product

The traditional, widely used strategy of Figure 3.1 have been proven by engineering companies to be effective when used to plan and organise the work that constitutes a product's production phase [Maylor 96]. However, it is considered to have a number of significant drawbacks at the *overall strategic level* as well as at the more *detailed functional level* when applied to a product's design-development phase.

Firstly, at the strategic level, because a network diagram is derived straight from a WBS without any intermediate analysis, the traditional strategy does not incorporate any functionality that would allow a user to investigate and analyse alternative sequences and schedules of activities. As such, opportunities for improvement and optimisation may be missed.

Again at the strategic level, whilst activity network diagrams represent activities and their *precedence relationships*, they do not explicitly represent the *dependence* of activities on one another for data. In order to understand the full implication of this observation, it is perhaps necessary to explain the difference between precedence relationships and data-dependencies.

Precedence relationships between activities define *relative timing constraints* and originate as a result of an *enabling condition* (*See* below for examples of enabling conditions). Three examples of precedence relationships are illustrated in Figure 3.2.



(*a*) The relative start time of Activity B is constrained to the completion time of Activity A or, more succinctly, Activity B cannot start until Activity A has finished; (*b*) Activity B cannot start until Activity A has started; and, (*c*) Activity B cannot finish until Activity A has finished.

Figure 3.2: Precedence Relationships

On the other hand, data-dependencies between activities *do not* define relative timing constraints and simply represent the specific enabling condition that there is a passage of data. In this respect, a data-dependency is perhaps best described as a *cause* of precedence, that is, the dependency of one activity on another for data results in a precedence relationship. However, a precedence relationship does not necessarily imply a data-dependency because the precedence relationship may not necessarily originate as a result of a data-dependency.

In production, precedence relationships originate as a result of the enabling conditions that certain production activities must precede others. For example, a sub-assembly cannot be assembled until it's discrete piece-parts have been manufactured. However, the sequencing of

design-development activities is not as rigidly fixed such that precedence relationships between design-development activities originate primarily as a result of data-dependence.

Ultimately, an activity network diagram of activities and precedence relationships is required for the purpose of scheduling. However, without the ability to explicitly represent data-dependencies, a network diagram derived straight from a WBS without any intermediate analysis or interpretation tends to be created in such a way that data-dependency and, perhaps more importantly, data interdependency are ignored.

When a model ignores significant and influential characteristics such as data-dependency and the iteration implied by inter-dependency, then important system behaviour remains unexplained and an over-simplified model, which does not reflect the real state of affairs results. In effect, because the traditional three-function strategy does not include the functionality to explicitly represent, analyse or interpret data-dependencies between activities and, more specifically because it cannot accurately represent the inter-dependency that implies iteration, it may be inappropriate as a strategy for modelling design-development.

At the more detailed functional level, it is useful to consider the difference between a product's production phase and it's design-development phase. Products in general, and complex products such as a warship in particular, can be defined in terms of two different, yet complementary *product definitions*. The first, which defines the product "as it will be when completed" is referred to as the *design-definition*. The second, which defines the product "as it will be manufactured and assembled", is referred to as the *production-definition*. A product's design-definition is essentially that of a *hierarchy of functional sub-systems*. On the other hand, it's production-definition is that of a *hierarchy of discrete piece-parts, minor-assemblies, sub-assemblies, assemblies, equipment modules and connectives*.

Perhaps because it is more common to use the traditional strategy to model a product's production phase, the concept of a WBS has evolved in such a way that users tends to focus their efforts on modelling activities based on end items. For example, a typical production-work breakdown structure is based on a hierarchy of activities related to the manufacture and assembly of physical and tangible end items such as discrete piece-parts, minor-assemblies, sub-assemblies, assemblies, equipment modules and connectives. However, the end items of a product's design-development phase are less tangible in physical terms. Consequently, it is

suggested that a WBS focussed solely on end items may not be appropriate for modelling the activities that constitute a product's design-development phase.

Again, at the functional level, there are issues specifically related to the resource-constrained scheduling of design-development activities that tend not to be catered for by the techniques adopted as part of traditional strategies. In the latter, based on the results of critical path analysis, resource-constrained scheduling is usually carried out by moving the start date of non-critical activities to remove excessive peaks of resource requirement and thus level requirements below capacity. This technique, known as *resource levelling* cannot always level requirements below the available capacity and, as such, overloads can still occur.

In the production phase, the resources applied to an activity tend to be groups of tradesmen; for example 10 welders may be required to complete a structural assembly. For the cases where overload occurs, the planner can usually make arrangements to overcome the problem of an over-load of required resources by allowing overtime or by assigning more welders from other parts of the production facility. However, in the design-development phase, the major resources applied to an activity tend to be individual members of the PDO. Such resources are scarce and consequently in the case of overload, it is less likely that a planner would be able to overcome the problem by assigning overtime or by assigning more designers. In this respect, the techniques prevalent in traditional modelling strategies, used in most part to derive resource-constrained schedules of production activities are usually inappropriate for deriving resource-constrained schedules of design-development activities.

3.5 The Requirements of a Modelling Strategy for Planning and Organising the Design-Development Phase of a Product

Based on discussions at a range of engineering companies (*See* Appendix I) and on the previously mentioned drawbacks of traditional strategies, it is suggested that a modelling strategy for planning and organising the design-development phase of a product should satisfy a number of *representational* and *analytical* requirements.

Ultimately, a PDO needs to design-develop products that are superior to those of it's competitors in terms of *quality*, *cost* and *time-to-market*. Therefore, the expanded aim of the research contained in this thesis is to develop a strategy for creating a model that can be analysed in ways that enable a PDO to (*i*) gain a better understanding of the work involved in

the design-development phase of a product; (*ii*) compare the effectiveness of sub-objectives and sub-strategies for planning and organising design-development activities; and ultimately, (*iii*) create a near optimal, resource-constrained schedule of design-development activities.

In order to gain a better understanding of the design-development phase of a product it is suggested that the models, upon which the strategy is based, should meet a number of *representational requirements*. Ultimately, when studying a product's design-development phase, the essential system elements to be represented by a model are the *activities defined by a product design-work breakdown structure*. These activities are inter-related by their dependence on each other for data and the resulting precedence relationships. Consequently, the essential inter-relationships to be represented by a model are the *data-dependencies* between activities and their *precedence relationships*. Additionally, in order to cater for iteration, the model should be capable of representing those instances where there is a *mutual inter-dependence of activities* on one another for data.

In order to use a model to analyse and compare a range of sub-objectives, it is suggested that it meet a number of *analytical requirements*: the derived model should be both *normative* and *quantitative* to enable the mathematical-based analysis and optimisation of a range of objectives. Furthermore, because *accumulated time* influences cost, and *elapsed time* influences both cost and time-to-market, the model should also be *dynamic*, that is it should be capable of using and deriving time-based information.

Finally, in terms of basic functionality, it is suggested that a modelling strategy for planning and scheduling a product's design-development phase should encapsulate techniques focussed on issues specifically related to design-development. Most relevant, are techniques for developing; (*i*) a product design-work breakdown structure (PDWBS); (*ii*) a near optimal sequence of activities based on data-dependencies; (*iii*) a network diagram that has been derived based on data-dependency; and, (*iv*) a near optimal resource-constrained schedule of design-development activities.

Based on these requirements, Appendix II includes a review of contemporary modelling techniques and concludes that two of the most appropriate models for representing the design-development phase of a product are the design structure matrix (DSM) system and activity network diagrams.

3.6 The Proposed Strategy for Modelling the Design-Development Phase of a Product

The following text proposes a modelling strategy for planning and organising a product's design-development phase. It has been developed in order to overcome the drawbacks of traditional strategies and encompass the proposed requirements as detailed previously in Sections 3.5 and 3.6 respectively. The strategy illustrated in Figure 3.3 varies in a number of ways from the traditional strategy of Figure 3.1.

Firstly, the proposed strategy introduces a new approach to creating a WBS of design-development activities namely the *product design-work breakdown structure.* Secondly, the proposed strategy introduces two additional functions that fit between those of creating a WBS and creating an activity network diagram. These two new functions, namely "*model the activities and their data-dependencies*" and "*derive a near optimal sequence of activities*" allow the user to model, investigate and analyse alternative sequences of activities at a level of detail not previously addressed by traditional strategies.

More specifically, the function of modelling activities and their data-dependencies allows the user to model at a level of detail where the inter-dependence of activities on one another for data, and the iteration that is implied by such, can be represented and it's effect interpreted. Based on this model, the function of deriving a near optimal sequence of activities allows the user to investigate, analyse and optimise alternative sequences of activities based on objectives such as *minimising iteration* and *maximising concurrency.*

Thirdly, the proposed strategy expands the function of deriving an activity network diagram by introducing a new procedure for interpreting and resolving any iteration that is implied by inter-dependant activities. The result is an activity network diagram that represents a truer definition of the product's design-development phase.

Finally, the proposed strategy expands the function of deriving a resource-constrained schedule of design-development activities by introducing a new scheduling technique that is focussed specifically on some of the issues that are peculiar to design-development. These issues relate primarily to the fact that in design-development more than production, resources are considered to be scarce.

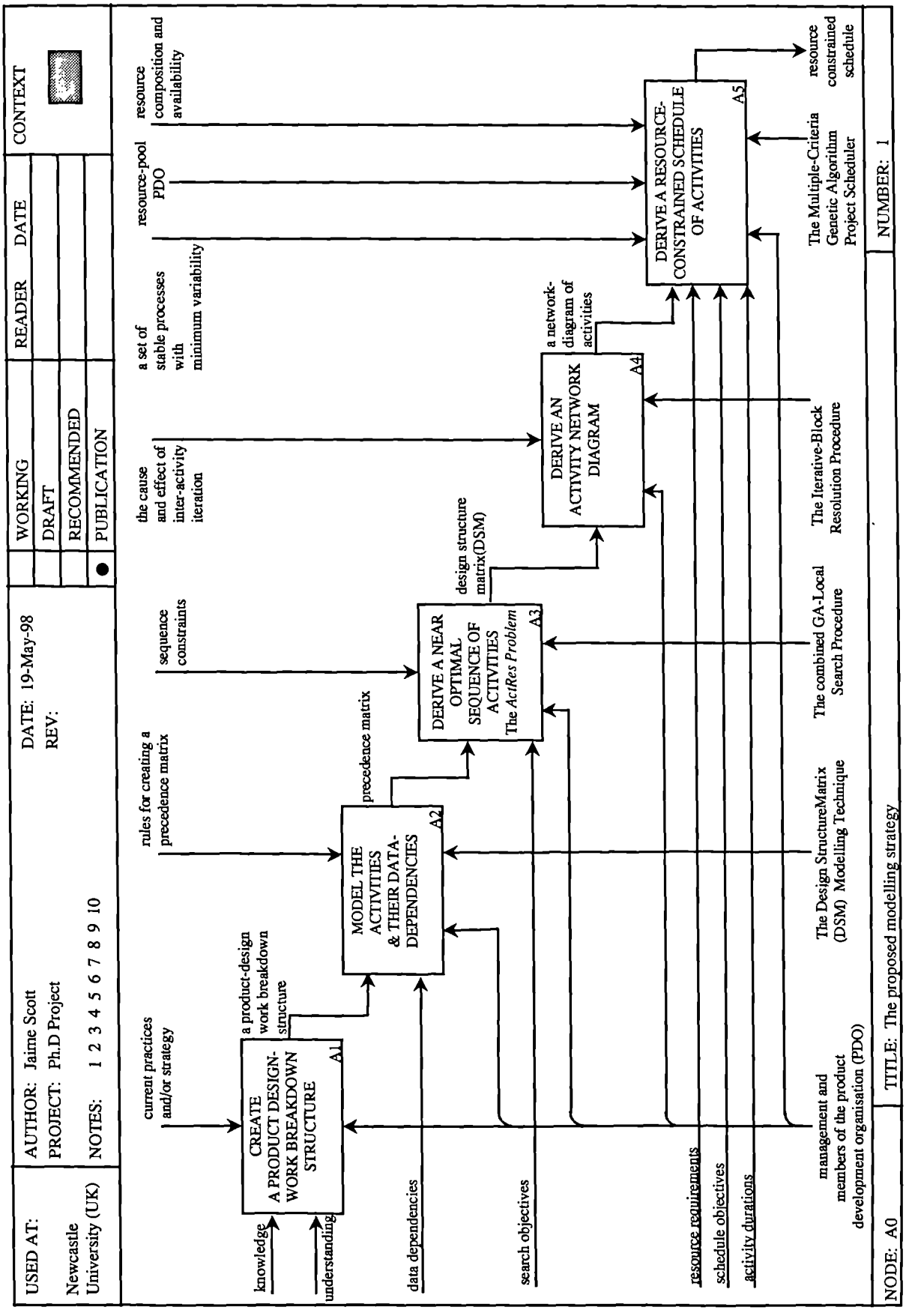


Figure 3.3: The proposed strategy for modelling the design-development phase of a product

In summary, the proposed strategy builds on the traditional strategy of Figure 3.1 by introducing additional functions and new techniques. It is considered that the two new functions namely "*model the activities and their data-dependencies*" and "*derive a near optimal sequence of activities*" form the core of the new strategy. This is because the ability to model at the most fundamental of levels enables PDOs to gain a better understanding of a product's design-development phase from the very outset.

In addition, such a model enables the user to investigate and optimise new sequences of activities. At this point, some readers may argue that since activities are sequenced as part of the function of deriving a schedule, a function that derives near optimal sequences separately duplicates the function of deriving a schedule. However, before a schedule can be derived, an activity network diagram of activities and precedence relationships is required. By sequencing activities and data-dependencies based on specific objectives as an earlier and separate function to scheduling, the precedence relationships that form part of a network diagram can be derived based on near optimal data-flows.

Having introduced how the proposed strategy varies from the more traditional approach, the proposed modelling strategy can now be introduced. Figure 3.3 illustrates the proposed strategy as an $IDEF_0$ model, and the sub-sections that follow address, in turn, each of the five functions (function boxes A1-A5 of Figure 3.3) which describe the strategy. Using the standard terminology of $IDEF_0$ (*See* Appendix II) each of the five functions can be described according to a set of (*i*) controls; (*ii*) mechanisms; (*iii*) inputs; and, (*iv*) outputs. Throughout the thesis a case study, based on *the design-development of the pre-contract design-definition of a warship* (*See* Appendix I) is used to demonstrate how the proposed modelling strategy may be applied.

3.6.1 Creating a Product Design-Work Breakdown Structure

"Create a product design-work breakdown structure" represents the first function of the *strategy for modelling the design-development phase of a product* and is represented in Figure 3.3 as function box A1.

The decomposition of design-work, to create a *product design-work breakdown structure*, is the cornerstone of an effective strategy for modelling the activities of a product's design-

development phase. When faced with a complex problem, the first step towards obtaining a solution is to decompose the problem into smaller, more manageable problems and to identify the corresponding work required in solving them. Based on this approach, a product design-work breakdown structure enables the PDO to decompose the design-development phase of a complex product into *a set of activities* that represents the design-development of the functional sub-systems which, together, constitute the product's design-definition.

The controls that govern this function are either (*i*) the company's existing product design-work breakdown structure; or alternatively, (*ii*) a set of guidelines for creating a product design-work breakdown structure. Based on the process of design-development described in Appendix I, the guidelines below have been followed to develop the PDWBS of Table 3.1.

These guidelines have been developed through discussions with designers, planners and project-managers at a range of engineering companies (*See* Appendix I) and by amalgamating the work of researchers including Austin *et al* [Austin 96] and Lewis *et al* [Lewis 98].

First decompose the *product system* into a hierarchy of constituent sub-systems (*See* Appendix I). Secondly, for each system and sub-system, define the design-work that relates to the design-development of each system's (*i*) architecture; (*ii*) full-scale 3-D geometry; and, (*iii*) functional / operational attributes. Thirdly, decompose this design work progressively into discrete activities. Finally, define each activity in terms of;

(1) the *form of presentation* and the *information content* of the *outputs* derived from each activity;

(2) the *form of presentation* and *information content* of the *inputs* to be processed, including inputs that are *dependent* on the product, and those that are *independent* of the product;

(3) *standardised design-development procedures* to be used to process the inputs and thus establish the outputs. These procedures can be formally recorded electronically in the *User Libraries* of a 3-D CAD/CAE/CAM System (*See* Sub-Section 2.3.1);

No.	Design-Development Activities (*outputs in italics*)	Functional / Operational Requirements			
		Float	*Move*	*Fight*	*Support Crew*
Activities relating to the ship's architecture					
1	develop *general arrangement*				
Activities relating to the full-scale 3-D geometry of the ship					
2	develop *hull form*				
3	determine *principal dimensions*				
Activities relating to the determination of intended principal functional/operational attributes					
4	determine *endurance*				
5	determine *weight*				
6	determine *stability*				
7	determine *propulsion power*				
8	determine *electrical loadings*				
9	undertake *magnetic analysis*				
10	undertake *structural assessment*				
11	determine *ship motions*				
12	determine *cost*				
Activities relating to the design-development of sub-system schema					
13	develop *hull structural sub-system schema*	●			
14	develop *paint & cathodic protection sub-system schema*	●			
15	develop *NBCD & damage control sub-system schema*	●			
16	develop *ballast sub-system schema*	●			
17	develop *boats, davits & handling sub-system schema*	●			
18	develop *bilge, salvage and sullage sub-system schema*	●			
19	develop *prime-mover sub-system schema*		●		
20	develop *transmission sub-system schema*		●		
21	develop *propulsor sub-system schema*		●		
22	develop *uptakes & downtakes sub-system schema*		●		
23	develop *fuel oil sub-system schema*		●		
24	develop *lub.oil sub-system schema*		●		
25	develop *starters sub-system schema*		●		
26	develop *electrical generation sub-system schema*		●		
27	develop *electrical distribution sub-system schema*		●		
28	develop *electrical conversion sub-system schema*		●		
29	develop *navigation sub-system schema*		●		
30	develop *steering sub-system schema*		●		
31	develop *motion control sub-system schema*		●		
32	develop *mooring sub-system schema*		●		
33	develop *communication sub-system schema*			●	
34	develop *fire-control sub-system schema*			●	
35	develop *gyro sub-system schema*			●	
36	develop *surveillance radar sub-system schema*			●	
37	develop *SSM sub-system schema*			●	
38	develop *guns sub-system schema*			●	
39	develop *SAM sub-system schema*			●	
40	develop *helicopter sub-system schema*			●	
41	develop *sonar sub-system schema*			●	
42	develop *degausing sub-system schema*			●	
43	develop *HVAC sub-system schema*				●
44	develop *fresh water sub-system schema*				●
45	develop *sewage sub-system schema*				●
46	develop *refrigeration sub-system schema*				●
47	develop *galley sub-system schema*				●
48	develop *laundry sub-system schema*				●
49	develop *hospital sub-system schema*				●
50	develop *accommodation sub-system schema*				●
51	develop *fire detection & control sub-system schema*				●

Note:
Activities 13-51 describe the design-development of each of the warship's sub-system schema at the highest level of description. At the next level of description each of these activities could be broken down into:
(i) Activities relating to the design-development of the sub-systems architecture
(ii) Activities relating to the design-development of the sub-systems full-scale 3-D geometry
(iii) Activities relating to the design-development of the sub-systems attributes

Table 3.1: A pre-contract product design-work breakdown structure for a warship.

(4) the *appropriate knowledge and experience* which is to be used in order to ensure that production, operational, and maintenance issues are explicitly reflected in the design-definition from the very outset. Again, typically as decision-support systems, *User Libraries* can be used to record this information;

(5) the *technical uncertainties* that remain and, hence must be addressed and resolved at a later design-development stage.

Note the introduction of the term "stage" in point 5, above. A significant part of the process of creating a design-definition constitutes *space management* in a very practical and realistic manner. Furthermore, it enables the PDO to create the design-definition using a number of clearly defined stages, such that the output from each stage links the design definition to a "level of detail".

For example, the pre-contract design-development of a warship, to produce a schematic design-definition, represents the *first stage* of it's design-development phase. Ultimately, prior to the award of contract, the schematic design-definition will be developed and defined in more detail as the customer's requirements are refined and modified through several rounds of tendering.

By linking an activity to a stage in this manner, the PDO, using the same product design-work breakdown structure, can define an activity's inputs, outputs, procedures, experience and technical uncertainties with respect to a specific stage. Based on this staged approach, delays outside the control of the PDO can be avoided, and design-development lead times can be reduced significantly. In a *non-staged approach*, these delays occur when the PDO has to wait for suppliers to provide "complete information" that includes detail that isn't required during preliminary stages.

Based on a *staged approach*, the activities in the product design-work breakdown structure will be processed more than once. Each occurrence of an activity implies differences in the form and content of an activity's inputs and outputs. In turn, this can affect the choice of the procedures and experience that are to be used to process the inputs into outputs. In addition, during each stage, more information is added to the design-definition and hence, the technical uncertainties associated with each stage are reduced between successive stages.

These changes in inputs, outputs, procedures, experience, and uncertainties between stages should ideally be considered prior to the creation of a plan of design-development activities. This is because, whilst the precedence relationships between activities may remain unchanged between stages, the processing of different inputs using different procedures, can affect the planned duration of an activity and it's specific resource requirements.

The inputs to the function of developing a PDWBS are (*i*) a clear understanding of the design-development of products; and, most specifically, (*ii*) knowledge of the design-work to be modelled.

The mechanisms of this function are those members of the PDO who are responsible for processing design-development activities. By involving members of the PDO during planning, there is a much higher probability that they will "come on board" and take "ownership" of the resulting model. Ultimately, all decisions should be reviewed and agreed with the appropriate management teams.

Finally, the output from this function is a breakdown of design-work into a set of activities, each of which can be defined in terms of a set of (*i*) data-inputs (*ii*) data-outputs; (*iii*) standardised design-development procedures; (*iv*) knowledge and experience; and, (*v*) technical uncertainties. However, whilst the definition of activities in this manner is desirable, it is not mandatory. Because of the limits on time and resource, the product design-work breakdown structure for the warship case study has not been defined in this detailed manner. Instead, Table 3.1, illustrated previously only lists a set of design-development activities without any further definition of their inputs, outputs, procedures, knowledge, experience or uncertainties.

3.6.2 Modelling the Design-Development Activities and their Data-Dependencies

"Model the design-development activities and their data-dependencies" represents the second function of the *strategy for modelling the design-development phase of a product* and is represented in Figure 3.3 as function box A2.

By decomposing design-work, a product design-work breakdown structure of design-development activities can be created from the *top-down*. Based on such a product design-

work breakdown structure, the four subsequent strategic functions can then be used to describe how a schedule of activities can be derived from the *bottom-up*.

Previously, in Section 3.5, the requirements of a design-development phase model were addressed. Based on a review of contemporary modelling techniques summarised in Appendix II, it has been concluded that, whilst contemporary modelling techniques may satisfy one or more of these requirements, *the design structure matrix (DSM) system* satisfies most of the requirements. As a result, whilst relevant members of the PDO act as one of the mechanisms of this function, the design structure matrix (DSM) system has been adopted as the principal mechanism.

The inputs to this function include (*i*) the set of design-development activities represented by the product design-work breakdown structure; and, (*ii*) their data-dependencies.

Ideally, by defining activities in terms of their specific data-inputs and outputs, the data-dependencies between activities can be determined simply by mapping the outputs from each activity to those activities which need such data as input. For example, suppose Activity A produces data-output α. If Activity B needs data-input α, then there exists a data-dependency between Activities A and B, that is, Activity B depends on Activity A.

A certain PDO, involved in the design-development of offshore oil and gas production platforms, has in fact developed a database application which, based on a set of activities defined in terms of their standardised data-inputs and outputs, automatically establishes the data-dependencies that exist between activities.

As reflected by the warship case study, when the specific data-inputs and outputs of each activity are not defined, the data-dependencies between activities need to be established by questioning relevant members of the PDO. For each activity, members are questioned as to (*i*) which other activities directly pass data to the activity in question; and, (*ii*) the strength of each data-dependency.

The controls that govern this function are the two rules that need to be followed when compiling each activity's data-dependencies. The first rule is that the dependencies must be based on the four-level dependency scheme described in Appendix II.

The second, and perhaps most important rule is that only *direct* data-dependencies should be represented. This rule is best explained by way of an example. Consider the activities in Table 3.1. Whilst the majority of the activities, numbered 13-51, *indirectly* pass data relating to their electrical loadings, to Activity 26: "develop electrical generation sub-system schema", they first *directly* pass this data to Activity 8: "determine electrical loadings". Therefore, the indirect dependencies of Activity 26 on data produced by Activities 13-51 should not be represented in the matrix. Instead, the dependencies are implicitly represented as the direct dependencies of Activity 8 on data produced by Activities 13-51 and, in turn, the direct dependency of Activity 26 on data produced by Activity 8. This rule avoids the unnecessary modelling of duplicate data-dependencies.

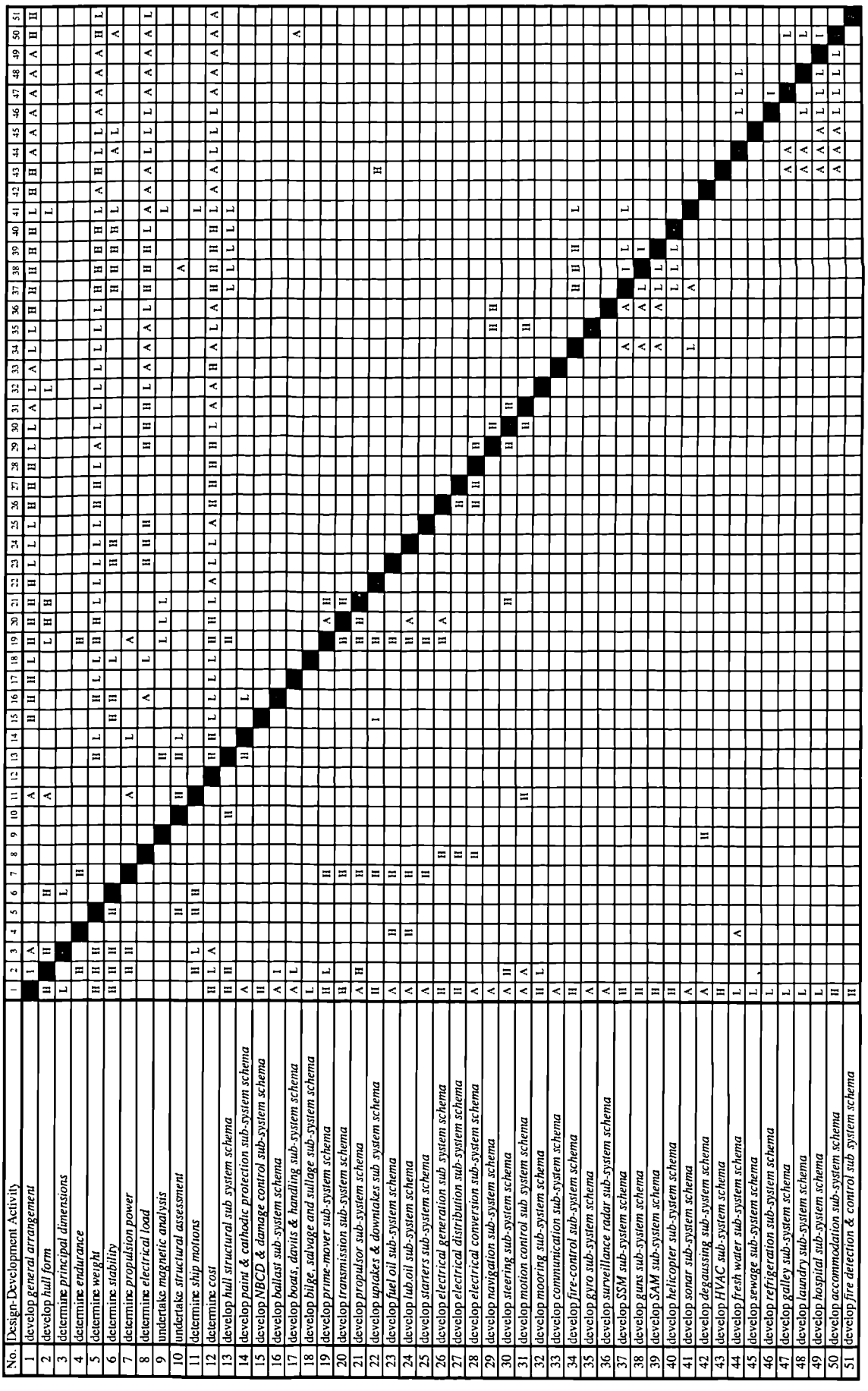
Based on the DSM System, the output from this function is a precedence matrix that models design-development activities and their data-dependencies. Figure 3.4 illustrates the precedence matrix for the warship case study.

3.6.3 Deriving a Near Optimal Sequence of Activities

"Derive a near optimal sequence of activities", more informally referred to as the *Activity Resequencing (ActRes) Problem* represents the third function of the *strategy for modelling the design-development phase of a product* and is represented in Figure 3.3 as function box A3.

Based on a precedence matrix, different sequences based on a range of different objectives can be derived and evaluated. In this respect, a precedence matrix, manipulated as part of the function of deriving a near optimal sequence of activities can act as a powerful *decision-support system.*

The controls that govern this function are the *sequencing constraints* that must not be violated when deriving new activity sequences. Such constraints relate to preferences based on experience or the need for certain activities to be sequenced at the beginning of design-development such that data relating to long-lead production items can be released as early as possible. Other sequencing constraints relate to the timed-release of data to design-assessment authorities such as classification societies.



No.	Design-Development Activity
1	develop *general arrangement*
2	develop *hull form*
3	determine *principal dimensions*
4	determine *endurance*
5	determine *weight*
6	determine *stability*
7	determine *propulsion power*
8	determine *electrical load*
9	undertake *magnetic analysis*
10	undertake *structural assessment*
11	determine *ship motions*
12	determine *cost*
13	develop *hull structural sub system schema*
14	develop *paint & cathodic protection sub-system schema*
15	develop *NBCD & damage control sub-system schema*
16	develop *ballast sub-system schema*
17	develop *boats, davits & handling sub-system schema*
18	develop *bilge, salvage and sullage sub-system schema*
19	develop *prime-mover sub-system schema*
20	develop *transmission sub-system schema*
21	develop *propulsor sub-system schema*
22	develop *uptakes & downtakes sub system schema*
23	develop *fuel oil sub-system schema*
24	develop *lub.oil sub-system schema*
25	develop *starters sub-system schema*
26	develop *electrical generation sub system schema*
27	develop *electrical distribution sub-system schema*
28	develop *electrical conversion sub-system schema*
29	develop *navigation sub-system schema*
30	develop *steering sub-system schema*
31	develop *motion control sub system schema*
32	develop *mooring sub-system schema*
33	develop *communication sub-system schema*
34	develop *fire-control sub-system schema*
35	develop *gyro sub-system schema*
36	develop *surveillance radar sub-system schema*
37	develop *SSM sub-system schema*
38	develop *guns sub-system schema*
39	develop *SAM sub-system schema*
40	develop *helicopter sub-system schema*
41	develop *sonar sub-system schema*
42	develop *degaussing sub-system schema*
43	develop *HVAC sub-system schema*
44	develop *fresh water sub-system schema*
45	develop *sewage sub-system schema*
46	develop *refrigeration sub-system schema*
47	develop *galley sub-system schema*
48	develop *laundry sub-system schema*
49	develop *hospital sub-system schema*
50	develop *accommodation sub-system schema*
51	develop *fire detection & control sub system schema*

Figure 3.4: The precedence matrix for the warship case study

The inputs to this function include (*i*) a precedence matrix; and, (*ii*) the objectives which are to be used as the basis for deriving near optimal sequences. As will be seen later in Chapters 5 and 6 such objectives include minimising iteration and maximising the concurrent processing of activities.

Again, whilst management and relevant members of the PDO represent one of the mechanisms of this function, the principal mechanism is a newly developed technique, detailed later in Chapter 4 and named the *Genetic Algorithm (GA)-Local Search Procedure.* This procedure, based on one or more objectives, searches for the sequence of activities that best satisfies the objectives within any pre-defined sequencing constraints. Finally, the output from this function is a near optimal sequence of activities and their data-dependencies as modelled using a *design structure matrix (DSM).*

Following a detailed description of the *GA-Local Search Procedure* and associated search objectives in Chapter 4, 5 and 6, the function of deriving a near optimal sequence of activities is demonstrated, later in Sub-Section 6.4.1, with the aid of the warship case study.

3.6.4 Deriving an Activity Network Diagram

"Derive an activity network diagram" represents the fourth function of the *strategy for modelling the design-development phase of a product* and is represented in Figure 3.3 as function box A4.

It is probable that a design structure matrix (DSM), as input to this function, will represent a sequence of activities where iteration is implied. However, by deriving a sequence of activities that is optimised, either partly, or solely on the objective of minimising iteration, unnecessary iteration, implied by poor sequencing, will be avoided. As such, the iteration implied by a DSM should be viewed as a necessary and integral part of the design-development phase of a product.

In order to derive a schedule of activities, the precedence relationships between activities must be known. Whilst the data-dependencies modelled by a DSM imply precedence relationships, such relationships are not explicitly modelled. On the other hand, whilst an activity network diagram does not explicitly represent data-dependencies, it does explicitly represent

precedence relationships. Therefore, prior to the creation of a resource-constrained schedule, the DSM must be converted into an activity network diagram.

Traditionally, design-development activity network diagrams are derived with an ignorance of the underlying data inter-dependencies that can lead to iteration. In contrast, the activity network diagram derived from this fourth strategic function is based on a full and complete understanding of this important and influential characteristic. Based on an interpretation of data-dependencies and inter-dependencies within the DSM, iteration can be identified and, prior to the derivation of an activity network diagram, the cause of any iteration can be determined and it's effect modelled. This is the principal reason why the derivation of a schedule based on precedence relationships is preceded by the derivation of a near optimal sequence.

The control that governs this function is an understanding of the specific cause and effect of each instance of iteration. Once again, whilst management and relevant members of the PDO represent one of the mechanisms of this function, the principal mechanism is the newly developed *iterative block resolution (IBR) procedure* which is used to derive an activity network from the design structure matrix (DSM).

Following a detailed description of the *iterative-block resolution procedure* in Sub-Section 6.4.2, the function of deriving an activity network diagram is demonstrated in the same sub-section with the aid of the warship case study.

3.6.5 Deriving a Resource-Constrained Schedule of Activities

"Derive a resource-constrained schedule of activities" represents the fifth and final function of the *strategy for modelling the design-development phase of a product* and is represented in Figure 3.3 as function box A5.

By assigning an estimated duration to each activity in an activity network diagram, a network-planning technique, such as *the critical path method*, can be used to analyse the network in order to create a time-based plan of activities. However, such techniques assume an infinite availability of resource and therefore do not accurately reflect those usual situations where resources are scarce.

Ultimately, the output from this function is a schedule of activities that matches resource requirements to resource availability in the most effective and efficient manner. The inputs to this function include (*i*) an activity network diagram defining activities and their precedence relationships; (*ii*) activity durations; (*iii*) activity resource requirements; and, (*iv*) a set of schedule objectives. These objectives, which are addressed later in Chapter 7, can include minimising the elapsed time required to complete all activities and maximising the utilisation of available resources.

In addition to those members of the PDO who are available to process activities, the principal mechanism of this function is the *Multiple Criteria Genetic Algorithm Project Scheduler* which is used to build a resource-constrained schedule of design-development activities.

Finally, the controls which govern this function include (*i*) the composition and availability of resources; (*ii*) a PDO based on the concept of a *resource-pool*; and, (*iii*) a set of stable design-development processes (*See* Chapter 2).

Following a detailed description of the *Multiple Criteria Genetic Algorithm Project Scheduler* in Chapter 7, the function of creating a resource-constrained schedule of activities is demonstrated with the aid of the warship case study.

3.7 Prior Research into Modelling a Product's Design-Development Phase

Over the last fifty years, a large number of models for representing a product's design-development phase have been put forward. Some of the most popular models include those developed by Asimow [Asimow 62], Hubka [Hubka 82], Pahl and Beitz [Pahl 84], French [French 85], Pugh [Pugh 90], and Ulrich and Eppinger [Ulrich 95].

Whilst research into the high-level modelling of a product's design-development phase, such as that mentioned above, is well documented, there are fewer references to be found which specifically address the aim of the research contained in this thesis. This observation is echoed by Jin *et al* [Jin 95] and, more recently, by Lewis and Cangshan [Lewis 98] when they state that "..in general there is a dearth of information on design management tools and their practical application to product development". One possible reason for this could be that

engineering companies sponsoring such research, aware of the commercial advantages of deriving an effective strategy, are keen to limit the dissemination of their work.

However, whilst published research is sparse, strategies aimed at providing PDOs with a means of modelling and optimising sequences and schedules of design-development activities can be found in the literature: For example, Kusiak and Park [Kusiak 90b] propose what, in effect, is a three-function strategy for managing a product's design-development phase.

- The first function focuses on decomposing a product's design-development phase into a product design-work breakdown structure (PDWBS) of activities. Based on the decomposition of a product into a hierarchy of *sub-systems* and *modules*, the activities that make up the design-development of each module are identified.

- The second function focuses on the matrix-based modelling of the relationships between the modules of a product and the activities that make up the design-development of each module. The matrix model represents the set of activities that constitutes the design-development of each module. Similar or identical activities may be performed in the design-development of different modules. In this respect, *matrix clustering analysis techniques* [Kusiak 87] are applied to cluster activities into *groups* of highly inter-related activities.

- The third function is based on *a knowledge-based system (KBS)* that is used to schedule and manage these groups of activities. Prior to the application of the KBS, a network diagram is created that represents the precedence relationships between activity groups. The KBS is then invoked to analyse these precedence relationships to determine whether they should be finish-to-start, start-to-start, etc. Ultimately, the KBS derives a critical path schedule for the activity groups.

Pourbabai and Pecht [Pourbabai 94] propose a strategy for creating a schedule of design-development activities which reflects the simple three function strategy summarised previously in Figure 3.1.

Larson *et al* [Larson 94] propose a strategy for creating a network diagram of design-development activities to be used as input to a scheduling procedure. Based on an IDEF

model (*See* Appendix II) of design-development activities and data-dependencies, a *precedence matrix* is created (*See* Appendix II). The precedence matrix represents how each activity in the matrix depends on each of the other activities in the matrix for input data. Through the application of a matrix *partitioning procedure* (*See* Sub-Section 5.3), a near optimal sequence of activities, based on the objective of minimising iteration, is generated. Finally, a *network diagram* is derived by modelling each *iterative block* of the partitioned matrix as a single aggregated activity and by representing each downstream data-dependency as a finish-to-start precedence relationship.

Building on the research of Larson *et al*, Belhe [Belhe 95] proposes a procedure for the *resource-constrained scheduling* of the type of network diagram created by Larson *et al's* research. The scheduling procedure still assumes that each downstream data-dependency represents a finish-to-start precedence relationship. However, in order to schedule the inter-related activities that make up an iterative block, Belhe assumes that it is possible to estimate the number of times a certain sequence of iterative activities is repeated. This number is then used to factor the duration of activities in the iterative block, and the block is replaced by the sequence of activities with the newly derived durations.

Austin *et al* [Austin 94, Austin 96 and Austin 97] have undertaken significant research based on the development of a strategy aimed at providing the construction industry with a means of modelling and optimising sequences and schedules of building design activities. Their strategy is very similar to that proposed by Larson *et al*. Based on a data-flow diagram (DFD) (*See* Appendix II) of design-development activities and data-dependencies, a precedence matrix is created. As before, the precedence matrix represents how each activity in the matrix depends on each of the other activities in the matrix for input data. Through the application of a matrix-partitioning procedure (*See* Sub-Section 5.3), a near optimal sequence of activities, based on the objective of minimising iteration, is generated.

Finally, a so-called *logic network diagram* is derived by interpreting the partitioned matrix. This network shows graphically, on a discipline basis, the inter-relationship of design-development activities (including the iterative blocks of inter-related activities) and data-flow on a pseudo timeline. Again, each downstream data-dependency in the matrix is represented as a finish-to-start precedence relationship in the logic network diagram.

Building on the research of Austin *et al*, Hassan [Hassan 95] proposes a simulation-based procedure for the resource-constrained scheduling of the type of network diagram created by Austin *et al's* research. The scheduling procedure still assumes that each downstream data-dependency is considered as a finish-to-start precedence relationship. However, in order to schedule the inter-related activities that make up an iterative block, Hassan assumes that each sequence of iterative activities is repeated twice. The time taken to complete the first iteration is related to the duration of the first activity in the sequence whilst the time taken to complete the second iteration is determined by the user.

Prasad *et al* [Prasad 98] describe a so-called *systematic concurrent workflow management (WM) process for integrated product development.* Workflow management is an analysis or study of a work process. In this case, the work process is defined as a product's design-development phase. The aim of workflow management is to optimise the workflow of *product, work, organisation* and *resource.* Applied to design-development, Prasad *et al* propose a four-function strategy for managing the design-development phase of a product.

- The first function focuses on modelling the product's design-development phase in terms of (*i*) a *product sub-model*; (*ii*) a *workflow sub-model* or a *work breakdown structure*; (*iii*) an *organisational sub-model*; and, (*iv*) a *resource sub-model.*

- The second function focuses on analysing the product's design-development phase based on the matrix-based workflow sub-model. More specifically, the function focuses on analysing the relationships among the activities of the workflow sub-model. These relationships include, but are not limited to the data-dependencies between activities.

- Based on the set of resulting models, the third function focuses on the adoption of strategies for re-engineering the design-development process to create a new work process model of activities. The potential benefits of each newly derived model are measured using the *analytic hierarchy process (AHP) method* as a means of choosing the most improved model.

- Finally, the fourth function focuses on the management of activities in the work process model.

Lewis and Cangshan [Lewis 98] propose what, in effect, is a three-function strategy for managing design-development activities.

- The first function focuses on describing the product's design-development phase. In particular, the phase is characterised by; (*i*) the inputs of data into the phase; (*ii*) the outcome from the phase in the form of plans and specifications of product configuration, components and materials of construction; (*iii*) the array of decisions (activities) by which the inputs are transferred into outcomes; and, (*iv*) the PDOs knowledge.

- The second function focuses on modelling the design-development phase using decision support tools including the *design structure matrix* (*See* Appendix II).

- Finally, the third function focuses on modelling the knowledge and skill base required to support activities and decisions. This information is encapsulated in a so-called *design resources chart*. This representation is constructed to give an overview of the design-development resources required for the successful development of the product. It identifies the resources (other than time) to be marshalled by the design-development manager and includes the knowledge generated and utilised during the product's design-development phase.

3.8 How the Research Behind the Proposed Modelling Strategy Varies from Prior Research

Models of a product's design-development phase such as those proposed by Asimow, Hubka, Pahl and Beitz, French, Pugh and Ulrich and Eppinger aim to communicate the underlying principal stages of a product's design-development phase and, as such, tend to be described at a high-level of abstraction.

However, in order to help a product development organisation (PDO) to *plan* and *organise* design-development work, it may be necessary to model at a level of abstraction where specific design-development activities, data-dependencies and precedence relationships are defined. In this context, the activities should be of *managerial significance*; that is, it should be possible to link a set of *clearly identifiable parameters* to each activity in the model. These parameters include inputs, outputs, duration, resource requirements, etc.

These sentiments are in agreement with those expressed by other researchers. For example, Hollins *et al* [Hollins 93] point out that models such as those mentioned above are insufficiently detailed to form the basis of a design-development planning technique. More recently, Austin *et al* [Austin 96] proffer the notion that design-development problems generally have their roots in detail and are usually caused through the poor communication of design data, information and knowledge. To adequately address such issues, a model is required that can represent the design-development phase of a product at the most fundamental of levels.

As mentioned previously, it is the principal aim of the research contained in this thesis to provide PDOs with a strategy for modelling and optimising sequences and schedules of activities such that the design-development phase of a product can be planned and organised in a more effective manner. Therefore, the research contained in this thesis differs in a number of ways from research related to the high-level modelling of a product's design-development phase. Firstly, this research is not focussed on the creation of a model; rather, it is focussed on the creation of a strategy for modelling. Furthermore, it is focussed on the representation of activities of managerial significance rather than the underlying principal stages of a product's design-development phase.

In terms of a modelling strategy, there are also significant differences between the research detailed in this thesis and prior research: As part of their early research, Kusiak *et al* do not specifically address how the activities of a *product design-work breakdown structure (PDWBS)* are to be generated. Later research [Kusiak 93] extends the functionality of the matrix-based modelling and analysis technique but still does not specifically address the function of creating a PDWBS.

Austin's research is focussed on *building-design-work.* As part of the function of creating a PDWBS, Austin proposes that a single model can cover the design of a general building such that the PDWBS model consists of a basic *generic framework* to which smaller, discrete sub-models are added. Austin's argument for the adoption of a generic approach relates to the fact that "..although parameters such as the magnitude and accuracy of the information may differ between two building designs, the category of information does not" [Austin 96]. When modelling the design-development phase of very similar products it may be appropriate for Austin to argue for the adoption of a generic model rather than the creation of a unique model.

However, when faced with the task of providing a strategy aimed at modelling the design-development of *any* engineered product, an approach that is focussed on identifying and uniquely modelling the design-development work associated with the specific sub-systems and components of the product may be more appropriate.

Prasad mentions the importance of developing an effective PDWBS but does not document any research on an approach for developing a PDWBS. Lewis focuses on what attributes should be linked to an activity but, again, does not document any research on an approach for developing a PDWBS.

The creation of a rational and systematic product design-work breakdown structure is perhaps the most important function of a strategy with the previously defined aim. Based on the author's experience with the industrial case study, all subsequent analysis is meaningless when an inappropriate PDWBS is used. In this respect, the research contained in this thesis differs from prior related research in that it proposes an approach for developing a PDWBS that can be used effectively as the basis of a model of any product's design-development phase.

Later, in Section 5.3, Kusiak's research into the matrix-based modelling and optimal sequencing of design-development activities is summarised. However, in the context of an encompassing strategy for managing design-development activities, Kusiak's work is focussed primarily on the matrix-based modelling and clustering of design-development activities. In the same way, Prasad's analysis tends to focus on clustering activities to satisfy the objective of maximising the integration and communication of data rather than on the development of a near optimal sequence of activities based on objectives such as minimising iteration.

Pourbabai and Pecht's strategy doesn't consider data-dependencies and, furthermore, does not address the central issue of iteration. In this respect, their strategy results in a network diagram that is derived from an incomplete model of a product's design-development phase and, as such, shares all of the drawbacks of the traditional three-function strategy described previously in Section 3.4

Lewis uses the design structure matrix (DSM) to model design-development activities and their data-dependencies, but does not appear to use the DSM to optimise data-flow and, as such, misses opportunities for improvement.

As indicated by the references cited later in Section 5.3, there has been a significant amount of prior research undertaken on the design structure matrix (DSM) system. However, there is limited documented research focussed on using the DSM specifically as a decision-support tool prior to the scheduling of design-development activities. As such, the application of the DSM in this way, as part of the proposed strategy, represents a contribution to the subject.

In terms of creating a network diagram, Kusiak's strategy only focuses on defining the precedence relationships between so-called groups of activities. They themselves admit that the strategy needs to be extended so that it addresses the co-ordination of activities within the groups.

Larson groups iterative blocks of activities into a single aggregated activity and goes no further towards the development of a function aimed at interpreting and resolving the intracies of iteration prior to the creation of a network diagram. Based on Larson's work, Belhe simply makes an assumption concerning the number of times an iterative block is to be repeated and factors iterative activity durations accordingly prior to the application of the proposed scheduling procedure.

Other examples of over simplified assumptions applied to the interpretation of a matrix model prior to the derivation of a network diagram include (*i*) the notion that the downstream data-dependencies of a matrix-model are directly comparable to finish-start precedence relationships in a network diagram; and, (*ii*) the notion that iterative blocks of activities in a matrix-model can simply be unravelled a pre-defined number of times to represent an extended sequence of activities in a network diagram.

It is considered that the over simplified interpretation of a model of activities and data-dependencies prior to the creation of a network diagram is a significant dysfunction of existing strategies. In this respect, the *iterative block resolution procedure*, which forms the basis of the fourth function of the proposed strategy, represents a significant contribution to overcoming this dysfunction.

Finally, the strategies of Kusiak *et al*, Austin *et al* and Pourbabai and Pecht address scheduling issues, but do not specifically address resource-constrained scheduling. Prasad *et al's* strategy is focussed more on monitoring a schedule rather than developing a schedule for the allocation of resources. In the same vein, Lewis *et al's* design resources chart models the key design decisions (activities) and resources. As such, their design resources chart acts as an aid to the high-level management of a product's design-development phase, but does not address detailed management issues such as the scheduling of activities.

In this respect, the research contained in this thesis differs by introducing the notion of resource-constrained scheduling as the ultimate function within the context of a strategy aimed at providing PDOs with a means of modelling and optimising sequences and schedules of design-development activities.

Finally, as well as those differences that exist at the strategic level, the five functions that constitute the proposed *strategy for modelling the design-development phase of a product* vary from the work of others in a number of ways. In the remainder of this thesis, prior research related to each specific strategic function and topic is documented and the differences between current and prior research are again highlighted as contribution.

3.9 Summary

Chapter 3 begins by making a case for modelling a product's design-development phase. In summary, a model helps to develop a better understanding of the fundamental elements of this life cycle phase of a product so that there is a higher probability that initiatives, aimed at improving the efficiency and effectiveness of design-development, will be focussed where they have the most beneficial effect. Consequently, the question is not whether a model should be used or not, but rather, what characteristics should be included in a strategy for modelling a product's design-development phase.

Over the last few decades, a traditional strategy for modelling a product's life cycle phases has evolved. This strategy, described in Chapter 3, consists of three functions, namely, (*i*) Derive a WBS; (*ii*) Derive an activity network diagram; and, (*iii*) Derive a resource-constrained schedule of activities.

Whilst the traditional strategy may be used most frequently to model a product's production phase, in the absence of an alternative, engineering companies tend to adopt the same strategy when faced with the task of planning and organising a product's design-development phase. However, the traditional strategy has several drawbacks when applied to design-development.

Based on discussions with designers, planners and project managers from a range of engineering companies, Chapter 3 proposes a set of requirements for a modelling strategy for planning and organising a product's design-development phase. Based on the drawbacks of the traditional modelling strategy and these requirements, a new strategy is proposed. The proposed strategy consists of five strategic functions, namely, (*i*) Derive a product design-work breakdown structure of activities; (*ii*) Model the activities and their data dependencies; (*iii*) Derive a near optimal sequence of activities; (*iv*) Derive an activity network diagram; and, (*v*) Derive a resource-constrained schedule of activities.

In Chapter 3, the first two functions of the proposed strategy are demonstrated with the aid of the warship case study. However, because the relevant techniques are yet to be described, the three subsequent strategic functions are merely summarised in Chapter 3 and demonstrated, with the aid of the warship case study, in later chapters of the thesis.

Ultimately, Chapter 3 concludes by describing prior research focussed on modelling the design-development phase of a product and details how the research behind the proposed strategy is different and, as such, forms a contribution to the subject. In summary, prior research has focussed on one or more functions of the proposed strategy, but until now to the best of the author's knowledge, no one has brought all of the functions together into a single strategy. In addition, little if any prior research has documented the application of a strategy (such as that proposed) to a real-life-engineering case study.

4. A Procedure for Deriving a Near Optimal Sequence of Activities

4.1 Introduction

At the end of Chapter 3, *a strategy for modelling the design-development phase of a product* was described. As part of this description, the warship case study was used to elaborate how the first two strategic functions (function boxes A1 and A2 of Figure 3.3) may be applied.

Of course, "creating a product design-work breakdown structure" of design-development activities, and "modelling the activities and their data-dependencies" using the design structure matrix (DSM) system, is only the beginning. As part of the third strategic function (function box A3 of Figure 3.3), a *search procedure* is subsequently applied to the matrix-model in order to "derive a near optimal sequence of activities". The derivation of a near optimal sequence of activities in this manner is less formally referred to as the *Activity Resequencing (ActRes) Problem.*

Near optimal sequences are derived by resequencing activities based on a number of objectives. Such objectives include *minimising the potential of iteration*, and *maximising the concurrent processing of activities.* These objectives are considered later in Chapters 5 and 6. Meanwhile the current chapter describes the search procedure used to derive a near optimal sequence of activities which, as the principal mechanism of the third strategic function, can be used to search for the sequence and the corresponding DSM which most closely matches any pre-defined objective.

The search procedure executes a search by resequencing activities, interchanging rows and swapping the corresponding columns, continually deriving new sequences and DSMs which are then evaluated using a mathematical-based measure of the search objective. New DSMs result from the fact that, whilst the data-dependencies between activities remain unchanged by sequence, their relative positions within the matrix change with each new sequence.

There exists a large range of search techniques upon which a specific procedure can be based. Section 4.2 summarises some of the more common techniques before detailing the rationale behind the choice of a *genetic algorithm (GA)* as the main component of a search procedure

for deriving a near optimal sequence of activities. Section 4.3 then goes on to describe the standard genetic algorithm, whilst Section 4.4 details how the standard has been modified to create the *Optimal Sequencer Genetic Algorithm.*

In order to improve efficiency and enhance solution quality further, a local search is applied to further improve on the sequences derived by the *Optimal Sequencer Genetic Algorithm.* This secondary search, based on a heuristic approach, is described in Section 4.5. In Section 4.6, the newly developed search procedure, which combines the genetic algorithm with the local search and, as such, has been named the *GA-Local Search Procedure*, is described and, as a result of experimentation, a set of guidelines for it's application is presented. Finally, Section 4.7 summarises the chapter and highlights how the research specifically documented in Chapter 4 represents a contribution.

4.2 The Choice Of Search Technique

When faced with the choice of which type of search technique to use, it is necessary to consider the nature of the search space associated with the problem at hand. The search spaces associated with combinatorial problems such as the *ActRes Problem* are very noisy and discontinuous. Furthermore, they grow exponentially as problem size increases. For these reasons, the *ActRes Problem* is a complex computational problem and is described in computer science terms as *NP-complete.* This means that *exact-mathematical approaches* based on dynamic programming and branch and bound search techniques that deterministically and exhaustively search for the global optimum solution will probably fail because of the prohibitively large amount of time that would be required.

The only way of ensuring that the *global optimum* is found is through the *complete enumeration* of all possible solutions. However, for large search spaces, such as those experienced with the *ActRes Problem*, complete enumeration is simply not feasible due to the excessive number of enumerations that would be necessary. For example, fifty activities can be mapped to 3.04×10^{64} (50 factorial) possible sequences. In this respect, it may be observed that, even for relatively short sequences, the complete enumeration of all possible sequences in a search for the global optimum would require an excessive amount of computer processing time.

Faced with a very large search space, *calculus-based search techniques* will tend to find a *local optimum* with a high probability of missing the global optimum completely. Once a local optimum has been found, further improvement can only be achieved through random restart. In addition, a noisy and discontinuous search space ensures that the mathematical-based measure of the search objective is not amenable to calculus. With these observations in mind, calculus-based search techniques are thus judged unsuitable for use in solving the *ActRes Problem.*

Using rules of thumb, *heuristic-based search techniques* are directed towards favourable regions of the search space. However, the search is always directed towards the *nearest* favourable regions. For excessively large search spaces, it is likely that the nearest favourable regions will be local optima. Therefore, whilst heuristic-based search techniques can be effective during the later stages of a search when the search space has been pruned down, they are unsuitable for solving the *ActRes Problem* in the first instance.

Using random choice, *random-based search techniques* avoid the certain prospect of convergence at a local optimum. However, such techniques are still highly inefficient since, without some form of direction, the global optimum will only be found by accident. For example, having evaluated 10,000 possible sequences of fifty activities at random, there is still only a minute probability, equal to 3.29×10^{-61} that the global optimum will be found.

Whilst purely random-based search techniques are inefficient, techniques that use *random choice* as a tool to *guide a highly explorative directed search* begin to overcome the inadequacies and inefficiencies associated with those search techniques already mentioned. In fact, it is considered that search techniques that incorporate such functionality offer the most effective and efficient solution to the *ActRes Problem.* Three specific search techniques that use random choice as a tool to guide a highly explorative directed search are *simulated annealing*, *tabu search*, and *genetic algorithms.*

Using the terminology associated with the *ActRes Problem*, simulated annealing [Kirkpatrick 83, Cleland 94] takes a sequence of activities and defines this arbitrary sequence as the *current state.* This state is invoked at a pre-defined start temperature. The current state is evaluated using an *energy function*, which is used to measure how closely the current state matches the search objective. The value of this energy function represents the *internal energy*

(E) of a given state such that the aim of the search is to find the state, that is, the sequence, with the most desirable internal energy. In the case of a minimisation problem, the most desirable internal energy is the lowest.

Using the current state, one of the activities is chosen at random and moved to a new position, again chosen at random. The internal energy of the resulting new state is then re-evaluated using the energy function. If the internal energy of the new state is lower than that of the current state, then the change in energy (ΔE) is negative. In such cases, the new state, that is, the new sequence, is automatically accepted as the current state.

However, if the internal energy of the new state is higher than that of the current state, then the change in energy (ΔE) is positive. In such cases, the new state is accepted as the current state, only with a probability as defined by Equation 4.1. In this way, newly derived states that represent a positive change in internal energy have a reduced chance of being accepted as the current state. Furthermore, for a given temperature, the probability of acceptance reduces exponentially with the increased magnitude of positive energy change (ΔE).

Probability of Acceptance = $e^{-\Delta E/T}$	*[Equation 4.1]*
T = Temperature ΔE = Difference in internal energy between the newly derived sequence and the current state	

The process of repositioning randomly chosen activities and evaluating the energy function of the resulting state is repeated a number of times using the start temperature. In this way, the current state changes continuously, accepting or rejecting newly derived states as the current state on the basis of their effect on the energy function.

Based on an annealing schedule, after the creation of a given number of new states, that is, new sequences, the temperature is reduced and the process is repeated using the previously saved current state. The annealing schedule also determines the rate of temperature reduction and therefore the number of times the annealing process is repeated. As can be seen from Equation 4.1, as the temperature reduces, so the probability of accepting higher energy states reduces exponentially. Therefore as the temperature is reduced, convergence begins to occur. The process can be stopped when no further reductions in energy are obtained.

Tabu search, described by Glover and Laguna [Glover 93] is very similar to simulated annealing, however, instead of repositioning activities to new random positions, each chosen activity is moved to an adjacent position. The *tabu list* contains a record of the sequences previously derived and is used to avoid the regeneration of such "tabu" sequences.

The previous paragraphs have described two of the three techniques, that is, simulated annealing & tabu search, which both use random choice as a tool to guide a highly explorative directed search. However, before describing the third technique - the genetic algorithm, the rationale behind choosing this technique as the main component of a technique for deriving a near optimal sequence of activities requires explanation.

Based on combinatorial problems such as the *ActRes Problem*, Reeves [Reeves 95] has compared the effectiveness of the genetic algorithm technique against the simulated annealing technique. Procedures based on both techniques have been applied to various sized problems, the results of which seem to indicate that, for relatively small problems, on average a genetic algorithm-based procedure performs only slightly better than a simulated annealing-based procedure. However as the size of the problem increases, the performance of the genetic algorithm is significantly better. In addition, the genetic algorithm-based procedure tends to reach convergence faster than the simulated annealing-based procedure.

Such comparisons are strongly related to the programming of the techniques into specific procedures and the range of examples used. In addition, the work of Reeves does not include a direct comparison between the genetic algorithm technique and tabu search. However, whilst specific evidence relating to the superiority of genetic algorithms over simulated annealing and tabu search is sparse, genetic algorithms do have a number of characteristics that offer significant advantages over simulated annealing and tabu search in terms of their flexibility, robustness and performance. In particular, there are two issues that highlight the superiority of genetic algorithms over the two alternatives.

(1) Genetic algorithms search from a population of solutions. Almost all optimisation search techniques move from one solution to the next, continually searching for improvement. This point-to-point strategy is dangerous because;

- When faced with noisy and discontinuous search spaces, point-to-point strategies tend to converge at a local optimum rather than the global optimum.

- Whilst the derived solution may be sited at the top of a peak or at the bottom of a trough in the search space, no insight to the *robustness* of the solution can be determined. For example, consider a solution sited at the top of a very steep peak, or at the bottom of a very narrow trough. Only a small change in value of one of the parameters that define this solution may be required in order to cause a significant displacement of the solution away from the optimal peak or trough. Such solutions are said to be *non-robust* since a small change in value of one of the parameters that define the solution can mean that the solution is no longer optimal. With respect to robustness, optimisation search techniques should return a solution that is sited at the top of a flat peak, or at the bottom of a shallow trough. In such cases, relatively small changes in values of the parameters that define the solution will not displace the solution significantly away from the peak or trough. Such solutions are said to be *robust.*

By sometimes accepting solutions which are inferior to the current state, the simulated annealing and tabu search techniques tend to avoid convergence at a local optimum, however they still converge towards a single solution. The genetic algorithm on the other hand, avoids convergence at a local optimum by continually moving, not between single solutions but between populations of solutions. In this way, genetic algorithms can return a final population of varied solutions, all close to the global optimum. Furthermore, whilst the genetic algorithm does not search specifically for robust solutions, the variance of the fitness of the solutions in the final population can at least be used to derive a measure of robustness.

(2) Genetic algorithms use a range of probabilistic, not deterministic transition rules. Genetic algorithms use random choice as a tool to guide a search towards regions of the search space with likely improvement. Whilst simulated annealing and tabu search techniques share such characteristics, genetic algorithms encompass a wider range of varied operators for guiding the search.

Before going on, attention needs to be drawn to the fact that, whilst genetic algorithms offer the most efficient and effective method for searching excessively large, noisy and discontinuous search spaces, they can never guarantee finding the global optimum solution.

For such large search spaces the only way of guaranteeing finding the global optimum is through full enumeration which, as previously stated, is infeasible. In such situations, Goldberg [Goldberg 89] proffers the notion that, the most important goal of an optimisation search technique should be efficient improvement over time by obtaining successive solutions that are better than previous solutions. Adopting this notion, the best search technique is the one that develops continual improvement, quickly converging towards a high quality solution that is, at best, the global optimum and, at worst, is very close.

4.3 The Standard Genetic Algorithm

The genetic algorithm is a search technique based on the axioms of natural selection and genetics. The technique combines survival of the fittest with some of the intuition of human search. The genetic algorithm begins with a single string. Using the terminology of genetics, a string is analogous to a ***chromosome***. Each position in the string is analogous to a ***gene*** and is used to represent one of a range of *parameters* that have been chosen to define the problem. The parameter associated with each string position can take on a value (***allele***) such that any one string of parameter values, that is, any one ***chromosome of alleles***, can be mapped to a single distinct solution. For the *ActRes Problem*, a given string of parameter values represents one possible sequence of activities.

Using the single input string, *a population* of new strings is developed by randomly resequencing the original string. This *initial population* should be spread over enough of the search space to represent as wide a variety of solutions as possible. The diversity so introduced prevents premature convergence to a local optimum. This population of randomly generated strings represents the *first generation*. The size of this population, that is, the number of generated strings is defined by the user and, once set, the *population size* remains constant throughout the search.

The first generation of strings then goes onto receive *genetic operation*, with the resulting development of a *second generation* of equal population size to the first. As a result of genetic operation, this second generation should contain fitter strings, that is, strings that more closely match the search objective. In turn, the second generation is operated on to develop a *third generation* that should contain even fitter strings. This process continues developing *successive generations* of fitter strings, converging towards the global optimum, until some stop condition is met. At this stage a population of the fittest strings is returned.

How closely a string matches the search objective, or more specifically the *fitness* of a string, is derived by first decoding the string into the combination of parameter values it represents. These parameter values are then plugged into a pre-defined search *objective function.* The resulting value represents the fitness of the associated string. The search objective function is so called since it represents a mathematical expression of the *search objective.* For the *ActRes Problem,* as will be seen later in Chapter 5, one of the search objectives is to derive a sequence of activities that incorporates minimum iteration. A very basic mathematical expression of this search objective is a procedure that counts the number of data-dependencies above the leading diagonal. Using the objective function, any one sequence (solution) can be evaluated and ranked according to how well it satisfies the search objective.

The aim of a maximisation problem is to find the combination of parameter values that, when plugged into the objective function, yield a fitness which is as large as possible. In such cases the string that represents this combination is the *fittest.* A genetic algorithm that yields good results in many practical problems is composed of three basic genetic operators; (*i*) *selection*; (*ii*) *crossover*; and, (*iii*) *mutation.* The following sub-sections detail these operators in relation to a very basic maximisation problem.

4.3.1 Selection

Using the input string, a population of strings is randomly created and the fitness of each string is derived using the objective function. Using the selection operator, strings begin the journey into the next generation based on their fitness. Fitter strings with a large associated fitness have a higher probability of contributing one or more offspring to the next generation.

The selection operator can be implemented as *a biased roulette wheel* where each string has a slot which is sized in proportion to it's fitness. Consider the following example, where the first generation, consisting of a population of four strings is developed by randomly resequencing an original input string. Let us assume that each string represents a list of parameter values that, when plugged into the objective function, yields a fitness as summarised in Table 4.1.

According to the logic of the selection operator, String 2 has a fitness equal to 576 which accounts for 49% (576 /1,170) of the total summed fitness values. As such, String 2 has a 49% share of the biased roulette wheel (*See* Figure 4.1). Every time the wheel is spun the notional

selector ball has a probability equal to 0.49 of landing in String 2's sector of the wheel. The roulette wheel is spun a number of times equal to the population size (in this example four times).

Number	String	Fitness	% TOTAL
1	01101	169	14%
2	11000	576	49%
3	01000	64	6%
4	10011	361	31%
TOTAL		1,170	100%

Table 4.1 Four strings used to illustrate how the selection operator works

Fitter strings have a higher probability of being selected since they occupy a larger share of the roulette wheel. Once a string has been selected, an exact replica of the string is entered into *a mating pool*, here referred to as the *Alpha Mating Pool*. The Alpha Mating Pool contains the strings that are going onto receive further genetic operation. In the example above, it is likely that the Alpha Mating Pool would contain two copies of String 2, and a copy each of Strings 4 and 1. It is most likely that String 3 would be lost, however the random nature of selection means that there is still a chance that strings with low fitness will survive into the Alpha Mating Pool.

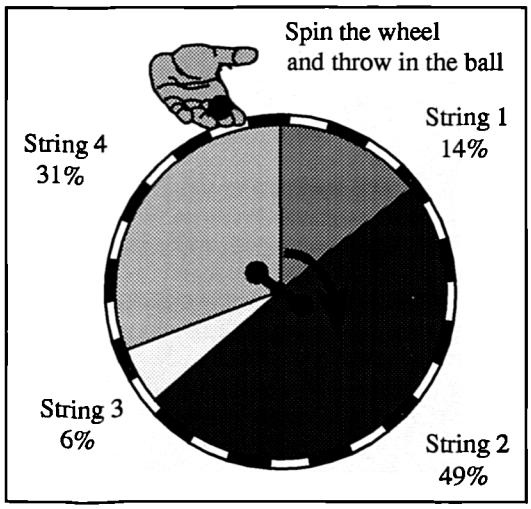


Figure 4.1: Selection using the Biased Roulette Wheel

An extension to the selection operator is *generation gap*, which was first introduced by DeJong [DeJong 75]. Generation gap allows populations to be overlapped. The parameter can take values in the range 0.0 - 1.0 where;

generation gap = 1 *represents non-overlapping populations*, and

$0 \leq$ generation gap < 1 *represents overlapping populations.*

For overlapped populations, a percentage of the fittest strings defined as (population size × [1-generation gap]), can be seeded straight into the Alpha Mating Pool. The remaining places in the Alpha Mating Pool are then filled as before using the biased roulette wheel selection operator. So for example, if generation gap = 0.9, then the first 10% of places in the Alpha Mating Pool would be filled immediately with the top 10% from a list of strings in the previous generation, ranked in descending order of fitness. The remaining 90% of places in the Alpha Mating Pool would be filled by randomly selecting strings from the list using the biased roulette wheel.

If the process of selection were to be repeated indefinitely without further genetic operation, then the final population would consist entirely of copies of the fittest string in the *initial population.* Therefore, in order to avoid premature convergence to a local optimum, mechanisms for change, such as crossover and mutation, are incorporated into the genetic algorithm.

4.3.2 Crossover

After selection, crossover is applied to the Alpha Mating Pool. Two member strings of the Alpha Mating Pool, defined as *parent strings* are randomly selected. For each pair of parent strings, an *integer crossover position* (x) is chosen at random. The integer crossover position can lie anywhere between the start and end of a string, but the same position applies to both strings. Two new *offspring strings* are then created by swapping all characters between the selected position and the end of the strings. Consider two parent Strings P_1 and P_2 and suppose the value $x = 4$ were generated at random

$$P_1 = 0110 \mid 1$$
$$P_2 = 1100 \mid 0$$

The resulting crossover yields the two offsprings;

$$O_1 = 0110 \mid 0$$
$$O_2 = 1100 \mid 1$$

The two parents are then removed from further consideration and the resulting two offsprings go into a second mating pool, here referred to as the *Beta Mating Pool.* The crossover operator continues to operate on pairs of strings from the Alpha Mating Pool, filling places in

the Beta Mating Pool, until crossover has been applied a pre-defined number of times. This number of times is related to the probability of crossover (P_c) which is set at a value between zero and unity. For example, if $P_c = 0.5$, then crossover is applied to 50% of the strings in the mating pool. Once crossover has been applied the pre-defined number of times, the remaining places in the Beta Mating Pool are filled with the remaining uncrossed strings from the Alpha Mating Pool.

4.3.3 Mutation

Occasionally selection and crossover can become overzealous and lose some potentially useful genetic material. Within the genetic algorithm, mutation is the occasional random alteration of the value of a string position. By itself, mutation is a random walk through the search space. When used sparingly with selection and crossover it is an insurance policy against the premature loss of potentially important genetic material.

After crossover, mutation is applied to the Beta Mating Pool. A string from the Beta Mating Pool is chosen at random. Two positions in the string are then chosen, again at random. The characters occupying the two positions are then swapped. The newly mutated string goes into a final pool of strings here referred to as the *Omega Pool.* The mutation operator continues to operate on strings from the Beta Mating Pool, filling places in the Omega Pool until mutation has been applied a pre-defined number of times. This number of times is related to the probability of mutation (P_m) which is set at a value between zero and unity. For example, if $P_m = 0.5$, then mutation is applied to 50% of the strings in the mating pool. Once mutation has been applied the pre-defined number of times, the remaining places in the Omega Pool are filled with the remaining unmutated strings from the Beta Mating Pool.

The Omega Pool is in fact the final population of the current generation. By the action of genetic operation the resulting population should contain a higher percentage of fitter strings than the population at the beginning of the current generation. The resulting population is then fed back into the procedure and subjected to selection, crossover and mutation all over again. This looped process continues developing successively fitter generations until some stop condition is met. Figure 4.2 illustrates the standard genetic algorithm

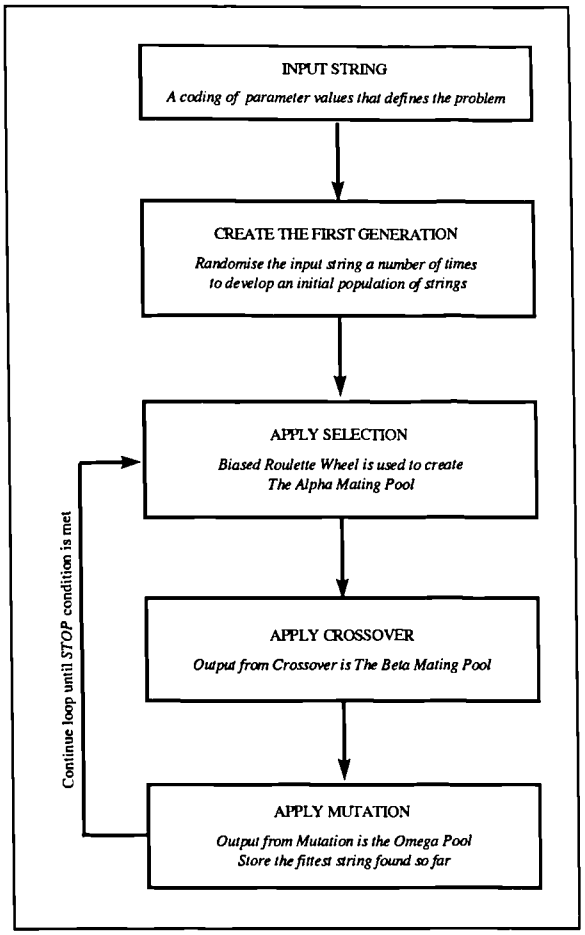


Figure 4.2: The standard genetic algorithm

4.4 Modifications and Extensions to the Standard Genetic Algorithm

Modifications to the standard genetic algorithm are necessary in order to apply the technique to combinatorial problems such as the *ActRes Problem.* In addition to these modifications, the technique can be extended in order to improve the efficiency and effectiveness of the search. The following sub-sections detail how the standard genetic algorithm has been modified and extended as part of the current research to create the *Optimal Sequencer Genetic Algorithm.*

4.4.1 String Representation

Traditionally, values of the parameters that define the problem are *represented* by a string of 0's and 1's. This *binary string representation* has the advantage of simplicity. However when dealing with combinatorial problems such as the *ActRes Problem,* the binary string representation proves to be inadequate. In a binary string representation of activities, each activity is represented as a binary number such that the string is a concatenation of these binary numbers. In such cases, the application of standard crossover and mutation operators can easily create new strings that include some activities more than once and others not at all.

In effect, invalid strings can result when standard crossover and mutation operators are applied to a binary representation of a combinatorial problem.

One solution to this problem relies on modifying the string representation such that standard crossover and mutation operators can be used. One example of such an approach is the *Ordinal Representation* [Grefensette 85]. In this case, the string is represented by a sequence of n integers within which the ith element can range from 1 to $(n\text{-}i+1)$. The Ordinal Representation uses two lists, the *Ordinal Sequence List* and the *Free List*. Using the example illustrated in Table 4.2, the actual sequence, a-d-c-e-b is represented by the ordinal sequence, 1-3-2-2-1, with the start activity fixed.

Ordinal Sequence List	Free List	Actual Sequence
( )	(a b c d e)	( )
(1)	(b c d e)	(a)
(1 3)	(b c e)	(a d)
(1 3 2)	(b e)	(a d c)
(1 3 2 2)	(b)	(a d c e)
(1 3 2 2 1)	( )	(a d c e b)

Table 4.2: The Ordinal Representation

The initial free list contains all of the activities in ascending order. An *ordinal sequence list* is derived as follows. The first integer in the *ordinal sequence list* is set equal to the position number in the free list of the first integer in the *actual sequence*. (E.g. 'a' is the first integer in the actual sequence and, because it occupies the 1st position number in the initial free list, the first integer in the ordinal sequence list is set equal to '1'). The free list is then revised by removing the 'a'. The second integer in the *ordinal sequence list* is then set equal to the position number in the free list of the second integer in the *actual sequence*. (E.g. 'd' is the second integer in the *actual sequence* and, because it occupies the 3rd position number in the revised *free list*, the second integer in the *ordinal sequence list* is set equal to '3'). This process continues until the *free list* has been depleted completely.

The advantage of the Ordinal Representation is that standard crossover and mutation operators can be used. However, in such cases the crossover operation is likely to develop offspring strings quite varied from the parent strings. This is because the section of the string to the right of the crossover point is affected more and, as a result, if crossover occurs near the start

of the string, the resulting offspring receives large disruption. For this reason, in many cases the Ordinal Representation does no better than a random search.

Another solution to the problem of representing strings associated with combinatorial problems relies on modifying the genetic operators such that a very simple string representation can be used. A number of researchers adopting such an approach have developed direct string representations to match their newly derived operators [Grefensette 85, Syswerda 91b].

For example, the direct string representation 1-2-3-4 represents a sequence of activities beginning with Activity Number 1, succeeded in order by Activity Numbers 2, 3, and 4. Using this representation, only the crossover and mutation operators require modification such that, with the exception of these operators, the standard search technique remains unchanged. Because a direct string representation is much easier to interpret and communicate, the *Optimal Sequencer Genetic Algorithm* uses a direct string representation in conjunction with modified crossover and mutation operators.

4.4.2 Modified Crossover

A number of modified crossover operators designed for use with combinatorial problems using direct representations have been proposed in the literature. In the *Alternating Edges Crossover Operator* [Grefensette 85] alternate activities are chosen from both parents and crossed. If an activity is repeated then it is replaced with a randomly selected activity that has not yet been used. This operator tends to disrupt good sub-sequences, that is parts of sequences that work well together. In the *Subtour Chunks Crossover Operator* [Grefensette 85], instead of choosing alternate activities, random length alternate sub-sequences are chosen from the two parents and crossed. If a sequence is repeated then it is replaced with a randomly selected activity that has not yet been used.

Of the crossover operators considered, the *Optimal Sequencer Genetic Algorithm* uses the *Partially Mapped Crossover* (PMX) *Operator* [Goldberg 85]. PMX chooses two positions at random. The sub-sequences between these two points on both parents are called *mapping sections*. These sections are then mapped together. Consider two parents, with PMX applied between the fourth and sixth positions;

P_1 = 9 8 4 | *5 6 7* | 1 3 2 10

P_2 = 8 7 1 | *2 3 9* | 10 5 4 6

After PMX

O_1 = 7 8 4 2 3 9 1 6 5 10

O_2 = 8 9 1 5 6 7 10 2 4 3

With crossover between the fourth and sixth positions, the mapping sections are 5-6-7 for P_1, and 2-3-9 for P_2. This means that in both P_1 and P_2 "5" is replaced by "2", and "2" is replaced by "5". The same applies for the other two mappings. In this way two new offsprings are developed with no repeated activities. PMX has been incorporated into the *Optimal Sequencer Genetic Algorithm* because of it's effectiveness "..in causing change with minimum disruption. In this way, potentially good sub-sequences are preserved" [Goldberg 85].

4.4.3 Modified Mutation

According to Reeves [Reeves 95], two types of modified mutation are possible. The first, an *Exchange Mutation* is a simple exchange of two activities in the string, chosen at random. The second, a *Shift Mutation*, moves a randomly chosen activity a random number of places to the right or left. However, a third and newly developed mutation operator, referred to as *Heuristic Mutation*, is incorporated within the *Optimal Sequencer Genetic Algorithm*. This *Heuristic Mutation* operator considers every string in the population in turn and operates as follows;

(1) For each string in the population evaluate;

> *Function (1):* The matrix corresponding to the sequence is developed. For each row *(i)* of the matrix, the data-dependencies above the leading-diagonal are summed and added to the sum of data-dependencies above the leading-diagonal in the corresponding column. This is repeated for each of the *n* row/column pairs of the $(n \times n)$ matrix. Figure 4.3 illustrates how Function 1 is evaluated.

(2) For each of the *n* activities in each string, the operator then evaluates the following conditional function;

> *Function (2):* If the result of *Function (1)* multiplied by the *Probability of Mutation* P_m (between 0-1, as defined by the user) is greater than a random

number chosen between zero and (n - 1), then the associated activity is swapped with another activity chosen at random. Else no swapping occurs.

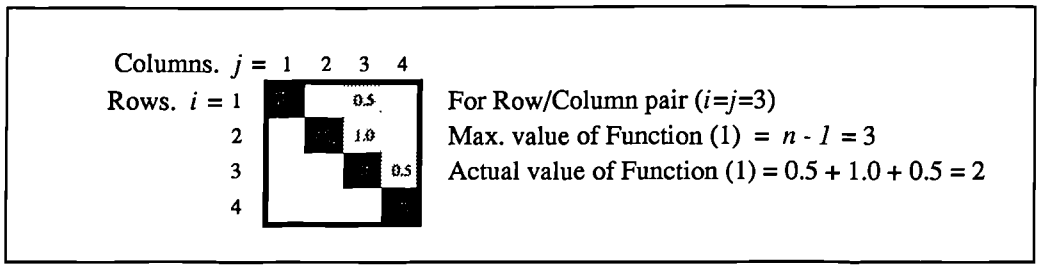


Figure 4.3: Evaluating Function (1)

This loop continues until *Function 1 and Function 2* have been evaluated for all n activities in each of the strings in the population. This operator uses the structure of the matrix in combination with the random adaption that works so well with genetic algorithm search procedures. By looking at both the activity row *(i)* and the corresponding activity column *(j=i)*, the maximum number of non-empty matrix positions above the leading diagonal is always the same. That is, for each activity's row/column pair, the maximum value of *Function 1* is always equal to *(n-1)*, corresponding to all of the matrix positions above the leading diagonal in the row/column pair filled with 1's (*See* Figure 4.3). The minimum value of *Function 1* is zero, corresponding to all the matrix positions above the leading diagonal in the row/column pair filled with 0's.

When dealing with the minimisation of iteration the objective is to reduce the number of data-dependencies above the leading diagonal by re-sequencing activities. Based on it's formulation, a high value of *Function (1)* indicates that an activity in it's current string position has a strong adverse effect on the fitness of the string under investigation. This is because a high value of the function implies that there are a large number of dependencies above the leading diagonal. It would therefore seem logical to try and reposition the activities contributing most to this degradation of fitness. However this movement should be random in nature, otherwise the search becomes excessively directed.

In summary, the choice of movement is carried out using conditional Function (2) as worded in Equation 4.2. Function (1) has a maximum value of n-1 and a minimum value of zero. The higher the value of Function (1) the greater the chance of a swap occurring. In addition, the higher the value of P_m, the higher the chance of swap. The advantage of such an operator

means that mutation occurs most vigorously at the beginning of the search where there are high values of Function (1). However, as the search progresses, and convergence begins to occur, the heuristic mutation occurs less frequently.

IF {*Function (1)* × P_m > a random number between 0 and *(n - 1)* }
THEN swap
ELSE no swap.
[Equation 4.2]

4.4.4 The Minimisation Problem

Genetic algorithms are based on natural selection, that is survival of the fittest. The fittest implies the largest, and so genetic algorithms naturally search for those strings that exhibit the largest fitness. In this way, genetic algorithms default to maximisation applications. In order to solve minimisation problems, some minor modifications to the standard genetic algorithm are necessary.

Minimisation problems can be adapted using a parameter referred to as *fitness measure.* The fitness measure is defined as the largest fitness in the Omega Pool plus a small increment. This small increment is derived using a second parameter referred to as *selection bias* and, typically, is chosen by the user to lie within the range (0 - 0.1). After ranking the strings in the Omega Pool according to their fitness, the fitness measure is calculated according to Equation 4.3.

Fitness measure = (largest fitness) × (1 + selection bias) *[Equation 4.3]*

Prior to selection, the fitness of each string in the Omega Pool is modified and recalculated as the difference between the fitness measure and the original fitness of each string, according to Equation 4.4.

Modified fitness = (fitness measure) - (original fitness) *[Equation 4.4]*

In this way, the *modified fitness* is large when the *original fitness* is small. Therefore, for minimisation problems, the genetic algorithm is "tricked" into believing that those strings exhibiting the lowest original fitness values are in fact the fittest. The selection bias is

necessary to ensure that even the string with the largest original fitness still has a chance of survival. Without this selection bias, such a string would have a modified fitness equal to zero, and would therefore have a zero probability of survival. In order for the genetic algorithm to work most effectively, all strings should have a chance of surviving into the next generation.

4.4.5 Termination Conditions

As illustrated previously in Figure 4.2, the genetic algorithm moves from generation to generation selecting, crossing and mutating strings until some condition for termination is met. The *Optimal Sequencer Genetic Algorithm* incorporates two such conditions.

Condition 1: Every *n* generations, the genetic algorithm stores the largest *modified* fitness found so far. It compares this value with the largest *modified* fitness that existed *n* generations ago. If the value of *stop check*, as defined in Equation 4.5, falls below a pre-set value which represents the lowest acceptable improvement, then the genetic algorithm search is terminated.

$$stop\ check = (\text{largest fitness}_{\text{n generations ago}} - \text{largest fitness}_{\text{now}}) / (\text{largest fitness}_{\text{n generations ago}}) \quad \textit{[Equation 4.5]}$$

Condition 2: If the genetic algorithm is not stopped by Condition 1 in the meantime, the genetic algorithm is terminated after a maximum number of pre-set generations.

4.4.6 Sequence Constraints

So far the genetic algorithm search technique has been described in terms of it's application to an *unconstrained* version of the *ActRes Problem*. However, the *Optimal Sequencer Genetic Algorithm* has been developed to incorporate sequence constraints. In the unconstrained version of the *ActRes Problem*, activities are sequenced purely on the basis of data-dependencies. However, certain problems may contain one or more constraints that must be satisfied in order to avoid the creation of an infeasible sequence. In this case, sequence constraints unrelated to data-dependency may exist such that certain activities must precede others.

Such constraints may relate to preferences based upon experience or the need for certain activities to be sequenced at the beginning of the product's design-development phase such that information related to long-lead production items can be released as early as possible. Other constraints may relate to the timed release of information to controlling authorities and classification societies for plan approval.

One way of dealing with constraints, suggested by Goldberg [Goldberg 89], is to use *penalties*. Based on such an approach, the constrained version of the *ActRes Problem* can be transformed to an unconstrained version by associating a *penalty value* with all derived sequences that contain constraint violations. For minimisation problems, this penalty is then added to the original fitness associated with the constraint-violating sequence such that, after modification using the fitness measure, the modified fitness of the constraint-violating sequence is reduced. In this manner, according to the rules of selection, such a sequence has a reduced chance of proceeding into the next generation.

However, whilst a specific constraint violating sequence contains unwanted genetic material in terms of two or more activities arranged in an unacceptable order, it may otherwise contain very desirable genetic material. Therefore the choice of penalty should be such that whilst constraint violating sequences have a reduced chance of surviving selection, such sequences still maintain a fair chance of surviving selection in order to go on to receive genetic operation.

One way of ensuring that constraint-violating sequences survive selection is to use a *penalty function* rather than the arbitrary *penalty value* previously discussed. The penalty function is represented by a probability distribution. Based upon this distribution, every time a sequence violation is encountered, the fitness is penalised through the incorporation of a randomly selected penalty value that varies about a mean within a specified variance. The mean is updated regularly and is set to half the fitness of the least fit sequence found so far. In addition, the distribution can be skewed towards higher penalty values.

4.5 The Heuristic Local Search

The genetic algorithm is effective when used to search an extremely large and noisy search space. As a result of it's random nature it has the ability to search over a wide spread of the search space without becoming trapped at local optima. As the genetic algorithm proceeds

through it's search, solutions begins to converge, primarily through the application of the selection operator.

During the search, crossover ensures propagation of solutions, whilst mutation improves the chances that genetic material will not be lost completely. However, as the search converges, the crossover and mutation operators become less effective as the diversity of the search space reduces and strings in the population begin to share very similar properties. In this respect, strings that share similar properties, when crossed and mutated do not yield significantly different strings.

In this way, the genetic algorithm produces the largest improvements during the *primary* stages of a search when the population is most diverse. Figure 4.4 represents a typical genetic algorithm *convergence plot* and can be used to illustrate this point.

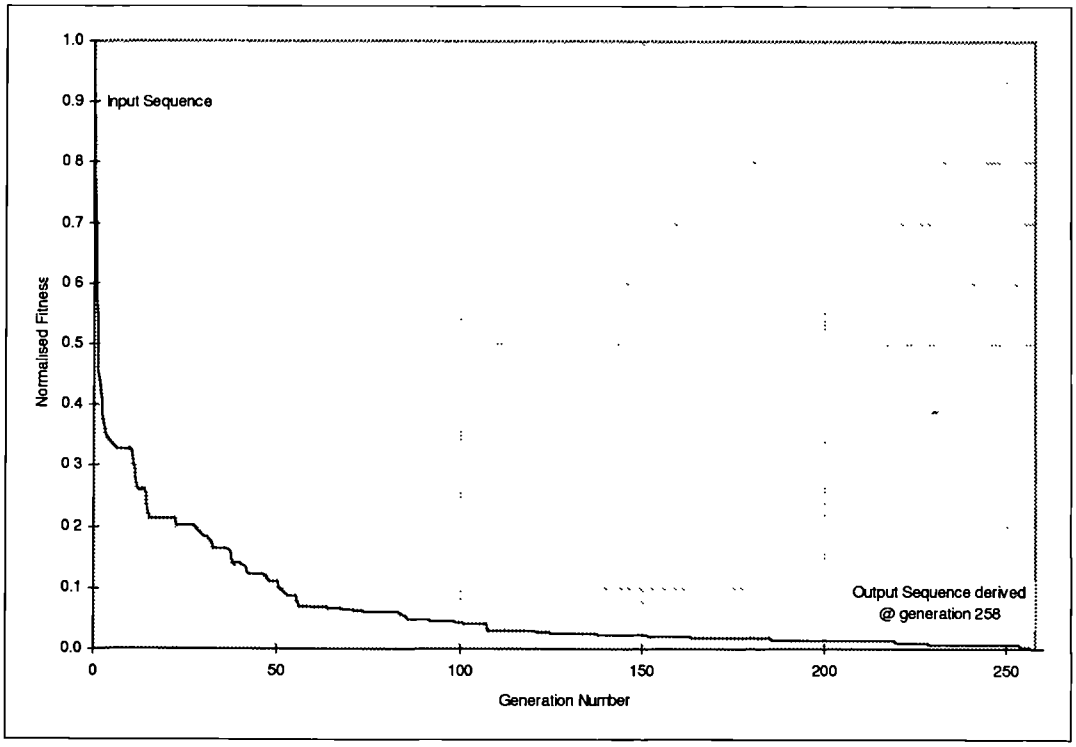


Figure 4.4: A typical genetic algorithm convergence plot (Based on experimentation - See Sub-Section 4.6.1)

It can be seen from the plot that, whilst it takes 258 generations for the search to converge to a solution, it is during initial generations of the search that the rate of convergence is highest. In fact, during the first 25 generations, the search completes 80% of it's journey towards convergence. After 50 generations 90% of the search has been completed, and after 75 generations 95% of the search has been completed. It then takes a further 183 generations to

complete the remaining 5% of the journey to convergence. Based upon these observations, it can be established that the genetic algorithm is very effective at pruning the search space, however it's efficiency falls off rather sharply with advancing generations.

On the other hand, local search techniques are not very effective when faced with large search spaces because they will tend to converge towards a local optimum. However once the search space has been pruned by a genetic algorithm, the subsequent application of a local search may offer the most efficient way of moving even closer to the global optimum.

Again, the choice of technique for the secondary local search is strongly related to the nature of the search space associated with the problem at hand. Whilst the search space may be dramatically pruned using a GA, it continues to be very noisy and discontinuous. Therefore, the inadequacies of calculus-based techniques, described at the beginning of Section 4.2, are still pertinent. However, as a result of search space pruning, the major inadequacy of a heuristic-based search technique is pre-empted.

Based upon these observations, the technique adopted for the secondary local search consists of a number of heuristics that, using the solution derived from the *Optimal Sequencer Genetic Algorithm* as a base, search locally for the global optimum. Since the performance of heuristics depends on problem characteristics and it is quite difficult to predict beforehand the most efficient heuristic for a given problem, eight different heuristics have been incorporated into the *Heuristic Local Search.* Each heuristic is actioned in order to derive a different *priority list.* Each priority list contains a ranked order of activities that are to be moved relative to the other activities, in turn, according to this ranking.

Heuristic 1: Starting at the right-hand side of the matrix and stepping backwards through the sequence of activities, if the activity contains a data-dependency above the leading diagonal in it's matrix-column, then the activity is placed into the next position in the priority list.

Heuristic 2: Starting at the right-hand side of the matrix and stepping backwards through the sequence of activities, regardless of whether or not a data-dependency exists above the leading diagonal in it's matrix-column, the activity is placed into the next position in the priority list. In this case, the priority list contains a list of all activities in reverse order.

Heuristic 3: Starting at the left-hand side of the matrix and stepping forwards through the sequence of activities, if the activity contains a data-dependency above the leading diagonal in it's matrix-column, then the activity is placed into the next position in the priority list.

Heuristic 4: Starting at the left-hand side of the matrix and stepping forwards through the sequence of activities, regardless of whether or not a data-dependency exists above the leading diagonal in it's matrix-column, the activity is placed into the next position in the priority list. In this case the priority list contains a list of all activities in order.

Heuristic 5: Starting at the top of the matrix and stepping down through the *sequence of* activities, if the activity contains a data-dependency above the leading diagonal in it's matrix-row, then the activity is placed into the next position in the priority list.

Heuristic 6: Starting at the bottom of the matrix and stepping up through the sequence of activities, if the activity contains a data-dependency above the leading diagonal in it's matrix-row, then the activity is placed into the next position in the priority list.

Heuristic 7: For each activity matrix-row, the procedure calculates the sum of the data-dependencies in the current row, each multiplied by the sum of the horizontal and vertical distances from the bottom left-hand corner of the matrix. The priority list is then created by ranking all activities in an ascending order of this summed product.

Heuristic 8: For each activity matrix-column, the procedure calculates the sum of the data-dependencies in the current column, each multiplied by the sum of the horizontal and vertical distances from the bottom left-hand corner of the matrix. The priority list is then created by ranking all activities in an ascending order of this summed product.

The heuristic-based local search can be described according to the following algorithm;

Step 1: Set Heuristic Number = 1

Step 2: Set Iteration Number = 1

Step 3: Set the sequence derived from the *Optimal Sequencer Genetic Algorithm* as the *best-sequence-so-far*. (The genetic algorithm-derived sequence is returned to before the application of a new heuristic since a heuristic applied to a converged solution that has been derived from a previous heuristic yields no further improvement).

Step 4: Create the ranked priority list by applying the current search heuristic to the sequence of activities in the *best-sequence-so-far*.

Step 5: Choose the first activity in the priority list and move the chosen activity to all possible positions in the *best-sequence-so-far*. Each movement results in a new sequence that can be evaluated for fitness. After the chosen activity has been moved to all possible positions in the sequence, store the resulting fittest sequence. If this sequence is fitter than the *best-sequence-so-far*, then store as the *best-sequence-so-far*.

Step 6: Choose the next activity in the priority list and move the chosen activity to all possible positions in the previously stored *best-sequence-so-far*. Again, after the chosen activity has been moved to all possible positions in the sequence, store the resulting fittest sequence. If this sequence is fitter than the *best-sequence-so-far*, then store as the *best-sequence-so-far*.

Step 7: Repeat Step 6 until all activities in the ranked priority list have been moved.

Step 8: If the *best-sequence-so-far*, as derived from Step 7 is superior to that derived in Step 5, then the current heuristic is still producing improved sequences. Therefore continue searching with the current heuristic, increment Iteration Number and go to Step 4, otherwise the current heuristic has not yielded any improvement and has reached convergence. In such a case it is necessary to apply the next heuristic to the genetic algorithm-derived sequence. Increment Heuristic Number. If Heuristic Number > 8, then Stop, otherwise go to Step 2

4.6 The GA-Local Search Procedure

The *GA-Local Search Procedure*, used to derive a near optimal sequence of activities, can be summarised as the *primary* application of the *Optimal Sequencer Genetic Algorithm* to deliver a *premature solution* which is subsequently used as input to the *Heuristic Local Search* in

order to deliver the *final solution.* Whilst the *Heuristic Local Search* operates without user interaction, the *Optimal Sequencer Genetic Algorithm* allows the user to vary five parameters which include;

1. Population Size
2. Number of Generations
3. Generation Gap
4. Probability of Crossover (P_c)
5. Probability of Mutation (P_m)

Returning to Section 4.3, where the standard genetic algorithm is explained; the technique begins with a single input string which is randomly resequenced a number of times in order to develop a population of strings. The number of times this initial resequencing occurs, and hence the number of newly developed strings equates to the population size. This population size should ideally be set to an appropriate number in order to ensure that the search is spread as wide as possible over the search space. The diversity so introduced helps to prevent premature convergence to local optima.

As previously highlighted, as the number of activities to be sequenced increases, the size of the problem and the associated search space increases exponentially. Therefore, in order to maintain sufficient diversity, population size has to be increased in relation to problem size. However, an increased population size entails more function evaluations and hence increased computational processing time since there are more strings to be evaluated.

If computational effort was not an issue, population size could be set to a very large number. However this is not feasible and therefore, for a given size of problem, a search for the most appropriate setting for population size should be focused upon investigating the trade-off between population size, computational effort, and solution quality. In this context, solution quality refers to the proximity of the derived solution to the global optimum.

The number of generations is determined largely by the population size. A large population size ensures a high degree of initial diversity. As the genetic algorithm proceeds through it's search, the search space begins to converge and hence diversity is reduced. Naturally, the larger the initial diversity, the more generations it will take to reach convergence.

Therefore, whilst final solution quality is likely to be higher with a large population size, a larger number of generations will be necessary in order to reach convergence. Conversely, a small population size with a reduced initial diversity reaches convergence more rapidly and hence requires fewer generations. This may be at the expense of solution quality since a less thorough search is involved. Here again, computational effort becomes an issue since an increase in number of generations entails more function evaluations. Therefore, for a given size of problem, a search for the most appropriate setting for number of generations should be focused upon investigating the trade-off between number of generations, computational effort, and solution quality.

Generation gap, probability of crossover (P_c), and probability of mutation (P_m), all relate to the degree of genetic operation that is applied during the genetic algorithm search. The choice of settings for these three parameters have very little impact upon computational effort and as a result, a search for the most appropriate settings for generation gap, P_c and P_m can be focused solely on finding those settings which consistently derive high quality solutions.

From this introduction it may be observed that, whilst the choice of settings for all five parameters have a strong impact upon solution quality, it is population size and number of generations which have the strongest impact upon computational effort and hence procedural efficiency. Furthermore, as may become clear in the following text, when compared against the sole application of the *Optimal Sequencer Genetic Algorithm*, it is observed that the *GA-Local Search Procedure* offers the potential for supporting a reduction in population size and number of generations.

Consequently, the *GA-Local Search Procedure* offers the potential for a significant improvement in procedural efficiency, without a loss of ultimate solution quality. This is due to the fact that, whilst a reduction in population size and number of generations means that the premature solution derived by the *Optimal Sequencer Genetic Algorithm* may be of a reduced quality, if the *Heuristic Local Search* is applied to a premature solution at the most appropriate stage of the genetic algorithm search, a high quality final solution can still result.

Consequently, for a specific problem size, a search for the most appropriate settings for population size and number of generations should be focused upon finding settings that result in an acceptable premature solution from the *Optimal Sequencer Genetic Algorithm* that,

when used as input to the *Heuristic Local Search*, ultimately leads to the derivation of consistently high quality solutions. Consider the convergence plots in Figure 4.5, which illustrate the point more clearly.

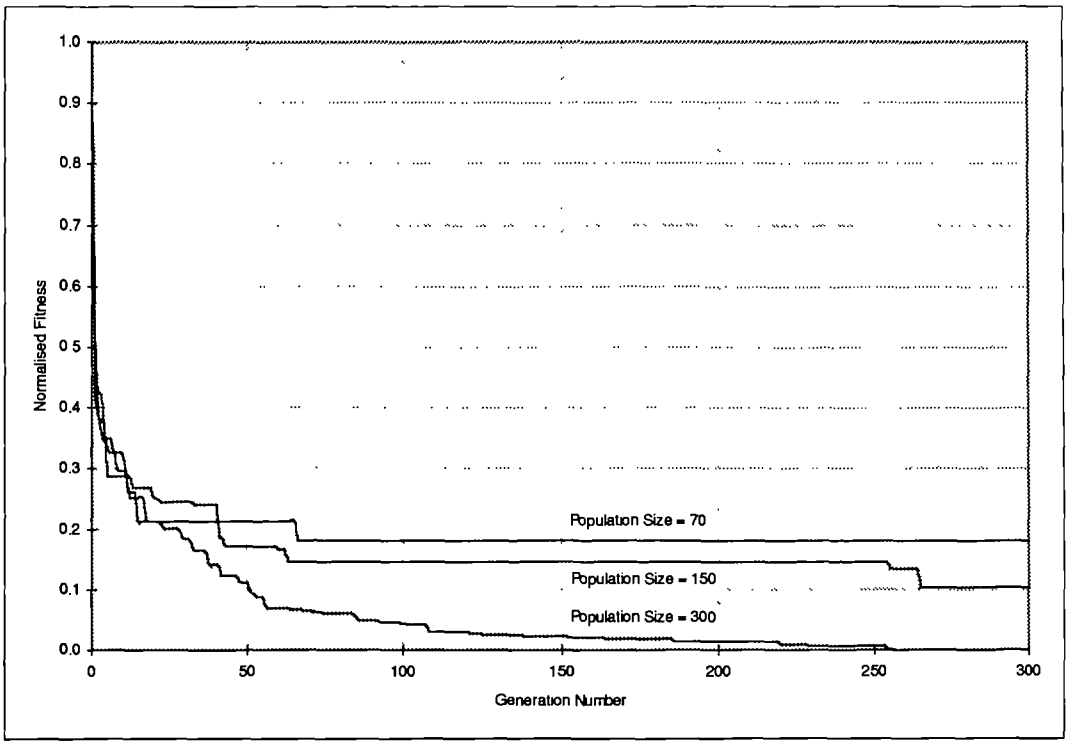


Figure 4.5: Variation of genetic algorithm solution quality with number of generations & population size

In order to derive the highest quality solution, as defined by a solution having a normalised fitness equal to zero, the *Optimal Sequencer Genetic Algorithm*, applied in isolation, requires over 250 generations and a population size equal to 300. From the graph, it may be observed that the *Optimal Sequencer Genetic Algorithm* is most effective during the first 75 generations. During these preliminary generations, on average, the search moves 87% of the way towards convergence.

Here, when the *Optimal Sequencer Genetic Algorithm* is stopped at the 75th generation, and the premature solutions, relating to each of the three population sizes, are fed as input into the *Heuristic Local Search*, all three ultimate solutions equate to the previously derived solution having a normalised fitness equal to zero. As a result, the *GA-Local Search Procedure* allows the number of generations to be reduced from 300 to 75, whilst the population size can be kept to a minimum of 70. Because the processing time associated with the *Heuristic Local Search* is much faster than the *Optimal Sequencer Genetic Algorithm*, these reductions in population size and number of generations represent a reduction in overall computational

effort, and hence lead to increased procedural efficiency. This observation is investigated further in the following sub-section.

4.6.1 Experimentation

Whilst there are established parameter settings for traditional genetic algorithm applications [Glover 93, Davis 89, Schaffer 89] few published notes are to be found relating to the most appropriate settings for combinatorial problems such as the *ActRes Problem.* In addition, as observed from the preceding text, the *GA-Local Search Procedure* can have a strong impact upon population size and number of generations.

Syswerda [Syswerda 91a] recommends that appropriate settings are to be found from experimentation. As a result, five test cases of varying size, summarised in Table 4.3, have been used in order to develop a clear understanding of the most effective and efficient genetic algorithm parameter settings for a wide range of problem sizes. Using the test cases it is possible to observe and analyse how the variation of parameter settings affects the results derived from the *GA-Local Search Procedure.*

Test case	Description	No. of activities
1	Gebala & Eppinger Numerical Matrix [Gebala 91]	12
2	Klein-Goldberger Binary Matrix [Steward 81a]	20
3	Black Brake Design Matrix [Steward 91]	34
4	Austin Building Design Matrix [Austin 89]	51
5	Stewards Thermo-Physics matrix [Steward 81a]	60

Table 4.3: Test cases

Later, in Chapter 5, it is explained how each of the five test cases has been used to derive a near optimal sequence of activities based on the objective of minimising iteration. However, for the current purpose of determining the most effective and efficient operational mode of the *GA-Local Search Procedure*, no further knowledge of the specific objective is required.

As a result of experience gained from early studies and experimentation not recorded here, a basic understanding of the most appropriate genetic algorithm parameter settings is already known. These preliminary settings have been used as a basis for further refinement. Using the input sequence and it's associated matrix, each test case is subjected to the test procedure detailed below;

Step1: Select a *population size* in order to ensure that the *Optimal Sequencer Genetic Algorithm* converges at the highest quality solution possible. This initial population size, which depends on the size of the problem and is therefore different for each of the five test cases, is derived based on experience.

Step 2: Set *number of generations* in order to ensure that the *Optimal Sequencer Genetic Algorithm* reaches convergence. Maintain this setting for all experiments and trials on the current test case. Again, the number of generations is derived based on experience.

Step 3: Using the pre-set population size implement the *Optimal Sequencer Genetic Algorithm* for all combinations of *generation gap* equal to 0.5, 0.7, and 0.9, *probability of crossover* equal to 0.9, 0.6, 0.3, and 0.0 and *probability of mutation* equal to 0.5, 0.1, 0.01, and 0.0. (A total of $3 \times 4 \times 4 = 48$ combinations)

Step 4: Because of the random nature of the *Optimal Sequencer Genetic Algorithm*, the same solution cannot be guaranteed between two separate trials of identical parameter settings. As a result, each of the 48 combinations is subjected to five trials.

Having completed Steps 1-4, five trials each of 48 parameter combinations yield a total of 240 (5×48) *Optimal Sequencer Genetic Algorithm*-derived solutions. It is now necessary to investigate whether, using these solutions as input to the *Heuristic Local Search*, further improvements can be found. In addition, it is necessary to investigate whether or not the application of the *Heuristic Local Search* to premature solutions yields equally high quality solutions. The objective here is to find the minimum number of generations for the current population size that still yields high quality solutions. Therefore;

Step 5: From the 240 solutions, a spread of five is chosen for further analysis. The best and worst *Optimal Sequencer Genetic Algorithm*-derived solutions are chosen and three other solutions are selected which, in terms of fitness, lie proportionally between these two extremes.

Step 6: For each of these five solutions, the solution derived at convergence, as well as the premature solutions derived at generation numbers corresponding to 75%, 50%, and 25% of the maximum number of generations are used as input to the *Heuristic Local Search.* In this way 20 premature solutions are used as input to the *Heuristic Local Search.*

Step 7: Each case is tested using three varied population sizes. Therefore, having completed Steps 1-6 for the initial population size, set population size equal to 50% of that selected in Step 1, and repeat Steps 3-6 a second time. Then set population size equal to 25% of that selected in Step 1, and repeat Steps 3-6 for a third and final time.

Having completed the first test case, repeat steps 1-7 for each of the remaining four test cases

4.6.2 Results & Analysis

Using the seven-step test procedure outlined above, a total of 780 solutions are derived for each test case (240 genetic algorithm trials for each of 3 population sizes equals 720, to which are added a further 20 local search trails for each of the 3 population sizes, i.e. 60 trials). The solutions are normalised in order to simplify the way in which the results of the experimentation are reported. Since we are dealing with a minimisation problem, of the 780 solutions, the highest quality solution when normalised is represented by zero, the lowest quality solution is represented by unity, and based on their relative fitness, the remaining 778 derived solutions are scaled between zero and unity accordingly.

Ultimately for each test case, the objective is to find the combination of generation gap, probability of crossover, and probability of mutation such that the combined application of the two search procedures, referred to as the *GA-Local Search Procedure*, consistently returns a normalised solution equal to zero. Furthermore, with procedural efficiency as an issue, such solutions should be derived using minimum population size and a minimum number of generations.

The final solution sequences and their associated design structure matrices are detailed in Chapter 5 where the results are compared against previously published solutions. The results of all test cases are summarised in a number of graphs that follow. In addition, for the sake of clarity, the complete set of results for Test 5 is presented in Tables 4.4a - 4.4d whilst the remaining results are documented in Appendix III.

Max. Gens:	700

Of the 780 experiments, Overall Worst Fitness = 16,025,200 (Normlsd = 1)
Overall Best Fitness = 2,757,320 (Normlsd = 0)
Normalised Fitness =(Derived Fitness-Overall Best Fitness) / (Overall Worst-Best)

Population size:	600

Set Number	Gen gap	Pc	Pm	Fitness					Best Fitness	Normlsd Best Fitness	Average Fitness	Normlsd Average Fitness
				Trial 1	Trial 2	Trial 3	Trial 4	Trial 5				
1	0.5	0.9	0.50	13,854,100	12,391,000	14,958,700	13,227,800	11,903,600	11,903,600	0.689	13,267,040	0.792
2			0.10	14,558,500	14,567,100	12,763,200	14,070,400	14,296,600	12,763,200	0.754	14,051,160	0.851
3			0.01	12,507,300	12,256,600	12,260,200	11,927,200	14,322,900	11,927,200	0.691	12,654,840	0.746
4			0.00	12,632,373	12,379,166	12,382,802	12,046,472	14,466,129	12,046,472	0.700	12,781,388	0.756
5		0.6	0.50	13,693,800	13,693,500	13,211,600	12,177,100	12,027,000	12,027,000	0.699	12,960,600	0.769
6			0.10	11,032,300	12,345,700	10,582,900	11,029,500	11,447,700	10,582,900	0.590	11,287,620	0.643
7			0.01	3,400,150	3,586,660	3,596,640	3,491,350	3,512,340	3,400,150	0.048	3,517,428	0.057
8			0.00	3,434,152	3,622,527	3,632,606	3,526,264	3,547,463	3,434,152	0.051	3,552,602	0.060
9		0.3	0.50	10,970,800	10,975,300	12,202,100	11,252,300	11,088,800	10,970,800	0.619	11,297,860	0.644
10			0.10	3,550,480	3,298,880	3,656,050	3,465,350	3,556,670	3,298,880	0.041	3,505,486	0.056
11			0.01	3,558,320	3,644,890	3,800,490	4,150,780	3,568,640	3,558,320	0.060	3,744,624	0.074
12			0.00	3,593,903	3,681,339	3,838,495	4,192,288	3,604,326	3,593,903	0.063	3,782,070	0.077
13		0.0	0.50	13,556,862	13,556,565	13,079,484	12,055,329	11,906,730	11,906,730	0.690	12,830,994	0.759
14			0.10	10,921,977	12,222,243	10,477,071	10,919,205	11,333,223	10,477,071	0.582	11,174,744	0.634
15			0 01	3,366,149	3,550,793	3,560,674	3,456,437	3,477,217	3,366,149	0.046	3,482,254	0.055
16			0.00	3,399,810	3,586,301	3,596,280	3 491 001	3,511,989	3,399,810	0.048	3,517,076	0.057
17	0.7	0.9	0.50	14,459,900	13,971,500	14,550,600	13,869,100	13,507,000	13,507,000	0.810	14,071,620	0.853
18			0.10	13,507,400	13,387,500	13,280,700	11,560,000	12,165,200	11,560,000	0.663	12,780,160	0.755
19			0.01	11,579,300	11,605,600	11,650,400	11,123,000	10,747,400	10,747,400	0.602	11,341,140	0.647
20			0.00	11,695,093	11,721,656	11,766,904	11,234,230	10,854,874	10,854,874	0.610	11,454,551	0.656
21		0.6	0.50	13,210,200	11,593,200	12,844,200	12,376,900	12,268,400	11,593,200	0.666	12,458,580	0.731
22			0.10	7 609,760	8,931,470	8,226,030	8,225,270	7,961,060	7,609,760	0.366	8,190,718	0.410
23			0.01	3,425,020	3,016,820	3,857,090	2,963,620	3,629,180	2,963,620	0.016	3,378,346	0.047
24			0 00	3,459,270	3,046,988	3,895,661	2,993,256	3,665,472	2,993,256	0.018	3,412,129	0.049
25		0.3	0.50	10,122,200	10,901,800	10,990,000	7,886,970	9,598,130	7,886,9"0	0.387	9,899,820	0.538
26			0.10	3,720,"60	3,685,100	3,054,790	3,363,290	3,610,720	3,054,790	0.022	3,486,932	0.055
27			0.01	3,517,320	3,817,590	4,065,250	3,537,170	3,617,980	3,51"320	0.057	3,711,062	0.072
28			0 00	3,552,493	3,855,766	4,105,903	3,572,542	3,654,160	3,552,493	0.060	3,"48,173	0.075
29		0.0	0.50	13,078,098	11,477,268	12,715,758	12,253,131	12,145,716	11,477,268	0.657	12,333,994	0.722
30			0.10	7,533,662	8,842,155	8,143,770	8,143,017	7,881,449	7,533,662	0.360	8,108,811	0.403
31			0.01	3,390,"70	2,986,652	3,818,519	2,933,984	3,592,888	2,933,984	0.013	3,344,563	0.044
32			0.00	3,424,677	3,016,518	3,856,704	2,963,324	3,628,817	2,963,324	0.016	3,378,008	0.047
33	0.9	0.9	0.50	13,812,100	13,580,700	12,596,800	13,534,900	13,702,800	12,596,800	0.742	13,445,460	0.806
34			0.10	13,132,200	13,066,200	13,680,100	12,457,600	12,444,400	12,444,400	0.730	12,956,100	0.769
35			0.01	11,285,800	11,387,000	10,612,800	11,822,100	11,429,200	10,612,800	0.592	11,307,380	0.644
36			0.00	11,398,658	11,500,870	10,718,928	11,940,321	11,543,492	10,718,928	0.600	11,420,454	0.653
37		0.6	0.50	10,910,500	12,937,400	11,471,600	12,483,500	12,009,000	10,910,500	0.615	11,962,400	0.694
38			0.10	3,029,230	3,437,940	3,416,720	3,520,320	3,113,650	3,029,230	0.020	3,303,572	0.041
39			0.01	4,082,390	3,666,480	4,554,480	3,784,080	3,542,660	3,542,660	0.059	3,926,018	0.088
40			0.00	4,123,214	3,"03,145	4,600,025	3,821,921	3,578,087	3,578,087	0.062	3,965,278	0.091
41		0.3	0.50	9,953,050	10,110,200	10,245,800	9,734,740	10,501,700	9,734,740	0.526	10,109,098	0.554
42			0.10	3,596,580	3,621,"50	3,028,040	3,622,360	3,505,620	3,028,040	0.020	3,474,870	0.054
43			0.01	3,602,240	4,024,140	3,441,320	3,190,790	3,511,920	3,190,790	0.033	3,554,082	0.060
44			0.00	3,638,262	4,064,381	3,475,"33	3,222,698	3,547,039	3,222,698	0.035	3,589,623	0.063
45		0.0	0.50	10,801,395	12,808,026	11,356,884	12,358,665	11,888,910	10,801,395	0.606	11,842,776	0.685
46			0.10	2,998,938	3,403,561	3,382,553	3,485,117	3,082,514	2,998,938	0.018	3,270,536	0.039
47			0.01	4,041,566	3,629,815	4,508,935	3,746,239	3,507,233	3,507,233	0.057	3,886,758	0.085
48			0.00	4,081,982	3,666,113	4,554,025	3,783,702	3,542,306	3,542,306	0.059	3,925,625	0.088

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after 100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
23	4	2,963,620	2,780,390	0.002	2,780,390	0.002	2,780,390	0.002	2,780,390	0.002
39	3	4,554,480	3,147,750	0.029	3,147,750	0.029	3,147,750	0.029	2,878,220	0.009
22	4	8,225,270	2,780,390	0.002	2,757,320	0.000	2,935,150	0.013	3,190,520	0.033
3	4	11,927,200	2,899,390	0.011	2,899,390	0.011	2,899,390	0.011	2,899,390	0.011
1	3	14,958,700	3,219,460	0.035	3,264,240	0.038	2,940,530	0.014	2,953,080	0.015

Table 4.4a: Complete Results for Test 5 (Population Size = 600)

Population size:	300

Set Number	Gen gap	Pc	Pm	Fitness					Best Fitness	Normlsd Best Fitness	Average Fitness	Normlsd Average Fitness
				Trial 1	Trial 2	Trial 3	Trial 4	Trial 5				
49	0.5	0.9	0.50	13,459,000	14,336,200	13,914,200	14,709,800	14,748,000	13,459,000	0.807	14,233,440	0.865
50			0.10	14,182,200	13,584,400	10,905,500	13,313,900	14,373,100	10,905,500	0.614	13,271,820	0.792
51			0.01	11,821,900	12,904,900	11,552,900	13,423,600	12,076,800	11,552,900	0.663	12,356,020	0.723
52			0.00	11,940,119	13,033,949	11,668,429	13,557,836	12,197,568	11,668,429	0.672	12,479,580	0.733
53		0.6	0.50	13,665,600	12,467,100	13,747,000	11,453,100	14,200,000	11,453,100	0.655	13,106,560	0.780
54			0.10	11,654,100	11,638,300	10,754,000	12,123,700	11,852,300	10,754,000	0.603	11,604,480	0.667
55			0.01	3,666,650	3,534,070	3,765,660	3,719,110	3,709,280	3,534,070	0.059	3,678,954	0.069
56			0.00	3,703,317	3,569,411	3,803,317	3,756,301	3,746,373	3,569,411	0.061	3,715,744	0.072
57		0.3	0.50	10,055,000	12,121,000	12,212,800	11,636,300	11,532,500	10,055,000	0.550	11,511,520	0.660
58			0.10	3,521,920	3,455,260	3,504,230	3,042,300	3,546,300	3,042,300	0.021	3,414,002	0.049
59			0.01	3,794,510	3,327,310	3,249,720	3,449,400	3,506,490	3,249,720	0.037	3,465,486	0.053
60			0.00	3,832,455	3,360,583	3,282,217	3,483,894	3,541,555	3,282,217	0.040	3,500,141	0.056
61		0.0	0.50	13,528,944	12,342,429	13,609,530	11,338,569	14,058,000	11,338,569	0.647	12,975,494	0.770
62			0.10	11,537,559	11,521,917	10,646,460	12,002,463	11,733,777	10,646,460	0.595	11,488,435	0.658
63			0.01	3,629,984	3,498,729	3,728,003	3,681,919	3,672,187	3,498,729	0.056	3,642,164	0.067
64			0 00	3,666,283	3,533,717	3,765,283	3,718,738	3,708,909	3,533,717	0.059	3,678,586	0.069
65	0.7	0.9	0.50	12,804,000	13,004,800	14,920,700	14,325,400	13,683,200	12,804,000	0.757	13,747,620	0.828
66			0.10	11,989,700	13,417,200	11,998,600	12,886,300	13,615,100	11,989,700	0.696	12,781,380	0.756
67			0.01	9,923,920	10,116,000	9,954,690	10,207,300	10,738,800	9,923,920	0.540	10,188,142	0.560
68			0.00	10,023,159	10,217,160	10,054,237	10,309,373	10,846,188	10,023,159	0.548	10,290,023	0.568
69		0.6	0.50	13,649,000	12,825,800	13,401,200	12,514,200	13,756,600	12,514,200	0.735	13,229,360	0.789
70			0.10	3,583,160	3,634,050	3,716,110	3,482,260	3,127,660	3,127,660	0.028	3,508,648	0.057
71			0.01	3,139,770	3,811,090	4,922,470	3,510,810	3,828,570	3,139,770	0.029	3,842,542	0.082
72			0 00	3,171,168	3,849,201	4,971,695	3,545,918	3,866,856	3,171,168	0.031	3,880,967	0.085
73		0.3	0.50	11,108,700	11,415,000	9,756,130	10,470,700	10,013,200	9,756,130	0.528	10,552,746	0.588
74			0.10	3,570,100	3,191,530	3,558,980	3,423,430	4,410,660	3,191,530	0.033	3,630,940	0.066
75			0.01	3,500,520	3,658,440	3,918,660	4,010,860	3,354,490	3,354,490	0.045	3,688,594	0.070
76			0.00	3,535,525	3,695,024	3,957,847	4,050,969	3,388,035	3,388,035	0.048	3,725,480	0.073
77		0.0	0.50	13,512,510	12,697,542	13,267,188	12,389,058	13,619,034	12,389,058	0.726	13,097,066	0.779
78			0.10	3,547,328	3,597,710	3,678,949	3,447,437	3,096,383	3,096,383	0.026	3,473,562	0.054
79			0.01	3,108,372	3,772,979	4,873,245	3,475,702	3,790,284	3,108,372	0.026	3,804,117	0.079
80			0 00	3,139,456	3,810,709	4,921,978	3,510,459	3,828,187	3,139,456	0.029	3,842,158	0.082
81	0.9	0.9	0.50	11,671,600	14,453,100	15,306,900	13,414,400	14,129,400	11,671,600	0.672	13,795,080	0.832
82			0.10	12,414,000	8,746,270	12,123,400	11,594,400	13,089,600	8,746,270	0.451	11,593,534	0.666
83			0.01	6,912,070	7,566,830	4,202,670	7,881,460	3,682,460	3,682,460	0.070	6,049,098	0.248
84			0.00	6,981,191	7,642,498	4,244,697	7,960,275	3,719,285	3,719,285	0.073	6,109,589	0.253
85		0.6	0.50	12,456,400	11,489,800	12,549,000	12,181,900	12,608,600	11,489,800	0.658	12,257,140	0.716
86			0.10	3,543,140	3,333,310	3,567,040	3,740,200	3,570,840	3,333,310	0.043	3,550,906	0.060
87			0.01	4,081,140	3,866,640	4,141,820	3,967,460	3,539,620	3,539,620	0.059	3,919,336	0.088
88			0.00	4,121,951	3,905,306	4,183,238	4,007,135	3,575,016	3,575,016	0.062	3,958,529	0.091
89		0.3	0.50	10,474,700	11,326,700	10,816,500	11,067,900	10,924,300	10,474,700	0.582	10,922,020	0.615
90			0.10	3,380,790	3,457,170	3,294,540	3,228,100	3,548,980	3,228,100	0.035	3,381,916	0.047
91			0.01	3,449,700	3,259,480	4,000,950	3,638,710	3,768,240	3,259,480	0.038	3,623,416	0.065
92			0.00	3,484,197	3,292,075	4,040,960	3,675,097	3,805,922	3,292,075	0.040	3,659,650	0.068
93		0.0	0.50	12,331,836	11,374,902	12,423,510	12,060,081	12,482,514	11,374,902	0.650	12,134,569	0.707
94			0.10	3,507,709	3,299,977	3,531,370	3,702,798	3,535,132	3,299,977	0.041	3,515,397	0.057
95			0.01	4,040,329	3,827,974	4,100,402	3,927,785	3,504,224	3,504,224	0.056	3,880,143	0.085
96			0.00	4,080,732	3,866,253	4,141,406	3,967,063	3,539,266	3,539,266	0.059	3,918,944	0.088

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after							
			100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
58	4	3,042,300	2,757,320	0.000	2,757,320	0.000	2,757,320	0.000	2,757,320	0.000
83	1	6,912,070	3,157,840	0.030	3,157,840	0.030	2,855,050	0.007	2,935,150	0.013
73	3	9,756,130	3,147,750	0.029	3,147,750	0.029	3,147,750	0.029	3,147,750	0.029
57	3	12,212,800	2,780,390	0.002	2,780,390	0.002	2,780,390	0.002	2,899,390	0.011
81	3	15,306,900	2,940,530	0.014	2,940,530	0.014	2,855,050	0.007	2,855,050	0.007

Table 4.4b: Complete Results for Test 5 (Population Size = 300)

Population size:	150

Set Number	Gen gap	Pc	Pm	Fitness					Best Fitness	Normlsd Best Fitness	Average Fitness	Normlsd Average Fitness
				Trial 1	Trial 2	Trial 3	Trial 4	Trial 5				
97	0.5	0.9	0.50	14,401,400	15,063,800	15,676,000	12,785,500	16,025,200	12,785,500	0.756	14,790,380	0.907
98			0.10	13,626,800	14,010,900	13,085,000	13,941,700	14,732,200	13,085,000	0.778	13,879,320	0.838
99			0.01	11,518,900	13,091,500	11,283,300	11,956,500	10,812,600	10,812,600	0.607	11,732,560	0.676
100			0.00	11,634,089	13,222,415	11,396,133	12,076,065	10,920,726	10,920,726	0.615	11,849,886	0.685
101		0.6	0.50	13,701,500	13,677,900	12,749,800	14,453,200	13,266,400	12,749,800	0.753	13,569,760	0.815
102			0.10	10,199,800	8,615,580	8,827,880	11,175,600	11,521,500	8,615,580	0.442	10,068,072	0.551
103			0.01	3,872,090	4,061,030	4,097,110	3,579,580	4,067,850	3,579,580	0.062	3,935,532	0.089
104			0.00	3,910,811	4,101,640	4,138,081	3,615,376	4,108,529	3,615,376	0.065	3,974,887	0.092
105		0.3	0.50	12,136,400	11,753,400	12,300,000	12,460,500	11,824,600	11,753,400	0.678	12,094,980	0.704
106			0.10	3,581,670	3,638,980	3,256,020	3,521,460	3,978,590	3,256,020	0.038	3,595,344	0.063
107			0.01	3,181,130	3,721,630	3,768,070	3,842,350	3,534,950	3,181,130	0.032	3,609,626	0.064
108			0.00	3,212,941	3,758,846	3,805,751	3,880,774	3,570,300	3,212,941	0.034	3,645,722	0.067
109		0.0	0.50	13,564,485	13,541,121	12,622,302	14,308,668	13,133,736	12,622,302	0.744	13,434,062	0.805
110			0.10	10,097,802	8,529,424	8,739,601	11,063,844	11,406,285	8,529,424	0.435	9,967,391	0.543
111			0.01	3,833,369	4,020,420	4,056,139	3,543,784	4,027,172	3,543,784	0.059	3,896,177	0.086
112			0.00	3,871,703	4,060,624	4,096,700	3,579,222	4,067,443	3,579,222	0.062	3,935,138	0.089
113	0.7	0.9	0.50	14,099,500	14,443,500	14,804,300	14,843,300	14,054,700	14,054,700	0.851	14,449,060	0.881
114			0.10	12,851,100	11,996,000	13,250,500	12,751,300	11,582,000	11,582,000	0.665	12,486,180	0.733
115			0.01	3,598,440	3,612,220	4,391,440	3,755,380	3,667,410	3,598,440	0.063	3,804,978	0.079
116			0.00	3,634,424	3,648,342	4,435,354	3,792 934	3,704,084	3,634,424	0.066	3,843,028	0.082
117		0.6	0.50	13,552,200	12,949,500	13,272,100	11,545,000	13,394,400	11,545,000	0.662	12,942,640	0.768
118			0.10	3,150,490	4,020,610	3,964,690	3,396,150	3,563,020	3,150,490	0.030	3,618,992	0.065
119			0 01	4,026,720	4,215,560	3,955,100	4,232,420	3,846,290	3,846,290	0.082	4,055,218	0.098
120			0.00	4,066,987	4,257,716	3,994,651	4,274,744	3,884,753	3,884,753	0.085	4,095,770	0.101
121		0.3	0.50	11,794,300	11,327,600	8,319,750	10,285,800	11,741,100	8,319,750	0.419	10,693,710	0.598
122			0.10	3,396,680	3,047,890	3,509,730	3,331,780	3,933,530	3,047,890	0.022	3,443,922	0.052
123			0.01	3,504,800	3,8"9,580	4,024,000	3,860,840	3,673,010	3,504,800	0.056	3,788,446	0.078
124			0.00	3,539,848	3,918,376	4,064,240	3,899,448	3,709,740	3,539,848	0.059	3,826,330	0.081
125		0.0	0.50	13,416,6"8	12,820,005	13,139,379	11,429,550	13,260,456	11,429,550	0.654	12,813,214	0.758
126			0.10	3,118,985	3,980,404	3,925,043	3,362,189	3,527,390	3,118,985	0.027	3,582,802	0.062
127			0.01	3,986,453	4,1"3,404	3,915,549	4,190,096	3,807,827	3,807,827	0.0"9	4,014,666	0.095
128			0.00	4,026,317	4,215,138	3,954,704	4,231,997	3,845,905	3,845,905	0.082	4,054,812	0.098
129	0.9	0.9	0.50	14,387,800	13,9"0,400	15,465,500	13,800,900	12,779,200	12,779,200	0.755	14,080,760	0.853
130			0.10	11,414,600	11,"22,800	11,005,600	12,912,000	11,951,200	11,005,600	0.622	11,801,240	0.682
131			0.01	3,944,400	3,563,940	3,623,730	4,553,990	3,513,660	3,513,660	0.057	3,839,944	0.082
132			0.00	3,983,844	3,599,579	3,659,967	4,599,530	3,548,797	3,548,797	0.060	3,878,343	0.084
133		0.6	0.50	12,451,900	12,"86,"00	9,966,520	13,371,300	13,699,200	9,966,520	0.543	12,455,124	0.731
134			0.10	3,582,400	3,088,880	4,625,910	3,969,360	3,463,610	3,088,880	0.025	3,746,032	0.075
135			0.01	4,063,040	4,326,500	3,773,640	3,666,580	3,751,860	3,666,580	0.069	3,916,324	0.087
136			0.00	4,103,6"0	4,369,765	3,811,376	3,703,246	3,789,379	3,"03,246	0.071	3,955,487	0.090
137		0.3	0.50	10,198,500	10,517,900	10,350,900	10,503,300	9,468,460	9,468,460	0.506	10,207,812	0.562
138			0.10	3,759,390	3,522,360	3,323,790	3,693,730	3,269,810	3,269,810	0.039	3,513,816	0.057
139			0 01	3,290,450	3,269,760	3,"81,510	4,141,820	3,819,240	3,269,760	0.039	3,660,556	0.068
140			0.00	3,323,355	3,302,458	3,819,325	4,183,238	3,857,432	3,302,458	0.041	3,697,162	0.071
141		0.0	0.50	12,327,381	12,658,833	9,866,855	13,237,587	13,562,208	9,866,855	0.536	12,330,573	0.722
142			0.10	3,546,576	3,057,991	4,579,651	3,929,666	3,428,974	3,057,991	0.023	3,708,572	0.072
143			0.01	4,022,410	4,283,235	3,735,904	3,629,914	3,714,341	3,629,914	0.066	3,877,161	0.084
144			0.00	4,062,634	4,326,067	3,773,263	3,666,213	3,751,485	3,666,213	0.069	3,915,932	0.087

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after 100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
122	2	3,047,890	2,780,390	0.002	2,780,390	0.002	2,780,390	0.002	2,780,390	0.002
102	3	8,827,880	2,855,050	0.007	2,855,050	0.007	2,964,360	0.016	3,147,750	0.029
137	2	10,517,900	3,157,840	0.030	3,219,460	0.035	2,780,390	0.002	2,935,150	0.013
114	3	13,250,500	2,940,530	0.014	2,940,530	0.014	2,780,390	0.002	2,780,390	0.002
97	5	16,025,200	3,177,690	0.032	3,177,690	0.032	2,855,050	0.007	2,935,150	0.013

Table 4.4c: Complete Results for Test 5 (Population Size = 150)

		Mean Normalised Best Fitness			
		Population Size			Mean
		600	300	150	Result
Generation Gap	0.5	0.398	0.384	0.385	0.389
	0.7	0.333	0.301	0.244	0.293
	0.9	0.298	0.224	0.220	0.248
Probabilty of Crossover	0.9	0.682	0.547	0.491	0.573
	0.6	0.267	0.252	0.241	0.253
	0.3	0.160	0.166	0.164	0.163
	0.0	0.263	0.247	0.236	0.249
Probabilty of Mutation	0.50	0.642	0.664	0.655	0.654
	0.10	0.347	0.266	0.262	0.292
	0.01	0.190	0.140	0.106	0.145
	0.00	0.194	0.143	0.109	0.149

Table 4.4d: Summary Results Table for Test 5

4.6.3 The Impact of Generation Gap, P_c and P_m on Solution Quality

As previously stated, the choice of settings for generation gap, probability of crossover (P_c), and probability of mutation (P_m), should be focused solely upon finding those settings which consistently derive high quality solutions. Figures 4.6-4.8 illustrate plots of *mean normalised fitness* against a range of settings for each parameter. A number of observations are consistent with all three plots;

- Because of the relatively small problem size, the results of Test 1 are somewhat inconsistent with the results associated with the other four test cases. Furthermore, the quality of solutions associated with Test 1 varies only marginally over the range of all three parameter settings. Again, this is considered to be a result of the relatively small search space associated with a matrix of only 12 activities. For this reason, subsequent observations are made without reference to the results of Test 1.

- As problem size increases, the variation of parameter settings begins to affect the quality of derived solutions more significantly. This is concluded from the fact that as problem size increases, the difference between the best and worst results increases.

- As problem size increases, the general trend of higher normalised fitness over the range of parameter settings illustrates an increased reliance upon the local search for the derivation of the highest quality solution. This is concluded from the fact that as problem size increases, the best results derived by the genetic algorithm are found at increasing distance from the fittest derived solution (normalised fitness = 0).

- Whether the matrix contains scaled dependencies (Test Cases 1 & 4), or the matrix simply contains binary dependencies (Test Cases 2, 3, and 5) appears to have no impact upon the results. (*See* Appendix II.1 for an explanation of binary and scaled dependencies)

- As the number of activities in the matrix increases with each test case, there is a general trend for the best choice of parameter to occur at higher values of normalised fitness with the exception of Test Case 5. However, it is considered that the anomaly of Test Case 5 does not require further investigation because, as may be observed from Figures 4.6- 4.8, the best choice of parameter value for generation gap, P_c, P_m are almost completely independent of the number of activities in the matrix.

An observation of each of the plots in turn leads to a number of conclusions.

Generation gap; DeJong [DeJong 75] suggests that the non-overlapping approach, that is the setting of generation gap equal to unity, is best for optimisation studies where ultimate convergence is the overriding concern. However, since the *Heuristic Local Search* is applied to a pre-convergent genetic algorithm solution, convergence of the genetic algorithm is not considered to be an overriding concern. As such, the suggestions of DeJong are not relevant here. In this respect a degree of overlap is desirable in order to ensure that a number of high quality solutions are maintained throughout successive generations. From an observation of Figure 4.6, the most effective choice of generation gap appears to lie within the range 0.7-0.9.

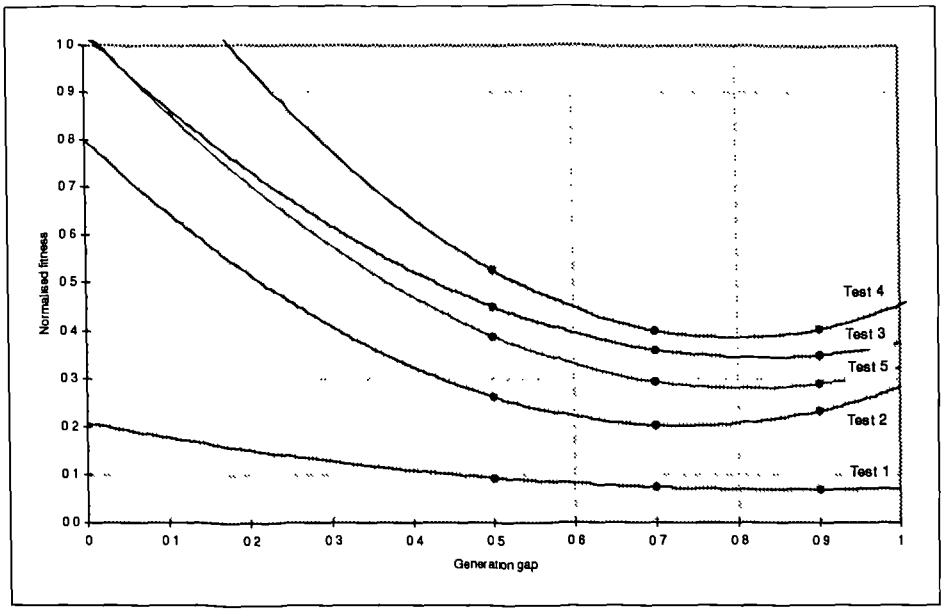


Figure 4.6: Variation of normalised fitness with generation gap

Probability of crossover (P_c); An observation of Figure 4.7 seems to indicate that the choice of P_c appears to have limited impact upon solution quality as observed by the relatively shallow plots. However, with the exception of Test 1, the most effective choice of this parameter centres on a value equal to 0.3

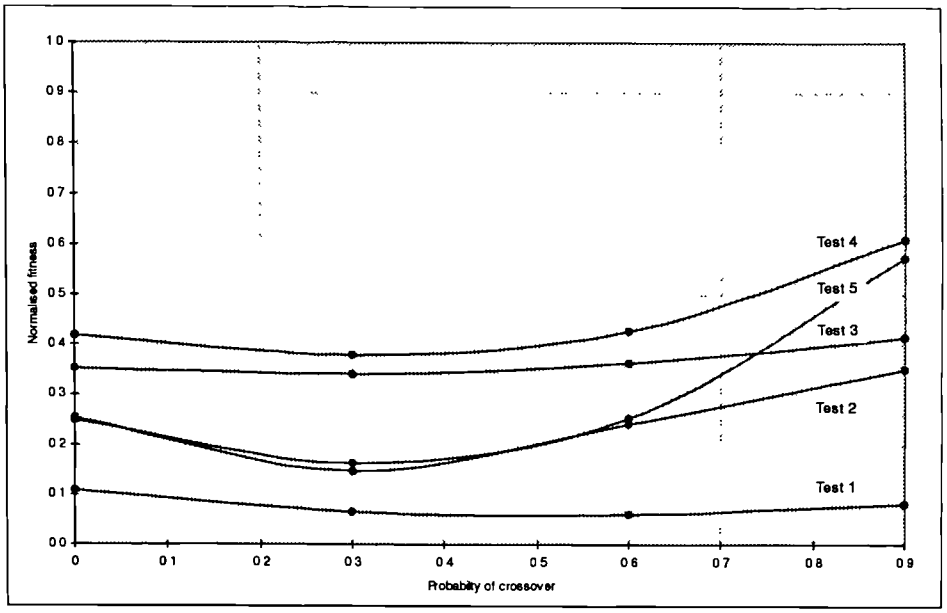


Figure 4.7: Variation of normalised fitness with probability of crossover

Probability of mutation (P_m); An observation of Figure 4.8 seems to indicate that, with the exception of Test 1, the choice of value for P_m has a significant impact upon solution quality with a clear indication that the most effective choice for this parameter centres on a value equal to 0.01

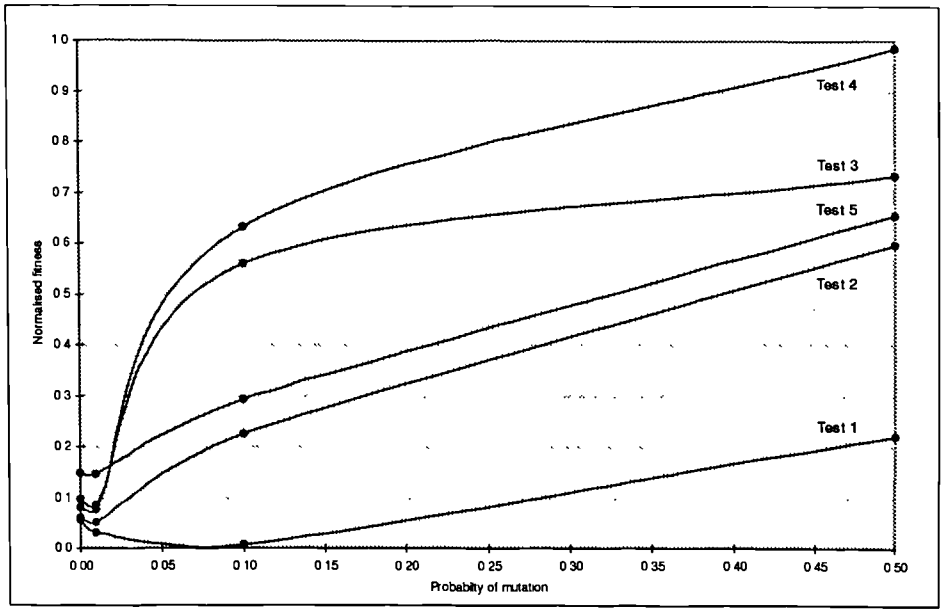


Figure 4.8: Variation of normalised fitness with probability of mutation

Whilst the plots for each test case are based upon mean results averaged over three population sizes, it has been observed for each test case, that the most effective parameter settings for generation gap, probability of crossover, and probability of mutation are consistent across all three population sizes tested, and therefore do not vary with population size.

4.6.4 The Impact of the GA-Local Search Procedure on Population Size and Number of Generations

The following tables detail, for each test case, the highest quality solutions of each of the five test cases derived using three search strategies. Table 4.5a details the results derived from the sole application of the *Optimal Sequencer Genetic Algorithm*, Table 4.5b details the results derived from the sole application of the *Heuristic Local Search*, whilst Table 4.5c details the results derived using the combined *GA-Local Search Procedure.*

Test Case	Input Fitness	Output Fitness	Improvement on input	Population Size	Number of Generations	Number of Evaluations
1	182,753	39,953	78%	30	100	3,000
2	1,809,280	372,874	79%	200	150	30,000
3	14,692,300	3,474,653	76%	300	300	90,000
4	14,715,000	3,534,330	76%	500	550	275,000
5	23,510,500	2,963,620	87%	600	700	420,000

Table 4.5a: Sole application of the Optimal Sequencer Genetic Algorithm

Test Case	Input Fitness	Output Fitness	Improvement on input	Population Size	Number of Generations	Number of Evaluations
1	182,753	39,953	78%	-	-	-
2	1,809,280	430,271	76%	-	-	-
3	14,692,300	3,600,590	75%	-	-	-
4	14,715,000	4,505,710	69%	-	-	-
5	23,510,500	2,899,390	88%	-	-	-

Table 4.5b: Sole application of the Heuristic Local Search

Test Case	Input Fitness	Output Fitness	Improvement on input	Population Size	Number of Generations	Number of Evaluations
1	182,753	39,953	78%	25	25	625
2	1,809,280	372,611	79%	50	38	1,900
3	14,692,300	3,434,500	77%	70	75	5,250
4	14,715,000	3,483,780	76%	120	138	16,560
5	23,510,500	2,757,320	88%	150	175	26,250

Table 4.5c: The combined GA-Local Search Procedure

On average, the sole application of the *Optimal Sequencer Genetic Algorithm* yields an improvement equal to 79%. In comparison, this represents a slightly more effective strategy than that associated with the sole application of the *Heuristic Local Search*, which yields a

smaller improvement equal to 77%. However, the most effective strategy is the *GA-Local Search Procedure*, which on average yields an improvement equal to 80%. The 2% difference in average solution quality between the sole application of the *Optimal Sequencer Genetic Algorithm* and the sole application of the *Heuristic Local Search* is at the significant cost of computational processing time and, as a result, it could be argued that the 2% benefit is not worth the cost.

However, as observed previously, the *GA-Local Search Procedure* can have a significant impact upon reducing population size and number of generations, which are the two parameters that drive the number of evaluations, and hence computational effort. The 3% difference in average solution quality between the *GA-Local Search Procedure* and the sole application of the *Heuristic Local Search* is at a small cost in computational effort. This is based on the relatively small number of evaluations that are necessary (*See* Table 4.5c). In comparison with the sole application of the *Optimal Sequencer Genetic Algorithm*, on average the *GA-Local Search Procedure* reduces the number of evaluations by 91%, and hence reduces computational effort by the same margin.

Both on the issue of ultimate solution quality and computational effort, and hence procedural efficiency, the combined *GA-Local Search Procedure* is superior to the sole application of either of the two individual search procedures. Furthermore, it is theorised that as problem size increases, the quality of solutions derived from the sole application of the *Heuristic Local Search* would deteriorate.

Adopting the *GA-Local Search Procedure*, based upon the values detailed in Table 4.5c, Figure 4.7 illustrates how the most effective setting for population size varies with problem size, whilst Figure 4.8 illustrates how the most effective setting for number of generations varies, again, with problem size.

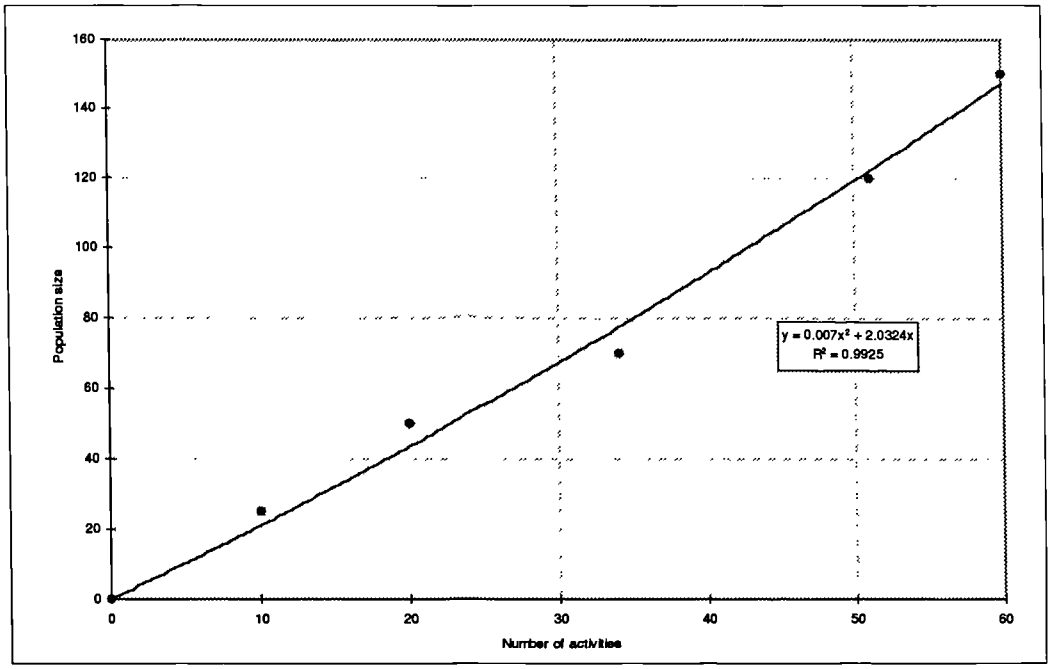


Figure 4.7: Variation of population size with problem size

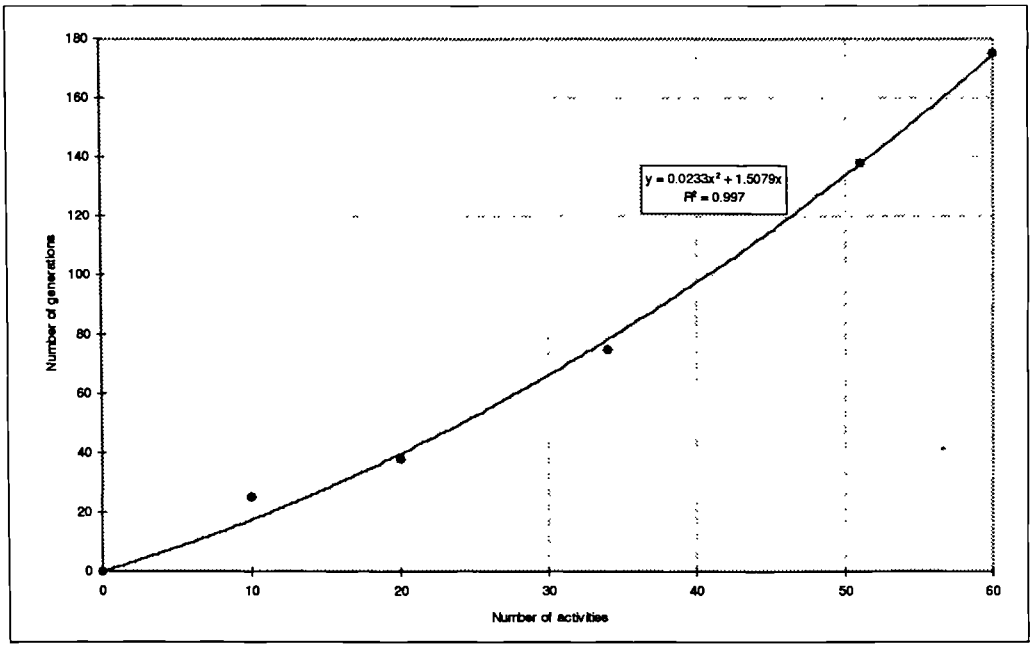


Figure 4.8: Variation of number of generations with problem size

4.6.5 Applying the GA-Local Search Procedure

Using the parameter settings derived from the experimentation, and summarised below as Step 1, each of the five test cases have been re-run for a total of ten trials each. Table 4.6 summarises the results. Such an analysis can help to derive conclusions relating to the number of trials that are necessary in order to ensure the derivation of the highest quality solution.

Test Case	% of times the highest quality solution was derived
1	80%
2	80%
3	60%
4	40%
5	40%

Table 4.6: Probability of highest quality result being returned

In summary, the *GA-Local Search Procedure* can be described as follows;

Step 1; Using the input sequence, apply the *Optimal Sequencer Genetic Algorithm*, using the following parameter settings;

- Population size = $0.007(\text{No. of activities})^2 + 2.0324(\text{No. of activities})$
- Number of generations = $0.0233(\text{No. of activities})^2 + 1.5079(\text{No. of activities})$
- Generation gap = 0.80
- Probability of crossover = 0.30
- Probability of mutation = 0.01

Step 2; Repeat Step 1 for Number of trials = 10. (Based on the results in Table 4.6, this helps to ensure that the highest quality result will be returned).

Step 3; Apply the *Heuristic Local Search* to each of the derived sequences

4.7 Chapter Summary

Chapter 4 focuses on the third function of the proposed modelling strategy and, in particular, describes the principal mechanism for solving the *ActRes Problem* of "deriving a near optimal sequence of activities" based on pre-defined objectives and proposes how it should be applied. The mechanism, which has been named the *GA-Local Search Procedure,* is based on the sequential application of two separate search procedures. The first, the *Optimal Sequencer Genetic Algorithm,* prunes the large, noisy and discontinuous search space to return an interim solution set which is then improved upon by the second procedure, the *Heuristic Local Search.*

New sequences are derived by resequencing activities based on a number of objectives including the *minimisation of iteration,* and the *maximisation of concurrency.* Having

described the *GA-Local Search Procedure*, these two objectives are examined in detail in Chapters 5 and 6.

Finally, the current research documented in Chapter 4 contributes to the subject in a number of areas. The *Optimal Sequencer Genetic Algorithm* component of the *GA-Local Search Procedure* is a modified version of the well-documented standard genetic algorithm. Whilst most of the modifications are based on published methods, the modified mutation operator and the incorporation of sequence constraints, detailed in Sub-Sections 4.4.3 and 4.4.6 respectively, have been newly developed for solving the *ActRes Problem.*

Perhaps the largest contribution of the research documented in Chapter 4 relates to the new development of the *Heuristic Local Search* procedure which, as the second component of the *GA-Local Search Procedure*, helps to ensure that solutions to the ActRes Problem are derived in the most efficient manner.

5. Deriving a Near Optimal Sequence of Activities Based on the Objective of Minimising Iteration

5.1 Introduction

Chapter 4 introduced the *GA-Local Search Procedure* which, as the principal mechanism of the third function of the proposed modelling strategy, can be used to derive a near optimal sequence of activities based on a mathematical expression of a given objective. Chapter 5 now addresses the third strategic function, that is, the *ActRes Problem* of deriving a near optimal sequence of activities based on the specific objective of *minimising iteration*

The notion of iteration, first introduced in Chapter 3, is a common thread that runs through most models of the design-development phase of a product. At the highest level of description, the term "iteration" is commonly used to describe a process that, via feedback loops, involves the repetition of a sequence of computations until an error term or uncertainty has been eliminated or, at least, minimised.

However, beyond this introductory description, many distinctive varieties of iteration exist. As a result, efforts to improve the efficiency of a product's design-development phase through the sequencing of activities in order to minimise iteration should perhaps be preceded by an understanding of the underlying role that iteration plays. In this context, the role that iteration plays as part of the design-development phase of a product is explained in Section 5.2. With a clearer understanding of the role iteration plays, Section 5.3 then reports on prior research which, based on the objective of minimising iteration, uses the *design structure matrix (DSM)* representation in conjunction with various search techniques to solve the ActRes Problem.

Section 5.4 elaborates on two of the most effective published approaches to deriving a near optimal sequence of activities and introduces a new *objective function* that measures the iteration implied by a given sequence of design-development activities. This new objective function, used in conjunction with the previously described *GA-Local Search Procedure*, represents a new approach to the derivation of a near optimal sequence of activities based on the minimisation of iteration. In order to demonstrate this new approach, Section 5.5 compares the solutions derived from it's application against previously published solutions for five

specific tests cases. Finally, Section 5.6 summarises the chapter and highlights how the research specifically documented in Chapter 5 represents a contribution.

5.2 The Role of Iteration in Design-Development

Before considering initiatives for improving the efficiency of a product's design-development phase based on the minimisation of iteration, it may be necessary to recognise the varied types of iteration, and fully understand the roles each type plays. Without such an understanding, the *effects caused* by iteration are likely to be misinterpreted.

It is difficult to describe iteration since the notion is rather subjective. Eisenhardt and Tabrizi [Eisenhardt 95] point out that iteration is conflictingly portrayed as either; (*i*) a costly problem that should be avoided; (*ii*) a useful means of improving a design-definition; or, (*iii*) a catalyst for innovation. These conflicting portrayals of iteration can make it difficult to interpret whether or not, in practice, any one observed instance of iteration exerts a net positive or negative effect.

That said, a product development organisation (PDO) can be aided in it's interpretation and understanding of iteration through the creation of a classification scheme. Such a scheme, using a hierarchy-based description, and classed according to differences in the *cause* of iteration, is outlined in the two sub-sections that follow.

5.2.1 A Hierarchy of Iteration

Three levels of iteration are proposed: (*i*) intra-activity; (*ii*) inter-activity; and, (*iii*) inter-product. The first level, namely, *intra-activity iteration* is the iteration that is at the heart of the various analytical procedures and low-level design problems that make up and are embodied *within* a design-development activity. By way of an example, consider the activity defined as "Create and analyse a finite element model of Component A". During the processing of this activity, several iterations may be necessary. If, for example, the first model is not defined to a sufficient level of completeness to obtain reasonable and believable results, then it is probable that a second, more accurate model will need to be created.

The second level of iteration, namely *inter-activity iteration* is an extension of the first and describes the iteration *between* one or more design-development activities. This type of

iteration is more meaningfully characterised by its cause, and is therefore discussed, in greater detail, later in Sub-Section 5.2.2.

The third level of iteration, namely *inter-product iteration* describes the iteration between different released versions of the same product and the activities associated with their design-development. For example, a piece of commercial software goes through several inter-product iterations as new versions of the code are developed and re-released onto the market.

5.2.2 The Cause of Iteration

It is proposed that there are three causes of iteration; (*i*) coupling; (*ii*) proposal-testing-modification (PTM) cycles; and, (*iii*) propagation of change. The following paragraphs address the three causes of iteration in turn.

Coupling

Coupling links design-development activities and their associated sub-problems which, because they all rely on one another for input data, are inter-related and therefore require to be processed and solved simultaneously. An example of *coupling* is illustrated in Figure II.3 of Appendix II, where Activities G, H, & I are linked by inter-activity iteration caused by *coupling*.

Proposal-testing-modification (PTM) cycles

Proposal-testing-modification (PTM) cycles, often less formally described as *'trial and error (elimination)*, cause iteration at all levels. Because a *PTM cycle* causes *intra-activity*, *inter-activity*, and *inter-product iteration*, and because it occurs continually throughout the design-development phase of a product, it is classically referred to as the *Anatomy of Design* [Rosenstein 67].

Figure 5.1 illustrates an interpretation, distilled from the observations of a number of researchers and practitioners, including [Asimow 62, Simon 75, Hubka 82] who view the Anatomy of Design, as embodied by the PTM cycle, as an iterative decision-making cycle. This cycle is applied continually, either formally or informally, throughout the design-development phase in order to solve each of the myriad of design problems which occur and need to be resolved.

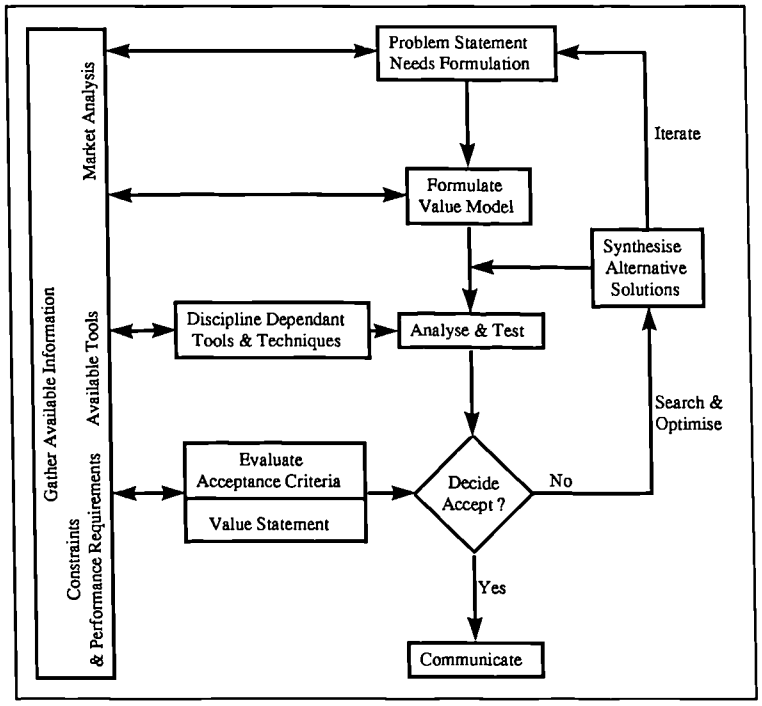


Figure 5.1: The Anatomy of Design, as embodied by the PTM cycle

Propagation of change

Finally, the third cause of iteration relates to changes that are propagated through the design-development of a new product. According to Steward [Steward 81a], such changes can arise from a number of sources;

- Perhaps a mistake has been identified. A major potential source of mistakes is the lack of integration and communication of shareable knowledge. Methods for avoiding this dysfunction have been described previously in Section 2.3.

- An opportunity to improve the product design may be perceived.

- The customer or the marketing department may propose changes in the specifications.

5.2.3 Design Reviews, Margins and Guesstimate Ranges

At the end of an iterative block of coupled activities or at the end of a PTM cycle, a *design review* is undertaken to determine, among other things, where any guesstimates have to be revised. If revision is required, then iteration will usually be necessary and those activities encapsulated within an iterative block or PTM cycle will have to be re-addressed and in some instances re-processed.

In addition to optimal sequencing, there are two other ways of reducing inter-activity iteration. The first is to assign a *margin (of acceptable error)* to data-output from activities that have to rely on guesstimates as data-input. The second relates to the use of a spread, or a *range of guesstimates*, rather than the use of a single point-value guesstimate.

Using margins, if data is found to be inaccurate, as long as it falls within the range of acceptable error, then iteration will not be necessary. Using point values, there is a high probability that guesstimates, when compared against subsequently derived data, will not be close enough to ensure that iteration is avoided. Therefore, the effective use of guesstimate ranges can help to ensure that there is a higher probability that any errors and omissions will not result in the need to revise calculations. In this way, the time delays caused by the need to repeat and iterate can be kept to a minimum.

In effect, the use of margins and guesstimate ranges acts to reduce the need to re-process activities and hence acts to weaken certain data-dependencies.

5.2.4 Modelling Iteration Using the Design Structure Matrix (DSM) System

Before reading this sub-section, the reader is directed towards Appendix II where the DSM system is explained. The design structure matrix (DSM) models the activities of the design-development phase of a product and, as such, is also capable of modelling *inter-activity iteration.* Using the DSM, any implied iteration between activities can be represented and manipulated in a manner that allows the effects of inter-activity iteration to be interpreted with maximum clarity. At the same time, the DSM also models iteration caused by *coupling* and *proposal-testing-modification cycles.*

Whilst unplanned changes, by their very nature, cannot be modelled in advance, the DSM can be used to show the effect of iteration caused when *a change is propagated.* A change, incorporated into a previously completed activity, does not simply change the output of the activity in question, it can affect all activities which rely on data from the changed activity. When the output of an activity changes, reading down the column of the affected activity in the DSM shows which activities, by virtue of their dependence on the affected activity, are also likely to be affected, and hence subject to change. Then, reading down the columns of these affected activities indicates what activities they affect in turn, and so on. This process

continues until no new activities with successors can be found, or until the effects on the successors that remain are considered to be of no consequence.

The maintenance of change in this manner is often referred to as *design change control* and whilst the DSM does not specifically model the effects of change, it shows how a specific change is propagated. Therefore, the DSM acts as a *decision-support system* in that it can be used to help identify those activities affected by a proposed change. In this manner, the full impact of a proposed change can be quantified prior to the *cost-benefit analysis* that is used subsequently to help determine whether or not to incorporate the proposed change.

5.3 Prior Research into the Minimisation of Iteration Based on the Design Structure Matrix (DSM) System

Previously published research on the design structure matrix (DSM) system reports on a range of varied solution approaches to the *ActRes Problem*, and generally is based on the objective of minimising iteration. This version of the *ActRes Problem*, where activities are sequenced in order to manipulate a matrix of data-dependencies, with the objective of finding the sequence of activities which incorporates minimum iteration, is commonly referred to as *partitioning*.

Using a precedence matrix as a starting point, *partitioning procedures* can be used to investigate new sequences. Each new sequence results in a manipulation of the precedence matrix as each activity's row and corresponding column of data-dependencies is shifted to the relevant new position in the newly derived sequence. Using a search strategy, such as the previously described GA-Local Search procedure, the aim then is to find a sequence of activities and the resulting DSM which, when evaluated using a suitable mathematical-based objective function, represents the sequence which implies minimum iteration.

The following sub-section demonstrates the concept of partitioning in more detail and summarises a range of published partitioning procedures that act as the basis for solution approaches to the minimum iteration *ActRes Problem*.

5.3.1 Partitioning the Design Structure Matrix (DSM)

Partitioning is the sequencing of activities in order to maximise the availability of input data required by each design-development activity. In doing so, the need for guesstimates is

minimised and hence iteration, which is necessary when guesstimates are subsequently found to be inaccurate, is also minimised.

Ideally, it would be possible to sequence activities by interchanging rows and swapping the corresponding columns of a matrix such that all input data required by each activity is made available when it is required. In such cases, there would be no need for guesstimates and the probability of iteration would be almost zero. If such a sequence existed, then the resulting DSM would be purely lower triangular, that is, all data-dependencies would be located below the leading-diagonal.

In reality, it is unlikely that partitioning will derive a sequence of activities that is completely free of iteration. However, by deriving a sequence of activities that is optimised based on the objective of minimising iteration, unnecessary iteration implied by poor sequencing will be avoided

The precedence matrix and the partitioned DSM, in Figures 5.2 and 5.3, help to demonstrate the previously described notions associated with inter-activity iteration. The matrices are based on an example taken from the literature [Steward 81a] and they detail a number of activities that make up the design-development of an electric car.

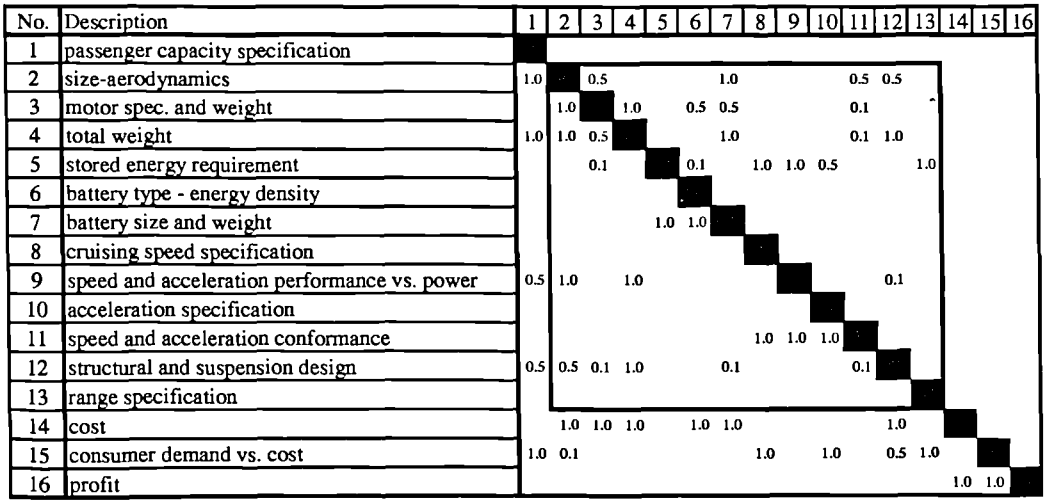


Figure 5.2: Precedence matrix for the design-development of an electric car

The precedence matrix in Figure 5.2 incorporates an iterative block that encapsulates activity numbers 2-13. PDO members, processing activities in the sequence in which they occur in Figure 5.2, have to use guesstimates for the input data represented by each data-dependency

found above the leading-diagonal. At the end of the iterative block, the PDO will have determined the data-outputs of each activity based on their guesstimates. A design review can then be used to compare the *values actually derived* with the *initial guesstimates*. If they are sufficiently close, then no iteration is necessary. If not, the values actually derived are used to make better guesstimates and the block, incorporating Activities 2-13 is repeated.

In Figure 5.3, the activities have been re-sequenced and their data-dependencies re-positioned accordingly such that the previously illustrated precedence matrix has been partitioned to create a design structure matrix (DSM). In this case, a smaller sub-set of data-dependencies has been re-positioned above the leading-diagonal such that different guesstimates need to be made.

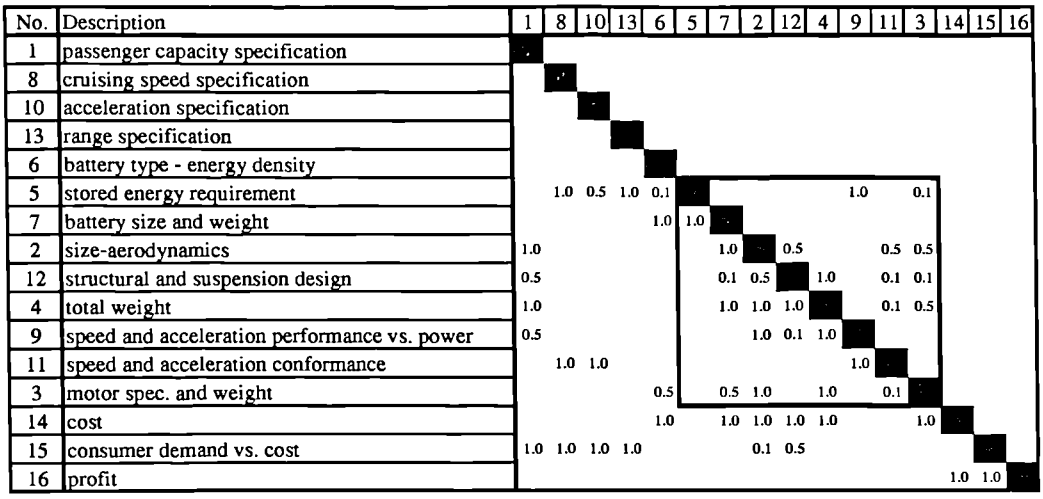


Figure 5.3: The precedence matrix, after partitioning, becomes the design structure matrix

In this case, fewer activities are involved in the resulting iterative block, and hence fewer guesstimates have to made. Furthermore, the lower values associated with those data-dependencies positioned above the leading-diagonal implies that the *risk* associated with the newly required guesstimates is lower. Such effects help to ensure that the iterative block in the DSM in Figure 5.3 will converge more rapidly than the iterative block in the precedence matrix in Figure 5.2.

In Figure 5.3, iteration, implied by the need for guesstimates, has not been eliminated completely. However, the chances are that the need for iteration has been reduced. This is primarily due to the fact that partitioning has removed a large number of the guesstimates that

had to be made in the poorly sequenced precedence matrix of Figure 5.2. (10 guesstimates in the re-sequenced matrix as opposed to 17 in the original).

Numerous partitioning procedures have been developed, and whilst these procedures vary in their approach they all share the same goal, that of sequencing activities in order to minimise iteration.

Sargent and Westerberg [Sargent 64] describe their partitioning procedure as part of a general-purpose computer program (SPEED-UP) for the analysis of complex chemical engineering systems. The procedure, referred to as *path searching*, operates by tracing data-dependencies between activities backward through the matrix until an activity is encountered twice. All activities between the two occurrences of the activity are then classed within an iterative loop. This operation is carried out until all activities have been considered.

Christensen and Rudd [Christensen 69] simplify the Sargent and Westerberg partitioning procedure by adding the ability of the search mechanism to trace forwards as well as backwards through the activity sequence in an attempt to find an optimal solution. The *Adjacency Matrix Technique*, first developed by Ledet & Himmelblau [Ledet 70], uses a revised representation of the precedence matrix known as the adjacency matrix. Kehat and Shacham [Kehat 73] describe a procedure that reduces the storage and computational requirements of Ledet and Himmelblau's procedure.

Steward's partitioning procedure [Steward 81a, Steward 81b] combines the principles of path searching and the adjacency matrix technique. Finally, Austin *et al* [Austin 97] undertaking research focussed on modelling and planning building design-work have developed a computer-based program that utilises Steward's partitioning procedure [Newton 95].

The partitioning procedures described so far represent pioneering work. However the solutions that result from such procedures are usually far from optimum, and are largely dependent on the input sequence of activities. Using heuristic rules, the procedures rely on a local manipulation of an input sequence to converge rapidly towards a solution. However the resulting re-sequence is likely to be a local optimum and, as a result, it is considered that such procedures are non-robust and only have a very small probability of finding near optimum solutions since different input sequences yield different solutions.

Computer technology has improved dramatically over recent years. This, combined with the development of efficient and effective optimisation search techniques, has resulted in changes to the basic workings of more recent partitioning procedures. This second generation of partitioning procedures uses the structure of the matrix to evaluate a mathematical expression of the objective (objective function). The resulting value represents a measure of the iteration implied by a given sequence of activities.

Using newly developed search strategies, these new partitioning procedures continually develop new sequences and resulting DSMs by interchanging matrix rows and the corresponding columns, each time re-evaluating the objective function for the resulting matrix. Such procedures continue to develop new sequences until the sequence incorporating minimum iteration is found. In this respect, these procedures vary in (*i*) their definition of the objective function used to measure iteration; and, (*ii*) the search technique.

Kusiak's *Triangularisation Partitioning Procedure* [Kusiak 90b, Kusiak 91], using a binary precedence matrix representation, develops new sequences and associated DSMs with the aim of finding a sequence that has the minimum number of non-zero data-dependencies above the leading-diagonal. The procedure searches for a sequence using a search strategy based on a branch and bound search technique [Kusiak 90a].

Using a numerical precedence matrix, the *Gebala Partitioning Procedure* [Gebala 91] rearranges activities in order to minimise any iterative backtracking required within iterative blocks by arranging the more important data-dependencies closer to the leading-diagonal. The objective of the Gebala Partitioning Procedure is the minimisation of the sum of the data-dependencies above the leading-diagonal, weighted by their distance from the leading-diagonal.

This partitioning procedure uses a heuristic-based search [Smith 94] that develops a random number of sequences that are evaluated such that the sequence that most closely matches the objective is stored as the best so far. The procedure then searches for further improvement by preserving the majority of the best-so-far sequence whilst moving one or more activities, in turn, relative to the others.

Rogers' *DeMAID (design manager's aid to intelligent decomposition) system* [Rogers 92] rearranges activities in order to minimise feedback loops and the total accumulated time and cost required to complete a given sequence of activities. Early versions used an expert system to search for new sequences, however more recently, a genetic algorithm has been incorporated [Rogers 96a, Rogers 96b].

In the past, the iterative blocks of a partitioned design structure matrix have been subjected to a process referred to as *tearing* [Kron 63]. The goal of tearing is to sequence the activities within each iterative block in order to find the best sequence of activities and thus a place to start the iteration. A *tear* implies the removal of a data-dependency between two activities such that the first activity begins with a guesstimate. Here the aim is to choose tears that represent a break of dependence that has minimum impact on the entire process. As in partitioning, various procedures have been developed to achieve tearing.

It is possible to sequence the activities within an iterative block using simple heuristics. Early developments of Rogers' *DeMAID system* used an expert system to make decisions concerning the optimal sequencing of coupled activities within iterative blocks. Kehat and Shacham [Kehat 73] also propose a procedure where the objective is to identify, and sequence earlier, those activities that rely least on the other activities in each iterative block. When more than one activity requires the same amount of input data, the one that supplies the most data to subsequent activities is sequenced first.

Steward uses *shunt diagrams* [Steward 81a] to rank the data-dependencies within an iterative block. With this ranking completed, the tearing procedure partitions each of the blocks in turn such that the activities within each block are sequenced with the dependencies having greatest impact repositioned below or as close to the leading-diagonal as possible. Based on Steward's tearing procedures, Austin *et al* have developed a tearing computer-program [Newton 95].

Smith *et al* have developed two methods for tearing. The first is the *sequential iterative model* [Smith 94] whilst the second is the *work transform model* [Smith 97]. Both models, which are numerical developments of Steward's tearing methodology, explicitly model iterative blocks of activities in order to find the sequence within each block that results in minimum block lead-time. The analysis uses Markov chains and matrix-eigenstructure analysis applied to the activity durations as well as the data inter-dependencies between activities.

By using activity durations, an additional dimension is added to the analysis. However, according to Smith and Eppinger themselves [Smith 94], whilst activity durations can have an effect on the optimal sequencing, the effect is minimal such that it is the positioning of data-dependencies which has the greatest impact.

As indicated previously in Section 3.7, early research into the use of the design structure matrix system incorporated the use of a binary precedence matrix as input to a partitioning procedure. Only after the precedence matrix has been partitioned to create the design structure matrix, would the dependencies be refined and ranked into scaled, numeric values. Using the scaled values, tearing procedures would then be applied in order to partition the iterative blocks that had been previously identified.

However, more recent research that uses a numerical precedence matrix as input to a partitioning procedure tends to obviate the need for the secondary tearing analysis [Smith 94, Smith 97]. The implication of this is that, an effective partitioning procedure, applied to a numerical precedence matrix, inherently performs tearing.

5.4 A New Objective Function for Measuring Inter-Activity Iteration

As previously stated, the goal of partitioning is the sequencing of activities in order to maximise the availability of input data required by each design-development activity, therefore minimising the probability of iteration. In order to apply the *GA-Local Search Procedure* previously described in Chapter 4, a mathematical expression that measures the scale of inter-activity iteration is required. With an objective function defined, the search procedure can then start searching for the sequence and associated design structure matrix that implies minimum iteration.

Of the partitioning procedures summarised previously, the objective functions associated with the *Kusiak Triangularisation Procedure* and the *Gebala Partitioning Procedure* are reviewed further prior to the definition of a new objective function for measuring inter-activity iteration.

The Kusiak Triangularisation Procedure

The Kusiak Triangularisation Procedure minimises an objective function that evaluates the number of non-empty matrix positions above the leading-diagonal. This is a very simple objective function, but has two drawbacks.

Firstly, the objective function incorporated within the Kusiak Triangularisation Procedure is based on the evaluation of a binary matrix representation. As previously stated, one shortcoming of the binary matrix representation is the assumption that all data-dependencies are of equal importance. No attempt is made to differentiate between the amount and importance of data transfer within the matrix. The Triangularisation Objective Function, therefore, works in such a way that it cannot take advantage of the additional analytical potential of a numeric precedence matrix.

Secondly, the Triangularisation Objective Function cannot differentiate between the positioning of data-dependencies above the leading-diagonal. Consider two different matrices, each with only one data-dependency above the leading-diagonal. The first matrix, as illustrated in Figure 5.4a, includes a data-dependency above the leading-diagonal in a position directly adjacent to the leading-diagonal. The second matrix, as illustrated in Figure 5.4b, contains it's single dependency above the leading-diagonal towards the top right-hand corner of the matrix. Using the Triangularisation Objective Function, both matrices are scored equally. However, the first matrix, in Figure 5.4a, is far superior since the iterative block is much smaller in that fewer activities are involved.

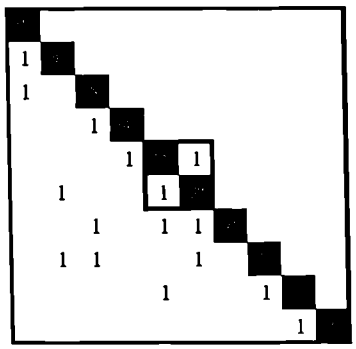


Figure 5.4a: A small iterative block

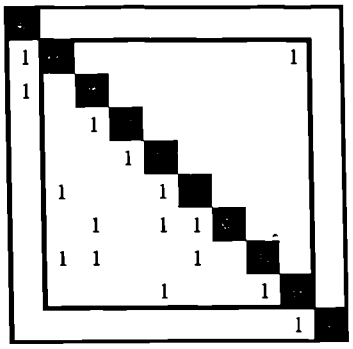


Figure 5.4b: A large iterative block

The Gebala Partitioning Procedure

The Gebala Partitioning Procedure minimises an objective function that evaluates the sum of the non-zero dependencies found above the leading-diagonal, each weighted by their distance from the leading-diagonal. The objective function incorporated within the Gebala Partitioning Procedure is based on an analysis of a numerical matrix representation.

The rules of partitioning can now consider rearranging activities in order to minimise any iteration by arranging the more important dependencies below the leading-diagonal as a first

priority. If this is not possible, the more important dependencies can be arranged as close to the leading-diagonal as possible. In this way, the Gebala Partitioning Procedure, with the combined utilisation of the numeric matrix representation and positional weightings, overcomes both of the previously highlighted drawbacks associated with the Kusiak Triangularisation Procedure.

However, whilst the Gebala Partitioning Procedure is superior to the Kusiak Triangularisation Procedure, both share a common failing, in that they both only consider those matrix positions above the leading-diagonal with a view to minimising iteration. In this way, these procedures do not consider the optimisation of the feedforward of data, as represented by the dependencies positioned below the leading-diagonal. In this respect, both procedures only consider half of the problem. This leads to a negative consequence in that a lack of consideration is given to the minimisation of *unplanned iteration* that may result from the *propagation of a change*. By considering the dependencies above the leading-diagonal, a basic partitioning procedure acts to minimise *planned iteration* caused by *coupling* and *PTM cycles*. The basic partitioning procedure aims to achieve this by searching for a sequence of activities whose associated DSM encapsulates the maximum number of data-dependencies below the leading-diagonal.

However, by considering the dependencies below the leading diagonal as well as those above, the new partitioning procedure acts to minimise *unplanned iteration*. The new partitioning procedure aims to achieve this by searching for a sequence of activities whose associated DSM encapsulates the maximum number of data-dependencies below the leading-diagonal, and also positions the strongest dependencies as far downstream as possible.

Consider a situation where a change is required in a previously completed activity. If a number of downstream activities have since been completed, then, if the activity to be changed has already passed data to these downstream activities, it may be necessary to revisit all such affected activities. Such a scenario represents *unplanned iteration* caused by *the propagation of a change*. By positioning the dependencies below the leading-diagonal, and as far downstream as possible, (i.e. as close to the bottom left-hand corner of the matrix) the number of activities caught in such an *unplanned iterative block* can be minimised.

A new search objective function for measuring inter-activity iteration, to be minimised using Chapter 4's *GA-Local Search Procedure* is summarised in words as "the sum of all of the data-dependencies of the matrix, each multiplied by a value which represents, for each matrix position, the scaled distance of the data-dependency from the bottom-left-hand corner of the matrix".

When this new function was first conceived, it was considered that the *distance values*, that the data-dependencies were to be multiplied by, could be formulated in a combination of different ways. Firstly, the vertical and horizontal components could each be multiplied by a separate or by the same fixed value. Secondly, the combined vertical and horizontal components could be raised to a power. Furthermore, it may have been necessary to apply added weight penalties to dependencies above the leading diagonal. Ultimately, the best formulation would be the one which ensured that as many data-dependencies as possible were re-positioned in those matrix positions close to the bottom-left-hand corner of the matrix. Based on these considerations, the generic function in Equation 5.1 was derived.

$$\text{Iteration-Based Objective Function} = \sum_{i=0}^{i=n-1} \sum_{j=0}^{j=n-1} \left((a_{ij}) \times (A \times [Bj + C(n-i)]^{D} \right)$$

[Equation 5.1]

a_{ij}	Data-Dependency in row i, column j. (Note: a_{00} represents the top left-hand corner of the matrix)
A	Above diagonal weighting, applied only to positions above the leading-diagonal ($j > i$)
j	Horizontal distance of a_{ij} from the left-hand side of the matrix
B	Weighting applied to distance j
n	Total number of activities in the sequence
$(n-i)$	Vertical distance of a_{ij} from the bottom of the matrix
C	Weighting applied to distance $(n-i)$
D	Combined horizontal & vertical distance power weighting

Some very basic sensitivity analysis has been carried out to investigate the most appropriate values for the variables embedded in Equation 5.1. The results of this analysis result in the simplified version of the objective function detailed in Equation 5.2.

$$\text{Iteration-Based Objective Function} = \sum_{i=0}^{i=n-1} \sum_{j=0}^{j=n-1} \left(a_{ij} \times \Omega_{ij} \right)$$

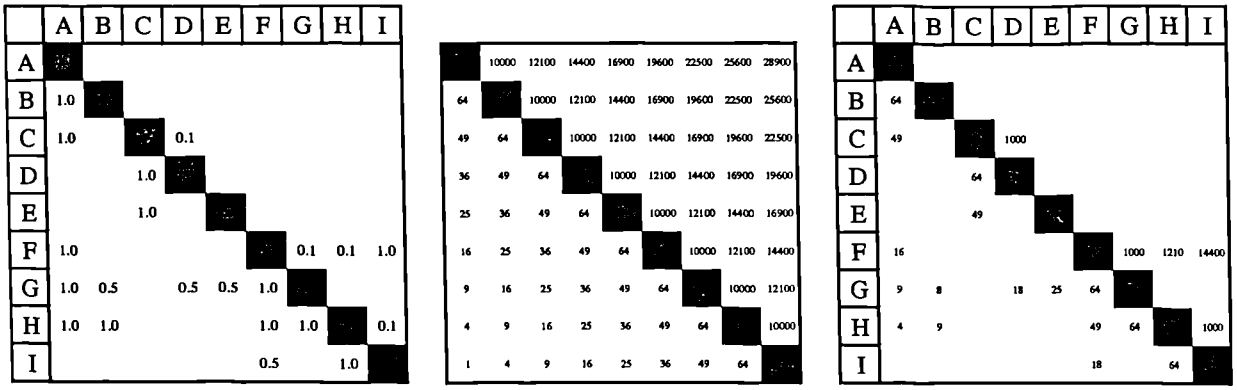
[Equation 5.2]

Weighted distance (Ω_{ij}) = ($1*[j + (n-i)]^2$)	for ($j < i$) matrix positions below the leading-diagonal
Weighted distance (Ω_{ij}) = ($100*[j + (n-i)]^2$)	for ($j > i$) matrix positions above the leading-diagonal
B = C = 1	

For the purposes of clarification, consider the example in Figure 5.5. The DSM in Figure 5.5a details 9 activities and their data-dependencies. The matrix in Figure 5.5b details the scaled distance of each matrix position from the bottom-left-hand corner of the matrix (Ω_{ij} in Equation 5.2).

By multiplying each data-dependency in Figure 5.5a by the value of Ω_{ij} in the corresponding matrix position in Figure 5.5b, the matrix of products in Figure 5.5c results. Summing these products yields a measure of iteration equal to 19,184 for the sequence of activities and the corresponding DSM in Figure 5.5a.

This new objective function, minimised using the previously described *GA-Local Search Procedure*, constitutes a new solution-approach to the *ActRes Problem*, and, as such, is referred to in the text, which follows as the *Scott Partitioning Procedure*.



(a) matrix of data-dependencies (a_{ij}) (b) matrix of scaled distances (Ω_{ij}) (c) Resulting sum of products equals 19,184

Figure 5.5: An illustration of how Equation 5.2 can be used to measure inter-activity iteration

5.5 A Comparison of Results Against Published Data

The five test cases detailed below are those used previously in Section 4.6 to develop parameter settings for the *GA-Local Search Procedure*. In the following sub-sections, the five DSMs that result from the application of the *Scott Partitioning Procedure* are compared against the DSMs which result from the application of previously published partitioning procedures. In all comparisons it is assumed that no sequence constraints exist and that the resulting DSMs are based purely on a search for the sequence of activities which implies minimum iteration.

The precedence matrices that act as input to the five test cases have been taken from the literature and use a number of varied schemes for the representation of data-dependencies. The input precedence matrix associated with Test Case 1 is based on a numeric dependency scheme derived using Krishnan's *importance ratio* [Krishnan 90]. Those input precedence matrices associated with Test Cases 2, 3 and 5 are based on a binary dependency scheme whilst the precedence matrix associated with Test Case 4 is based on the four-level dependency scheme proposed by Austin *et al* [Austin 96] (*See* Appendix II).

In this respect, none of the five test cases use the newly derived four-level dependency scheme proposed in Appendix II. However, since the aim here is simply to compare procedural effectiveness, it is considered acceptable to focus merely on a comparison of the DSMs that are derived using the new *Scott Partitioning Procedure*, with those that are published in the literature.

5.5.1 Test Case One

Test Case One is taken from Reference [Gebala 91]. The precedence matrix illustrated in Figure 5.6a, when evaluated using the new objective function yields a measure of iteration equal to 182,753.

	1	2	3	4	5	6	7	8	9	10	11	12
1									0.3			
2							0.9		0.2	0.7		0.5
3						0.1			0.9			
4	0.7				0.4							
5			0.6					0.5				
6			0.5									
7		0.9			0.2							0.4
8				0.7								
9										0.2		
10	0.2											
11						0.6						
12	0.5						0.2				0.4	

Figure 5.6a: Test Case 1: Precedence Matrix [Gebala 91]

Figure 5.6b illustrates the design structure matrix (DSM) which results from the application of the *Gebala Partitioning Procedure*. This DSM, when evaluated, using the new objective function, yields a measure of iteration equal to 50,004. In comparison with the precedence matrix, this represents a reduction in iteration equal to 73%.

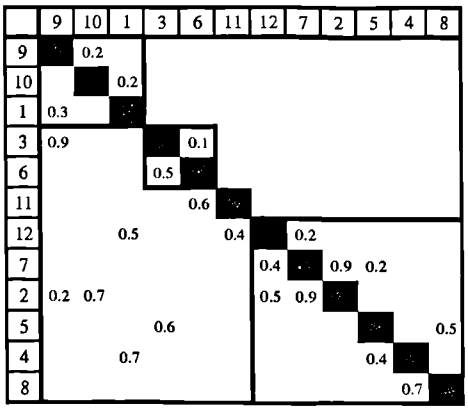


Figure 5.6b: Test Case 1: DSM derived using the Gebala Partitioning Procedure

Figure 5.6c illustrates the DSM that results from the application of the new *Scott Partitioning Procedure*. This DSM, when evaluated using the new objective function, yields a measure of iteration equal to 39,953. In comparison with the precedence matrix, this represents a reduction in iteration equal to 78% which, in turn, represents an additional 5% improvement on the DSM derived using the *Gebala Partitioning Procedure*.

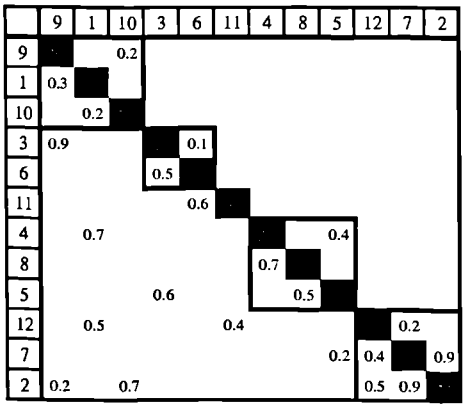


Figure 5.6c: Test Case 1: DSM derived using the new Scott Partitioning Procedure

The iterative blocks of activities implied by each of the two resulting DSMs share a number of similarities. Both procedures yield a block containing activities 1, 9, and 10, as well as a block containing activities 3 and 6, and in both cases, activity 11 is sequenced outside of any iterative block. The biggest difference lies in the sequencing of activities 2, 4, 5, 7, 8, and 12. The *Gebala Partitioning Procedure* sequences the activities in a single block, whilst the *Scott Partitioning Procedure* sequences 2 smaller blocks, each containing 3 activities. (In Chapter 6, a procedure for identifying iterative blocks is detailed.)

At the end of an iterative block, a design review is used to determine which guesstimates have to be revised and, hence, whether or not the iteration implied by the block will in fact be necessary. The use of guesstimates can have a cumulative error effect on all downstream activities that use such data, either directly or indirectly. Design reviews are invoked such that these cumulative error effects are limited only to those activities within the same iterative block. In this respect, smaller iterative blocks imply smaller cumulative error effects, which, in turn, can go some way towards reducing the probability of iteration.

Based on this notion, there is a higher probability that the iterative block in Figure 5.6b, containing activities 12, 7, 2, 5, 4, and 8, will induce more iteration than the two smaller blocks in Figure 5.6c, containing the same activities. Two design reviews are required in the second case, as opposed to a single review in the first case. However, it is likely that the two design reviews associated with the two blocks in Figure 5.6c would merely constitute informal discussion between the relevant members of the product development organisation (PDO).

The single review, associated with the single block in Figure 5.6b, involves more data, and more input from different individuals and, as a result, it is likely that a more complex, time-consuming and formal discussion will be required. Based on these observations, it is considered that the DSM derived using the *Scott Partitioning Procedure* has advantages over the DSM derived using *the Gebala Partitioning Procedure.*

5.5.2 Test Case Two

Test Case Two is taken from Reference [Steward 81a]. The precedence matrix illustrated in Figure 5.7a, when evaluated using the new objective function, yields a measure of iteration equal to 1,809,280.

Figure 5.7b illustrates the DSM that results from the application of the *Steward Partitioning Procedure.* This DSM, when evaluated using the new objective function, yields a measure of iteration equal to 1,264,881. In comparison with the precedence matrix, this represents a reduction in iteration equal to 30%.

No.	Description	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	consum exp			1			1				1							1			
2	captl form																				
3	corp sav				1																
4	corp profit																	1			
5	cap consum																1			1	
6	emp comp					1											1				
7	no. wage ern					1											1			1	
8	wage indx							1													
9	imports						1				1								1	1	
10	farm income			1			1					1						1	1		
11	ag prices indx																		1		
12	per assets			1			1				1				1			1			
13	ent assets						1									1			1		
14	bond yield																				
15	comm yield																				
16	nat income	1	1			1				1											
17	oth income						1				1						1				
18	price indx						1	1	1												
19	priv cap		1			1															
20	corp surplus			1																	

Figure 5.7a: Test Case 2: Precedence Matrix [Steward 81a]

No	Description	2	1	3	4	5	6	7	8	9	10	11	16	17	18	19	14	12	15	13	20
2	captl form																				
1	consum exp			1			1				1			1							
3	corp sav				1																
4	corp profit													1							
5	cap consum												1			1					
6	emp comp					1							1								
7	no wage ern					1							1			1					
8	wage indx							1													
9	imports						1				1				1	1					
10	farm income			1			1					1		1	1						
11	ag prices indx														1						
16	nat income	1	1			1				1											
17	oth income						1				1		1								
18	price indx						1	1	1												
19	priv cap	1				1															
14	bond yield																				
12	per assets			1			1				1			1			1				
15	comm yield																				
13	ent assets						1								1				1		
20	corp surplus			1																	

Figure 5.7b: Test Case 2: DSM derived using the Steward Partitioning Procedure

Figure 5.7c illustrates the DSM that results from the application of the new *Scott Partitioning Procedure*. This DSM, when evaluated using the new objective function, yields a measure of iteration equal to 372,611. In comparison with the precedence matrix, this represents a reduction in iteration equal to 79% which, in turn, is more than double the improvement associated with the DSM derived using the *Steward Partitioning Procedure*.

No.	Description	2	15	14	19	5	16	6	7	8	18	11	17	4	3	10	9	1	20	13	12
2	captl form																				
15	comm yield																				
14	bond yield																				
19	priv cap	1				1															
5	cap consum				1		1														
16	nat income	1				1											1	1			
6	emp comp					1	1														
7	no. wage ern				1	1	1														
8	wage indx								1												
18	price indx							1	1	1											
11	ag prices indx										1										
17	oth income						1	1									1				
4	corp profit												1								
3	corp sav													1							
10	farm income							1			1	1	1		1						
9	imports				1			1			1					1					
1	consum exp							1					1		1	1					
20	corp surplus														1						
13	ent assets		1					1			1										
12	per assets			1				1					1		1	1					

Figure 5.7c: Test Case 2: DSM derived using the new Scott Partitioning Procedure

In comparison, the two procedures yield significantly different sequences. Whilst both identify a single iterative block containing the same activities, the iterative block in Figure 5.7b contains 18 data-dependencies above the leading-diagonal which, in turn, represents 18 guesstimates. In comparison, the iterative block in Figure 5.7c represents a far superior sequence where only 5 guesstimates have to be made.

In this respect, the DSM derived using the *Scott Partitioning Procedure* incorporates a reduced amount of uncertainty. Fewer guesstimates and reduced uncertainty imply smaller cumulative error effects which, as stated previously, can go some way towards reducing the probability of iteration. Based on reduced uncertainty, the DSM derived using the *Scott Partitioning Procedure* has advantages over the DSM derived using *the Steward Partitioning Procedure.*

5.5.3 Test Case Three

Test Case Three is taken from Reference [Steward 91]. The precedence matrix illustrated in Figure 5.8a, when evaluated using the new objective function, yields a measure of iteration equal to 14,692,300.

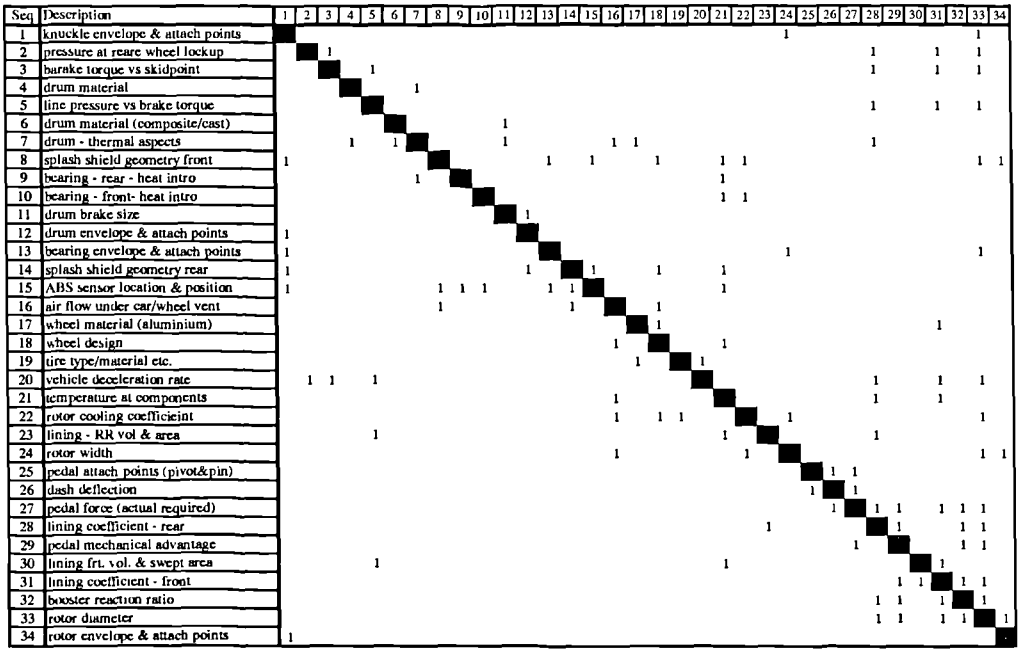


Figure 5.8a: Test Case 3: Precedence Matrix [Steward 91]

Figure 5.8b illustrates the DSM that results from the application of the *Black Partitioning Procedure*. This DSM, when evaluated using the new objective function, yields a measure of iteration equal to 5,342,840. In comparison with the precedence matrix, this represents a reduction in iteration equal to 64%.

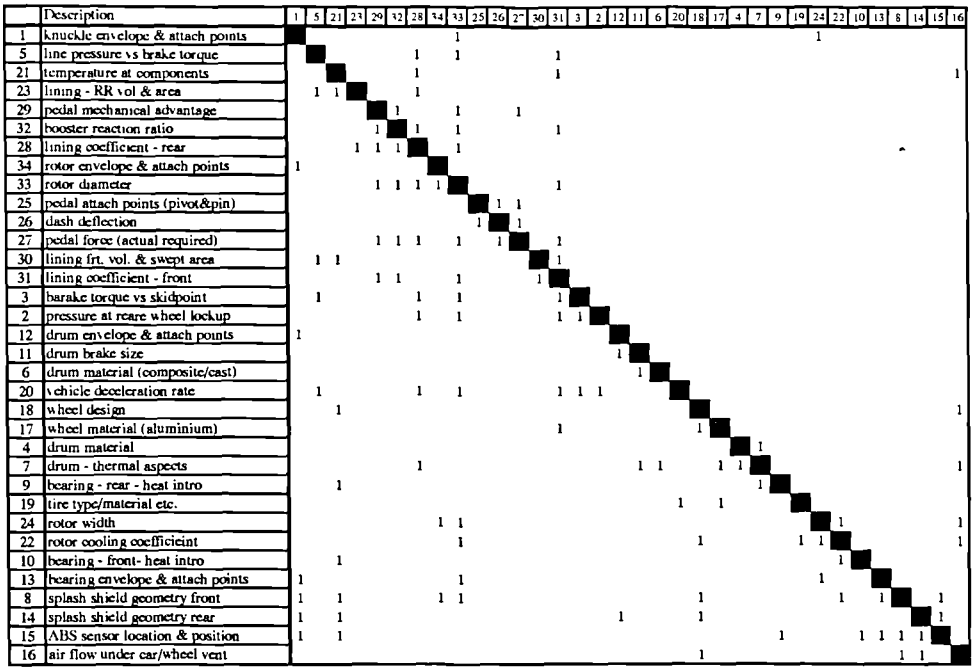


Figure 5.8b: Test Case 3: DSM derived using the Black Partitioning Procedure

Figure 5.8c illustrates the DSM that results from the application of the new *Scott Partitioning Procedure.*

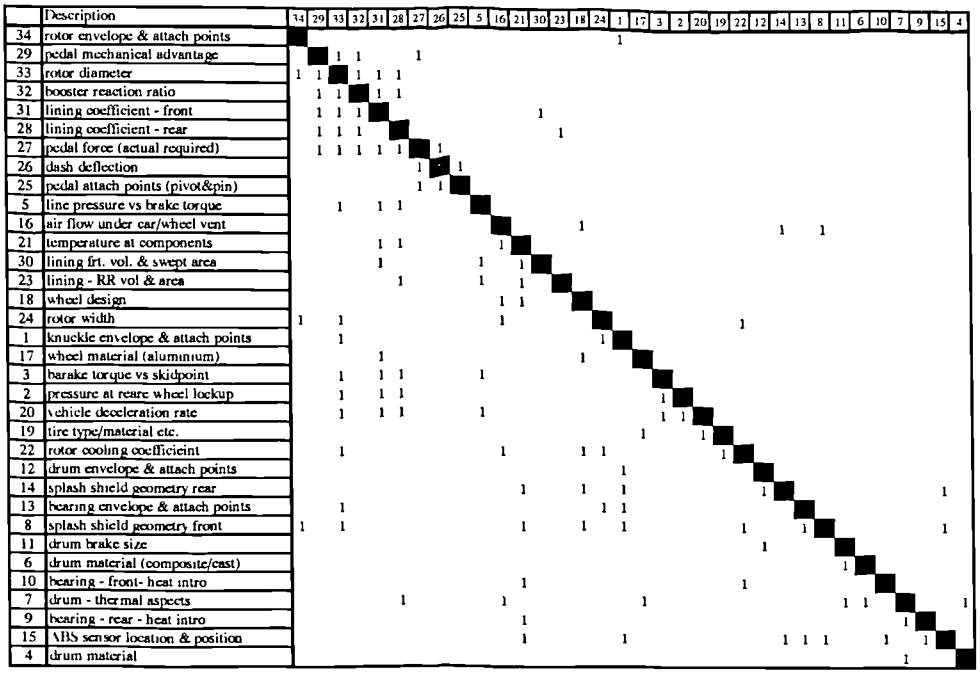


Figure 5.8c: Test Case 3: DSM derived using the new Scott Partitioning Procedure

This DSM in Figure 5.8c, when evaluated using the new objective function, yields a measure of iteration equal to 3,434,500. In comparison with the precedence matrix, this represents a reduction in iteration equal to 77% which, in turn, represents an additional 13% improvement on the DSM derived using the *Black Partitioning Procedure.*

The procedures derive significantly different sequences, although neither can develop a sequence that allows the partitioning of the 34 activities into smaller iterative blocks. In the DSM derived using the *Black Partitioning Procedure,* there are 30 data-dependencies above the leading-diagonal, compared to 20 in the DSM derived using the *Scott Partitioning Procedure.* Like Test Case Three, based on reduced uncertainty, the DSM derived using the *Scott Partitioning Procedure* has advantages over the DSM derived using *the Black Partitioning Procedure.*

5.5.4 Test Case Four

Test Case Four is taken from Reference [Austin 89]. The precedence matrix illustrated in Figure 5.9a, when evaluated using the new objective function, yields a measure of iteration equal to 14,714,973.

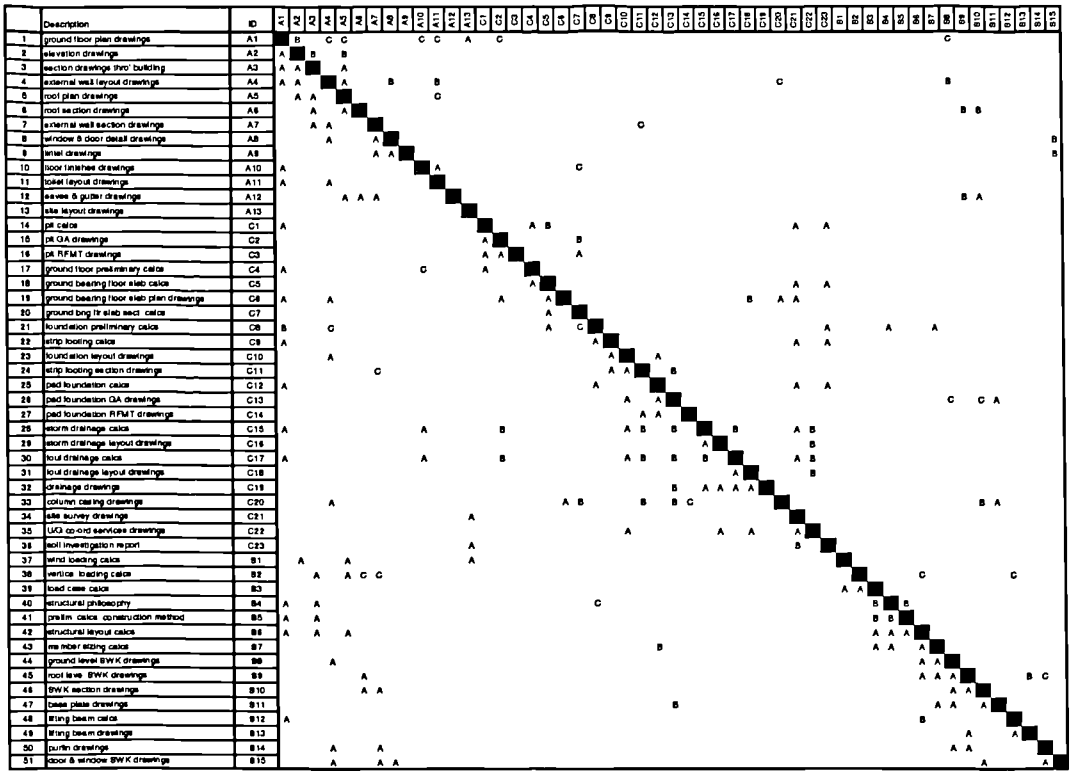


Figure 5.9a: Test Case 4: Precedence Matrix [Austin 89]

Figure 5.9b illustrates the DSM that results from the application of the *Austin Partitioning Procedure*. This DSM, when evaluated using the new objective function, yields a measure of iteration equal to 3,719,681. In comparison with the precedence matrix, this represents a reduction in iteration equal to 75%.

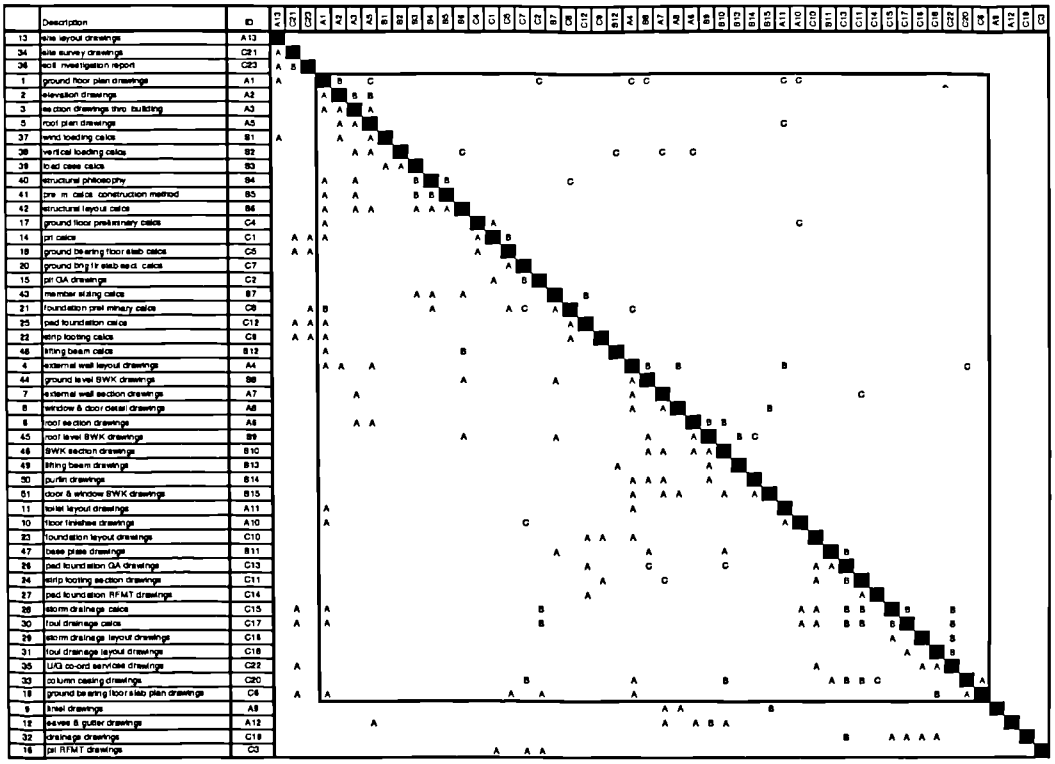


Figure 5.9b: Test Case 4: DSM derived using the Austin Partitioning Procedure

Figure 5.9c illustrates the DSM that results from the application of the new *Scott Partitioning Procedure*. This DSM, when evaluated using the new objective function, yields a measure of iteration equal to 3,483,780. In comparison with the precedence matrix, this represents a reduction in iteration equal to 76% which, in turn, represents an additional 1% improvement on the DSM derived using the *Austin Partitioning Procedure*.

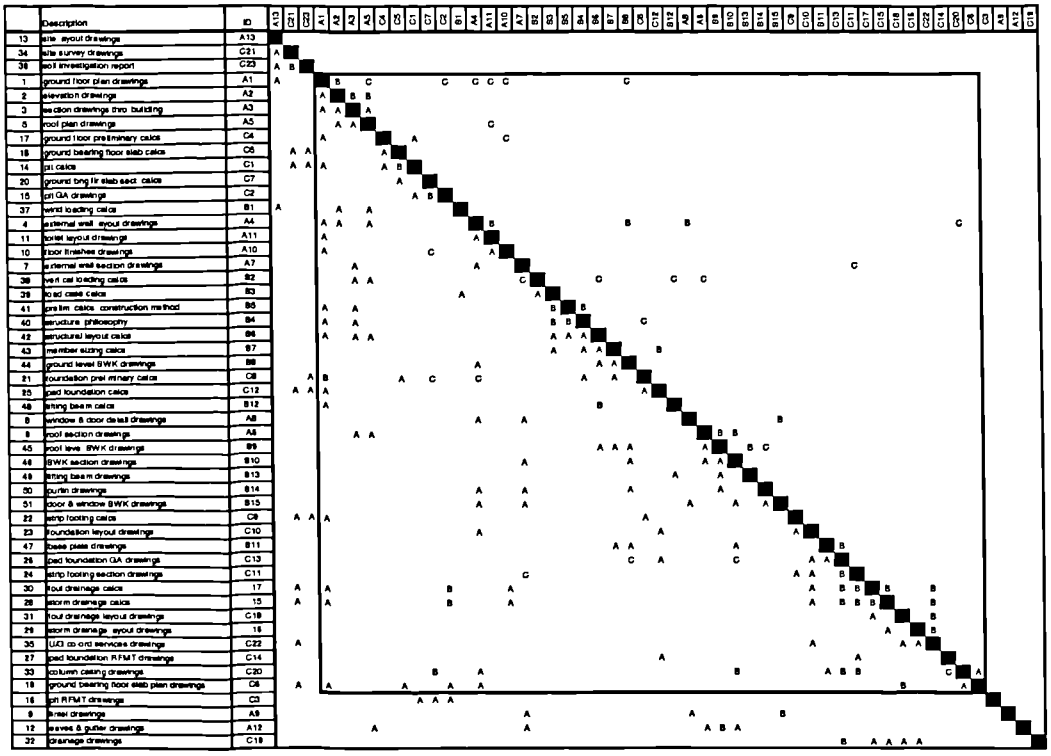


Figure 5.9c: Test Case 4: DSM derived using the new Scott Partitioning Procedure

The DSMs derived by the two partitioning procedures do not vary significantly either in terms of sequence or the associated measures of iteration. Both matrices incorporate a large iterative block that varies in sequence, yet contains the same activities.

As highlighted previously in Sub-Section 5.3.1, the *Austin Partitioning Procedure* is a computer-based program that utilises Steward's partitioning and tearing procedures. However, whilst the program incorporates an automatic tearing procedure, Austin indicates that the process of tearing a partitioned matrix using such a procedure is not as effective as a more subjective approach to tearing based on feedback from engineers with relevant knowledge and experience. [Austin 96].

Whilst the DSMs in Figures 5.9b and 5.9c do no vary significantly either in terms of sequence or the associated measures of iteration, the DSM of Figure 5.9c has been derived using a fully

automatic procedure without any knowledge of the modelled system. The fact that, in order to derive the DSM of Figure 5.10b, user intervention has been required, may indicate that the fully automatic *Scott Partitioning Procedure* is more efficient than the *Austin Partitioning Procedure.*

5.5.5 Test Case Five

Test Case Five is taken from Reference [Steward 81a]. The precedence matrix illustrated in Figure 5.10a, when evaluated using the new objective function, yields a measure of iteration equal to 23,510,500.

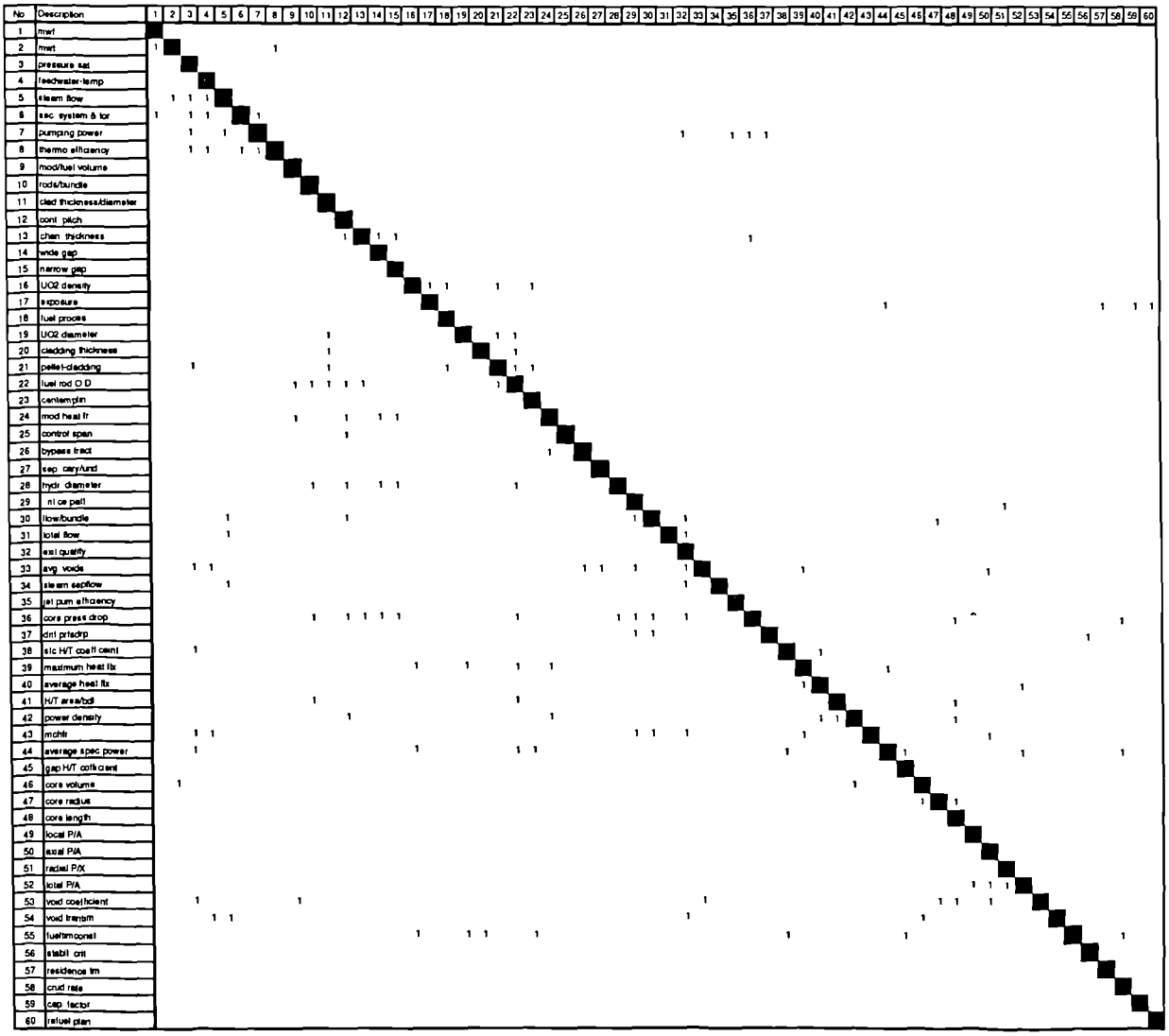


Figure 5.10a: Test Case 5: Precedence Matrix [Steward 81a]

Figure 5.10b illustrates the DSM that results from the application of the *Steward Partitioning Procedure.* This DSM, when evaluated using the new objective function, yields a measure of

iteration equal to 5,672,083. In comparison with the precedence matrix, this represents a reduction in iteration equal to 76%.

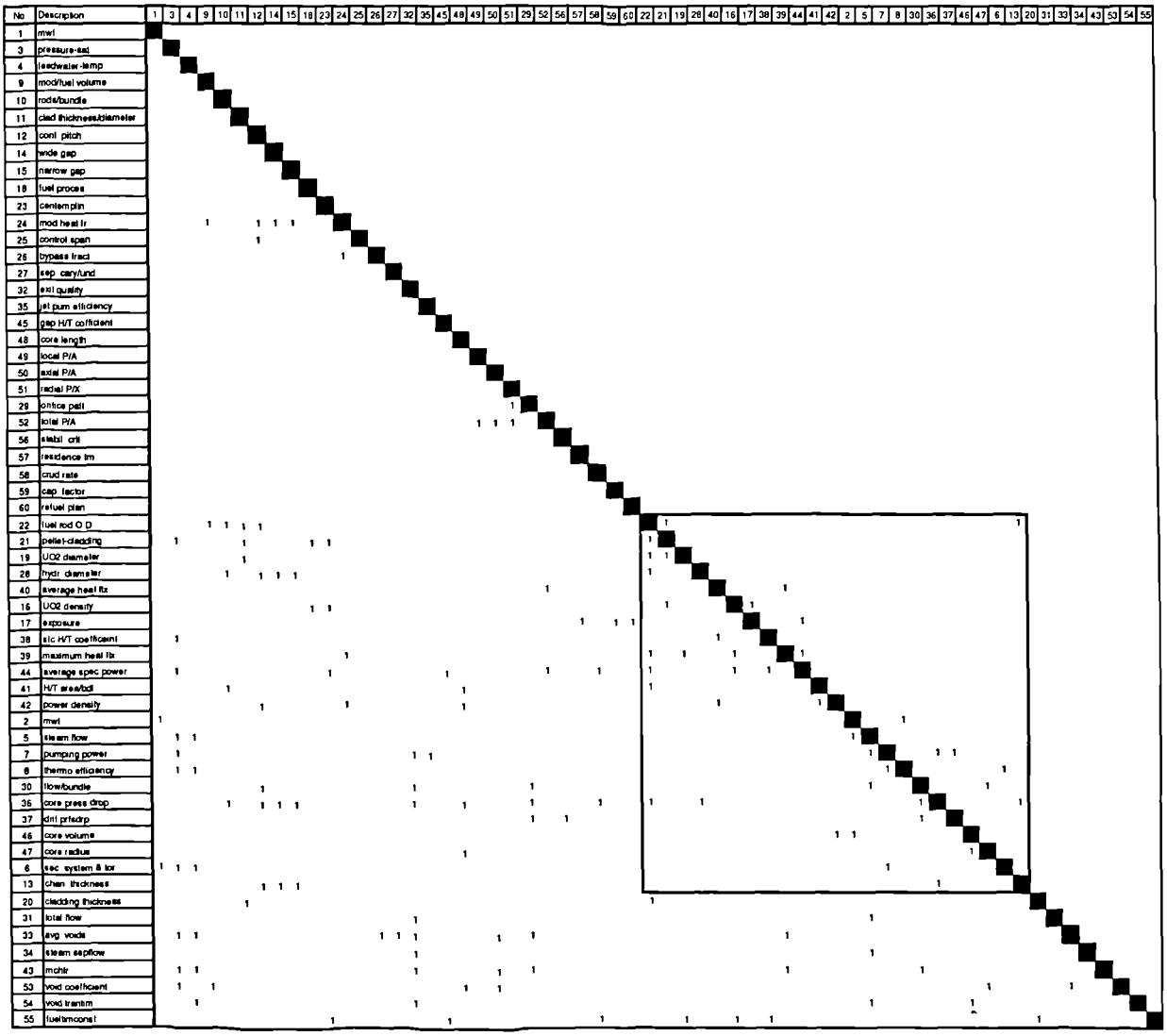


Figure 5.10b: Test Case 5: DSM derived using the Steward Partitioning Procedure

Figure 5.10c illustrates the DSM that results from the application of the new *Scott Partitioning Procedure*. This DSM, when evaluated using the new objective function, yields a measure of iteration equal to 2,757,320. In comparison with the precedence matrix, this represents a reduction in iteration equal to 88% which, in turn, represents an additional 12% improvement on the DSM derived using the *Steward Partitioning Procedure*.

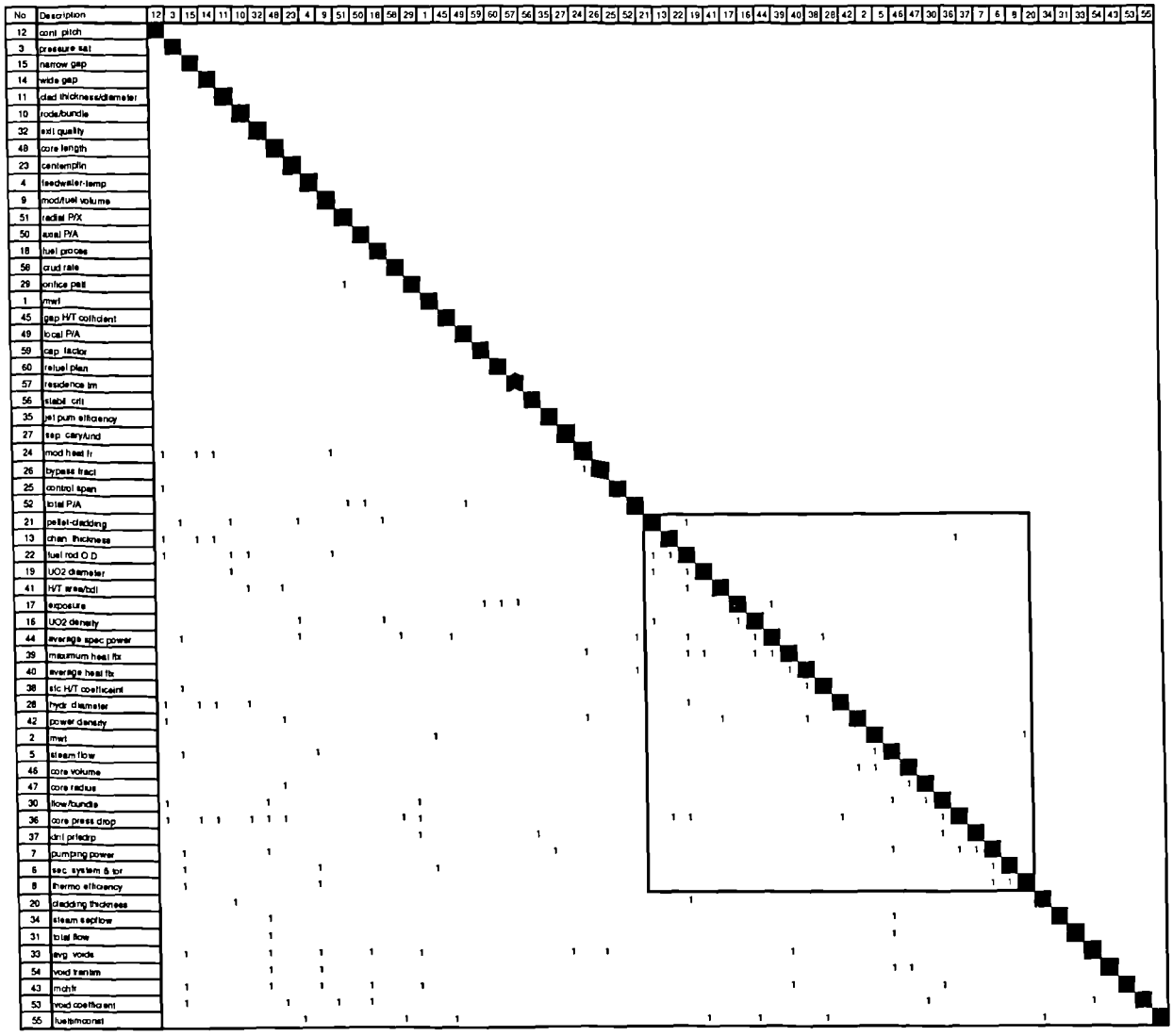


Figure 5.10c: Test Case 5: DSM derived using the new Scott Partitioning Procedure

The procedures derive significantly different sequences, although both derive a single iterative block containing the same 23 activities. However, whilst the major block associated with both design structure matrices contains the same activities, that derived by the *Scott Partitioning Procedure* arranges the 23 activities in a sequence that relies on a reduced number of guesstimates. (5 in comparison to the 12 found in the DSM derived using the *Steward Partitioning Procedure*).

From an observation of Figure 5.10c, with the exception of Activity No. 29, the absence of data-dependencies below the leading-diagonal indicates that the first 25 activities (Activity No. 12 to Activity No. 27) could be processed in parallel, resource availability permitting. In turn, this information would become clear from the creation of a network plan, using the data-dependencies as precedence constraints, however prior to any subsequent planning analysis, a

DSM can be used to communicate, to the PDO, the rationale behind any subsequent plan. In this respect, the *Scott Partitioning Procedure*, because of it's new *objective function* and it's efficient *GA-Local Search Procedure*, derives DSMs which are easier to read, communicate and interpret, prior to any subsequent planning analysis.

5.6 Chapter Summary

Chapter 5, using the *GA-Local Search Procedure*, addresses the *ActRes Problem* of deriving a near optimal sequence of activities based on the specific objective of *minimising iteration.* The chapter begins by considering the role iteration plays in the design-development phase of a product and introduces a classification scheme to describe iteration. Following this introduction, the chapter goes on to report on prior research which, based on the objective of minimising iteration, uses the design structure matrix (DSM) representation in conjunction with various search techniques to derive a near optimal sequence of activities.

Towards the end of the chapter, a new solution approach to the *ActRes Problem* of deriving a near optimal sequence of activities based on the specific objective of *minimising iteration* is introduced. Using five test cases, this new solution approach is tested and the results are compared against prior solutions published in the literature.

Finally, the current research documented in Chapter 5 contributes to the subject in a number of areas. Most specifically, a new partitioning procedure is introduced. The *Scott Partitioning Procedure* is based on the previously described *GA-Local Search Procedure* and a newly developed objective function. This new function is a mathematical expression that can be used to measure the inter-activity iteration associated with a given sequence of activities and varies from those developed previously since it aims to measure unanticipated iteration as well as that which is implied through the need for guesstimates.

Having demonstrated the new partitioning procedure by comparing five results derived using the procedure against those derived by others, it could be concluded that the new procedure offers a number of advantages. In terms of functionality, it can develop sequences that; (*i*) incorporate smaller iterative blocks (*See* Test 1); (*ii*) incorporate the requirement for fewer guesstimates (*See* Tests 2 & 3); (iii) are derived automatically and yet still reflect reality (See Test 4); and, (*iv*) matrices that are easier to read, communicate and interpret (*See* Test 5).

6. The Sequencing of Activities Based on Multiple Criteria and the Derivation of Activity Network Diagrams

6.1 Introduction

Chapter 4 first introduced the *GA-Local Search Procedure* which, as the principal mechanism of the third function of the proposed modelling strategy can be used to derive a near optimal sequence of activities based on a given objective. Chapter 5 then went on to addresses the third strategic function, that is, the *ActRes Problem* of deriving a near optimal sequence of activities based on the specific objective of *minimising iteration.*

Whilst all prior research based on the design structure matrix (DSM) system has been focused on the objective of minimising iteration, Chapter 6 now highlights that, under an appropriate set of conditions, there is another objective which should be used in conjunction with that of minimising iteration. This second objective is that of *maximising concurrency.*

Concurrency, that is, the simultaneous processing of activities, is desirable since it implies being able to reduce lead-times. As will be explained in Section 6.2, the appropriate set of conditions for adopting this new objective relate to *design reuse* and, in particular, are appropriate to a design-development environment where the reuse of historic design-data is facilitated. Consequently, Section 6.2 introduces the notion of design reuse and describes a number of initiatives aimed at encouraging the reuse of design-data.

With a better knowledge of historic data that represents relevant experiences, there exists the potential for making more accurate guesstimates. As a result, whilst it is inevitable that the adoption of the objective of maximising concurrency will be at the expense of having to use more guesstimates, the increased accuracy of guesstimates means that iteration and it's resulting cost continues to be kept to a minimum.

Section 6.3 investigates the relationship between iteration and concurrency in more detail and illustrates that, in addition to influencing the amount of technical uncertainty and iteration, the optimal sequencing of activities can also have a significant influence on the lead-time and accumulated-time associated with a set of activities. Section 6.3 concludes with a new

combined objective function which, optimised using the previously described *GA-Local Search Procedure*, can be used to derive sequences which imply minimum iteration and, at the same time, incorporate the maximum potential for the concurrent processing of activities.

Previously, in Chapter 3, a *strategy for modelling the design-development phase of a product* was introduced and merely summarised with respect to the last three strategic functions. Having described the enabling research in the preceding text, Section 6.4 demonstrates the third and fourth functions of the modelling strategy using the warship case study. Finally, Section 6.5 summarises the chapter and highlights how the research specifically documented in Chapter 6 represents a contribution.

6.2 Design Reuse

In today's business environment, where gaining competitive advantage increasingly means getting new products to market faster, engineering companies are continuously developing and implementing strategies that enable them to design-develop products faster. Chapter 2 has already focused on one such strategy by summarising some of the most influential initiatives associated with concurrent engineering (CE).

Through the improved integration and communication of shareable data, information and knowledge, CE-based initiatives facilitate the concurrent processing of both product and production-process design-development activities. In addition, CE ensures that the *unnecessary rework* associated with the traditional "over-the-wall" approach to design-development and production is avoided.

The notion of avoiding unnecessary rework also lies at the centre of initiatives which, based on *design reuse*, are gaining popularity. Such initiatives assume that, within traditional design-development, there is much work that should not have to be done.

Based on the view of a product, first introduced in Section 3.2, as that of a unique aggregation of multi-level standard physical objects, there exists great potential for design reuse. By re-using part of a solution from a previous product design, time and effort can be saved on the current product design. Re-useable solutions, such as specific and parametric standard design modules and components, can be incorporated into a 3D CAD/CAE/CAM system that, in addition to having a 3D geometry modelling capability, can interact with a set of *object-*

oriented user libraries of preferred design solutions. Such libraries should be readily accessible to members of the product development organisation (PDO) such that selected contents can be efficiently incorporated into newly adapted product schema.

Design reuse tends to imply the reuse of *design form and function* as represented by specific and parametric standard modules and components. However, within the context of the proposed modelling strategy, in the absence of real data yet to be derived for the current design-development, design reuse is interpreted as the reuse of historic design-data to facilitate the improved accuracy of guesstimates.

Amongst other forms of media, this historic data can be stored as (*i*) empirical and heuristic design-development procedures in handbooks and manuals; (*ii*) standard scientific and engineering data series; and, (*iii*) catalogues. In this respect, a comprehensive design-development environment should be created which, electronically or otherwise, should provide access to this historic data, facilitate it's use, and generally help the PDO to organise and reuse what they learn.

Having introduced the notion of design reuse and, in particular the reuse of historic design-data, the reader is directed to [Duffy 98] for an overview of a number of computer-based systems that support the reuse of historic design-data.

6.3 The Multiple Criteria ActRes Problem

6.3.1 The Relationship Between Iteration and Concurrency

As stated at the beginning of Section 5.3, prior research based on the design structure matrix (DSM) system focuses on the criterion of iteration and, more specifically, on the objective of minimising iteration. However, whilst the minimisation of iteration has been viewed by prior research as the overriding objective associated with the *ActRes Problem*, other objectives may be appropriate. In particular, it will be demonstrated here that, whilst the minimisation of iteration reduces technical uncertainties by maximising the availability of *real and accurate data* to each activity, it can be at the cost of reducing the potential for *concurrency*.

Concurrency is desirable because it implies being able to reduce lead-times. Therefore, it is hypothesised that, by sequencing activities based on the sole objective of minimising iteration

without adequate attention to the potential for concurrency, opportunities for reducing design-development lead-times are likely to be missed. In order to explain this hypothesis further, it may be necessary to review the design structure matrix representation and it's interpretation.

Based on an observation of the simple binary precedence matrix of Figure 6.1, it may be observed once again that (*i*) *sequential activities* can be identified by data-dependencies located in matrix positions below and immediately adjacent to the leading diagonal; (*ii*) *parallel activities* are identified by the absence of data-dependencies in such positions; and, (*iii*) *coupled activities* are identified by data-dependencies located above the leading diagonal.

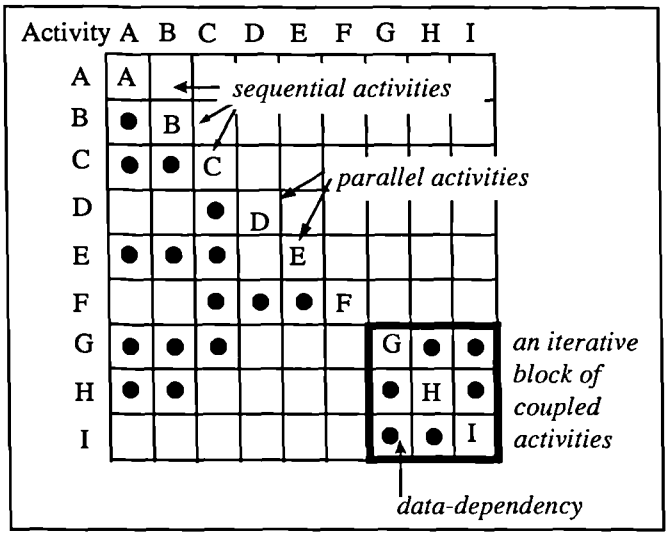


Figure 6.1: A precedence matrix of design-development activities and their data-dependencies

From an observation of the design structure matrices (DSMs) illustrated previously in Figures 5.6c - 5.10c, it can be concluded that, when activities are sequenced based on the sole objective of minimising iteration, a pattern of data-dependencies indicative of sequential activities tends to result. That is, the activities are sequenced in a way that results in a large number of data-dependencies being re-positioned below and immediately adjacent to the leading-diagonal.

In this respect, the DSMs associated with activities that are sequenced based on the sole objective of minimising iteration tend to incorporate a large number of activities that are linked sequentially and therefore cannot be processed in parallel. In this respect, whilst iteration is minimised, it is at the cost of reducing the potential for concurrency. Consequently, when a near optimal sequence is derived based on the sole objective of

minimising iteration, some opportunities for reducing design-development lead-times can be missed.

Consider a simple example that illustrates this hypothesis. Figure 6.2a illustrates a sequence of four activities (A, B, C and D) represented in a DSM. The data-dependencies are defined according to the four level dependency scheme detailed in Appendix II. (High dependency = 1.0, Average dependency = 0.5, Low dependency = 0.1, No dependency = 0.0). In this example, the values on the leading diagonal correspond to activity durations.

Case 1: Derive a sequence based on the sole objective of minimising iteration

In this case, the activities have been sequenced based on the sole objective of minimising iteration. In actuality, the resulting sequence incorporates no iteration at all. Figure 6.2b illustrates the Gantt chart that is based on the DSM of Figure 6.2a. Note that the strong data-dependency between Activities A and B, and that between Activities C and D, implies that Activity B cannot start until the completion of Activity A. In the same way, Activity D cannot start until the completion of Activity C. However, the weak data-dependency between Activities B and C implies that Activity C can begin prior to the completion of Activity B.

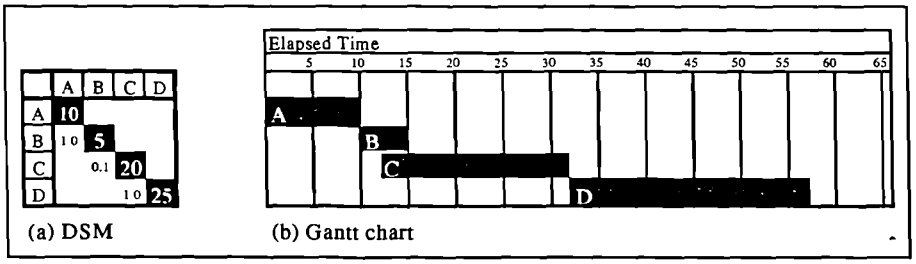


Figure 6.2. Activities sequenced on the basis of minimising iteration

Case 2: Derive a sequence based on the dual objectives of maximising concurrency and minimising iteration

Figure 6.3a illustrates the same activities and data-dependencies as illustrated in Figure 6.2a. However, in this case, the activities have been re-sequenced and their data-dependencies have been re-positioned based on the dual objectives of maximising concurrency as well as minimising iteration. Whilst the example is trivially small, it illustrates how, using multiple objectives, a sequence of activities and associated DSM, different to that illustrated in Figure 6.2a, can be derived.

Figure 6.3b illustrates the Gantt chart that is based on the DSM of Figure 6.3a assuming that the iteration, implied by the data-dependency above the leading diagonal, is in fact necessary. The DSM of Figure 6.3a dictates that Activity A is processed first. Activity C is next and, because Activity C does not depend on data from Activity A, both activities can be processed concurrently, with Activity C using a guesstimate of the data that is derived subsequently by the processing of Activity B. Activity B depends on data from Activity A, and therefore, has to wait for the completion of Activity A before it can begin. On completion of Activity B, it is found that the original guesstimate used by Activity C was not sufficiently accurate. In this case Activity C is stopped prematurely and re-started using the real data derived from Activity B (hence the shorter duration of the first occurrence of Activity C in Figure 6.3b). As in the previous case, because Activity D depends on data from Activity C, Activity D cannot begin until the second occurrence of Activity C has been completed.

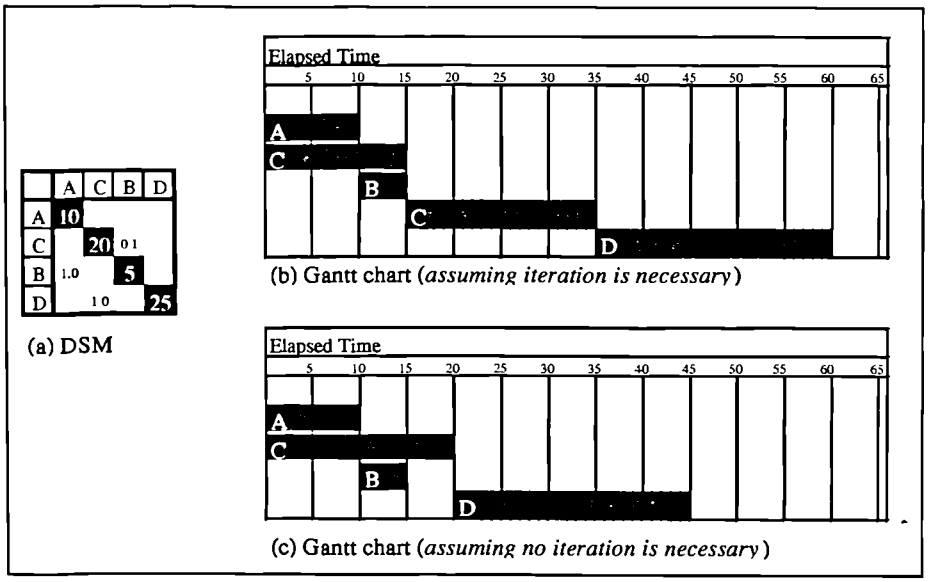


Figure 6.3. Activities re-sequenced on the basis of minimising iteration and maximising concurrency

On the other hand, Figure 6.3c illustrates the Gantt chart that is derived assuming that, whilst iteration is implied by the data-dependency above the leading diagonal, it is not in fact necessary. This could be a result of the guesstimate used by Activity C being sufficiently close to the real data subsequently derived by the processing of Activity B such that iteration in the form of the re-processing of Activity C is avoided.

Hence, if an accurate guesstimate can be made of the data that is required by Activity C from Activity B, then the precedence relationship implied by the data-dependency between the two activities can be ignored. The *decoupling* of a data-dependency in this way means that, in

effect, Activity C is no longer constrained by the precedence of Activity B. As a result, the lead-time associated with the processing of the four activities can be reduced from 57 time units (Figure 6.2b) to 45 time units (Figure 6.3c). This represents a saving of 21% on lead-time.

However, if an accurate guesstimate cannot be made, then iteration will be necessary and, as a result, the lead-time associated with the processing of the four activities may increase as a result of iteration. In the example above, as a result of iteration the lead-time associated with the processing of the four activities increases from 57 time units (Figure 6.2b) to 60 time units (Figure 6.3b). Furthermore, the *accumulated-time*, or the total time spent processing all activities, increases from 60 time units to 75 time units. This represents an increase of 42% in terms of accumulated time.

Using this simple example, it can be observed that, in addition to influencing the amount of technical uncertainty and iteration, the near optimal sequencing of activities can also have a significant influence on the lead-time and accumulated-time associated with a set of activities.

Under *appropriate conditions*, the near optimal sequencing of activities based on multiple objectives can maximise the opportunity for reducing lead-time and still minimise technical uncertainty and any resulting iteration. These so-called appropriate conditions apply to design-development where (*i*) reducing design-development lead-time is desirable; (*ii*) accurate guesstimates can be made; and, (*iii*) the availability of resources permit concurrency.

The accuracy of guesstimates is improved by management strategies, such as those mentioned previously in Section 6.2, that encourage and facilitate the effective storage and reuse of historic design data and solutions. Furthermore, whilst resources such as members of the product development organisation (PDO) may be scarce, the application of the resource-constrained project scheduling technique, detailed later in Chapter 7, ensures that, through the optimal allocation of resources to activities, opportunities for concurrency are fully realised.

6.3.2 An Objective Function for the Multiple Criteria ActRes Problem

The consideration of concurrency as well as iteration converts the *ActRes Problem* into a multiple-criteria optimisation problem. There are a number of approaches to dealing with multiple criteria. When the criteria are expressed in broadly commensurate terms, as is the

case here, the simplest approach is to *weight* and then *combine* the individual criteria within a single objective function.

The *combined objective function*, used to help solve the multiple-criteria *ActRes Problem*, thus combines two weighted objective functions. The first function to be combined measures the iteration inherent to a given sequence whilst the second measures the concurrency inherent to the same sequence.

(1) The objective function used to measure iteration

The objective function used to measure iteration is detailed in Section 5.4. In order to minimise the value of this objective function, activities will be re-sequenced such that their associated data-dependencies are re-positioned in those matrix positions close to the bottom-left-hand corner of the matrix. In this way, the function in Equation 6.1, repeated from Chapter 5, and minimised using the previously described *GA-Local Search Procedure*, returns activity sequences that imply minimum iteration.

For the purposes of clarification, consider the simple example in Figure 6.4, repeated from Section 5.4. The DSM of Figure 6.4a details 9 activities and their data-dependencies. The matrix of Figure 6.4b details the scaled distance of each matrix position from the bottom-left-hand corner of the matrix (Ω_{ij} in Equation 6.1). By multiplying each data-dependency of Figure 6.4a by the value of Ω_{ij} in the corresponding matrix position of Figure 6.4b, the matrix of products of Figure 6.4c results. Summing these products yields a measure of iteration equal to 19,184 for the sequence of activities and the corresponding DSM of Figure 6.4a.

Objective Function for measuring *Iteration* = $\sum_{i=0}^{i=n-1} \sum_{j=0}^{j=n-1} \left(a_{ij} \times \Omega_{ij} \right)$ *[Equation 6.1]*

a_{ij}	Data-Dependency in row i, column j. (Note: a_{00} represents the top left-hand corner of the matrix)
n	Total number of activities in the sequence
$\Omega_{ij} = (1*[j + (n - i)]^2)$	for ($j < i$) matrix positions below the leading diagonal
$\Omega_{ij} = (100*[j + (n - i)]^2)$	for ($j > i$) matrix positions above the leading diagonal

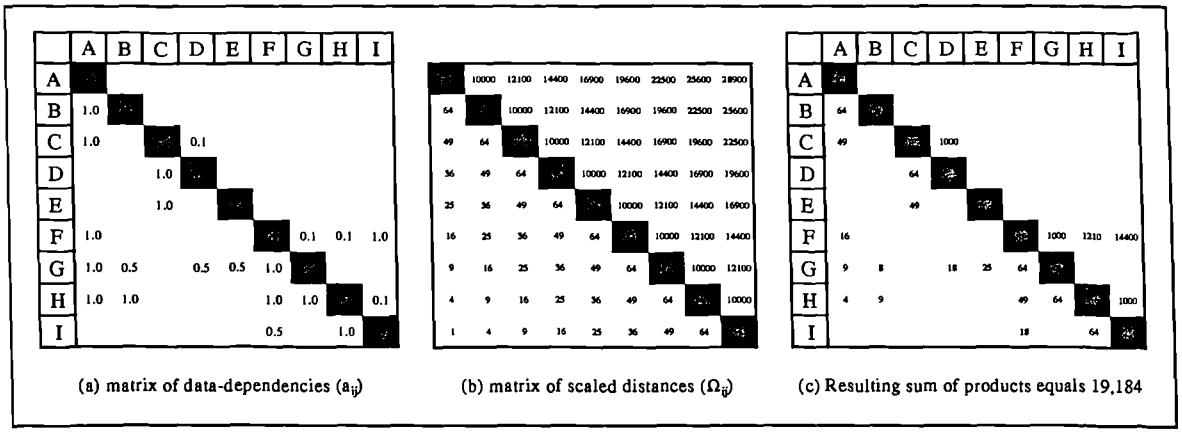


Figure 6.4: An illustration of how Equation 6.1 can be used to measure iteration

(2) The objective function used to measure concurrency

Referring back to the observations about the DSM made at the beginning of Sub-Section 6.3.1, *parallel activities* or *concurrent activities* are identified by the absence of data-dependencies in matrix positions immediately adjacent to the leading diagonal. Therefore, concurrency can be measured according to the position of data-dependencies relative to the leading diagonal. In this respect, a sequence and resulting DSM where there are few data-dependencies in matrix positions immediately adjacent to the leading diagonal could be considered indicative of a sequence that incorporates potential for concurrency.

The objective function used to measure concurrency can be expressed in words as, "the sum of all of the data-dependencies of the matrix, each multiplied by a value which represents, for each matrix position, the scaled distance of the data-dependency from the bottom-left-hand corner of the matrix for data-dependencies below the leading diagonal, and from the leading-diagonal for data-dependencies above the leading diagonal".

In order to minimise the value of this objective function, activities will be re-sequenced such that their associated data-dependencies are re-positioned in those matrix positions close to the bottom left-hand corner of the matrix or, alternatively, in those positions directly above the leading diagonal. In this way, the objective function detailed in Equation 6.2, minimised using the previously described *GA-Local Search Procedure*, returns activity sequences which incorporate maximum concurrency.

Objective Function for measuring *Concurrency* = $\sum_{i=0}^{i=n-1} \sum_{j=0}^{j=n-1} \left(a_{ij} \times \Phi_{ij} \right)$

[Equation 6.2]

a_{ij} Dependency in row i, column j. (Note a_{00} represents the top left-hand corner of the matrix)
n Total number of activities in the sequence
$\Phi_{ij} = (j + [n - i])^2$ for ($j < i$) matrix positions below the leading diagonal
$\Phi_{ij} = (j - i)^2$ for ($j > i$) matrix positions above the leading diagonal

Again, for the purposes of clarification, Figure 6.5a includes the scaled distance of each matrix position as represented by Φ_{ij} in Equation 6.2. By multiplying each data-dependency of Figure 6.4a by the value of Φ_{ij} in the corresponding matrix position of Figure 6.5a, the matrix of products of Figure 6.5b results. Summing these products yields a measure of concurrency equal to 584 for the sequence of activities and the corresponding DSM of Figure 6.4a.

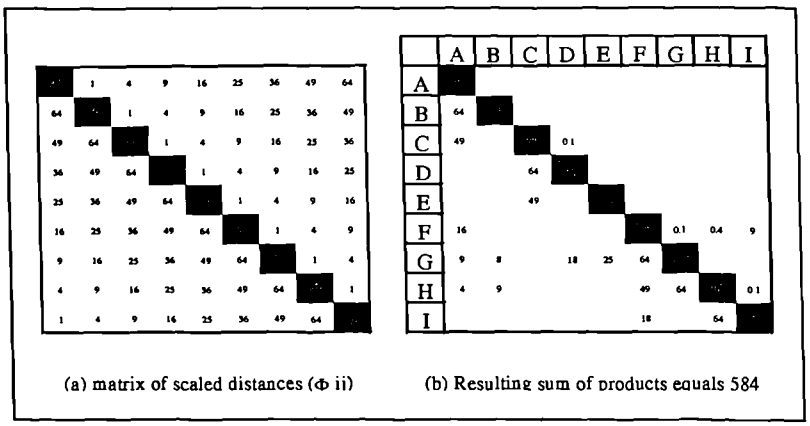


Figure 6.5: An illustration of how Equation 6.2 can be used to measure concurrency

(3) A combined objective function

The combined objective function is simply a weighted sum of the functions used to measure iteration (Equation 6.1) and concurrency (Equation 6.2). However prior to the allocation of *weights*, the measures of iteration and concurrency associated with a given DSM must be *normalised.*

As previously illustrated, the DSM of Figure 6.4a incorporates a measure of iteration equal to 19,184 and a measure of concurrency equal to 584. If the two criteria were to be assigned equal weights, then the combined measure of iteration and concurrency would equal 19,768 (19,184 + 584). In this case, the *GA-Local Search Procedure* would tend to focus on minimising the iteration component simply because the combined measure is dominated by the measure of iteration.

Therefore, in terms of the *GA-Local Search Procedure*, each time a new sequence is derived, prior to combining the two measures into a single measure, the measure of iteration is normalised by dividing the derived measure of iteration by the worst measure of iteration found in the current generation. In the same way, the measure of concurrency is normalised by dividing the derived measure of concurrency by the worst measure of concurrency found in the current generation.

Consider the sequence of activities and the associated DSM illustrated in Figure 6.4a to be one of a population in the current generation. If the worst measure of iteration, associated with one of the sequences in the current generation equals 40,010, then the normalised measure of iteration associated with the sequence and DSM of Figure 6.4a is re-calculated as 0.479 (19,184 / 40,010). In the same way, if the worst measure of concurrency, associated with one of the sequences in the current generation, equals 2,283 then the normalised measure of concurrency associated with the sequence and DSM of Figure 6.4a is re-calculated as 0.255 (584 / 2,283).

By updating the worst measures of iteration and concurrency in each generation, the normalised measures always lie between zero and unity. Therefore when the two values are combined based on an equal weighting, both criteria impact comparably on the overall fitness of each sequence and corresponding DSM.

With both measures normalised, weights can be applied which represent the user's preferences. If the user wants to focus solely on iteration, then the concurrency weight is set to zero and vice-versa. User preferences relating to the setting of weights depend on whether or not the appropriate conditions, discussed previously in Sub-Section 6.3.1, prevail.

The resulting multiple-objective function detailed in Equation 6.3, minimised using the previously described *GA-Local Search Procedure*, is capable of returning activity sequences which incorporate either, or both minimum iteration and maximum concurrency. This constitutes a multiple-criteria solution-approach to the *ActRes Problem*.

Combined Function for measuring *Concurrency & Iteration* = $\sum_{i=0}^{i=n-1}\sum_{j=0}^{j=n-1}\left(a_{ij}\times\left[A\Omega_{ij}+B\Phi_{ij}\right]\right)$ *[Equation 6.3]*

A= Weighting applied to the iteration component of the combined function
B = Weighting applied to the concurrency component of the combined function

6.4 Mapping the Design Structure Matrix to a Time-Line

Previously, in Section 3.6, *a strategy for modelling the design-development phase of a product* was introduced, and the first two strategic functions (function boxes A1 and A2 of Figure 3.3), namely "create a product-design work breakdown structure", and "model the activities and their data-dependencies", were both described and demonstrated using the warship case study.

The third, fourth and fifth strategic functions (function boxes A3-A5 of Figure 3.3) were merely summarised in Section 3.6. The fifth function, namely "derive a resource-constrained schedule", is not demonstrated until Chapter 7. However, based on the enabling research reported in Chapters 4 and 5, as well as that reported in the current chapter, the third and fourth strategic functions, namely "derive a near optimal sequence of activities (the *ActRes Problem*)" and "derive an activity network diagram" are demonstrated, using the warship case study, in the following two sub-sections.

6.4.1 Deriving a Near Optimal Sequence of Activities

"Derive a near optimal sequence of activities", informally referred to as the *Activity Resequencing (ActRes) Problem* represents the third function of the *strategy for modelling the design-development phase of a product* and is represented in Figure 3.3 as function box A3.

The text that follows elaborates on that which is summarised previously in Sub-Section 3.6.3. The inputs to this function are (*i*) a precedence matrix; and, (*ii*) the objectives which are to be used as the basis for deriving a near optimal sequence. Based on the precedence matrix for the warship case study, illustrated previously in Figure 3.4, the equally weighted dual objectives of minimum iteration and maximum concurrency have been used to derive a near optimal sequence of activities.

There are two sequencing constraints associated with the warship case study. Based on Section I.2 in Appendix I, which describes the process of developing the pre-contract design-definition of a warship, the first sequencing constraint ensures that Activities 1-12 in the precedence matrix of Figure 3.4 are sequenced first. These activities relate to the design-development of the warship's overall architecture, it's 3-D geometry and it's principal functional/operational attributes to create an *outline warship solution schema.*

Once an outline warship solution schema has been created, Activities 13-51 in the precedence matrix of Figure 3.4, which relate to the design-development of a schema for each functional sub-system schema, can be processed. The second sequencing constraint ensures that these 39 activities are sequenced such that the schema of those sub-systems which most strongly influence the warship's overall architecture, 3-D geometry and principal functional / operational attributes, are design-developed as early as possible. These constraints are addressed according to the procedure outlined previously in Sub-Section 4.4.6.

Based on these sequencing constraints and the previously mentioned objectives, the activities in the precedence matrix of the warship case study in Figure 3.4 can be optimally re-sequenced. Using the *GA-Local Search Procedure*, the design structure matrix (DSM) for the warship case study, illustrated in Figure 6.6, is derived.

6.4.2 Deriving an Activity Network Diagram

"Derive an activity network diagram" represents the fourth function of the *strategy for modelling the design-development phase of a product* and is represented in Figure 3.3 as function box A4.

The text that follows elaborates on that which is summarised previously in Sub-Section 3.6.4. As indicated by the data-dependencies above the leading-diagonal in Figure 6.6, it is unlikely that a DSM, as input to this function, will model a sequence of activities such that all the input data for each activity will be available when it is required. As a result, guesstimates will be necessary and inaccurate guesstimates can result in iteration.

The newly developed *iterative block resolution (IBR) procedure*, used to resolve the iteration implied by a DSM after re-sequencing, and used to convert the DSM into an activity network diagram, is best described as a step-by-step procedure.

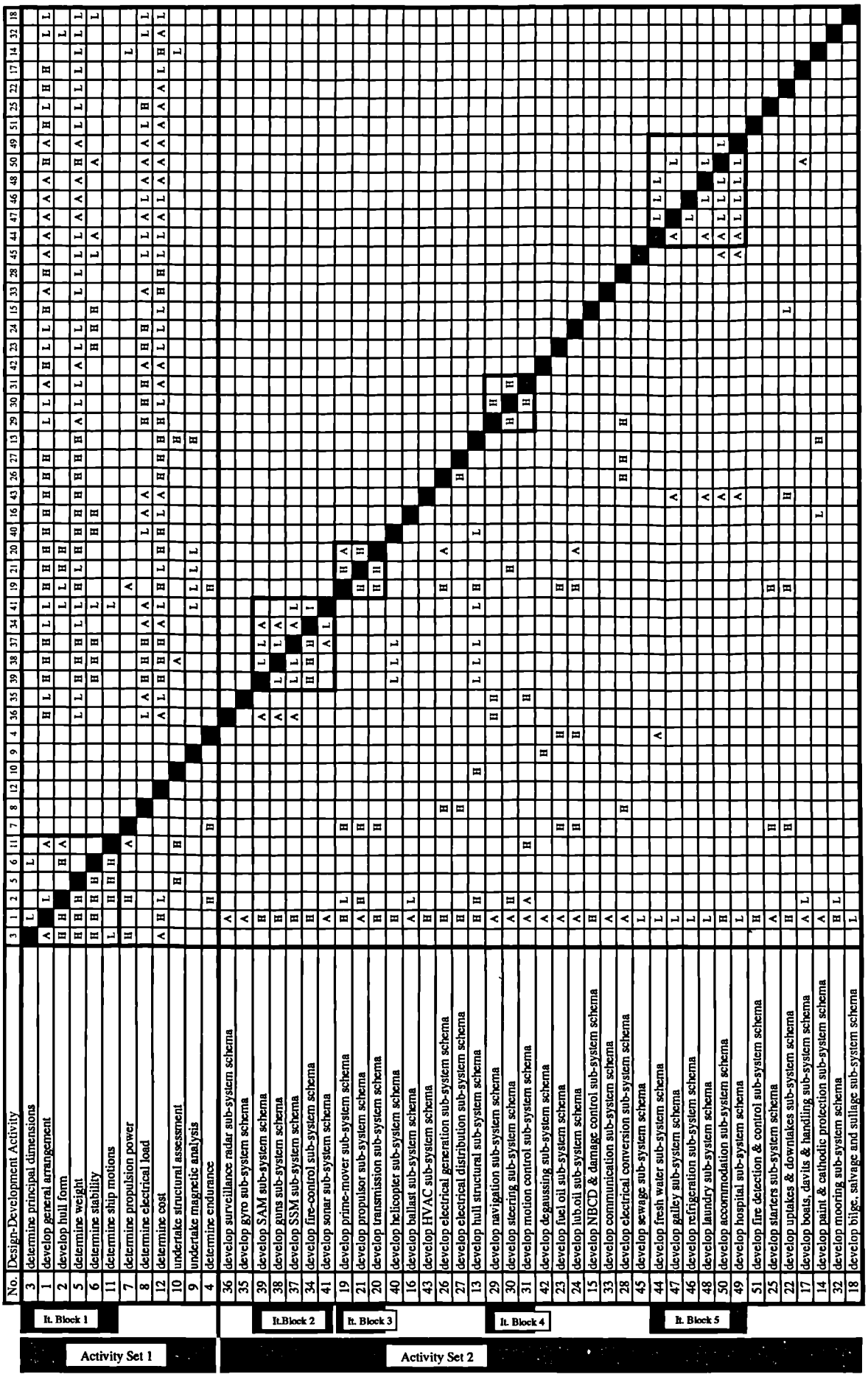


Figure 6.6: The design structure matrix (DSM) for the warship case study

Step 1: Separate the DSM into Activity Sets

To decompose the problem of resolving iteration into smaller and more manageable sub-problems, the DSM is first separated into *activity sets* which group activities according to the product's hierarchy of functional sub-systems.

For example, the DSM for the warship case study of Figure 6.6 is separated into two activity sets. Activity Set 1 encapsulates Activities 1-12, which together describe the design-development of an outline warship solution schema. In this respect, Activity Set 1 groups activities which relate specifically to the *highest level* in the product's hierarchy of functional sub-systems which is the warship system itself. In the same way, Activity Set 2 encapsulates Activities 13-51, which together describe the design-development of a schema for each of the warship's principal sub-systems. In this respect, Activity Set 2 groups activities which relate specifically to the *first level* in the product's hierarchy of functional sub-systems.

Step 2: Identify Iterative-Blocks of Activities

An *iterative-block* groups activities that are mutually inter-dependent on one another for data. Therefore, an iterative-block encapsulates activities that are linked by data-dependencies that are positioned above as well as below the leading-diagonal of a DSM. In this respect, all activities in an iterative-block are either directly, or indirectly dependent on all other activities in the same iterative-block.

Based on this definition, all 51 activities in the DSM of Figure 6.6 can be grouped into one large iterative-block, but perhaps more notably, five localised iterative-blocks of highly inter-dependant activities can be identified. Iterative-Block 1 (*See* Figure 6.6) encapsulates 6 activities that relate to the design-development of the warship's outline solution schema. Iterative-Block 2 encapsulates 5 activities which all relate to the design-development of weapons sub-system schema. Iterative-Block 3 encapsulates 3 activities which all relate to the design-development of propulsion sub-system schema. Iterative-Block 4 encapsulates 3 activities which all relate to the design-development of motion control sub-system schema. Finally, Iterative-Block 5 encapsulates 5 activities which all relate to the design-development of crew-support sub-system schema.

Step 3: Model and Interpret the Iteration implied by each Iterative-Block of Activities

Because the activities grouped together within an iterative-block are mutually inter-dependent on one another for data, it is not possible to derive a sequence of activities that does not result in the need for guesstimates. As an example, consider the 6 possible sequences associated with the 3 activities of Iterative-Block 3 of Figure 6.6. As can be seen from the 6 partial DSMs of Figure 6.7, each of the 6 possible sequences (19-20-21, 19-21-20, 20-19-21, 20-21-19, 21-19-20, 21-20-19) result in the need for guesstimates (as indicated by the data-dependencies above the leading-diagonal).

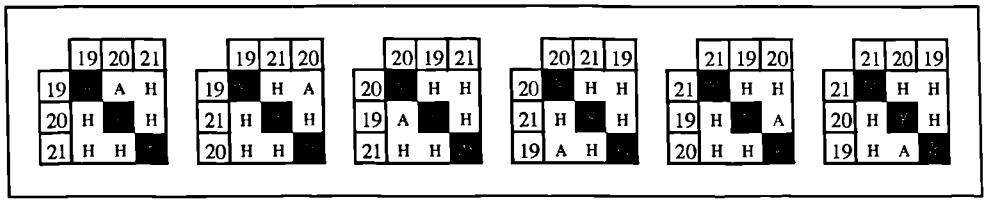


Figure 6.7: The partial DSMs associated with the 6 possible sequences of the 3 activities in Iterative Block 3

As stated previously, the use of guesstimates implies iteration because, if a guesstimate is found to be unacceptably inaccurate when it is compared with the real and accurate data derived subsequently, then the activities which previously used the guesstimate will have to be re-processed.

It is the ability of the design structure matrix (DSM) system to model data inter-dependencies as well as the iteration that is implied by the use of guesstimates, which makes it an effective technique for modelling design-development activities. However, it is the sub-modelling of iterative blocks and their subsequent interpretation that ensures that the effect of iteration is fully understood such that, ultimately, it is accurately reflected in a schedule of activities.

Activities grouped together within an iterative-block can be modelled as either (*i*) sequential activities; or, (*ii*) parallel activities. In a *sequential iterative-block*, activities are processed one after the other in the sequence dictated by the DSM. In a *parallel iterative-block*, activities are processed concurrently.

Smith's *sequential iteration model* [Smith 94] assumes that, after each activity in an iterative-block has been processed, a probabilistic choice determines whether an earlier activity is repeated or whether the next activity in the block is processed.

On the other hand, Steward's *sequential iteration model* [Steward 81a] assumes that, only after all activities in the block have been processed once, is a choice made whether or not to repeat the block or not. Based on this assumption of how activities are processed, Rogers [Rogers 96b] has developed a procedure that evaluates an *iteration-factor* which relates the number of times the block is to be repeated to the number of data-dependencies above the leading-diagonal in an iterative-block.

Smith's *parallel iteration (work transformation) model* [Smith 97] assumes that within an iterative-block, activities are processed concurrently over a period of time. During this period, activities create rework for one another such that they are almost continually re-iterated until no further rework is required or until time runs out, whichever occurs first.

Ultimately, each iterative-block needs to be resolved into a set of discrete activity occurrences such that resources can be accurately allocated. Ideally, the iteration implied by an iterative-block is minimised by assigning a team of co-located members of the product development organisation (PDO) to process each activity in the block concurrently. Assuming the concurrent processing of activities, a parallel iteration model can be used to model the activities in an iterative-block. However, as may become clear in the following text, the concurrent processing of activities is not always possible. Therefore, the first decision to make when interpreting the iteration that is implied by iterative-blocks is whether the activities in an iterative-block should be processed, and hence modelled, as either sequential activities or parallel activities.

A Sequential Iteration Model

Because the number of human resources that would be required to process a large number of activities in parallel is prohibitive, a large iterative-block such as the one that encapsulates all 51 activities of Figure 6.6, can only be processed sequentially and should therefore be modelled using a sequential iteration model.

For the cases where a sequential iteration model is considered appropriate, it would appear that there is a choice to be made between the application of Smith's model and Steward's model. According to Smith's sequential iteration model, after the completion of each activity in an iterative-block, if an unacceptable inconsistency between a guesstimate and some derived data is identified, then *only one* of the previously completed activities in the

block is re-processed. However, this assumption that only one previously completed activity is to be re-processed implies that it is independent of all other activities. In reality, it is very likely that the re-processing of one activity in an iterative-block will result in the need to re-process other previously completed activities which are dependent on the re-processed activity for data.

Based on these shortcomings of Smith's model, and on the experience gained with the warship case study, for those instances where a sequential iteration model is considered appropriate, it is postulated that a variant of Steward's model should be used. In this variant the activities of a sequential iterative-block are processed according to the sequence reflected by the DSM and, whenever necessary, guesstimates are used. The activities, which use guesstimates, are modelled as *multiple occurrence activities*. Such activities are processed full-time for the pre-defined duration during their first occurrence and, based on the feedback of real and accurate data from downstream activities, are partially re-processed, as necessary, intermittently until all activities in the iterative-block have been processed.

Once all activities in a sequential iterative-block have been processed, a *design acceptance review* is undertaken. The purpose of this review is to (*i*) review the technical uncertainties which remain and their implications; and, (*ii*) identify the inputs, procedures and knowledge to be used at the next stage of design-development in order to resolve the technical uncertainties which remain.

Decisions made as a result of this review ultimately determine whether or not the iterative-block is to be repeated in full. In the case of the large sequential iterative-block, which encapsulates all 51 activities of the DSM of Figure 6.6, more often than not, the regular feedback of real and accurate data from activities in Activity Set 2 to those in Activity Set 1, ensures that an acceptable level of convergence has been reached prior to the design acceptance review. As a result, a second complete iteration of the large sequential iterative-block in the DSM of Figure 6.6 is usually unnecessary.

A Parallel Iteration Model

For small iterative-blocks, such as Iterative-Blocks 1-5 of Figure 6.6, it is more likely that there will be sufficient resources to process activities concurrently. Therefore, small

iterative-blocks should be modelled using a parallel iteration model. On completion of a small iterative-block, an informal design-review can be undertaken in order to ensure that an acceptable level of convergence has been reached.

Therefore, the activities in a parallel iterative-block should begin and end at the same time, and the human resources assigned to process the activities should be co-located in order to encourage the efficient and effective integration and communication of inter-dependant data.

In summary, the large iterative-block of Figure 6.6, which encapsulates all 51 activities, is processed, and therefore modelled, sequentially. The 12 activities in Activity Set 1 are modelled as multiple occurrence activities which are processed full-time for their pre-defined durations during their first occurrence and, based on the feedback of real and accurate data from activities in Activity Set 2, are partially re-processed, as necessary, intermittently until all activities in the DSM of Figure 6.6 have been processed.

On the other hand, smaller iterative-blocks like Iterative-Blocks 1-5 are processed in parallel and therefore the activities in each iterative-block are modelled as parallel activities which are processed concurrently, all beginning and ending at the same times. During this processing period, the activities in each block create rework for one another such that they are almost continually re-iterated until no further rework is required or until time runs out, whichever occurs first.

Step 4: Convert the data-dependencies into precedence relationships

Precedence relationships between activities ultimately determine the earliest time that activities can be scheduled to start. Therefore, prior to the derivation of a time-based plan and schedule, the data-dependencies, represented by the DSM, needs to be converted into activity precedence relationships. In this respect, the DSM, where activities are linked by data-dependencies, needs to be converted into an activity network diagram, where activities are linked by precedence relationships.

A number of guidelines, for the conversion of data-dependencies into precedence relationships are proposed in the text that follows. There are two major groupings of data-

dependencies to be considered; (1) data-dependencies above the leading-diagonal of the DSM; and, (2) data-dependencies below the leading-diagonal of the DSM.

Furthermore, the data-dependencies below the leading diagonal can be considered according to 3 sub-groupings; (*i*) data-dependencies between *activity sets*; (*ii*) data-dependencies within *parallel iterative-blocks*; and, (*iii*) all other data-dependencies. As an example, the groupings of data-dependencies for the warship case study are illustrated in Figure 6.8.

(1) Data-dependencies above the leading-diagonal of the DSM

The data-dependencies above the leading-diagonal represent guesstimates of the real and accurate data that will be derived through the processing of downstream activities. Therefore, there is no need to convert data-dependencies above the leading-diagonal into precedence relationships.

(2) Data-dependencies below the leading-diagonal of the DSM

It is the data-dependencies below the leading diagonal that imply precedence relationships.

(*i*) Data-dependencies between activity sets

Activities relating to the design-development of the product's overall architecture and it's principal functional/operational attributes (e.g. Activities 1-12 in Activity Set 1 of Figure 6.6) are initially processed to create an outline product solution schema. Following the completion of Activity Set 1, and before the commencement of activities relating to the design-development of a schema for each sub-system (e.g. Activities 13-51 in Activity Set 2 of Figure 6.6) a *concept design review* is undertaken.

The purpose of this review is to (*i*) determine the viability of the proposed outline product solution schema; (*ii*) identify the most influential technical uncertainties; and, (*iii*) develop preliminary material technical specifications (MTS) to be sent to specialist equipment suppliers. Following the review, providing an acceptable level of convergence has been reached, the PDO can begin processing the activities in Activity Set 2 of Figure 6.6.

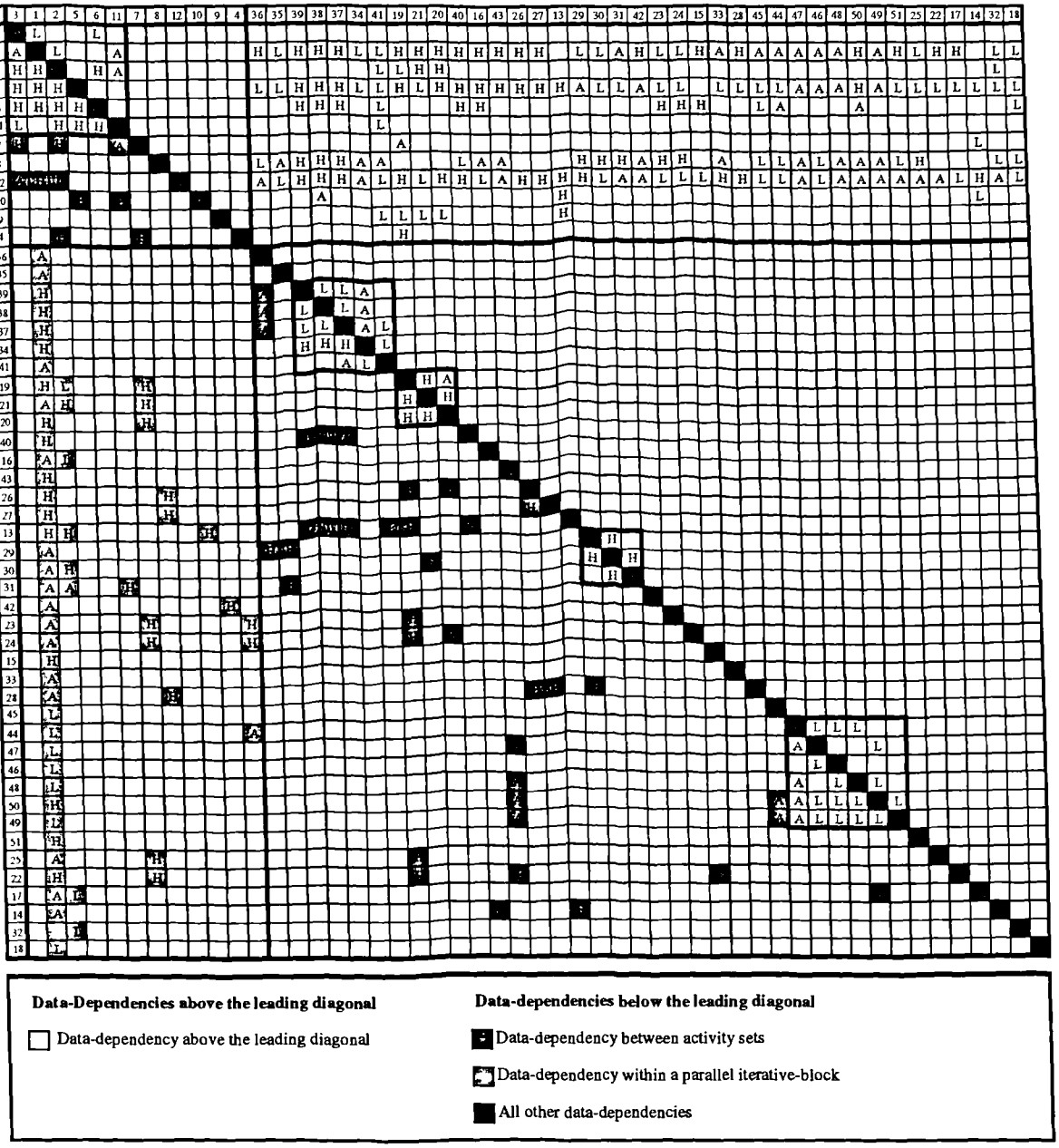


Figure 6.8: Groupings of data-dependencies

When creating an activity network diagram, the data-dependencies, which represent the downstream passage of data from activities in Activity Set 1 to activities in Activity Set 2, are all replaced by a precedence relationship which delays the latter set of activities for a period of time. This period relates to the time taken to develop an outline product solution schema and undertake a concept design review. For the warship case study, this period of time amounts to approximately 2 weeks.

(*ii*) Data-dependencies within parallel iterative-blocks

As indicated previously in Step 3, activities in a parallel iterative-block are processed concurrently and begin at the same time. Therefore, when creating an activity network

diagram, the below-diagonal data-dependencies within each parallel iterative-block are replaced by *a start-start precedence relationship* between activities of the same block.

(*iii*) All other data-dependencies

A data-dependency that links two design-development activities below the leading-diagonal does not necessarily imply a finish-to-start precedence relationship like that illustrated in Figure 6.9.

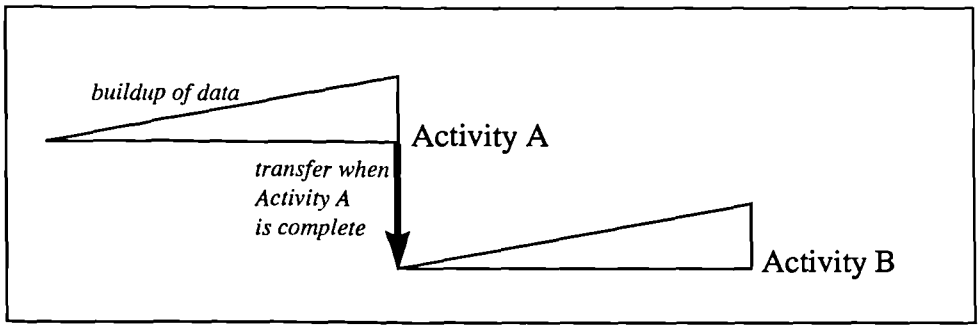


Figure 6.9: A finish-to-start precedence relationship

In Figure 6.10, two activities, which are linked by a data-dependency, are overlapped. By overlapping activities, the data-dependency between Activity A and B is not removed, but is *decoupled* such that it does not necessarily imply a finish-to-start precedence relationship. Because Activity A is still incomplete when Activity B begins, the data available to Activity B is, by necessity, incomplete. However, it is considered that there is enough data to commence Activity B prematurely.

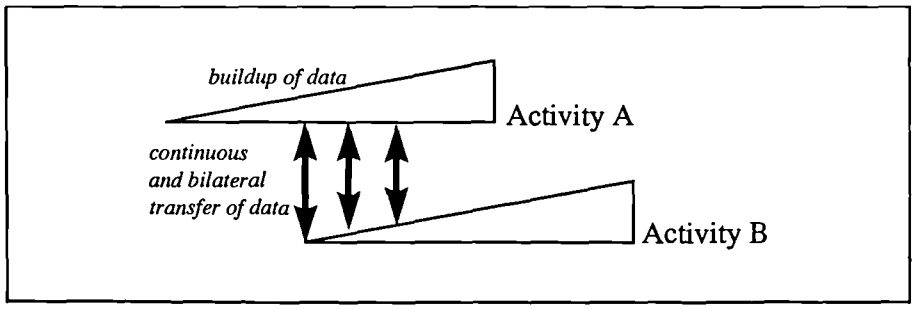


Figure 6.10: Decoupling a data-dependency

Because the data passed from Activity A to Activity B is incomplete, effective and efficient integration and bilateral communication between members of the product development organisation (PDO) responsible for both activities, is imperative. In Sub-

Section 2.3.1 a number of tools and techniques which facilitate and encourage effective and efficient communication between members of a PDO processing concurrent activities are summarised.

Several researchers have investigated the decoupling of data-dependencies in order to facilitate the overlapping of design-development activities. Takeuchi and Nonaka [Takeuchi 86] liken design-development to a game of rugby in which a multi-disciplinary team passes a rugby ball between themselves, altogether moving forward as a single unit. This, it is claimed, describes an overlapped, *concurrent engineering* approach, which is significantly different to the traditional finish-start sequential approach, which they liken to a relay race where individuals pass a baton to each other in succession.

Clark and Fujimoto [Clark 91], based on their extensive research into the world automotive industry, show, quite categorically, that faster design-development is achieved by overlapping activities. In agreement with the foregoing text, they recommend frequent face-to-face, bilateral communication of data. Smith and Reinertsen [Smith 91] recognise the merits, as well as the costs of overlapping activities and cite such an approach as one of the keys to "developing products in half the time".

Whilst Takeuchi and Nonaka, Clark and Fujimoto, and Smith and Reinertsen all extol the virtues of decoupling data-dependencies, and describe initiatives which facilitate the overlapping of design-development activities, they do not present any clearly defined strategies to help PDOs make decisions about which data-dependencies should be decoupled and, hence, which activities should be overlapped.

Krishnan *et al* [Krishnan 94] introduce such a strategy and present a model-based framework to manage the overlapping of design-development activities. However, it is speculated here that Krishnan's strategy, which relies on a large amount of data collection, tends to over-complicate what is essentially a simple decision. The decision should focus on one question, which needs to be asked of each data-dependency modelled within the DSM; "does this data-dependency really imply a finish-to-start precedence relationship?"

Based on the design structure matrix (DSM), the PDO not only has a clear knowledge of each activity's data-dependencies, but also has knowledge of the strength of each dependency. Therefore, based on the four-level dependency scheme, first introduced in Appendix II, a "High" data-dependency in a matrix position corresponding to Activity A's column and Activity B's row indicates that it is essential that data from Activity A is available to Activity B before its commencement. An "Average" dependency indicates that it is not essential that data is available before the commencement of Activity B, but it would be preferable, and a "Low" dependency indicates that it is simply not essential that data is available before the commencement of Activity B.

Based on such an understanding of data-dependencies, any data-dependencies not yet converted, can be converted into precedence relationships as follows; a "High" data-dependency implies a finish-to-start precedence relationships, whilst an "Average" and "Low" data-dependency implies that the successor activity is started when the predecessor activity has been processed for a period of time corresponding to half it's duration. This is an issue for discussion in the context of the application.

In this respect, the matrix model once again acts as a *decision-support system* and can help the PDO to develop *strategies for overlapping activities* in a logical and structured manner. Overlapping strategies created in such a way help to ensure that the benefits of reduced design-development lead-time are achieved without the cost of rework which can result from badly planned overlapping strategies.

It should be noted that, whilst all of the data-dependencies represented in the DSM are not directly convertible to precedence relationships, nevertheless they still offer significant insight into the data transfers which form an integral part of the design-development of a product. In this respect, whilst the resulting activity network diagram, derived as output from this process, does not explicitly represent iteration, it has in fact been created with knowledge of such. Based on the information summarised in the preceding text, an activity network diagram of activities for the warship case study is illustrated in Figure 6.11.

6.4.3 A Critical Path Analysis of the Warship Case Study

In Chapter 7, a resource-constrained scheduling procedure is introduced. This procedure can be used to derive schedules by optimally matching resource requirements to resource

availability. *Critical path analysis (CPA)* assumes that resource availability is infinite and therefore, in today's product development organisations (PDOs) where resources are scarce, the simple time-based analysis offered by CPA yields results that tend not to reflect reality. However, solely for the sake of completeness, the following text, based on the activity network diagram of Figure 6.11, includes a critical path analysis of the activities associated with the warship case study.

The time taken to design-develop the pre-contract design-definition of a warship is approximately 3 months. Based on the activity network diagram of Figure 6.11, the *critical path analysis* of activities for the warship case study is detailed, alongside a Gantt chart of activities, in Figure 6.12. The analysis assumes that Activities 1-12 are processed full-time during the initial 2 weeks and then intermittently over the remaining time, and that the estimated durations of activities 13-51 are as presented in Table 6.1.

No	Sub-system schema;	Dur. (wks)	No	Sub-system schema;	Dur. (wks)	No	Sub-system schema;	Dur. (wks)
36	surveillance radar	3	26	electrical gen.	3	44	fresh water	2
35	gyro	1	27	electrical distbn.	3	47	galley	2
39	SAM	3	13	hull structure	3	46	refrigeration	2
38	guns	3	29	navigation	2	48	laundry	2
37	SSM	3	30	steering	2	50	accommodation	2
34	fire-control	3	31	motion control	2	49	hospital	2
41	sonar	3	42	degaussing	2	51	fire det. & control	2
19	prime-mover	3	23	fuel oil	2	25	starters	2
21	propulsor	3	24	lub.oil	2	22	uptks & downtks	2
20	transmission	3	15	NBCD & dmg cont.	2	17	boats, dav.& hand.	2
40	helicopter	2	33	communication	2	14	paint/ cath. prot.	2
16	ballast	2	28	electrical conv.	2	32	mooring	2
43	HVAC	3	45	sewage	2	18	bilge, salv. & sull.	2

Table 6.1: Durations (in weeks) for the 39 activities in Activity Set 2 that relate to the design-development of sub-system schema

Activity Nos. 13-51 should be processed as early as possible, beginning any time between their *early start* and *late start* times. Whilst the actual timing of design-development activities is determined ultimately by their precedence relationships and the availability of PDO members, the process of deriving a near optimal sequence of activities to create a design structure matrix (DSM) is invaluable since the DSM ultimately represents a *priority sequence* according to which activities should be considered for processing. Therefore, according to each activity's resource requirements and the availability of resources, activities should be considered for processing in the order of priority as they are listed in Figure 6.12.

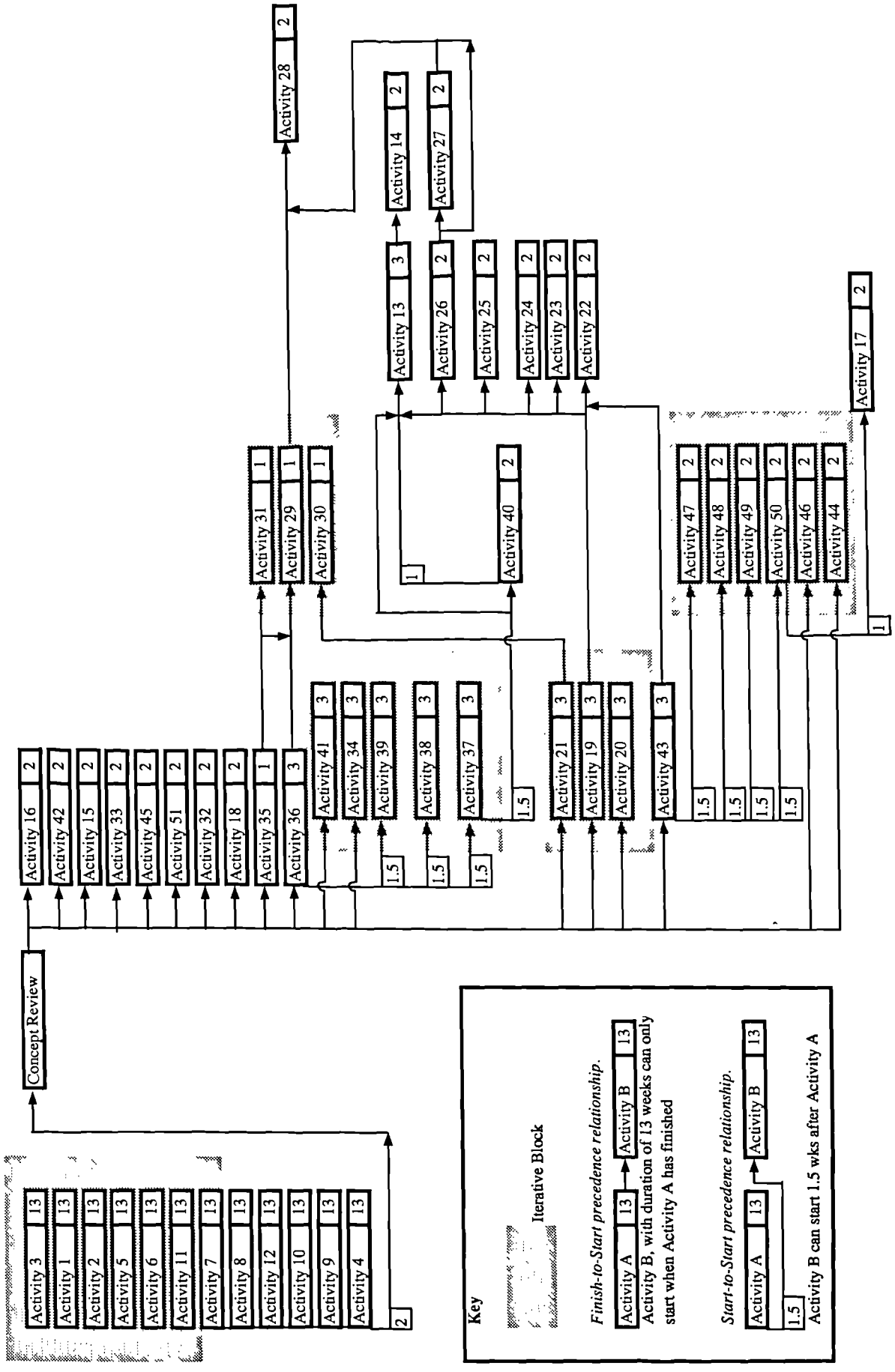
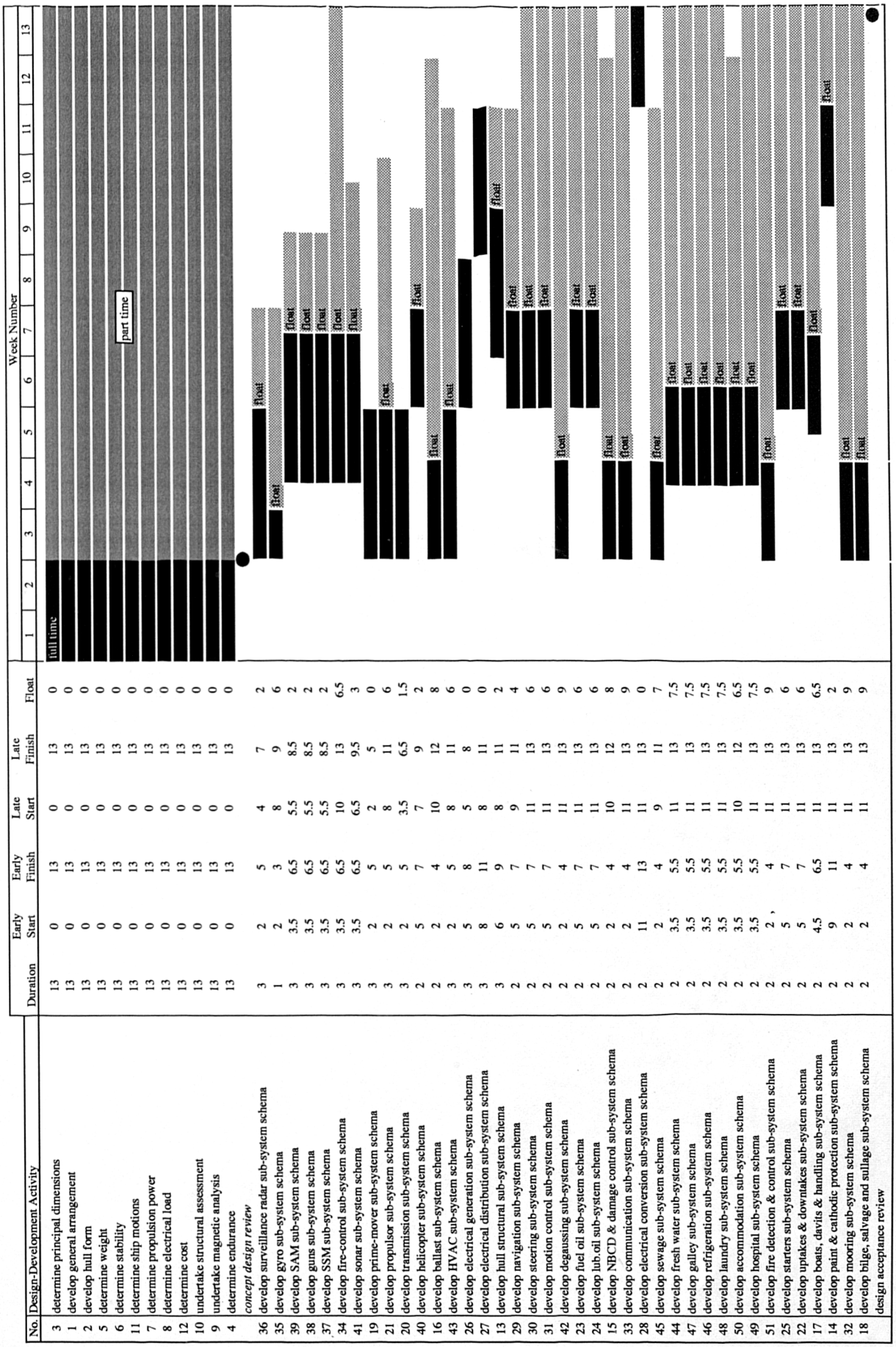


Figure 6.11: An activity network diagram for the warship case study.



No.	Design-Development Activity	Duration	Early Start	Early Finish	Late Start	Late Finish	Float
3	determine principal dimensions	13	0	13	0	13	0
1	develop general arrangement	13	0	13	0	13	0
2	develop hull form	13	0	13	0	13	0
5	determine weight	13	0	13	0	13	0
6	determine stability	13	0	13	0	13	0
11	determine ship motions	13	0	13	0	13	0
7	determine propulsion power	13	0	13	0	13	0
8	determine electrical load	13	0	13	0	13	0
12	determine cost	13	0	13	0	13	0
10	undertake structural assessment	13	0	13	0	13	0
9	undertake magnetic analysis	13	0	13	0	13	0
4	determine endurance	13	0	13	0	13	0
	concept design review						
36	develop surveillance radar sub-system schema	3	2	5	4	7	2
35	develop gyro sub-system schema	1	2	3	8	9	6
39	develop SAM sub-system schema	3	3.5	6.5	5.5	8.5	2
38	develop guns sub-system schema	3	3.5	6.5	5.5	8.5	2
37	develop SSM sub-system schema	3	3.5	6.5	5.5	8.5	2
34	develop fire-control sub-system schema	3	3.5	6.5	10	13	6.5
41	develop sonar sub-system schema	3	3.5	6.5	6.5	9.5	3
19	develop prime-mover sub-system schema	3	2	5	2	5	0
21	develop propulsor sub-system schema	3	2	5	8	11	6
20	develop transmission sub-system schema	3	2	5	3.5	6.5	1.5
40	develop helicopter sub-system schema	2	5	7	7	9	2
16	develop ballast sub-system schema	2	2	4	10	12	8
43	develop HVAC sub-system schema	3	2	5	8	11	6
26	develop electrical generation sub-system schema	3	5	8	5	8	0
27	develop electrical distribution sub-system schema	3	8	11	8	11	0
13	develop hull structural sub-system schema	3	6	9	8	11	2
29	develop navigation sub-system schema	2	5	7	9	11	4
30	develop steering sub-system schema	2	5	7	11	13	6
31	develop motion control sub-system schema	2	5	7	11	13	6
42	develop degaussing sub-system schema	2	2	4	11	13	9
23	develop fuel oil sub-system schema	2	5	7	11	13	6
24	develop lub.oil sub-system schema	2	5	7	11	13	6
15	develop NBCD & damage control sub-system schema	2	2	4	10	12	8
33	develop communication sub-system schema	2	2	4	11	13	9
28	develop electrical conversion sub-system schema	2	11	13	11	13	0
45	develop sewage sub-system schema	2	2	4	9	11	7
44	develop fresh water sub-system schema	2	3.5	5.5	11	13	7.5
47	develop galley sub-system schema	2	3.5	5.5	11	13	7.5
46	develop refrigeration sub-system schema	2	3.5	5.5	11	13	7.5
48	develop laundry sub-system schema	2	3.5	5.5	11	13	7.5
50	develop accommodation sub-system schema	2	3.5	5.5	10	12	6.5
49	develop hospital sub-system schema	2	3.5	5.5	11	13	7.5
51	develop fire detection & control sub-system schema	2	2	4	11	13	9
25	develop starters sub-system schema	2	5	7	11	13	6
22	develop uptakes & downtakes sub-system schema	2	5	7	11	13	6
17	develop boats, davits & handling sub-system schema	2	4.5	6.5	11	13	6.5
14	develop paint & cathodic protection sub-system schema	2	9	11	11	13	2
32	develop mooring sub-system schema	2	2	4	11	13	9
18	develop bilge, salvage and sullage sub-system schema	2	2	4	11	13	9
	design acceptance review						

Figure 6.12: A critical path analysis and Gantt chart representation for the warship case study

6.5 Chapter Summary

Chapter 6, using the *GA-Local Search Procedure* addresses the *ActRes Problem* of deriving a near optimal sequence of activities based on the newly considered objective of *maximising concurrency*. Whilst the minimisation of iteration is ever important, Chapter 6 demonstrates that, under the appropriate conditions, the maximisation of concurrency is also desirable since it implies being able to reduce lead-times.

Consequently, Chapter 6 demonstrates that, supported by a product development organisation where the reuse of design-data is facilitated, the best approach to solving the *ActRes Problem* of deriving a near optimal sequence of activities is to use a *multiple criteria approach* based on iteration and concurrency. Such an approach helps to ensure faster design-development whilst ensuring that the added cost due to iteration is still as small as possible.

Chapter 6 also brings together the research detailed in Chapters 4, 5 and the first half of Chapter 6 to demonstrate, using the warship case study, the third and fourth strategic functions of the proposed *strategy for modelling the design-development phase of a product.*

Finally, the current research documented in Chapter 6 contributes to the subject in a number of areas. Of course, whilst the objective of maximising concurrency is not new, its utilisation in the context of deriving a near optimal sequence of activities based on a DSM is original. Furthermore, the approach to deriving sequences based on the multiple objectives of maximising concurrency and minimising iteration is original. In total, the new objective functions for measuring iteration, concurrency (both separately and together), optimised using the *GA-Local Search Procedure*, contribute to the subject of deriving near optimal sequences of design-development activities using the DSM system.

In addition, the development of the *iterative block resolution (IBR) procedure* represents a significant contribution to the subject of deriving a network diagram of design-development activities as a prelude to deriving a schedule. Whilst the contribution of the IBR procedure at the strategic level has been highlighted previously in Section 3.8, the following text highlights its contribution at the functional level by describing the drawbacks of prior research.

In order to derive a schedule of activities, the precedence relationships between the activities need to be defined. However, the majority of published research on the DSM system does not

proceed beyond obtaining a re-sequenced matrix with a definition of iterative blocks of inter-dependant activities. Such analysis is useful for high-level project management; for example, the end of iterative blocks can be used to identify where design reviews are needed. However, it was clear from the beginning of the research that DSM analysis could be further enhanced if it were to be used as part of a process for developing schedules of design-development activities.

Whilst examples may be limited in number, there are a number of published research papers that addresses the DSM in terms of time and planning issues. Smith *et al* [Smith 94, 97] estimate the total elapsed time required to do an iterative block using mathematical techniques such as Markov chains and eigenvector analysis. However, in this approach the iterative block is considered as a single entity and no information can be derived on the start times of individual activities within the block.

Rogers *et al* [Rogers 96b] address accumulated time but not elapsed time. In this case, accumulated time relates to the summation of all the durations of the coupled activities in each iterative block. However, whilst accumulated time can be related to cost, this concept of time cannot be used to derive elapsed time because the precedence relationships between activities are not considered.

Steward [Steward 81a] researches the application of the DSM as an input to critical path scheduling. However, the approach is flawed by some of the oversimplified assumptions, concerning the interpretation of the DSM, that have been described previously in Section 3.8.

In summary, little published research is available that details the iterative block resolution (IBR) that is necessary in order to resolve the intricacies of iteration prior to commencing the scheduling of product development activities. Consequently, the newly developed *IBR Procedure*, described in Chapter 6, is largely original and forms a contribution by linking the DSM system to the subject of planning.

7. The Resource-Constrained Scheduling of Design-Development Activities

7.1 Introduction

Scheduling can be defined as the planning of work where the resources *required* to process the work are matched with the resources that are *available*. When resource availability is limited, the scheduling of work is constrained to periods of time during which resource requirements do not exceed availability. In this respect, *resource-constrained scheduling* decisions are based on issues relating to the *configuration*, *sequencing* and *timing* of work.

As part of the strategy for modelling the design-development phase of a product, the *configuration* of work is first addressed in Chapter 3 where the notion of creating a product design-work breakdown structure is detailed. In Chapter 5, and later on in Chapter 6, the *sequencing* of work is addressed. In particular, Chapter 5 considers the sequencing of design-development activities in order to minimise iteration whilst Chapter 6 focuses on the dual objectives of maximising concurrency and minimising iteration. Because the notion of concurrency is closely related to the relative *timing* of work, issues relating to the timing of design-development activities are also addressed in Chapter 6.

Chapter 7 now focuses on the *resource-constrained project scheduling (RCPS) problem* and brings together the research described in previous chapters in order to describe, in detail, the fifth and final function of the proposed modelling strategy, namely, "derive a resource-constrained schedule of activities". (*See* function box A5 of Figure 3.3)

As may become clear in the following section, the general RCPS problem can be viewed as a generalisation of the *machine scheduling problem*. It is for this reason that the machine scheduling problem is introduced and summarised in Section 7.2. With the scene set, Section 7.3 reviews resource-constrained project scheduling and, using a classification scheme, examines a number of solution-approaches to the problem that have been reported in the literature.

Section 7.4 begins by highlighting the fact that the majority, if not all solution-approaches to the RCPS problem are, in fact, focused on the resource-constrained scheduling of the

activities associated with a product's *production phase*. For the reasons described in Chapter 3, there is still significant resistance to the modelling and subsequent scheduling of activities associated with a product's design-development phase and, as stated previously, this resistance is due in part to the lack of solution-approaches specifically focused on design-development. With this observation in mind, Section 7.4 goes on to classify a new solution-approach to the resource-constrained scheduling of design-development activities.

Based on this solution-approach, Section 7.5 introduces a newly developed tool named the *Multiple Criteria Genetic Algorithm (MCGA) Project Scheduler*. In Section 7.6, the MCGA Project Scheduler is demonstrated using a number of applications of the tool to a problem that is based, in part, on the warship case study. Finally, Section 7.7 summarises the chapter and highlights how the research specifically documented in Chapter 7 represents a contribution.

7.2 The Machine Scheduling Problem

In the classical machine scheduling problem, *a set of jobs* is to be processed on *a set of machines* such that some managerial objective is optimised; the most popular objective being the minimisation of the elapsed time taken to complete all jobs. The two assumptions, common to all versions of the problem, are that, (*i*) a job cannot be processed by more than one machine at any instant in time; and, (*ii*) each machine is only capable of processing a single job at each instant in time.

This problem has been studied and developed extensively over the last thirty years and the surveys conducted by Graham *et al* [Graham 79], Lawler *et al* [Lawler 82], Blazewicz *et al* [Blazewicz 88], Cheng & Sin [Cheng 90] and Tzafestas & Triantafyllakis [Tzafestas 93] summarise the literature on 'state of the art' solution-approaches to the machine scheduling problem pertinent to the times in question.

Figure 7.1 illustrates a breakdown of the machine scheduling problem. The problem is first broken down into two sub-problems; (*i*) the *single-machine scheduling problem*; and, (*ii*) the *multiple-machine scheduling problem.*

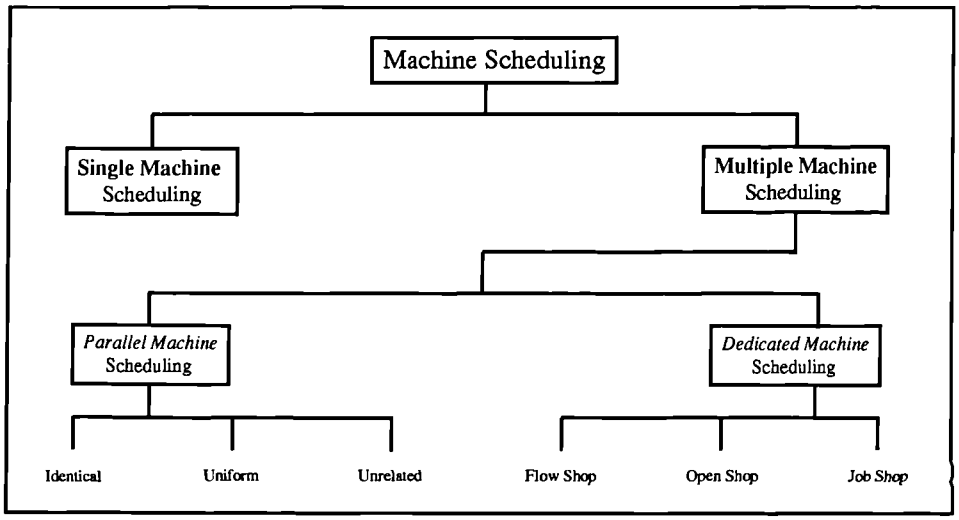


Figure 7.1: A breakdown of the machine scheduling problem.

In the basic single-machine scheduling problem, the set of machines does, in fact, only consist of a single machine. This single machine can process a set of jobs in any order such that the problem is to determine the processing sequence of jobs so as to optimise the managerial objective(s). In the more complex multiple machine scheduling problem, more than one machine is available such that the problem, now, is to determine which machines are to be allocated to which jobs and, in addition, to determine the processing sequence of jobs on each machine.

Because of it's simplicity, the single machine scheduling problem cannot be broken down further, however, the multiple-machine scheduling problem can be broken down into (*i*) the *parallel-machine scheduling problem*; and, (*ii*) the *dedicated machine scheduling problem.*

The parallel-machine scheduling problem assumes that each job is processed on a single machine only once and that all machines perform the same function such that each job can be processed on any one of the machines. Furthermore, for the parallel-machine scheduling problem, the machines are described in terms of their relative processing speeds. *Identical parallel machines* have the same fixed processing speed. *Uniform parallel machines* have different, yet fixed processing speeds. Finally, *unrelated parallel machines* have processing speeds which are not fixed and which differ according to job.

Conversely, the dedicated-machine scheduling problem assumes that each machine is specialised only for the processing of certain jobs such that each machine is only capable of

processing a sub-set of the total set of jobs. In this case, each job is processed a number of times by more than one machine and, consequently, the sub-problem is classified further according to how jobs are routed between machines. In a *flow shop*, each job must be processed by all machines in the same pre-defined sequence. In an *open shop*, each job must be processed by all machines in any order. Finally, in a *job-shop*, each job must be processed by a pre-defined sub-set of machines. In the job-shop, the sequence of processing for each job is arbitrary and, whilst it may be different for each job, it must be pre-defined.

By definition, all versions of the dedicated machine scheduling problem are based on the decomposition of each job into a set of sequential stages. For the basic flow shop and open shop versions, each job consists of m stages, where m equals the total number of machines. In the basic job-shop version, each job consists of n stages, where n can vary between 1 and m.

Based on an understanding of the machine scheduling problem, the general resource-constrained project scheduling (RCPS) problem can now be introduced. The RCPS problem is defined in the following section but first, as an introduction, a comparison of the job-shop scheduling problem and the RCPS problem is summarised in Table 7.1.

It may be observed from Table 7.1 that the RCPS problem can be viewed as a modified version of the job-shop scheduling problem. Here, *jobs* (now referred to as *activities*) can be processed by more than one *machine* (now referred to as *project resources*), and may be linked by *precedence relationships*. A precedence relationship between two activities implies that the *preceding activity* must at least begin prior to the commencement of the *succeeding activity*.

The majority of solution-approaches to the various machine scheduling sub-problems assume that all jobs are ready for processing at time zero and, as such consider the *static case*. Conversely, the more complex *dynamic case* assumes that not all jobs are ready for processing at time zero. In manufacturing, the static case does not always apply, hence problems which are formulated using such assumptions can sometimes derive infeasible solutions. However, for any type of project scheduling, the static case is always valid since, at the beginning of a project, or at time zero, all activities are theoretically ready for processing.

The Job-Shop Scheduling Problem	The RCPS Problem
The problem is to schedule a set of manufacturing jobs onto a limited set of machines such that some managerial objective is optimised.	The problem is to schedule a set of project activities using a limited amount of project resource, including labour, machines & equipment, such that some managerial objective is optimised
A manufacturing job is processed by a machine.	A project activity is processed through the application of project resources.
A job can only be processed by one machine at a time.	An activity can be processed by more than one project resource at a time.
A machine can only process one job at a time	A project resource can only process one project activity at a time
Each job is unrelated, such that there are no precedence relationships between jobs.	Each activity may be related to others, such that precedence relationships exist between activities.
A job consists of a number of stages that are processed by different machines in a pre-defined sequence.	An activity consists of a number of stages that are processed by different project resources, either sequentially or in parallel, according to a pre-defined pattern of resource allocation.

Table 7.1: Comparing the job-shop scheduling and RCPS problems

7.3 Resource-Constrained Project Scheduling

The development of PERT and CPM in the late 1950s meant that, for the first time, a formal model of any life-cycle phase of an engineered product, comprising a network of activities linked by precedence relationships, could be used to plan and schedule work.

However, the modelling and subsequent scheduling of activities using such techniques, focuses solely on the *time aspect* and no consideration is given to *resource constraints* which may include limits on the availability of equipment, labour and cash, as well as time. In many real-life situations, delays to the start of an activity can occur if the resources required by the activity in question are not available in sufficient quantities during the time interval the activity was originally scheduled to be processed.

As demonstrated by Just and Murphy [Just 94], the difference between project-schedules that are resource-constrained, and those that are not, can be significant. Unanticipated delays and the resulting overtime and cost overruns are all symptomatic of project scheduling undertaken without adequate consideration for resources and their limits.

Project scheduling that considers resource constraints, including constraints on time, is commonly referred to as *resource-constrained project scheduling (RCPS)* and the general *RCPS problem* can be stated as follows. Given a set of activities, linked by precedence relationships, where each activity can be performed in one of several *operating modes*, and each operating mode is characterised by a pre-defined duration and resource requirement, when should each activity be scheduled to begin, and in which mode should the activity be processed such that some managerial objective is optimised.

Since the pioneering work of Kelly [Kelly 63] and Wiest [Wiest 63], the RCPS problem has occupied many researchers. The surveys conducted by Davis [Davis 73], Elmaghraby [Elmaghraby 77], Willis [Willis 82] and Icmeli *et al* [Icmeli 93] summarise the literature on 'state of the art' solution-approaches to the RCPS problem pertinent to their time. More recently, Ozdamar & Ulusoy [Ozdamar 95] survey fifty-seven solution-approaches to the RCPS problem.

Figure 7.2 illustrates a scheme by which solution-approaches to the RCPS problem can be classified. In this respect, each solution-approach addresses the RCPS problem in terms of it's approach to (*i*) *search strategy*; (*ii*) *search objective*; (*iii*) *resource allocation*; (*iv*) *type of resource*; (*v*) *pre-emption condition*; and, (*vi*) *number of projects*. It should be noted that, whilst solution-approaches to the RCPS problem add a new dimension to project scheduling, like CPM and PERT, they are still based on a model of the project in terms of an activity network diagram of project activities linked by precedence relationships.

The following sub-sections explain the varied notions and concepts behind each of the six classifications used to describe solution-approaches to the RCPS problem and highlight some of the most relevant research to be found in the literature.

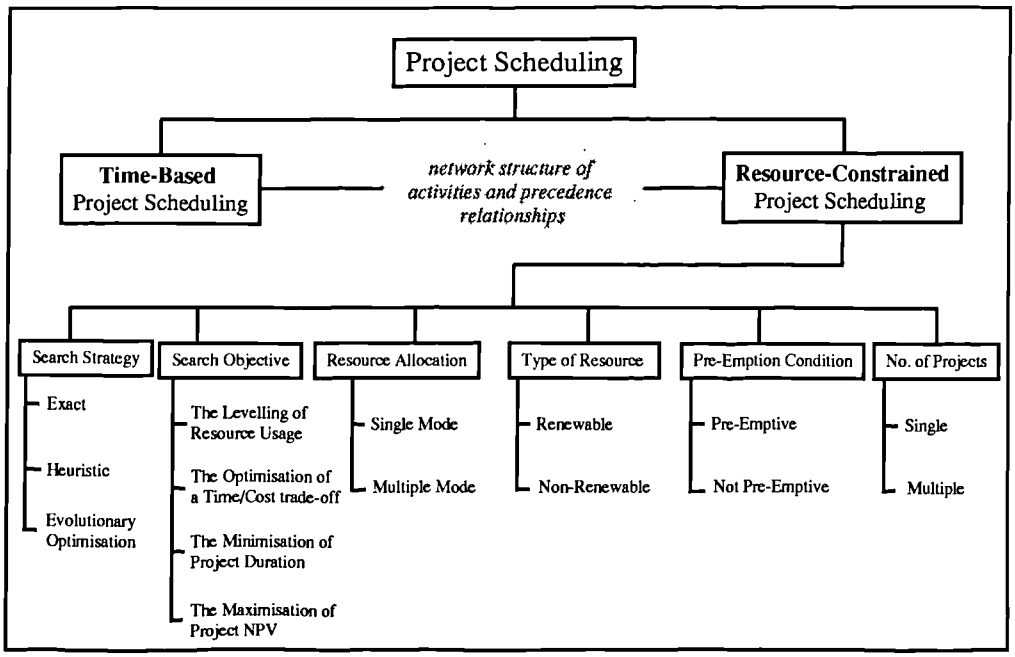


Figure 7.2: A scheme by which solution-approaches to the RCPS problem can be classified

7.3.1 Search Strategy

Using the results of Blazewicz [Blazewicz 78], it has been shown that the RCPS problem is *NP-complete*. This means that *exact mathematical approaches*, based on dynamic programming and branch and bound search techniques, that deterministically and exhaustively search for the global optimum solution, will probably fail because of the prohibitively large amount of time that would be required.

Exact mathematical search strategies

There have been a number of attempts to find global optimum solutions through the development of exact mathematical approaches which, due to NP-completeness, tend to employ *branch and bound* search techniques. Branch and bound search techniques, based on an integer programming formulation of the RCPS problem [Pritsker 81, Brand 64] are reported by Bell & Park [Bell 90], Christofides *et al* [Christofides 87] and Demeulemeester & Herroelen [Demeulemeester 92].

The basic branch and bound procedure builds a feasible schedule by trying, at each decision point, to put in process, all unscheduled activities that satisfy precedence relationships. In this manner, the procedure builds a complete schedule by successively adding unscheduled activities to a partial schedule. The only decision points considered are those points in time at which one or more activities finish.

At each decision point, only the unscheduled activities with predecessors that have finished are considered as candidates for inclusion into the partial schedule. The candidate activities are plugged into the partial schedule, to test for resource overload, in an order determined by a priority list which ranks candidate activities in a descending ranking of $L(\alpha)$. This parameter is commonly defined as the length of the critical path from "*activity* α" to the end of the project.

These unscheduled activities can be scheduled to begin at the current decision point if there is no resource overload. That is, an unscheduled *candidate activity* can be scheduled to begin at this time if it's resource requirements do not exceed the available resources left by the activities already scheduled. If a resource overload does occur at the decision point, then the overload must be resolved. There are two ways of achieving the resolution; either (*i*) delay the candidate activity; or, (*ii*) delay one or more activities that have already been scheduled. The set of activities that could be delayed is referred to as an *alternative set.* In this way, each candidate activity has an alternative set of activities containing those activities, which in various combinations, if delayed, would enable the scheduling of the candidate.

Activities are delayed by adding notional precedence relationships between the appropriate activities. Whenever resource overload is encountered, a branching of the search tree begins. Each branch of the tree represents a local scheduling of the candidate and it's alternatives, such that the resource overload encountered at the current decision point is resolved. Ultimately, at each decision point there will be a number of possible combinations such that each possible combination (branch) relates to a local scheduling of the candidate and it's alternatives.

In order to determine whether a particular branching will yield a good solution, a lower bound is applied to each branch. A simple lower bound would be determined as follows. If the branch indicates that "*activity* β" should be delayed until the completion of "*activity* α", then, $LB = f_\alpha + L(\beta)$. In words, LB is equal to the finishing time of "*activity* α" plus the length of the longest path from "*activity* β" to the end of the project. For each branch, the lower bound is calculated for all added precedence relationships, and the largest (worst) value is stored as LB*.

If the LB* of a particular branch is less than or equal to the previously stored best LB* (*incumbent*), then the current LB* becomes the incumbent and the branch is further searched to develop the partial schedule at the next decision point. Conversely, if the LB* of a particular branch is greater than the incumbent, then no further search down this particular branch is undertaken. In such a case the branch has been fathomed and is removed from further consideration.

A branch reaches conclusion when all activities have been scheduled, or it has been fathomed. When a branch reaches conclusion, the search backtracks up the search tree to the next level. The added notional precedence relationships corresponding to the last alternative are removed and new ones that represent the next alternative are added. If no further alternatives exist at a given level, the search further backtracks up the search tree to the next level. The process is completed when the search returns to level zero.

Heuristic search strategies

Heuristic search strategies, first developed for the RCPS problem by Kelly [Kelly 63] and Weist [Wiest 64, Wiest 65], begin by ranking activities according to an assigned *priority value*, derived for each activity by invoking a *heuristic*, more commonly referred to as a *priority rule*.

Priority values are assigned in order to resolve *resource conflicts*. A resource conflict occurs at a point in time when, precedence relationships satisfied, two or more candidate activities may begin but there are insufficient resources to allow all of the candidate activities to begin. By scheduling an activity with a higher priority value in preference to other activities that may be competing for the same resource, potential conflict is resolved.

The basic heuristic scheduling procedure builds a complete schedule by successively adding unscheduled activities to a partial schedule. The procedure begins with a dummy start activity of zero duration. Each time an activity finishes (*decision point*) those unscheduled activities that satisfy precedence relationships are considered as candidates for inclusion into the partial schedule. Starting at the top of a list, ranked according to priority value, candidate activities are scheduled in turn until the available resources are depleted. The procedure then updates the resource levels and the set of activities in progress, and the decision point is advanced to

the earliest completion time among the activities in progress. The procedure continues until all activities have been scheduled.

There exists today a large number of heuristic scheduling procedures and priority rules. The specific rules incorporated into commercial computer packages remain closely guarded secrets, however most have been discussed in the literature. Generally, priority rules prioritise activities based on activity parameter-values derived from a critical path analysis of an activity network. For example, activities may be prioritised according to their available float, late finish time, or a combination of other values derived from critical path analysis. Other examples include priority given to the most 'resource greedy' activities, and priority given to activities chosen at random. Where competing activities share the same priority value, multiple priority rules can be applied.

Priority rules are defined as either *serial* or *parallel.* Using serial priority rules, the priority values assigned to activities remain fixed throughout the scheduling procedure. Using parallel priority rules, the priority values assigned to activities are updated each time an activity is scheduled.

o

The performance of any one priority rule is strongly affected by the topology of the activity network as well as the number and tightness of resource and other project constraints. As a result, a priority rule has yet to be found which performs consistently better than other rules. Davis & Patterson [Davis 75] and Alvarez-Valdes & Tamrit [Alvarez-Valdez 89] report research on the performance evaluation of different priority rules.

One way of coping with the performance variability of individual priority rules is to develop procedures that incorporate several rules. The rules are invoked, in turn, in order to develop a range of schedules. The best schedule, amongst those derived by each priority rule, is then selected as the final solution. Boctor [Boctor 90] suggests such *a composite priority rule approach* and shows that this approach derives superior solutions to those derived by procedures based simply on an individual priority rule.

Another alternative approach is to use priority rules iteratively, so as to use the resulting schedule derived using a priority rule as the input to a second procedure which, again using priority rules, improves further on the original solution. Bell & Han [Bell 91] and Li & Willis

[Li 92] have developed so-called *multi-pass heuristic procedures* that employ iterative improvement procedures and show that this approach also derives superior solutions to those derived by the previously described basic single-pass heuristic procedure.

Evolutionary optimisation search strategies

Recent improvements in computer hardware have encouraged the development *of evolutionary optimisation techniques.* The most popular of these techniques are *simulated annealing, tabu search* and *genetic algorithms.* These techniques, summarised in Chapter 4, using a measurement of the goodness of a schedule (objective function) and a degree of randomness to guide a highly explorative search, continually develop new priority lists which are used to create new schedules which are then evaluated in turn. These optimisation-based search strategies continue searching for a schedule that is better than the previously stored best schedule until some stop condition is met.

Whilst numerous examples documenting the use of *simulated annealing* to solve a range of optimisation problems exist, perhaps with the exception of Boctor's work [Boctor 96], few references are to be found on the application of such techniques to the RCPS problem. Applications of *tabu search* to the RCPS problem are reported by Nabrzyski & Werglarz [Nabrzyski 96], Pinson *et al* [Pinson 94] and Lee & Kim [Lee 96]. Research into the application of *genetic algorithms* to the RCPS problem is gaining popularity, and applications are reported by Nakano & Yamada [Nakano 80], Bruns [Bruns 85], Bagchi *et al* [Bagchi 91], Davis [Davis 91], Kanet & Sridharan [Kanet 91], Syswerda & Palmucci [Syswerda 91b], Lee & Kim [Lee 96] and Ramat *et al* [Ramat 97].

7.3.2 Search Objective

The first complete survey of solution-approaches to the RCPS problem, undertaken by Davis [Davis 73], considered three versions of the problem. The first version focused on the *levelling of resource usage* as an objective. The second considered the problem as a *trade-off between time and cost* whilst the third version considered the *general RCPS problem* which focused on scheduling under fixed resource limits. The scheme illustrated previously in Figure 7.2 covers the general RCPS problem as Davis saw it, whilst the versions focusing on the levelling of resource usage and the trade-off between time and cost are covered by Figure 7.2's scheme as specific objectives to be optimised.

Based on a critical path analysis of an activity network diagram, the concept behind *the levelling of resource usage* as an objective, is to reschedule non-critical activities within their available float in order to achieve a better distribution of resource usage. Solution-approaches that address the levelling of resource usage tend to be based on integer linear programming procedures [Ahuja 76, Easa 89] and on the use of heuristics [Shaffer 65].

The *optimisation of a time/cost trade-off* involves the reduction of certain activity durations through the allocation of additional resources. This tends to reduce project durations but results in higher project costs. The problem then, is to find the optimal trade-off between the reduced time and the increased cost. The objective, here, is to determine which activities should be modified in duration by applying additional resource such that an acceptable trade-off between project cost and project duration is derived. A number of solution-approaches that aim to optimise the time/cost trade-off are to be found in the literature and include Siemens [Siemens 71], Philips & Dessouky [Philips 77] and Tufekci [Tufekci 82].

Solution-approaches to the RCPS problem, based on the two previously described objectives, do consider resources and, therefore, are constrained by resource considerations. However, because such solution-approaches allow the manipulation and the application of additional resources, they are not *constrained by the limits* of resource availability. In today's business climate, cost-cutting initiatives have often resulted in resource cutbacks and, as such, project resources are often scarce. In this respect, more recent solution-approaches to the RCPS problem are based on the optimisation of managerial objectives subject to a pre-defined limit of resource availability.

The majority of such research has focused on the optimisation of time-based objectives where the objective is to schedule activities, under resource-constraints, in such a way that the elapsed time required to complete all activities is minimised. [Christofides 87, Bell 90, 91, Demeulemeester 92, Li 92]. Sometimes, schedules optimised purely on a time basis are inefficient in terms of cost. In many cases, the inflows and outflows of cash to a project can be quantified such that the objective is to schedule activities in such a way that the *net present value (NPV) of the project's cash flow is maximised* [Russell 70, Doersch 77, Patterson 90].

Most prior research has addressed a single objective, whether it is time- or cost-based. More recently, multiple objective approaches have begun to emerge [Slowinski 81, Daniels 89,

Dean 92, Speranza 93]. However, examples of such research are still scarce; of the fifty-seven approaches surveyed by Ozdamar & Ulusoy [Ozdamar 95], only five adopt a multiple-objective approach.

7.3.3 Resource Allocation

There are two cases of resource allocation commonly addressed by solution-approaches to the RCPS problem. In the first case, referred to the *single mode* case, each activity has a single *operating mode* such that each activity can only be processed by one pre-defined group of resources. In the second case, referred to as the *multiple mode* case, each activity has more than one operating mode such that each activity can be processed by any one of a pre-defined group of resources.

Ultimately, the incorporation of multiple mode resource-allocation increases the size and complexity of the problem since the solution-approach now has the flexibility of allocating an operating mode, as well as a starting time, to each activity.

7.3.4 Type of Resource

Two types of resource are described in the literature. *Renewable resources* are constrained on a time-period basis such that the resource is limited over a given time-period. However, at the end of the time period the whole of the resource is once again available. Human resources can be viewed as renewable resources. On the other-hand, *non-renewable resources* are constrained on a project basis such that the resource is limited over the whole duration of the project. As the project proceeds, the resource is gradually used up. Raw material and cash can be viewed as non-renewable resources.

7.3.5 Pre-Emption Condition

A schedule is pre-emptive under the condition that activities, once started, can be interrupted. An activity is deemed to be pre-empted when it is stopped and it's previously allocated resources are re-assigned. Once the re-assigned resources become free again, the pre-empted activity can be resumed. Conversely, a schedule is non pre-emptive under the condition that activities, once started, cannot be interrupted.

7.3.6 Number of Projects

The majority of reported solution-approaches to the RCPS problem focus on single projects. A number of solution-approaches based on multiple projects are to be found in the literature and include Speranza & Vercellis [Speranza 93], Slowinski [Slowinski 81] and Tsubakitani & Deckro [Tsubakitani 90].

7.3.7 An Example Formulation of the General RCPS Problem

The following example has been included to illustrate how the RCPS problem is usually formulated. The example, illustrated in Figure 7.3, is based on a single project that consists of 10 activities, including two dummy activities (The start and finish activities).

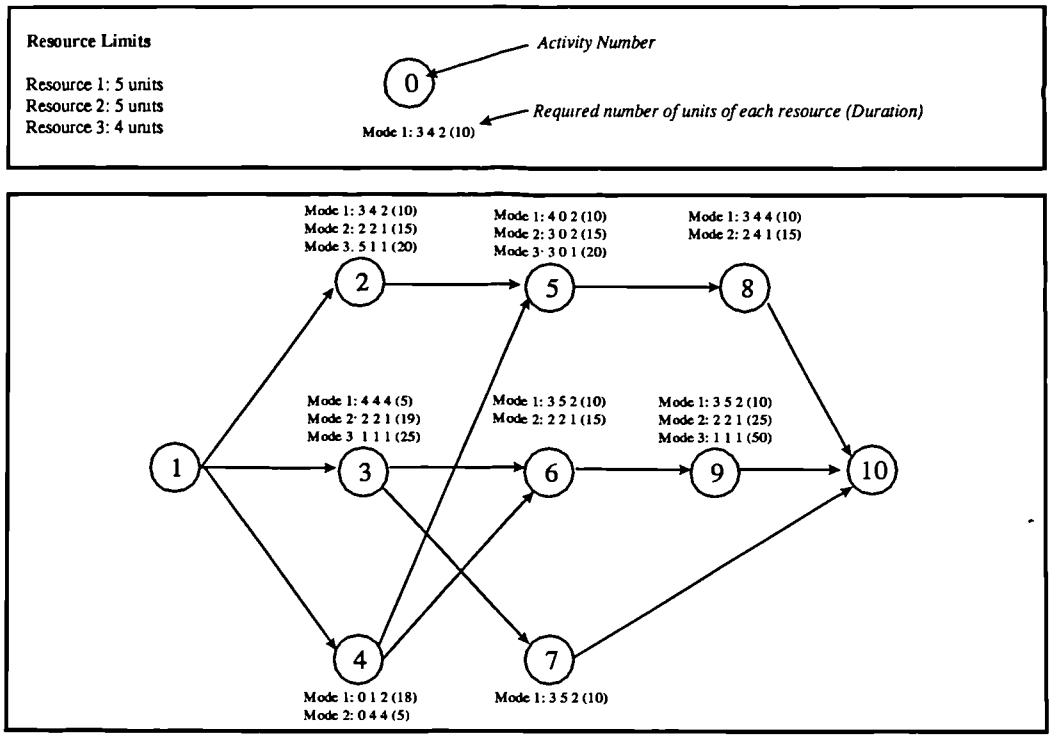


Figure 7.3: The general multiple-mode formulation of the RCPS problem

Each activity can be processed in one or more alternative operating modes. Each operating mode is characterised by fixed multiple unit requirements of three different resources and corresponding activity duration. Resource availability is constant, renewable, and limited over the duration of the project. The aim then, is to determine the start-time and the operating mode of each activity such that resource limits are not exceeded and a pre-defined managerial objective is optimised. Pre-emption is not allowed.

The example illustrated represents the general multiple-mode formulation of the RCPS problem. In the single mode case, each activity would be defined using only a single operating mode and single activity duration. In this simpler case, the aim would be to determine only the start-time of each activity such that resource limits are not exceeded and a pre-defined managerial objective is optimised.

7.4 A New Solution-Approach to the Resource-Constrained Scheduling of Design-Development Activities

When addressing the subject of project scheduling, most researchers focus their solution-approaches on the production phase of a product. For the reasons described in Chapter 3, there is still significant resistance to the modelling and subsequent scheduling of activities associated with the design-development phase of a product and, as stated previously, this resistance is due, in part, to the lack of solution-approaches specifically focused on the design-development phase of a product.

Whilst it is accepted that, because of the pressures of reducing time to market, design-development activities should be integrated with production activities, different objectives and strategies, relating to the scheduling of these two types of activities, need to be adopted. Again, for the reasons first described in Chapter 3, there are a number of differences, between the production phase of a product and the design-development phase of the same product, which make prior solution-approaches to the RCPS problem unsuitable for the scheduling of design-development activities.

Using the classification scheme first introduced in Section 7.3, the following sub-sections begin to formulate a new solution-approach to the resource-constrained scheduling of design-development activities.

7.4.1 Search Strategy

By far, the most researched solution search strategies are those based on heuristics. However the performance of heuristic-based solution-approaches is problem dependent and they do not provide good solutions consistently. In fact, "...the performance of heuristics depends on problem characteristics and it is quite difficult to predict, beforehand, the most efficient heuristic for a given problem..." [Boctor 90].

Whilst it is acknowledged that recent research, based on the development of the previously mentioned *composite* and *multi-pass heuristic procedures*, has gone some way towards improving the quality of heuristic-based solutions, such procedures still cannot guarantee finding good solutions consistently. At the same time, Ozdamar & Ulusoy [Ozdamar 95] conclude that, whilst exact mathematical search strategies based on branch and bound procedures go some way towards improving the solution efficiency, they are observed to be insufficient for problems of practical size.

Lee & Kim [Lee 96] have recently developed procedures based on evolutionary optimisation techniques. Simulated annealing, tabu search, and genetic algorithms have been applied to single-mode resource constrained project scheduling problems. Results of computational tests on the performance of procedures based on each of these three search techniques appears to indicate that all three evolutionary optimisation techniques work better than existing branch and bound and heuristic-based search strategies [Lee 96].

It is true to say that when the search space is not excessively large, such as is the case for the single-mode RCPS problem, then simulated annealing is the best approach. However, for more complex resource allocation cases, where there exists a choice of resource allocation, the size and complexity of the problem increases dramatically since the scheduling procedure now has the flexibility of allocating an operating mode, as well as a starting time, to each activity. In such cases, Lee & Kim find that a genetic algorithm-based search strategy is the best approach.

As the name implies, the search strategy incorporated into the new MCGA Project Scheduler is a *multiple-criteria genetic algorithm (MCGA)*. The MCGA search mechanism is based on the work of Todd [Todd 97c] and is summarised, as part of the newly developed MCGA Project Scheduler, in Section 7.5 and detailed in Appendix IV.

7.4.2 Search Objective

Again, as the name implies, the MCGA Project Scheduler can in fact optimise on the basis of multiple criteria. Ultimately, the choice and combination of criteria depends on user preferences.

Criterion: Elapsed time required to complete all activities.

Objective: Minimise elapsed time.

As indicated in Chapter 2, many engineering companies need to get their products to market faster. In this respect, minimising elapsed time is the principle objective in many cases.

In addition to the principle objective of minimising elapsed time, a number of additional criteria have been considered. The purpose of introducing these additional criteria is twofold. In the first instance, they form the basis of *explorative research* into the relationship between *organisational learning* [Senge 90] and *scheduling issues*. In the second instance, when combined with the criterion of minimising elapsed time, they *illustrate* the multiple-criteria capabilities of the MCGA Project Scheduler.

Criterion: Utilisation of members of the product development organisation (PDO).

Objectives: Maximise the utilisation of all members.

Maximise the utilisation of the most experienced members.

Maximise the utilisation of the least experienced members.

The major resources associated with design-development activities are the skilled personnel who make up the PDO. Invariably, such resources are scarce and, as such, it is important that they are used in the most efficient and effective manner. Like all organisations, a PDO consists of personnel with differing levels of skills and experience. As may become clear in Section 7.7, by expressing the *relative skill level* of PDO members quantitatively as an index, it is possible to derive schedules based on the three objectives mentioned above.

Criterion: The relative improvement of the PDO's skill level.

Objective: Maximise the relative improvement of the PDO's skill level.

Elapsed time can be minimised by repeatedly using the most experienced PDO members. However, it is considered that a strategy for training lesser experienced members is also required such that knowledge and valuable experience is not lost by the PDO when it's most experience members leave or retire.

Again, as may become clear in Section 7.7, by expressing the *relative skill level* of PDO members quantitatively as an index, and by introducing a function that updates this index as and when members are assigned to activities, it is possible to derive schedules based on the objective mentioned above.

The MCGA Project Scheduler does not specifically address net Present Value (NPV) as a criterion to be optimised. Unlike production projects, there is a reduced amount of cash flow associated with design-development projects. Furthermore, by minimising the elapsed time required to complete all activities in a design-development project, there is an increased chance that products will get to market faster and return on investment will be realised earlier. Hence, finishing a design-development project early improves NPV by default. In this respect, for design-development projects, cost-based objectives such as the maximisation of NPV have limited relevance.

7.4.3 Resource Allocation

For design-development projects, the major resources correspond to skilled personnel rather than fixed multiple unit requirements of different resource-types. This leads to an *assignment-type* problem, that is, the project is processed by assigning an individual, or a group of individuals, to each activity from a *resource pool* of PDO members.

Reported research into the scheduling of project activities, based on activity assignment, is limited. Drexl [Drexl 91], using a hybrid branch and bound / dynamic programming procedure, formulates the problem by linking *a set of alternative individual resources* to each activity such that the assignment of any one of an activity's alternatives implies a distinct activity duration and cost. The problem then is, given a set of activities linked by precedence relationships, where each activity can be performed by one of a number of skilled individuals and each individual performs the activity at a different speed and incurs a different cost, when should each activity be scheduled to begin, and which individual should be chosen to perform the activity, such that elapsed time and project cost are both minimised.

Salewski *et al* [Salewski 97] using a randomised heuristic solution-approach, present a formulation of the problem which they refer to as the *mode identity resource-constrained project scheduling problem (MIRCPSP).* Mode identity refers to a generalisation of the

general RCPS problem where the complete set of project activities is grouped into sub-sets of activities, each sub-set processed by the same fixed individual resource or set of resources.

Both Drexl's and Salewski's approaches have their relative merits and shortcomings. Drexl's approach allows the problem to be defined in the most comprehensive manner such that, by defining an alternative set of individual resources for each activity, a wider range of solutions can be investigated, and the solution space of viable schedules is enriched. However, for large projects, the definition of a set of alternative resources for each activity is time consuming and becomes increasingly reliant upon a database of resources, activities, processing times and costs which is also time consuming and difficult to maintain.

By grouping activities together, Salewski's approach links a pre-defined resource to specific activity sub-sets rather than to individual activities. This approach rationalises the problem and reduces the amount of data, time and effort required to fully define the problem. However, the problem is defined in such a way that only one pre-defined individual resource is capable of processing each sub-set of activities. Ultimately, such an approach limits the range of solutions to be examined and, as such, somewhat over constrains the problem.

The MCGA Project Scheduler addresses resource allocation in a manner that avoids these shortcomings and, at the same time, combines the advantages of both Drexl's and Salewski's, approaches. Here, activities are grouped into sub-sets. Each sub-set is then linked to a set of alternative individual resources. This new solution-approach to resource allocation, detailed in Sub-Section 7.5.1(2), rationalises the problem such that the time and effort required for definition is kept to a minimum. In addition, by allowing alternative individual resources to be linked to each sub-set, it ensures that a wide range of solutions can be investigated.

Before moving on, it should be highlighted that the RCPS problem, defined using such an approach to resource allocation, and assuming no precedence relationships between activities, is analogous to a problem category documented in the literature as the *labour tour scheduling problem* [Bechtold 91] and the basic *staff scheduling problem* [Shaffer 91, Easton 93, Ashley 95].

7.4.4 Type of Resource

As previously stated, the major resources associated with a design-development project are the skilled personnel who make up the PDO. Because of limits on the number of PDO members assigned to any one project, such resources are limited over a given time-period. However, at the end of the time period the whole of each individual resource is once again available. In this respect, the MCGA Project Scheduler considers resources as renewable.

7.4.5 Pre-Emption Condition

Having assigned one or more PDO members to process a design-development activity, work should not be interrupted. The practice of re-assigning PDO members between incomplete activities has a detrimental effect on productivity and should be avoided. Consequently, the MCGA Project Scheduler does not allow pre-emption. However, whilst pre-emption is not allowed, the MCGA Project Scheduler can accommodate periods of time during which PDO members are not available. Such periods may relate to vacation or sickness.

7.4.6 Number of Projects

Assuming that project activities make calls on a common resource pool of PDO members, there is little if any distinction between individual projects. Consequently, when dealing with a multi-project environment, all project activities can be treated as members of one large *meta-project*

7.5 The Multiple-Criteria Genetic Algorithm Project Scheduler

As the name implies, the Multiple Criteria Genetic Algorithm (MCGA) Project Scheduler incorporates a modified version of the standard genetic algorithm search technique first introduced in Chapter 4. However, instead of using a single search criterion, the MCGA generates near optimal solutions based on a range of different and sometimes conflicting objectives.

The MCGA, like the standard GA, can be adapted and applied to derive near optimal solutions to a wide range of problems. The most significant difference between applications is the way in which solutions to the problem are encoded as a string, decoded as a solution and evaluated within an *objective function evaluation sub-routine*. For the *containership loading problem* [Todd 97a] each string, representing a loading configuration of containers onboard a ship, is

decoded into a loading condition and evaluated within the evaluation sub-routine based on criteria such as the number of unloads.

In the same way, each string in the *job-shop scheduling problem* [Todd 97b], representing a schedule of jobs to be manufactured, is decoded into a schedule and evaluated within the evaluation sub-routine based on criteria such as the elapsed time taken to complete all jobs. Newly developed for it's application to the RCPS problem, the MCGA Project Scheduler incorporates a new evaluation sub-routine referred to as the *Project Schedule Builder (PSB).* Figure 7.4 summarises the MCGA Project Scheduler, which is summarised here, and detailed in the following sub-sections.

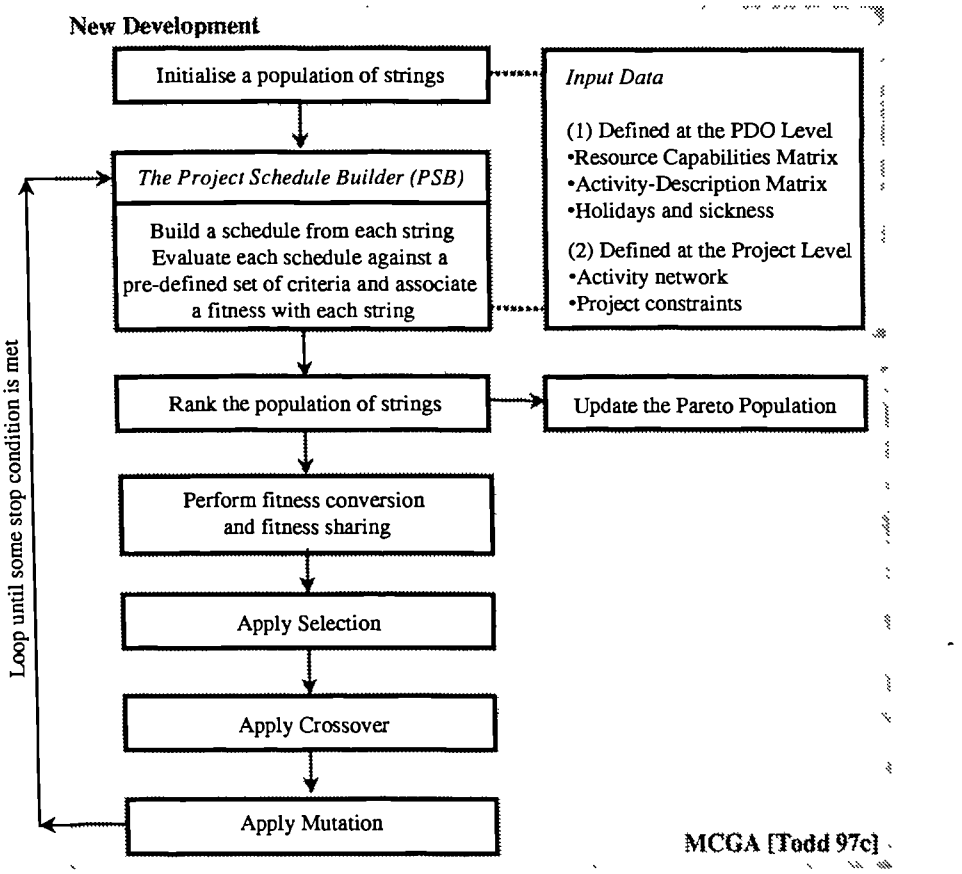


Figure 7.4: The MCGA Project Scheduler.

Based on a pre-defined set of input data, the MCGA Project Scheduler encodes a set of possible *solution definitions to the problem* as a population of strings. Each *string* represents a *solution definition* that can be built into a *schedule* using the Project Schedule Builder (PSB) evaluation sub-routine. In this respect, a string equates to a solution definition which, in turn, equates to a schedule. Once the PSB has built a schedule from each string, the resulting schedules can be evaluated against pre-defined criteria to associate a fitness with each string.

Once evaluated, the strings and their associated fitness are returned to the MCGA for genetic operation with the aim of deriving fitter strings which, when evaluated in turn, represent superior schedules.

Based on this summary, the following sub-sections detail; (*i*) the input data required by the MCGA Project Scheduler; (*ii*) how a solution definition which defines a schedule is represented by a string; (*iii*) how the Project Schedule Builder (PSB) decodes a string to build a schedule. (Appendix IV details how the MCGA works to derive near optimal schedules).

7.5.1 Input Data

The MCGA Project Scheduler uses four sets of input data:

(1) An activity network diagram

As output from the fourth function of the modelling strategy, the resulting activity network diagram defines a set of activities, durations and precedence relationships.

(2) A resource capabilities matrix

As introduced previously in Sub-Section 7.4.3, the MCGA adopts a new approach to resource allocation. Based on the similarity of their work content, activities can be grouped together into *sub-sets*. Furthermore, each sub-set of activities can be linked to a set of alternative resources via *work-type categories*. These work-type categories describe the work associated with a product's design-development phase. (E.g. Stress Analysis=Work-Type A, Engineering Drafting=Work-Type B, etc.).

As part of the *activity description matrix* described next, a work-type category is assigned to each activity. Subsequently, activity sub-sets are formed by grouping activities according to their associated work-type category. The *resource capabilities matrix* then defines those PDO members capable of processing each work-type and thus defines those members capable of processing each activity.

As mentioned previously, this new approach to resource allocation rationalises the problem such that the time and effort required for definition is kept to a minimum. This relates to the fact that, instead of having to define a set of alternative PDO members for each new project activity, it is only necessary to assign a work-type to each new project activity. Furthermore,

the capabilities of members can be updated at any time since they are defined independent of project definition.

As way of example, Table 7.2 illustrates a partial *resource capabilities matrix*, which relates the capabilities of three members of a PDO to five work-types.

	Work-Type				
PDO Member	A	B	C	D	E
Dave	0	1	1	1	0
Frank	1	0	0	1	1
Mary	1	0	0	1	0

Table 7.2: A partial resource capabilities matrix

From an observation of Table 7.2, the PDO member known to his friends as Dave is not capable of processing activities described by Work-Types A and E. However, he is capable of processing activities described by Work-Type B, C and D.

(3) An activity-description matrix

The *activity-description matrix* describes an activity in terms of it's associated work-type. For added versatility, each activity can be broken down into *stages* to allow the representation of variable and/or multiple resource allocation over the duration of an activity. The MCGA Project Scheduler allows the user to define staged activities in one, or a combination of two cases; (*i*) activities containing *parallel-linked stages*; and, (*ii*) activities containing *sequential-linked stages*.

(i) Activities containing parallel-linked stages

Because only a single resource can be assigned to each stage, parallel-linked stages are used primarily to enable more than one resource to be assigned to the same activity. Stages of the same activity that have been defined to begin at the same time must be delayed until all of the individual resources assigned to each of the activity's parallel-linked stages are available. Such delays are minimised as follows; as soon as one of the parallel-linked stages can begin, it's assigned PDO member is set to "waiting". At the same time, the other stages that are parallel-linked to this *trigger stage* are re-prioritised in each of the relevant PDO member's priority list of activities to be processed. When all parallel-linked stages have an assigned PDO member set to waiting, all of the activity's parallel-linked stages can begin.

(ii) Activities containing sequential-linked stages

Stages of the same activity that have been defined as a sequential group are prioritised in order to minimise the time lags between the finish and start of consecutive stages. Making PDO members wait in the same way as they are made to wait for the case of parallel-linked stages could reduce any potential time lags. However, it is assumed that in practice, project managers would not make an available PDO member wait in such situations. As a result, the MCGA Project Scheduler assumes that, once available, the first stage is allowed to begin and downstream sequential-linked stages of the same activity are re-prioritised in each of the other relevant PDO member's priority lists.

Whilst this modification does not guarantee that consecutive stages of the same activity will be processed immediately one after the other, the incorporated functionality implicitly assures that time lags between consecutive sequential-linked stages of the same activity are kept to a minimum. Figure 7.5 illustrates four examples of how project activities can be defined by stage.

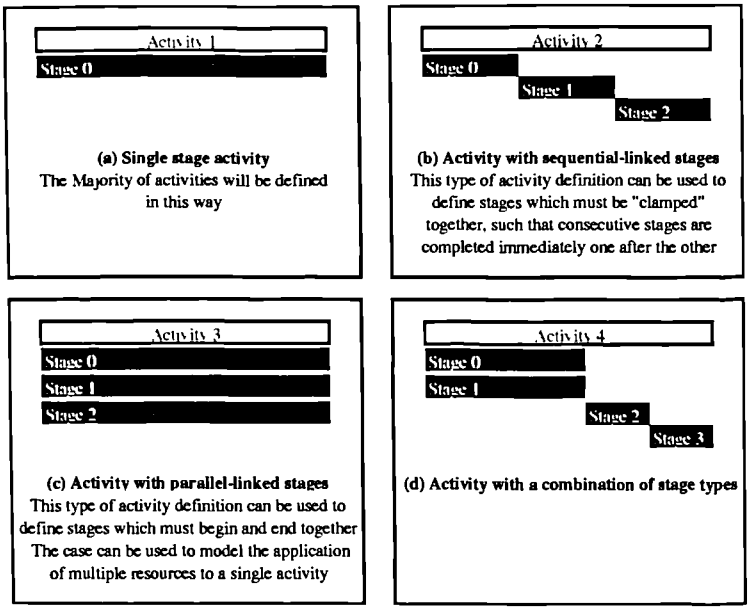


Figure 7.5: Examples of activity definitions that can be handled by the MCGA Project Scheduler.

Table 7.3 below, based on the four activity examples in Figure 7.5, represents a partial *activity-description matrix*, in which a work-type is linked to each activity stage. As way of explanation, consider Activity No. 3 in Table 7.3. This activity is made up of three stages, numbered consecutively as Stage Numbers 0, 1, and 2. The three stages should be scheduled to begin at the same time and, therefore, are assigned the same *parallel group (PG) number.*

In order to complete Activity No. 3, Stage 0, (referred to as Activity 3.0), a PDO member capable of Work-Type D must be assigned. From the partial *resource capabilities matrix* in Table 7.2, it can be observed that all PDO members are capable of processing activities of Work-Type D.

Activity	Stage No.	Work Type	PG No.	Stage No.	Work Type	PG No.	Stage No.	Work Type	PG No.	Stage No.	Work Type	PG No.
1	0	A	0	-	-	-	-	-	-	-	-	-
2	0	B	0	1	B	1	2	C	2	-	-	-
3	0	D	0	1	A	0	2	E	0	-	-	-
4	0	E	0	1	B	0	2	A	1	3	C	2

Table 7.3: A partial activity-description matrix

In the same way, Activity 3.1 requires a PDO member capable of Work-Type A which, again from an observation of the partial *resource capabilities matrix* in Table 7.2, implies using either Frank or Mary. Activity 3.2 requires a PDO member capable of Work-Type E. Only Frank is capable of processing activities of Work-Type E, which implies that Frank must be assigned to Activity 3.2.

Before moving on, it should be noted that, whilst the maximum number of activity stages illustrated in the partial *activity-description matrix* in Table 7.3 is four, there is in fact no limit on the number of stages into which an activity can be sub-divided.

(4) Constraints

The MCGA Project Scheduler can incorporate periods of time during which activities cannot be scheduled, as well as any constrained start and finish times associated with individual activities.

7.5.2 String Representation

There are two main classes of string representation used in GA-based scheduling applications; (*i*) *direct* representation; and, (*ii*) *indirect* representation. As the name implies, a direct string representation can be interpreted directly as a schedule, whereas an indirect string representation needs to be processed through some intermediate decoding stage in order to convert it into a schedule.

When dealing with simple resource allocation problems, a direct string representation can be used. However, in order to cope with the complex resource allocation issues associated with the scheduling of design-development activities, the MCGA Scheduler uses an indirect string representation.

Before illustrating an example string representation, Table 7.4 summarises the information which can be deduced from the partial *resource capabilities matrix* of Table 7.2 and the partial *activity-description matrix* of Table 7.3. This information is fully encoded within the string representation which is made up of integers, and is constructed in two main sections, *the resource choice section*; and, *the priority list section.*

Activity	Work-Type	Capable member(s) of the PDO
1.0	A	Frank, Mary
2.0	B	Dave
2.1	B	Dave
2.2	C	Dave, Frank, Mary
3.0	D	Dave, Frank, Mary
3.1	A	Frank, Mary
3.2	E	Frank
4.0	E	Frank
4.1	B	Dave
4.2	A	Frank, Mary
4.3	C	Dave, Frank, Mary

Table 7.4: Information deduced from Tables 7.2 & 7.3.

The resource choice section

In the example there are six activities (1.0, 2.2, 3.0, 3.1, 4.2, and 4.3) for which a choice of resource has to be made, so there are six genes in the *resource choice section.* The first gene is allocated to the first activity for which there exists a choice of resource (Activity 1.0), the second gene is allocated to the second activity for which there exists a choice of resource (Activity 2.2), and so on.

The value (allele) of each gene in the *resource choice section* defines the specific PDO member who, from the list of capable members, is actually assigned to process the activity associated with the gene. Therefore, in this example, because the first activity for which there exists a choice of resource (Activity 1.0) can be processed by either Frank or Mary, the first gene can take on the value (allele) of '1' which is the integer identity for Frank, or take on the

value '2' which is the integer identity for Mary. In the same way, the second gene can take on the value (allele) of either '0' (Dave), '1' (Frank), or '2' (Mary), and so on.

In the example string representation of Figure 7.6, Frank (represented by his integer identity '1') is assigned to process the first activity for which there exists a choice, Activity 1.0; Dave (represented by his integer identity '0') is assigned to Activities 2.2 and 3.0; Mary (represented by her integer identity '2') is assigned to Activities 3.1 and 4.2; and Frank (again represented by his integer identity '1') is assigned to process the final activity for which there exists a choice, Activity 4.3.

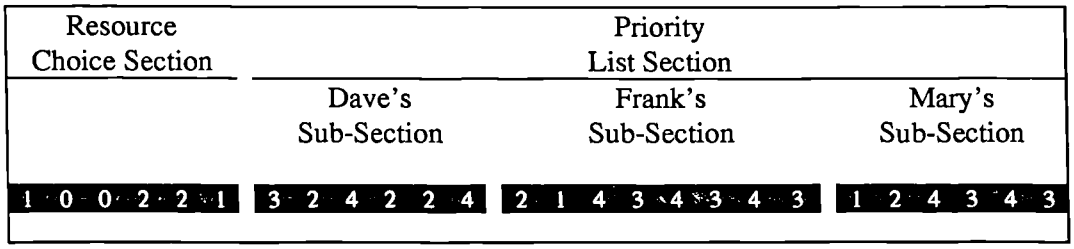


Figure 7.6: An example string

The priority list section

As first mentioned in Section 7.3.1, priority lists are used in order to resolve potential resource conflict. Remember that a resource conflict occurs at a point in time when, precedence relationships satisfied, two or more candidate activities may begin but there are insufficient resources to allow all of the candidate activities to begin. By scheduling an activity with a higher priority value in preference to other activities that may be competing for the same resource, potential conflict is resolved.

The *priority list section* of the string consists of a number of sub-sections. Each sub-section represents the full list of activities that could be processed *conceivably* by each member of the PDO, in an order of priority. In this example, the *priority list section* is divided into three sub-sections corresponding to the three members of the PDO listed in Table 7.2.

Reiterating the information summarised in Table 7.4, the full list of activities that could be processed conceivably by Dave is 2.0, 2.1, 2.2, 3.0, 4.1, and 4.3. In the example, in Figure 7.6, one of Dave's possible priority list permutations; (3.0)-(2.0)-(4.1)-(2.1)-(2.2)-(4.3), is represented, by the first *priority list sub-section*, as 3-2-4-2-2-4. The stage numbers are not required to be represented in the string because the MCGA implicitly assumes that, within

each sub-section, stages of the same activity are sequenced in ascending order. This limits the number of possible priority list permutations.

The priority list for Dave, represented by the first *priority list sub-section* in Figure 7.6, implies that at the start of a project, Dave should be used to process Activity 3.0 first. If this activity cannot be processed, because it's predecessors have not been completed, then Dave should be used to process Activity 2.0, and so on.

Depending on the values (alleles) of the *resource choice section* genes, Dave may not be chosen to process Activity 3.0 and therefore, before a schedule can be built, the *priority list sub-sections*, that only represent those activities that could be processed *conceivably* by each member of the PDO, must be adjusted to a final priority list of activities for each PDO member.

In order to create each PDO member's final priority list of activities, the string is first decoded by separating it into it's constituent *priority list sub-sections*. Each activity, for which a choice exists, is then allocated, according to the value assigned to the appropriate *resource choice section* gene, to the appropriate PDO member's final priority list of activities, and removed from all others. In this way the example string illustrated in Figure 7.6 can be decoded into the three final priority lists of activities, summarised in Table 7.5.

Dave	Frank	Mary
3.0	1.0	3.1
2.0	3.2	4.3
4.1	4.0	-
2.1	4.3	-
2.2	-	-

Table 7.5: The example string decoded into a priority list for each PDO member.

Theoretically, the maximum possible size of the *search space* for the example, which includes 2.6×10^9 possible strings, is determined as follows. The six genes in the *resource choice section* can take on the following values; Gene No.1 ('1' or '2'); Gene No.2 ('1', '2', or '3'); Gene No.3 ('1', '2', or '3'); Gene No.4 ('1' or '2'); Gene No.5 ('1' or '2'); Gene No.6 ('1', '2', or '3'). Therefore, based on resource choice alone, there are 216 ($2 \times 3 \times 3 \times 2 \times 2 \times 3$) varied string permutations.

In terms of the *priority list sub-sections*, remembering that the MCGA implicitly assumes that, within each sub-section, stages of the same activity will always be sequenced in ascending order, the 6 activities which Dave could conceivably process, could be prioritised 60 (6 factorial / [3 factorial][2 factorial]) different ways. In the same way, Frank's sub-section represents 1,120 (8 factorial / [3 factorial][3 factorial]) conceivable priority list permutations whilst Mary's sub-section represents 180 (6 factorial / [2 factorial][2 factorial]). Therefore, based on the three priority list sub-sections alone, there are 12,096,000 (60 × 1,120 × 180) varied string permutations.

Combining all of these permutations together means that the data represented in Tables 7.2 and 7.3 can be mapped to a search space containing the previously stated 2.6×10^9 (216 × 12,096,000) strings.

Whilst each of the 216 possible resource choice permutations implies 12,096,000 different strings, when they are adjusted according to one of the 216 specific resource choice permutations, the number of final priority list permutations is much smaller. For example, the *resource choice section*, in Figure 7.6, representing one of the 216 possible resource choice permutations as detailed in Table 7.5, can only be mapped to 480 ([5 factorial/3 factorial] × [4 factorial/2 factorial] × 2 factorial) final priority list permutations.

In the absence of any precedence relationships, the 480 string permutations for the example in Figure 7.6 would map directly to an equivalent number of distinct schedules. However, because of precedence relationships, different priority lists can in fact result in the same schedule. As such, depending on the number of constraints, the size of the *solution space* can be significantly smaller than the full theoretical search space.

7.5.3 The Project Schedule Builder (PSB)

The Project Schedule Builder (PSB) builds a valid schedule from a string. In the manner described in Section 7.5.2, each string is decoded into a priority list of activities for each PDO member. The priority lists, in combination with a knowledge of; (*i*) activity durations; (*ii*) activity precedence relationships; (*iii*) any pre-defined release/due date timing constraints; and, (*iv*) any periods of time, during which, PDO members are not available; can then be used to build a valid schedule.

The Project Schedule Builder (PSB) is a clock-based scheduling mechanism. This means that at the end of every unit-time interval, beginning with the first PDO member, the PSB determines the current state ('idle', 'busy', or 'waiting') of the PDO member. If a PDO member's current state is 'idle', then the PDO member's priority list is examined, and the highest priority unscheduled activity is considered for processing. If the precedence relationships and any other pre-defined constraints are not satisfied then the PSB moves down the priority list until a constraint satisfying unscheduled activity is found. If found, the constraint satisfying unscheduled activity can begin, and the status of the relevant PDO member set to 'busy' for the period of time it takes to complete the activity. If a constraint satisfying unscheduled activity is not found, then the relevant PDO member is either set to 'idle', or 'waiting', depending on which constraints are not satisfied, and the PSB moves onto consider the next PDO member.

If a PDO member's current state is 'busy', then the Project Schedule Builder (PSB) automatically moves onto consider the next PDO member. Once all PDO members have been considered for the current clock interval, the PSB simply ticks onto the next clock interval and re-considers the status of the first PDO member. This process continues until all activities have been scheduled.

The process described above details the basic version of the Project Schedule Builder (PSB). In addition, the PSB has been further developed in order to form the basis of explorative research into the relationship between organisational learning and scheduling issues. This exploration, which forms the basis of further ongoing research, is focused on the ability to define the *relative skill level* of PDO members quantitatively as an index.

(1) Relate activity duration to a PDO member's skill level.

The basic *resource capabilities matrix* simply defines those PDO members capable of processing each work-type and thus, via an *activity-description matrix*, defines those members capable of processing each activity. However, at a lower level of abstraction, it may be possible to define each PDO member's capabilities in terms of a relative level of skill. For example, instead of simply defining whether or not a member is capable of a specific work-type, it may be possible to define on a scale of 0 to 1, exactly how skilled the PDO member is for the work-type in question. It may then be possible to relate the time it takes for a member to process an activity described by a specific work-type to his or her skill level for the work-

type in question. As way of example, Table 7.6 represents a modified version of the simple *resource capabilities matrix* of Table 7.2.

PDO Member	Work-Type				
	A	B	C	D	E
Dave	0.0	0.8	1.0	0.5	0.0
Frank	1.0	0.0	0.8	1.0	1.0
Mary	0.5	0.0	0.5	0.5	0.0

Table 7.6: A modified version of the resource capabilities matrix

Referring back to the partial *activity-description matrix* of Table 7.3, Activity 1.0 is of Work-Type A which, from an observation of Table 7.6, can be processed by either Frank or Mary. By relating duration to skill level, if Activity 1.0 were to be processed by Frank, then the duration of the activity would be equal to the activity's pre-defined *base-duration* since Frank has a skill-level equal to 1.0 for Work-Type A. However, if Activity 1.0 were to be processed by Mary, then the duration of the activity would be equal to (*base-duration* / 0.5). In this case, the increased duration relates to the fact that Mary has a skill-level equal to 0.5 for Work-Type A.

Irrespective of whether or not this option is applied in the derivation of a schedule, the variation in derived schedules results from the variation in resource allocation and activity priorities, as defined by the string. However, if the option to relate activity duration to skill level is chosen, then the choice of resource allocation has a stronger impact on overall project duration.

(2) Updating skill levels as part of the scheduling function

Every time a PDO completes an activity described by a specific work-type, his or her skill level for the work-type in question may improve. Based on this assumption, if the option to update skill levels as part of the scheduling function is chosen then each PDO member's work-type skill level is updated every time he or she completes an activity described by the work-type in question according to one of the two learning curves illustrated in Figure 7.7.

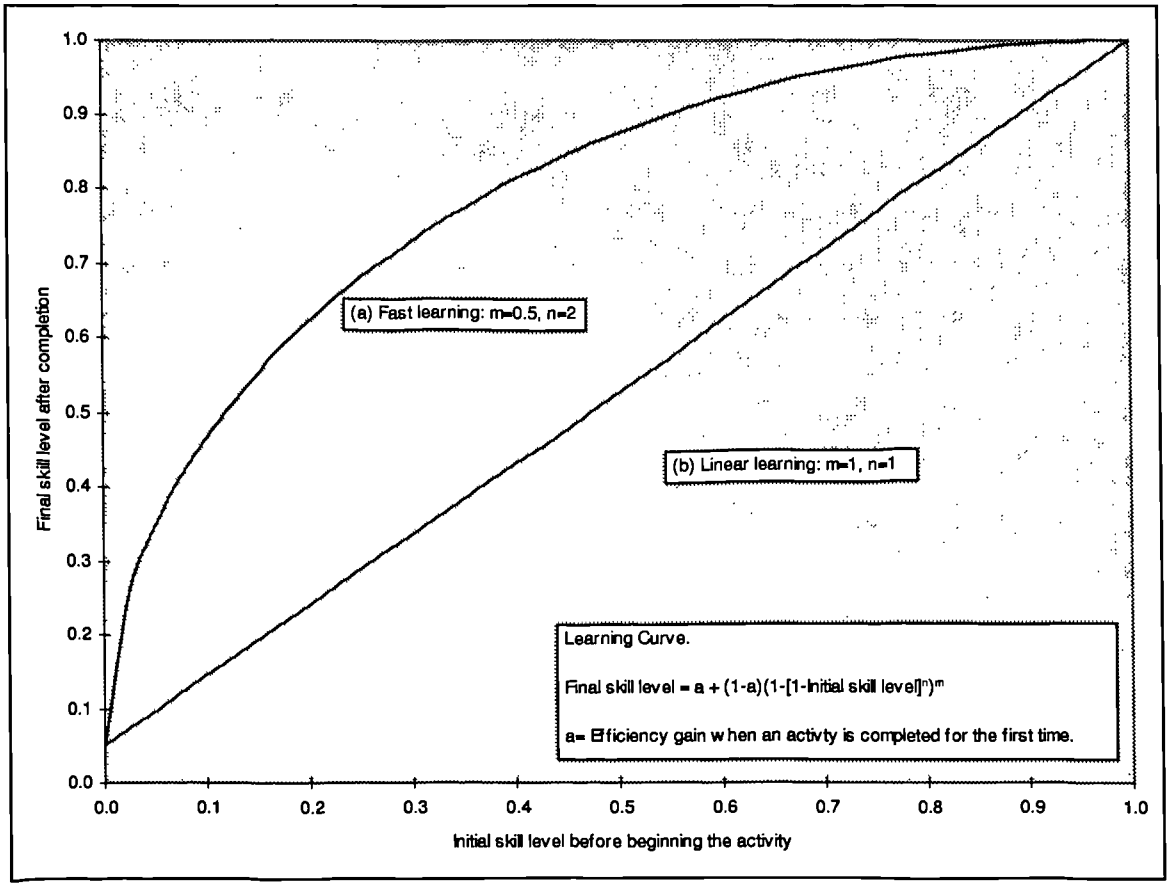


Figure 7.7: Learning curves.

(3) Assign *apprentice* PDO members to work alongside *journeyman* PDO members

If chosen as an option, every time a PDO member is assigned to an activity described by a specific work-type for which the PDO member has a maximum skill level (1.0), the PDO member may be used as a *journeyman*. This implies that if, at the decision point, another less experienced PDO member is idle, then he or she is also assigned to the activity as an *apprentice*.

This option can be invoked using a *scaled switch*. If the switch is set to "off" (0), then no apprentices are assigned. If set to "on" (1), then apprentices are assigned whenever possible. However if the switch is set between 0-1, then apprentices are assigned, wherever possible, with a probability, on average, equal to the switch value. For example if the switch is set to 0.5, then, apprentices are assigned, on average, once in every two possibilities.

7.6 Testing the MCGA Project Scheduler

Various aspects of the *MCGA Project Scheduler* have been tested using an example that is loosely based on the warship case study. This test-example is illustrated in Figure 7.8 as an activity network diagram where each activity is defined in terms of it's duration and precedence relationships.

In all of the tests that follow, the output from the MCGA Project Scheduler has been closely inspected and checked for errors in order to ensure that the computer program operated as intended. For each test, a population size of 500 was run over 100 generations. A typical run takes approximately 40 minutes using a Sun UltraSparc workstation.

Assuming that all of the activity precedence relationships in the activity network diagram of Figure 7.8 are described as finish-to-start, then a *critical path analysis* of the network indicates that the minimum elapsed time required to complete all activities is 20 time units.

In reality, however, there will only be a finite number of PDO members. As previously indicated, the MCGA Project Scheduler considers resource-constraints and the capabilities of PDO members based on the information contained in a *resource capabilities matrix.* Table 7.7 details the *resource capabilities matrix* that is to be used in the tests which follow. As observed from Table 7.7, the matrix represents a PDO limited to 6 members whose capabilities can be defined with respect to 5 standard *work-types.*

PDO member	Capabilities for Work-Type				
	A	B	C	D	E
1	1	1	1	1	1
2	1	1	1	0	1
3	1	1	1	1	1
4	1	1	1	0	1
5	1	1	1	0	1
6	1	1	0	1	1

Table 7.7: The resource capabilities matrix which forms part of the test-example.

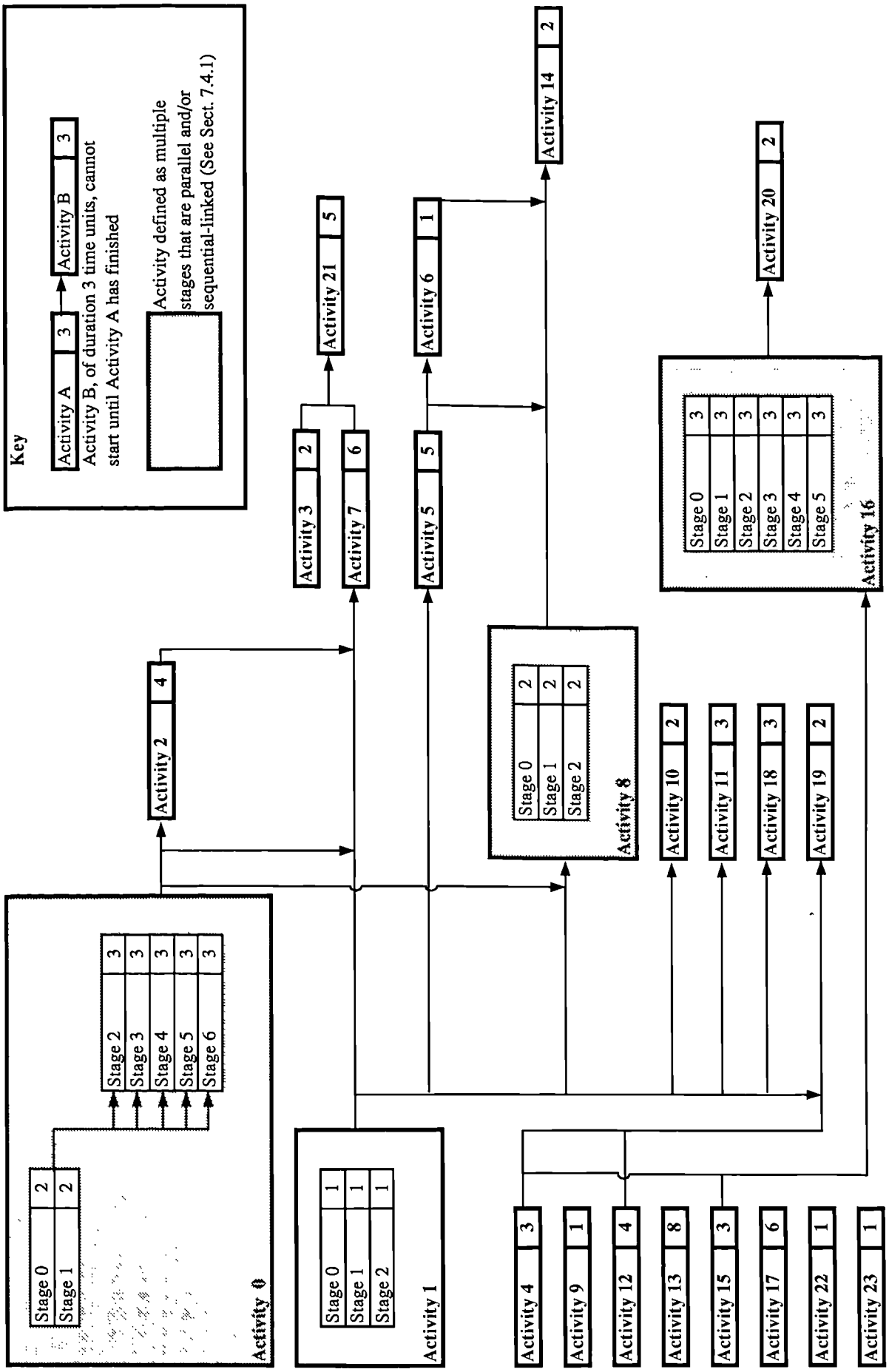


Figure 7.8: The test-example: an activity network diagram detailing each activity's estimated duration and it's precedence relationships.

In order to run the MCGA Project Scheduler, two additional pieces of input data are required; (*i*) an *activity-description matrix*; and, (*ii*) any *constraints*, for example, periods of time when PDO members are on holiday. The *activity-description matrix* in Table 7.8 can be created using information derived from the activity network diagram of Figure 7.8. For the purposes of this example, one of the 5 standard work-types has been assigned arbitrarily to each activity and no constraints on the availability of PDO members have been included.

Activity	S	W	P	S	W	P	S	W	P	S	W	P	S	W	P	S	W	P	S	W	P
0	0	A	0	1	A	0	2	A	1	3	A	1	4	A	1	5	A	1	6	A	1
1	0	B	0	1	B	0	2	B	0												
2	0	A	0																		
3	0	B	0																		
4	0	B	0																		
5	0	C	0																		
6	0	C	0																		
7	0	D	0																		
8	0	E	0	1	E	0	2	E	0												
9	0	E	0																		
10	0	B	0																		
11	0	B	0																		
12	0	E	0																		
13	0	C	0																		
14	0	C	0																		
15	0	B	0																		
16	0	B	0	1	E	0	2	B	0	3	E	0	4	E	0	5	E	0			
17	0	E	0																		
18	0	B	0																		
19	0	B	0																		
20	0	E	0																		
21	0	E	0																		
22	0	E	0																		
23	0	B	0																		

S: stage W: work-type P: parallel-group

Table 7.8: The activity-description matrix that forms part of the test case.

7.6.1 Deriving a resource-constrained schedule based on the objective of minimising elapsed time

In this first test, the MCGA Project Scheduler has been used to derive a *resource-constrained schedule* based on the PDO defined by the *resource capabilities matrix* of Table 7.7, and by adopting the sole objective of minimising the elapsed time required to complete all activities.

For the test, it is assumed that the time taken to complete each activity is equal to the activity's base-duration as defined in Figure 7.8. The resulting resource-constrained schedule is illustrated in Figure 7.9 whilst Figure 7.10 illustrates the convergence curve which represents how the average fitness of the solutions in each population improves over 100 generations.

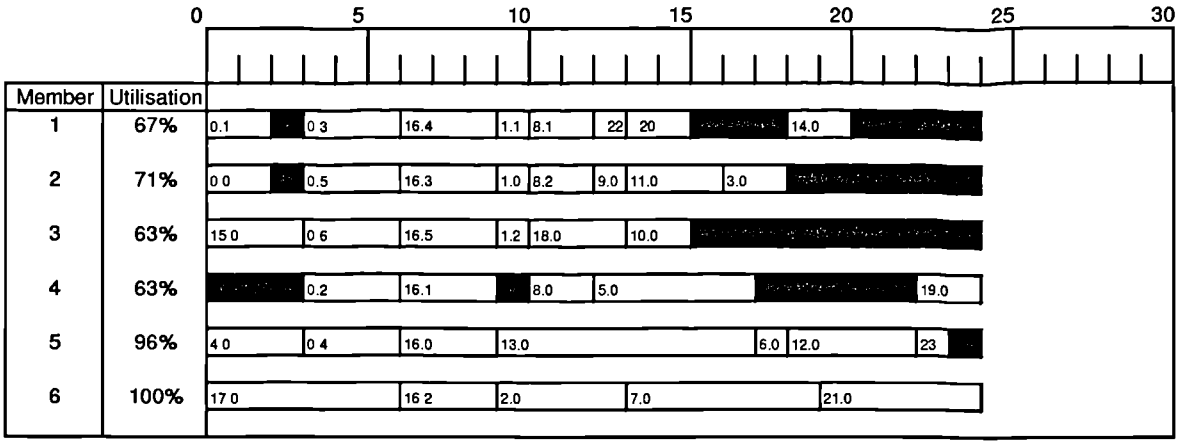


Figure 7.9: The resource-constrained schedule for the test-example, based on the sole objective of minimising the elapsed time required to complete all activities (The black bars indicate time when the PDO member is not working on a project activity)

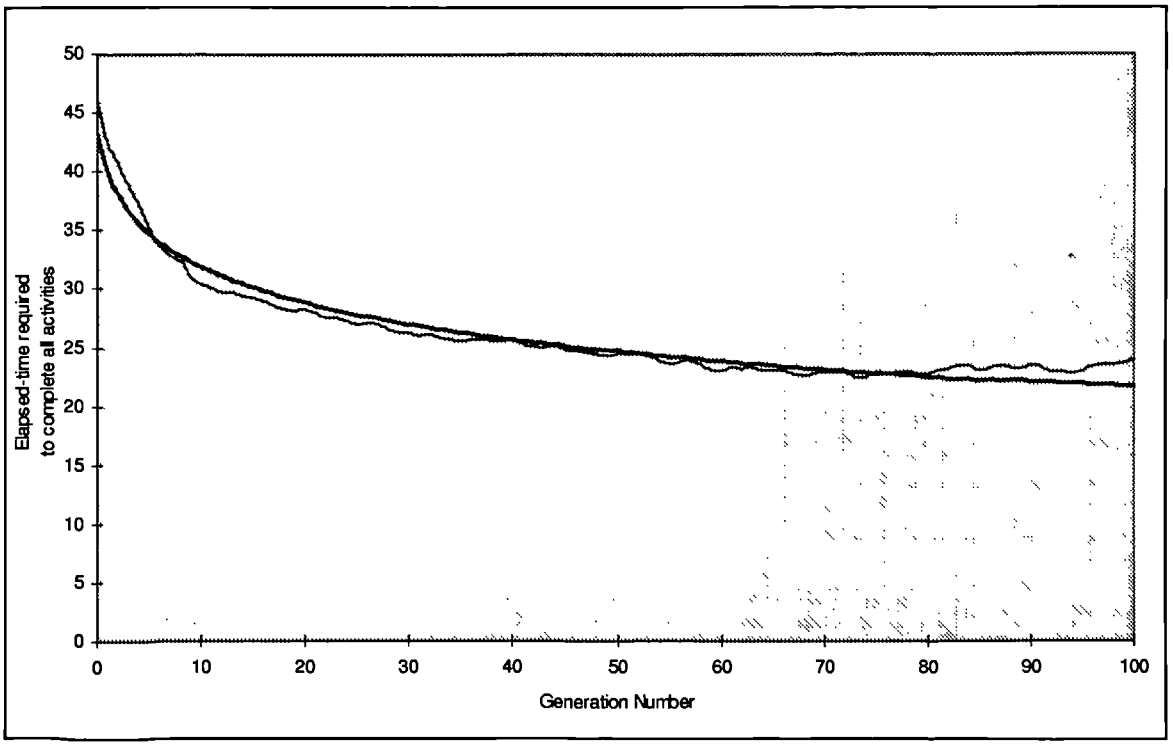


Figure 7.10: The genetic algorithm convergence curve representing how the average fitness of each population of solutions improves as the search proceeds from generation 0 to generation 100.

As indicated by Figure 7.9, the elapsed time required to complete all activities is 24 time units in this case. Because of resource-constraints the elapsed time is 20% longer than that of the 20 time units predicted by the critical path analysis. However, whilst the schedule of Figure 7.9 may be longer than that derived from critical path analysis, it represents a more accurate indication of how the project will progress.

7.6.2 Deriving a resource constrained schedule based on multiple criteria

Having illustrated the functionality of the MCGA Project Scheduler with respect to the principle objective of minimising elapsed time, the tests that follow have two purposes. Their first purpose is to form the basis of *explorative research* into the relationship between organisational learning and scheduling issues. Their second purpose is to *illustrate* the multiple-criteria capabilities of the MCGA Project Scheduler.

For the tests that follow, the simple *resource capabilities matrix* of Table 7.7 has been detailed further in order to represent the relative skill levels of the six PDO members with respect to the five work-types (*See* Table 7.9). All other input data, as detailed previously in Figure 7.8 and Table 7.8, remains the same.

PDO member	Skill Level for Work-Type					Average Skill Level
	A	B	C	D	E	
1	1.0	1.0	0.5	1.0	1.0	0.90
2	1.0	0.5	1.0	0.0	1.0	0.70
3	0.5	0.5	0.5	1.0	1.0	0.70
4	0.2	1.0	0.5	0.0	0.2	0.38
5	0.2	0.2	0.5	0.0	0.2	0.22
6	0.2	0.2	0.0	0.2	0.2	0.16

Table 7.9: A resource capabilities matrix that defines skill levels

Test 2:
Based on resource-constraints, minimise elapsed time and relate activity durations to skill levels

In this test, as before, the MCGA Project Scheduler has been used to derive a *resource-constrained schedule* based on the PDO defined by the *resource capabilities matrix* of Table 7.9, and by adopting the sole objective of minimising the elapsed time required to complete all activities.

However, for the test, it is assumed that the time taken to complete each activity is equal to the activity's base duration, as defined in Figure 7.8, divided by the relevant PDO member's skill level for the work-type which describes the activity in question. Finally, each time a member of the PDO completes an activity, his or her skill level is updated for the work-type in question using the linear learning curve, illustrated previously as Curve (b) of Figure 7.7.

The resulting resource-constrained schedule is illustrated in Figure 7.11 whilst Figure 7.12 illustrates the convergence curve which represents how the average fitness of the solutions in each population improves over 100 generations.

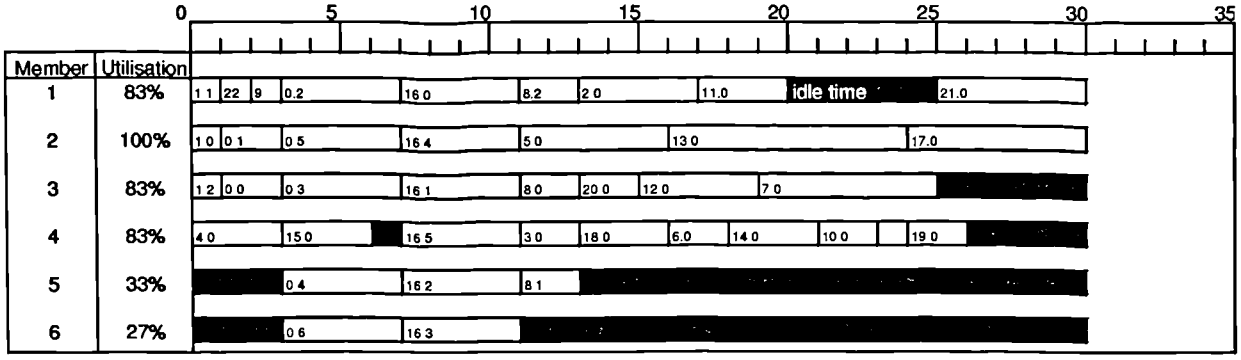


Figure 7.11: The resource-constrained schedule for the test-example, based on the sole objective of minimising the elapsed time required to complete all activities

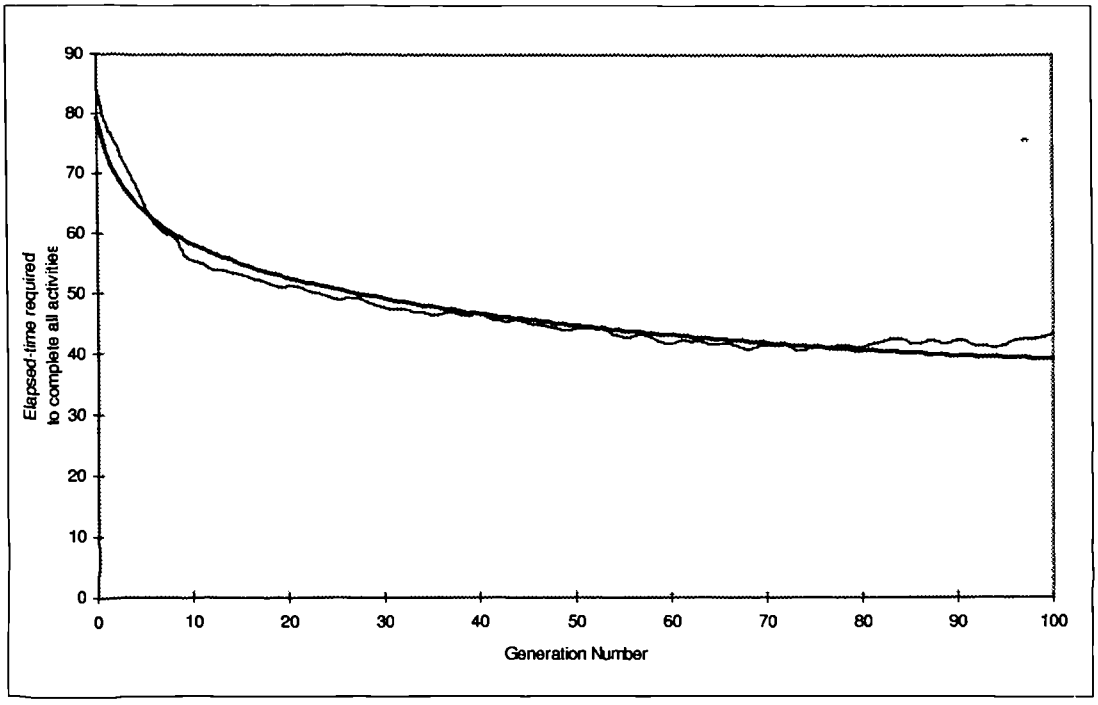


Figure 7.12: The genetic algorithm convergence curve representing how the average fitness of each population of solutions improves as the search proceeds from generation 0 to generation 100.

As indicated by Figure 7.11, the elapsed time required to complete all activities is 30 time units in this case. This is different to that indicated previously in Figure 7.9 because, in this case, PDO members are no longer assumed to posses the same level of skills. As such, some PDO members are in fact slower than others at completing the same activity.

By adopting the sole objective of minimising the elapsed time required to complete all activities, it can be observed from the utilisation values detailed in Figure 7.9 that the MCGA Project Scheduler favours the use of the most highly skilled members of the PDO such that PDO members 1-4 are used more frequently than members 5-6. This is because the higher skilled members can process activities faster.

For products that need to get to market as fast as possible, deriving a schedule based on the sole objective of minimising elapsed time is perhaps the best approach. However, the next test demonstrates a different scenario.

Test 3:
Based on resource-constraints, maximise the relative improvement of the PDOs skill level

For certain products, it may be possible to forego the objective of *minimising the elapsed time required to complete all activities* in favour of *maximising the relative improvement of the PDO's skill level.* Through experience and "on-the-job" learning, improvements in the general level of skills can result when members of the PDO process activities described by work-types for which they initially have a low skill level.

Consider the following example where the objective of maximising the relative improvement of the PDO's skill level may be appropriate: a *warship engineering company* may be asked to develop a pre-contract warship design-definition by a foreign navy. If, for whatever reasons, the company assigns a low probability that the enquiry will lead to a full contract, it may take the opportunity of using lesser experienced members of the PDO to work on the requested design-definition.

The Schedule illustrated in Figure 7.13 illustrates the resource-constrained schedule that is derived through the application of the MCGA Project Scheduler to the previously described test-example. This test is exactly the same as the previous one with the exception that, in this

case, the schedule has been derived based on the sole objective of maximising the relative improvement of the PDO's skill level.

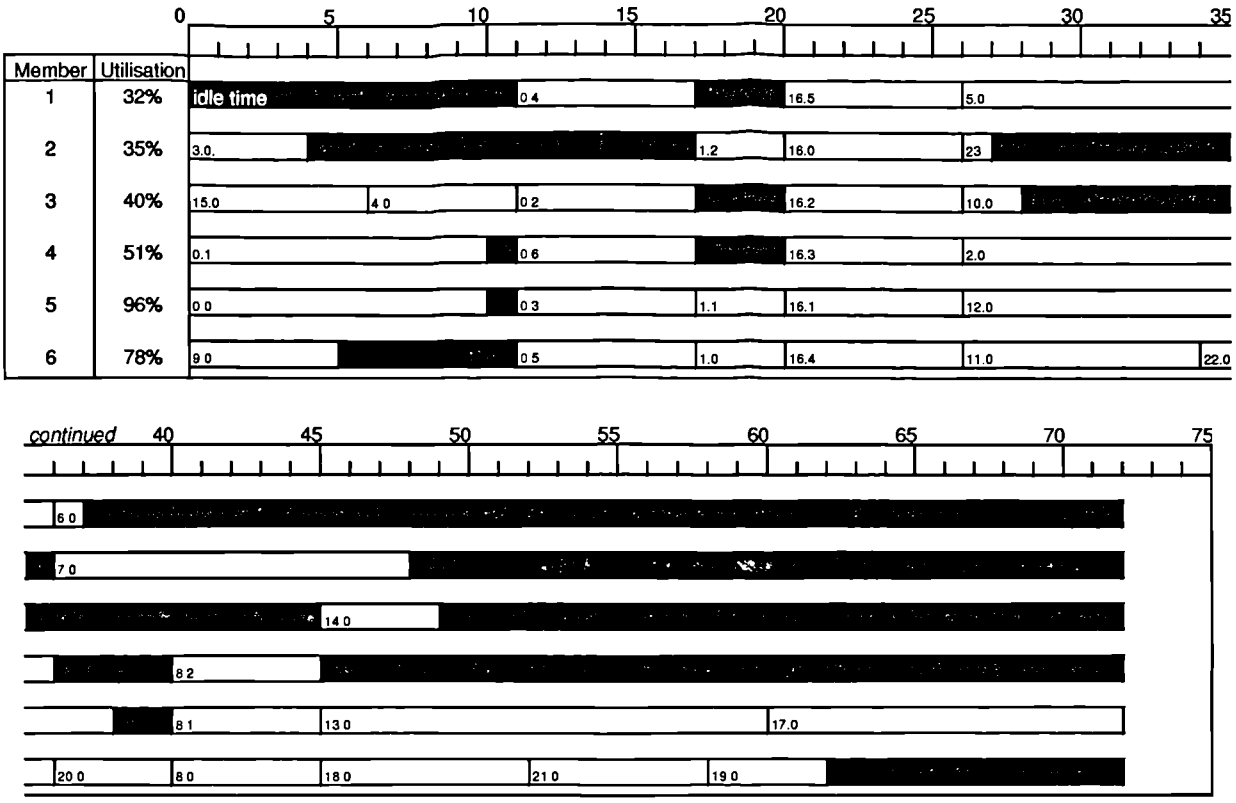


Figure 7.13: The resource-constrained schedule for the test-example, based on the sole objective of maximising learning (The black bars indicate time when the PDO member is not working on a project activity)

Figure 7.14 illustrates the convergence curve that represents how the average fitness of the solutions in each population improves over 100 generations. Here, the relative improvement of the PDOs skill level is calculated as the change in the average skill level of the whole PDO between the start and end of a product's design-development.

Whilst the schedule in Figure 7.13 represents a relative improvement of the PDO's skill level equal to 22%, it is at the cost of extending the elapsed time required to complete all activities. In fact, the elapsed time for this test result is more than double that associated with the previous test, that is, 72 time units compared to 30 time units.

As would be expected, by adopting the objective of maximising the relative improvement of the PDO's skill level, it can be observed from the utilisation values detailed in Figure 7.13, that the MCGA Project Scheduler favours the use of the least skilled members of the PDO

somewhat more than in the previous test. This is because the least skilled members offer the best opportunities for improvement in their skill levels.

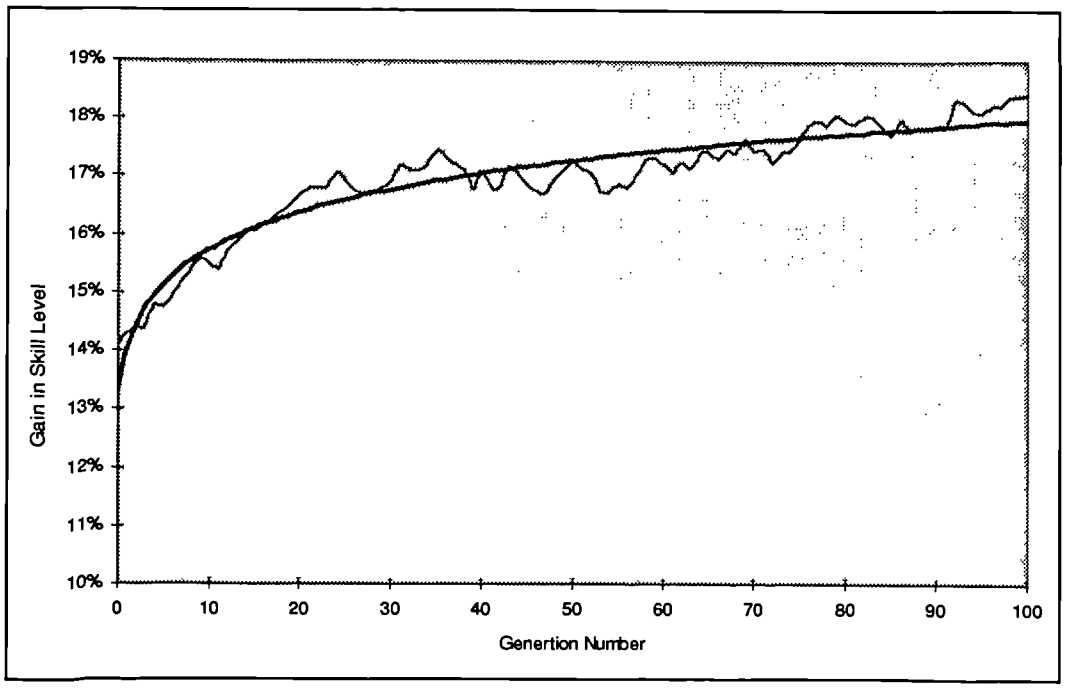


Figure 7.14: The genetic algorithm convergence curve representing how the average fitness of each population of solutions improves as the search proceeds from generation 0 to generation 100.

In order to optimise a schedule using the dual criteria of elapsed time and the relative improvement of the PDO's skill level, the MCGA Project Scheduler can be used to it's full potential. In the following test, the MCGA Project Scheduler is used to derive resource-constrained schedules based on multiple criteria.

Test 4:
Multiple Criteria

Because the elapsed time required to complete the design-development of a new product depends on the overall skill level of the PDO, engineering companies should be resourcing design-development work with a balance of (*i*) highly skilled PDO members who can process design-work effectively and efficiently; and (*ii*) lower skilled PDO members who need to be trained in order to improve their skill levels for a range of work-types so that the efficiency and effectiveness of the PDO is continuously improved.

Figure 7.15 illustrates the resource-constrained schedule that is derived through the application of the MCGA Project Scheduler to the previously described test-example. Whilst

the previous two tests have been based on a single criterion, for this test a resource-constrained schedule has been derived based on the multiple objectives of minimising the elapsed time required to complete all activities as well as maximising the relative improvement of the PDO's skill level.

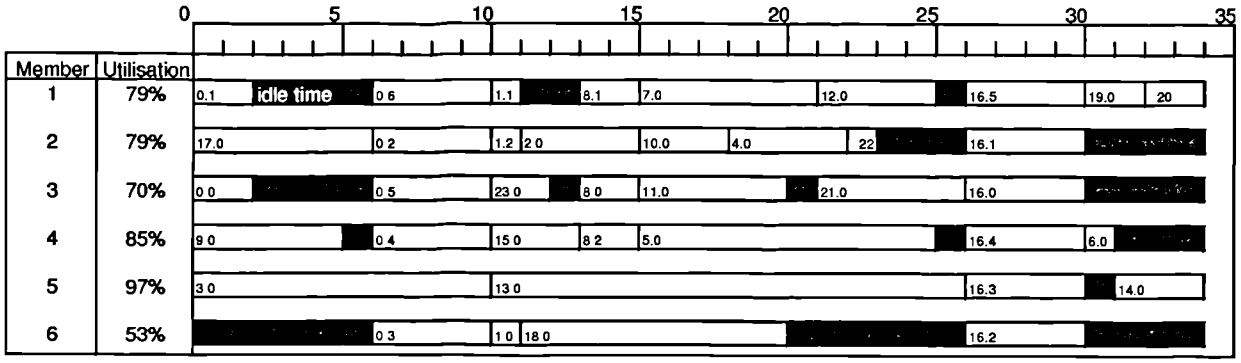


Figure 7.15: The resource-constrained schedule for the test-example, based on multiple objectives (The black bars indicate time when the PDO member is not working on a project activity)

Figure 7.16 illustrates how the distribution of solutions in the population changes over successive generations, whilst Figure 7.17 illustrates how the Pareto front changes over successive generations.

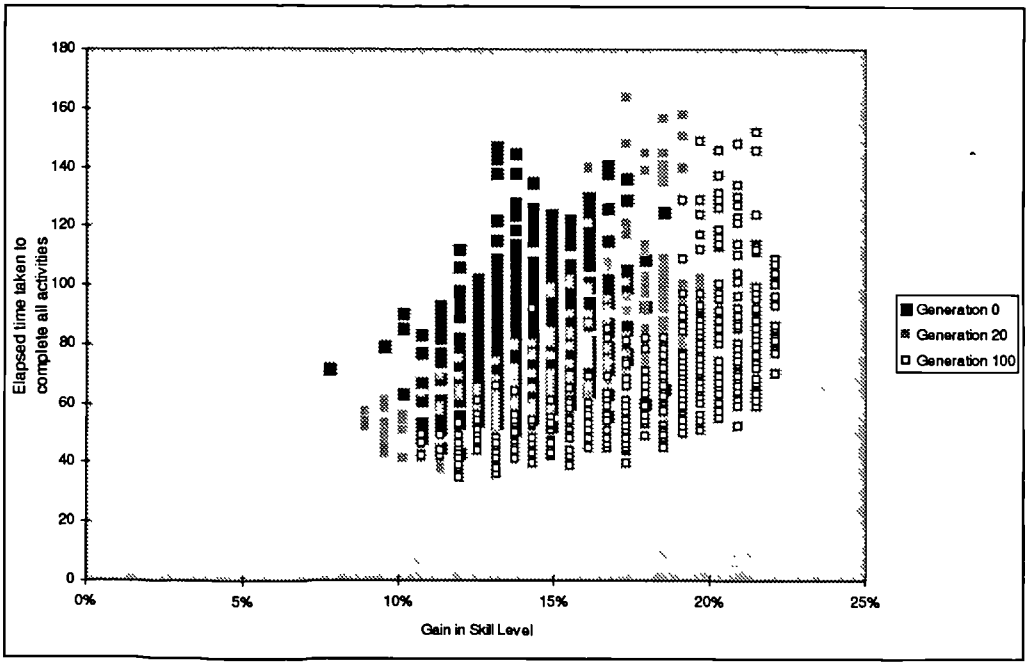


Figure 7.16: The distribution of solutions over successive generations

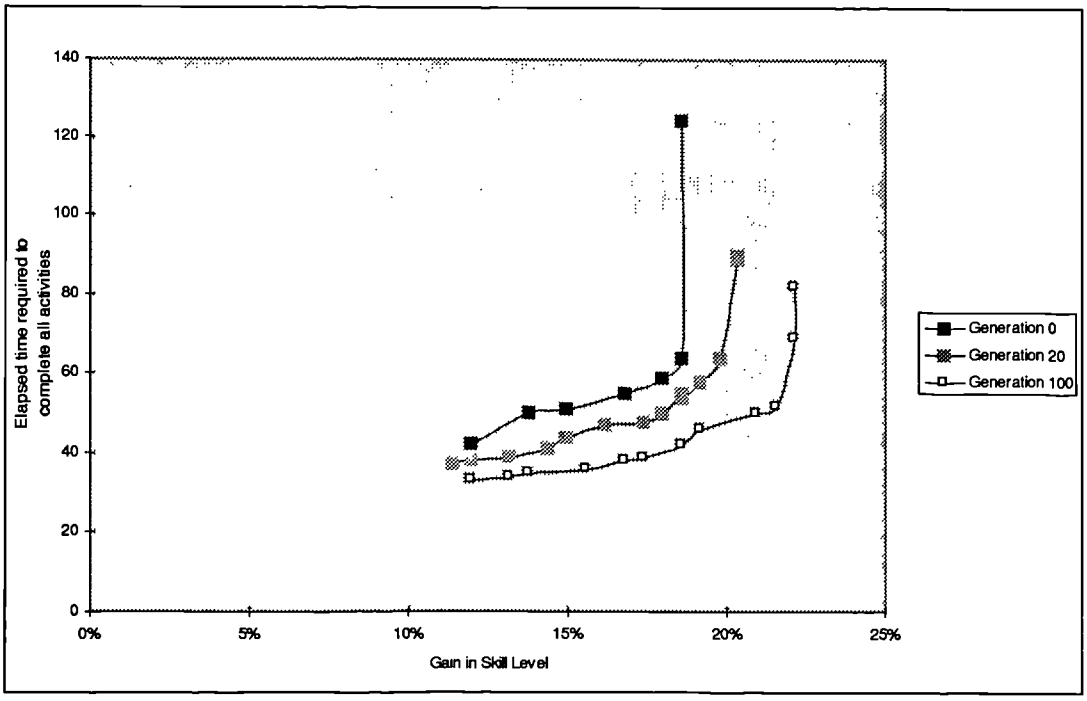


Figure 7.17: How the Pareto front moves over successive generations

Based on an observation of Figures 7.16 and 7.17, it can be seen how the MCGA Project Scheduler successively derives fitter solutions with each generation. As the search proceeds with each generation, the distribution of solutions and the associated Pareto front both shift towards higher values of gain in skill levels and lower values of elapsed time. This shift is indicated on the graphs by the movement of data-points towards the bottom-right-hand corner of each graph.

By deriving a schedule that is optimised using dual objectives, an acceptable compromise can be achieved. In this case, the elapsed time required to complete all activities is 34 time units, whilst the relative improvement of the PDO's skill level equals 13%. The schedule derived in this case represents a compromise and consequently, with respect to each of the objectives in isolation, this latest schedule is not as efficient as either of the two schedules derived previously. However, it does represent an acceptable balance of *improved skill levels* and *fast design-development* and, consequently, based on both criteria, it could be considered to be superior to either of the two previously derived schedules.

Test 5:
Assigning journeymen and apprentices

Whilst the multiple-criteria approach ensures that gains in the PDOs skill levels are facilitated with only a small detrimental effect on elapsed time, the schedules derived so far have

assumed that, when lower skilled PDO members are assigned to an activity, they learn and improve their level of skill simply by processing the activity.

In reality, learning and the subsequent improvement of the PDOs skill levels is most effective with the help of a teacher. Therefore, the MCGA Project Scheduler can build schedules that incorporate the notion of apprentices and journeymen. As stated previously in Sub-Section 7.5.3(3), based on this notion, every time a PDO member is assigned to an activity described by a specific work-type for which the PDO member has maximum efficiency, the PDO member may be used as a journeyman. This implies that if, at the decision point, another less experienced PDO member is idle, he or she is also assigned to the activity as an apprentice.

Figure 7.18 illustrates the resource-constrained schedule that is derived through the application of the MCGA Project Scheduler to the previously described test example. As in the previous test, the multiple objectives of minimising elapsed time required as well as maximising the relative improvement of the PDOs skill levels have been used. However, in this test, journeymen have been assigned wherever appropriate.

The elapsed time required to complete all activities in this final test equals 34 time units, the same as in the previous test. However, in this case, because journeymen are used, there are more opportunities for assigning lesser-experienced members of the PDO to activities. This has the effect of increasing the average utilisation of PDO members as well as improving on the relative improvement of the PDO's skill level, which in this case equals 18%.

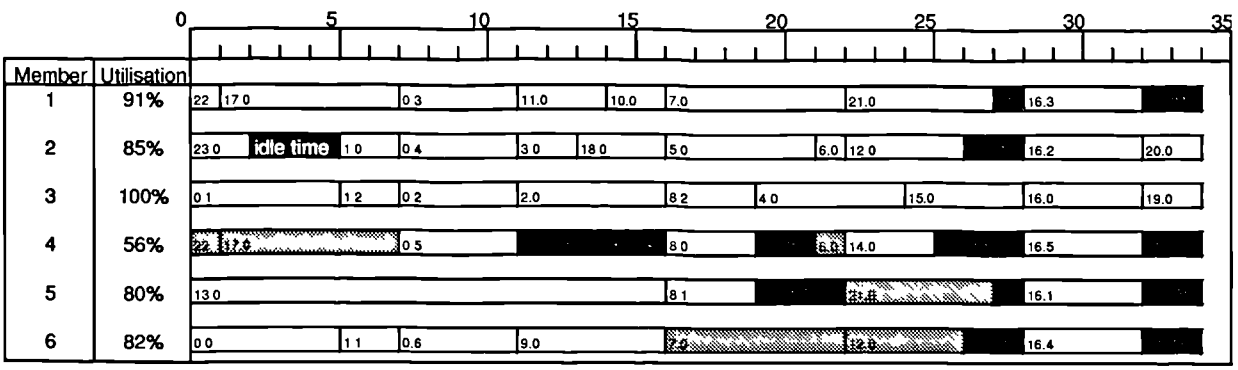


Figure 7.18: The resource-constrained schedule for the test-example, based on multiple objectives using journeymen. (The black bars indicate time when the PDO member is not working on a project activity, whilst the grey bars indicate the PDO member working as an apprentice)

7.7 Chapter Summary

Chapter 7 focuses on the fifth and final function of the proposed modelling strategy, namely, "derive a resource-constrained schedule".

Because the resource-constrained project scheduling (RCPS) problem can be viewed as a generalisation of the machine scheduling problem, Chapter 7 begins with a summary of the machine scheduling problem. Using a classification scheme, Chapter 7 also details various aspects of the RCPS problem and reviews previously published solution-approaches to the problem before introducing a new solution approach to the resource-constrained scheduling of design-development activities.

Having explained the concepts behind this new solution-approach, the Multiple Criteria Genetic Algorithm (MCGA) Project Scheduler, which acts as the principal mechanism of the fifth strategic function, is described. Using an example based on the warship case study, a number of tests have been presented that illustrate the functionality of the MCGA Project Scheduler.

Ultimately, the current research documented in Chapter 7 contributes to the subject in a number of areas. Whilst the MCGA Project Scheduler is based on the machine-scheduling tool developed by Todd [Todd 97b], it's application to the resource-constrained scheduling of design-development activities is original. Furthermore, the tool has been modified for it's application to the resource-constrained project scheduling (RCPS) problem. These modifications incorporate the newly developed *Project Schedule Builder* and a new definition of input data (*See* Figure 7.4). Specifically, these enable the user to represent variable and/or multiple resource allocation over the duration of an activity.

Finally, by relating activity duration to a PDO member's level of skill, and upgrading these levels of skill as a result of on-the-job training, the explorative research detailed in Chapter 7 contributes in a small way towards developing a relationship between organisational learning and scheduling.

8. Conclusions, Contribution and Further Work

8.1 Conclusions

The aim of this research has been to provide product development organisations with a strategy for modelling and optimising sequences and schedules of design-development activities such that the design-development phase of a product can be managed and controlled in a more effective manner than before. The following text now summarises the conclusions that have been drawn from the research in pursuit of this aim.

It is very difficult to write up a set of conclusions for the type of research summarised in this thesis. Few tests and experiments have been carried out and, as such there are few results to form the basis of conclusions. In this respect, the following text summarises the principal conclusions that have been made throughout the thesis. Ultimately, conclusions have to be drawn regarding whether or not the proposed strategy can be implemented such that the design-development phase of a product can in fact be managed and controlled in a more effective manner than before. Ultimately, the only way of truly drawing such conclusions will be as a result of the complete application of the strategy to an industrial project. This issue is being addressed as part of ongoing research.

In today's highly competitive business environment of rapid technological change, increased market segmentation and reduced product life cycles, it has been concluded that the best opportunities for gaining competitive advantage lie in the improvement of *product design-development performance* and, in particular, *faster* design-development. As a result, engineering companies are increasingly adopting new initiatives aimed at reducing time to market. One of the most popular of such initiatives is *concurrent engineering (CE).*

Concurrent engineering is defined here as the *concurrent* and *faster processing* of *design-development and production activities* supported by the *improved integration and communication of data, information, and knowledge.* Whilst it is accepted that the application of CE principles can go a long way towards reducing time to market, it has been re-iterated in the thesis that, like all new initiatives, there are costs as well as benefits to be considered.

Recalling the aim of the thesis, the proposed modelling strategy is used to create schedules "..such that the design-development phase of a product can be managed and controlled in an

effective manner..." In order for schedules to be effective, they should be based on activity durations that are accurate. If these durations are not accurate then the schedule will not reflect reality and, consequently, this principal management and control mechanism will be delusive. In order for activity durations to be accurate, they need to be based on processes that are stable and subject to small variability. As a result, it has been concluded that a system's approach to the management of business processes that are engineered to be stable and subject to contained variability is of fundamental importance if the proposed modelling strategy is to be used to maximum advantage.

Prior to the creation of a schedule of activities, a clear understanding of the product's design-development phase is required. By defining and then representing the most appropriate, essential system elements and their inter-relationships, a model can be used to gain a clear understanding of a product and it's design-development phase. In this respect, it has been concluded that the essential system elements to be represented by such a model are the *activities defined by a product design-work breakdown structure* and the essential inter-relationships to be represented are the activities' *data-dependencies, inter-dependencies* and *precedence relationships*. Furthermore, in order to use a model to analyse and compare a range of sub-objectives, it has been concluded that it should be *normative, quantitative* and *dynamic*.

A large number of models, relating to the design-development phase of a product, have been published. However, in order to manage and control a product's design-development phase in an effective manner, it has been concluded that a model should be created that is defined in terms of activities of *managerial significance*.

As suggested in the introduction to the thesis and later in Section 3.2, there is still some resistance from industry to the modelling of a product's design-development phase for the purpose of management and control. In this respect, a number of reasons commonly used to justify this resistance are discussed in the thesis. However, it has been concluded that all such reasons can be countered and, as such, a case for modelling has been justified and presented.

The first function of the proposed modelling strategy is to *create a product design-work breakdown structure* of design-development activities whilst the second is to *model the design-development activities and their data-dependencies* using the design structure matrix

system. These are perhaps the two most important functions of the strategy because they form the basic model, upon which, each of the three subsequent functions is built.

One of the conclusions drawn from experience with the industrial case study is that the matrix-model, and the product design-work breakdown structure, upon which it is based, should be created from scratch whenever the proposed modelling strategy is *first* used. It should always be remembered that data-dependence is at the core of the modelling strategy. Therefore, based on the discussion presented in Section 3.4, because existing modelling techniques such as activity network diagramming are limited in their representation of data-dependence, it has been concluded that the creation of a matrix-model based on an existing activity network diagram of activities and precedence relationships tends to unwittingly over-constrained any subsequent analysis. This conclusion is based on the fact that, whilst a data-dependency implies a precedence relationship, a precedence relationship does not necessarily imply a data-dependency (*See* Section 3.4).

The third function of the proposed modelling strategy is the *activity resequencing (ActRes) problem of deriving a near optimal sequence of activities*. Much of the published research carried out in this area has been focused on the development of procedures that search for the sequence of activities that implies the least amount of iteration. These procedures, commonly referred to as *partitioning procedures*, vary in their approach to; (*i*) how the iteration inherent to a given sequence of activities is measured; and, (*ii*) the search strategy used to derive an optimal sequence based on minimum iteration. Because the ActRes problem is NP-complete and the search space is extremely noisy and discontinuous, it has been concluded that the most effective and efficient approach to solving the problem is to use a genetic algorithm-based search strategy in the first instance to derive an interim solution-set. This is then further improved upon through the application of a heuristic-based local search to derive the final solution.

In terms of measuring the iteration inherent to a given sequence, a new objective function has been presented. This new objective function, combined with the *GA-Local Search Procedure* constitutes the *Scott Partitioning Procedure*. Having tested this new partitioning procedure against previously published solutions to five different ActRes problems, it has been concluded that the Scott Partitioning Procedure demonstrates a number of advantages over previously published procedures. In terms of functionality, it can develop sequences that; (*i*)

incorporate smaller iterative blocks; (*ii*) incorporate the requirement for fewer guesstimates; (iii) are derived automatically and yet still reflect reality; and, (*iv*) matrices that are easier to read, communicate and interpret (*See* Test 5).

In terms of procedural efficiency, no information is available as to the CPU time required to derive the previously published solutions. However, since the CPU time required to derive the new solutions on a Sun UltraSparc workstation is at most 7.2 minutes, it is considered that even if the previously published procedures were found to be faster, the saving in time would be of little significance.

As well as minimising iteration, it has been concluded that, when reducing design-development lead-time is desirable, accurate guesstimates of design-data can be made and the availability of resources permit concurrency, then an additional objective, namely, *maximising the concurrent processing of activities* should be considered. In such cases, because iteration and concurrency are somewhat conflicting, a multiple-objective approach is necessary that ensures that both objectives can be satisfied together in an acceptable manner.

Based on a weighted multiple objective-search function, different priorities can be given to each objective. Furthermore, it has been concluded that the choice of which objective to prioritise depends on the product that is being design-developed. If, for example, the product is completely novel and the company has little or no previous experience relating to the design-development of such a product, then the minimisation of iteration might be favoured. However, if the company has a wide range of previous experience and this experience is stored and easily retrievable as part of an effective *design reuse system*, then the maximisation of concurrency might be favoured. In the latter case, more guesstimates will be necessary, however, based on experience, it is more likely that these guesstimates will be sufficiently accurate so as to avoid the iteration that would otherwise be necessary.

The fourth function of the proposed modelling strategy is to *derive an activity network diagram.* In order to derive a schedule of activities, the design structure matrix that links activities by their *data-dependencies* and *inter-dependencies*, should be converted into an activity network diagram that links activities by their *precedence relationships*. Ordinarily, an activity network diagram is derived as soon as a product design-work breakdown structure has been created. In such cases, no cognisance is taken of the underlying data-dependencies

which, imply and are responsible for the precedence relationships. With the ability to model data-dependencies, to derive optimal sequences based on such, and to resolve data inter-dependencies, it is has been concluded that the proposed modelling strategy offers product development organisations with a new approach to the investigation and improvement of their overall design-development process.

The fifth and final function of the proposed modelling strategy is to *derive a resource-constrained schedule of activities*. In today's business environment, where engineering companies are under increasing pressure to design-develop new products faster with fewer resources, the scheduling of design-development activities with full cognisance of resource constraints may be considered to be imperative.

Much research into the resource-constrained project scheduling (RCPS) problem has been carried out. Based on the classification scheme illustrated previously in Figure 7.2, solution approaches to the RCPS problem vary in their approach to (*i*) search strategy; (*ii*) search objective; (*iii*) resource allocation; (*iv*) type of resource; (*v*) pre-emption condition; and, (*vi*) number of projects.

Based on a review of published research and feedback from managers, designers and planners, at a range of engineering companies (*See* Appendix I), a new solution approach to the RCPS problem has been developed. Because the problem is NP-complete, whilst heuristic-based search strategies are the most common, it is generally accepted that a genetic algorithm-based search strategy offers the most effective way of developing near optimal solutions. Most existing solution approaches tend to focus on a single objective. In particular, the minimisation of the elapsed time required to complete all activities is most favoured since reducing time to market is now viewed as the principal goal of most engineering companies involved in the design-development and production of products.

However, when deriving a schedule based on the sole objective of minimising the elapsed time required to complete all activities, a manager or automatic procedure will tend to favour the use of the more experienced members of the PDO who are usually the most efficient. This practice may have serious implications for the company in the medium to long term in that, the lesser experienced members do not receive enough of the on-the-job training which is necessary in order to develop their own skills.

Consequently, it has been concluded that a solution approach to the RCPS problem may be able to handle multiple objectives that could include maximising the skills gained by members of the PDO as well as minimising elapsed time. Furthermore, because such an approach adds additional complexity to a problem which is already NP-complete, it has been concluded that the only viable option is to base the solution approach on a genetic algorithm search strategy - hence the development of the *Multiple-Criteria Genetic Algorithm (MCGA) Project Scheduler.*

In order to apply the MCGA Project Scheduler effectively, a number of organisational issues have to be addressed. In particular, the MCGA Project Scheduler works on the basis that members of the PDO are assigned, from a *resource pool*, to a product when, and only when, their specific skills are required. Based on conversations with managers and designers, such a suggestion results in a range of responses. Most accept that, in a multi-project environment where resources are particularly scarce, it may be necessary to adopt such an approach. However, such an approach subjects each *product manager* to an increased amount of pressure to ensure that all members working on a specific product at any one time all share a common understanding. In response, it has been concluded that the adoption of the tools and techniques first mentioned in Section 2.3 could help to ensure that a common understanding is achieved by ensuring the effective integration and communication of data, information and knowledge.

8.2 Contribution

The research contained in this thesis contributes to making significant practical advances to the subject of modelling the design-development phase of a product at the strategic level as well as at the more detailed functional level.

At the strategic level, a strategy for modelling the design-development phase of a product has been introduced. This new strategy focused specifically on design-development (*See* Figure 3.3) differs from traditional unfocused strategies (*See* Figure 3.1) in a number of ways. These differences, detailed in Section 3.4, can be summarised as follows:

- Firstly, the proposed strategy introduces a new approach to creating a WBS of design-development activities namely the *product design-work breakdown structure.*

- Secondly, the proposed strategy introduces two additional functions that fit between those of creating a WBS and creating an activity network diagram. These two new functions, namely "*model the activities and their data-dependencies*" and "*derive a near optimal sequence of activities*" allow the user to model, investigate and analyse alternative sequences of activities at a level of detail not previously addressed by traditional strategies.

- Thirdly, the proposed strategy expands the function of deriving an activity network diagram by introducing a new procedure for interpreting and resolving any iteration that is implied by inter-dependant activities. The result is an activity network diagram that represents a truer definition of the product's design-development phase.

- Finally, the proposed strategy expands the function of deriving a resource-constrained schedule of design-development activities by introducing a new scheduling technique that is focused specifically on some of the issues that are peculiar to design-development. These issues relate primarily to the fact that in design-development, more than production, resources are considered to be scarce.

At the strategic level, the proposed strategy also differs in a number of ways from prior research specifically focused on modelling the design-development phase of a product. Because these differences need to be described with specific reference to prior research, a summary here would tend to be verbose and repetitive. As a result, the reader is referred back to Section 3.8 for a detailed description of how the proposed strategy differs from prior research on modelling the design-development phase of a product.

In addition to it's differences to prior research and practice, at the strategic level the research addresses a number of issues that need to be considered if any modelling strategy is to be successful. These issues which, relate specifically to the notion of process stability and variability, have been described as invisibles because, all too often, engineering companies ignore them. It is considered that by presenting a summary of such issues as background, future work relating to the modelling, scheduling, management and control of design-development activities will be carried out with a cognisance and a better understanding of the implications of such issues.

Finally at the strategic level, the research is qualified, to an extent, through the application of the proposed modelling strategy to an industrial case study based on the pre-contract design-development of a warship (*See* Appendix I).

The research contained in this thesis also contributes to making significant practical advances at the more detailed functional level. These contributions, highlighted at the end of Chapters 4 to 7, can be summarised as follows;

- The *Optimal Sequencer Genetic Algorithm* component of the *GA-Local Search Procedure* is a modified version of the well-documented standard genetic algorithm. Whilst most of the modifications are based on published methods, the modified mutation operator and the incorporation of sequence constraints have been newly developed for solving the *ActRes Problem.*

- The *Heuristic Local Search* procedure is completely new and, as the second component of the *GA-Local Search Procedure*, helps to ensure that solutions to the ActRes Problem are derived in the most efficient manner.

- The *Scott Partitioning Procedure* is based on the *GA-Local Search Procedure* and a newly developed objective function. This new function is a mathematical expression that can be used to measure the inter-activity iteration associated with a given sequence of activities. It differs from those developed previously since it aims to measure unanticipated iteration as well as that which is implied through the need for guesstimates.

- Whilst the objective of maximising concurrency is not new, its utilisation in the context of deriving a near optimal sequence of activities based on a DSM is original. Furthermore, the approach to deriving sequences based on the multiple objectives of maximising concurrency and minimising iteration is original. In total, the new objective functions for measuring iteration, concurrency (both separately and together), optimised using the *GA-Local Search Procedure*, contribute to the subject of deriving near optimal sequences of design-development activities using the DSM system.

- Little published research is available that details the iterative block resolution (IBR) that is necessary in order to resolve the intricacies of iteration prior to commencing the

scheduling of product development activities. Consequently, the newly developed *IBR Procedure* is largely original and forms a contribution by linking the DSM system to the subject of planning.

- Whilst the MCGA Project Scheduler is based on the machine-scheduling tool developed by Todd [Todd 97b], it's application to the resource-constrained scheduling of design-development activities is original. Furthermore, the tool has been modified for it's application to the resource-constrained project scheduling (RCPS) problem. These modifications incorporate the newly developed *Project Schedule Builder* and a new definition of input data

- By relating activity duration to a PDO member's level of skill, and upgrading these levels of skill as a result of on-the-job training, the explorative research detailed in Chapter 7 contributes in a small way towards developing a relationship between organisational learning and scheduling.

8.3 Further Work

Whilst valuable feedback has been derived through the use of the industrial case study, further validation of the proposed modelling strategy needs to be undertaken. This issue is to be addressed by an ongoing research project based at a large engineering company involved in the design-development and production of warships. Whilst the modelling strategy has been demonstrated within the thesis using a warship, it is considered that the strategy is equally applicable to modelling the design-development phase of any engineered product, although further research with a wider range of engineering companies would indicate whether this expectation is valid.

As indicated at the very beginning of the thesis, the notions of process stability and variability have a very significant influence on the successful outcome of any modelling strategy. Most work in this field has been focused on production processes, but one area of further work could be focused on the development of research into studying and improving the stability and reducing the variability of pre-production processes and, in particular, design-development processes.

Because a product design-work breakdown structure has such a significant impact on the success of the modelling strategy, more work needs to be undertaken in order to develop the approach which is merely introduced within the thesis. In particular, research into the viability of defining activities in terms of their inputs, outputs, standard procedures, experience and technical uncertainties should be undertaken.

In terms of modelling design-development activities and their data-dependencies, the examples used in the thesis are relatively small; the largest contains 60 activities. As the number of activities increases it may be necessary to use a hierarchical approach to the creation of a set of inter-related design structure matrices. For a specific engineering company it may also be possible to derive a set of generic models that cover a product range, and which, are simply customised for specific products.

Based on the ActRes problem of deriving a near optimal sequence of design-development activities, the thesis proposes a dual-objective approach focused on the maximisation of concurrency as well as the minimisation of iteration. Concurrency is desirable since it implies reduced lead times, however it also results in the increased use of guesstimates which, if proven to be inaccurate, will cause iteration. It is concluded in the thesis that design re-use initiatives can help to ensure that guesstimates based on past experience, knowledge and information are more accurate. One area for further research could be focused on how design re-use initiatives can be used to support the sequencing of activities based on the maximisation of concurrency.

In terms of deriving an activity network diagram from a design structure matrix, the iterative-block resolution (IBR) procedure is described in rather subjective terms. The guidelines proposed need further refinement through increased exposure to real-life case studies and, in the future, it may be possible to integrate such guidelines into a semi-automatic knowledge-based system.

Further work also needs to be undertaken so that the explorative research detailed in Chapter 7 can be developed and a more discerning link between organisational learning and scheduling can be made.

To conclude, the specific techniques introduced within the thesis, such as the GA-Local Search Procedure and the MCGA Project Scheduler, themselves offer a number of possibilities for further enhancement. Streamlining the computer code, further investigation into the optimisation of the various GA parameters, and parallelisation of the code can all help to provide the improvement in procedural efficiency which is required in order to allow the techniques to tackle ever-increasing industrial sized problems.

I. The Industrial Case Study

As part of the research detailed in this thesis, time was spent discussing and studying design-development alongside designers, planners, researchers and project-managers at a range of engineering companies. These companies are involved in the design-development and production of a range of different types of engineered products including DIY consumer products, air-filters, electronic jacquards for weaving-looms, offshore production platforms, surface warships, merchant ships and naval submarines.

Based on this experience, a case study based on the *pre-contract design-definition of a warship* was chosen to demonstrate how the proposed strategy could be applied. This choice was influenced by a *number of factors including*; (*i*) the author's educational background, training and practical experience in the shipbuilding industry; (*ii*) an avid interest of the collaborating shipyards to become involved in the research; and finally, (*iii*) the need for a case study that was of sufficient complexity to demonstrate the proposed modelling strategy without being complex to the point that it overshadowed the strategy itself.

The following text describes a process of design developing a warship based on a number of sources. The principal sources are designers, planners, researchers and project-managers from the collaborating shipyards, who over the period of the research helped to develop a description of their design-development processes. In addition, the process description also incorporates the views of several researchers [Andrews 88, Andrews 96, Andrews 97, Brown 86, French 85, Snaith 82, Ulrich 95].

I.1 The Notion of the Dual Definition of a Product

The following text develops the notion first introduced in Section 3.4, that every product has at least two complementary definitions. The first, which defines the product "as it will be when completed" is referred to as the design-definition, whilst the second, which defines the product "as it will be manufactured and assembled" is referred to as the production-definition.

(1) The Product's Design-Definition

A warship is composed of a set of *"on-board" engineering sub-systems* arranged within a *structural containment sub-system.* Each of the sub-systems include *functional objects* the characteristics of which are established so that they satisfy the warship's *functional /*

operational requirements. These requirements relate to the functional and operational processes that contribute wholly or in part to a warship's functional and operational performance and, at the highest level of description, include "*float*", "*move*", "*fight*" and "*support crew*".

Based on this set of key functional / operational requirements, *solution schema* can be created that satisfy these requirements. In the first instance, the schema are represented in terms of *diagrammatics* that represent (*i*) the warship's *functional sub-systems*; and, (*ii*) the *inter-dependencies between sub-systems*, expressed in terms of the *transfer of energy*, *information* and *material*.

Each sub-system of a product is composed of *physical objects*. Thus, a warship's structural containment sub-system is composed of *surfaces* that define a hull envelope, suitably divided internally by *decks* and *bulkheads*, all of which are clad with an appropriate structural schema composed of *interconnected structural objects* including plates, web frames and longitudinal stiffeners.

In addition, a warship's "on-board" engineering sub-systems are composed of (*i*) *modules* which represent either a single piece of equipment, or a rational grouping of functionally-related equipment, interconnected by local connectives, and supported on a common foundation; (*ii*) *long-distance connectives*, comprising piping, cabling and trunking, which typically interconnect modules with storage tanks; and, (*iii*) *fittings* including pipe clips and cable trays.

Based on this understanding of physical objects, solution schema, initially represented in terms of diagrammatics, can be developed into a hierarchy of sub-schema based on physical objects. Beginning with the warship as a whole, then addressing each sub-system in turn, a solution schema is developed and is expressed in terms of (*i*) a *configuration* that depicts the logical and physical connectivity between functional/physical objects; (*ii*) *a spatial arrangement* which depicts the disposition and location of physical objects relative to each other; (*iii*) the corresponding *accurate full-scale geometry* of their circumscribing envelopes; and, (*iv*) an associated set of *derived* and *assigned attributes*.

Attributes are either inherently derived properties of a physical object or are arbitrarily assigned according to some rationale. Typically, the former include mass-properties, as well as properties that describe how the object behaves in its operational environment (e.g. stability characteristics) and predictions relating to operational performance, while the latter include make, catalogue references, price, due-dates and part numbers.

In summary, a product's design-definition expresses *the architecture of the product*, and is represented as a schema that ultimately *embodies* the functional sub-systems of a product in terms of interconnected physical objects disposed and located in 3-D space.

Usually, a number of different candidate solution schemas are created and developed as part of an iterative *proposal-testing-modification (PTM) cycle* (*See* Sub-Section 5.2.2). Ultimately, based on analysis, preferences and experience, one schema, which may in fact be a combination of the best parts of a number of other schema, is chosen. Based on this choice, the solution schema that satisfies functional / operational requirements in the most cost-effective way is then developed and defined in more detail. This process, of proposing, testing and modifying a number of different solution schemas is commonly referred to as *concept generation and selection.*

In order to ensure that the solution schema that is ultimately selected satisfies the functional and operational requirements in the most efficient and cost-effective way, the PDO first needs to understand and apply beneficially (*i*) the alternative component technology that could be incorporated into the product's sub-systems; (*ii*) alternative materials; and, (*iii*) the analytical modelling techniques, such as finite element analysis (FEA), that can be used to predict structural responses, functional efficiency and operational performance. Such techniques can be used to identify potential weakness in a product's ability to sustain it's physical integrity and functionality.

Based on this understanding and application, the PDO then needs to *engineer-out* any weaknesses by either re-configuring or re-sizing the product globally or locally. At the same time, any uneconomic over-provision of materials must be identified and removed.

(2) The Product's Production-Definition

Once the product's design-definition has reached a sufficient level of detail, it can be disaggregated to create the product's complementary production-definition. This second definition relates to a dissagregation of the product's design-definition into a hierarchy of assemblies, sub-assemblies, minor-assemblies, equipment modules, connectives and piece-parts, all of which correspond with the planned stages of assembly and manufacture.

I.2 Developing the Pre-Contract Design-Definition of a Warship

The following text summarises the design-development of the pre-contract design-definition of a warship. In the first instance the *functional / operational requirements* of the *Warship Product System* may be grouped according to (*i*) Float; (*ii*) Move; (*iii*) Fight; and, (*iv*) Support Crew. Based on these requirements, the four steps that follow represent the iterative loop of design-developing the pre-contract design-definition of a warship. During preliminary iterations of this stepped loop, a number of "concepts" will be subjected to the process.

(1) Synthesise an *outline product solution schema* that satisfies the functional / operational requirements. This solution schema is typically represented as diagrammatics of the warship's functional sub-systems and their inter-dependencies in terms of the transfer of energy, information, and material. Figure I.1 summarises the principal functional sub-systems of a warship.

(2) Develop the solution schema to represent a physical outline in terms of the warship's (*i*) *external configuration* (e.g. number of hulls); (*ii*) *full-scale 3-D geometry* (e.g. hullform, principal dimensions, etc.); (*iii*) *internal configuration* and *spatial arrangement* of *ship zones (compartments)* subdivided by decks and bulkheads.

(3) Predict, using analytical techniques, the warship's *intended principal functional / operational attributes* (e.g. endurance, electrical load, weight, etc.)

(4) Synthesise and develop a schema for the structural containment sub-system and each of the warship's "on-board" engineering sub-systems.

(a) The structural containment sub-system is initially represented, as part of the outline product solution schema, in terms of a hull envelope (external configuration) which is

suitably divided by decks and bulkheads represented, in the first instance, as plane surfaces (internal configuration). As more information becomes available, these plane surfaces are clad with an appropriate structural schema composed of interconnected structural objects such as plates, webs and longitudinal stiffeners.

(b) The equipment that makes up each "on-board" engineering sub-system is generally design-developed by specialist equipment suppliers. Thus, each "on-board" sub-system is initially represented as a schema that includes (*i*) a configuration of physical objects (*modules* composed of *equipment* and *local connectives* supported on a *common foundation*); and, (ii) that part of the diagrammatic that represents the long-distance connectives between modules and storage tanks.

The equipments are chosen from the options provided by the supplier and are; (*i*) sized and rated ultimately on the basis of the ship's intended principal functional / operational attributes; and, (*ii*) have their own *sub-system functional / operational attributes* which can be used to revise and update the analysis relating to the product's intended principal functional / operational attributes.

Equipment, local connectives and structural foundations are represented as an integral part of the module to which they belong. The circumscribing *space-envelope*, detailing the spatial limits of the module and the positions of any *input/output interfaces* with long distance connectives, is then defined by it's full-scale geometry and relative location with respect to specific ship zones.

As will be demonstrated in the example that follows, this approach to developing the pre-contract design-definition of a warship is essentially an exercise in the management of space. Traditionally, those parts of the diagrammatics that represent the long-distance connectives between modules and storage tanks are developed as 2-D drawings largely divorced from the internal geometry and spatial arrangement of ship zones. Using the previously described approach, that part of the diagrammatics that represent long-distance connectives can be defined by their relative location with respect to specific ship zones.

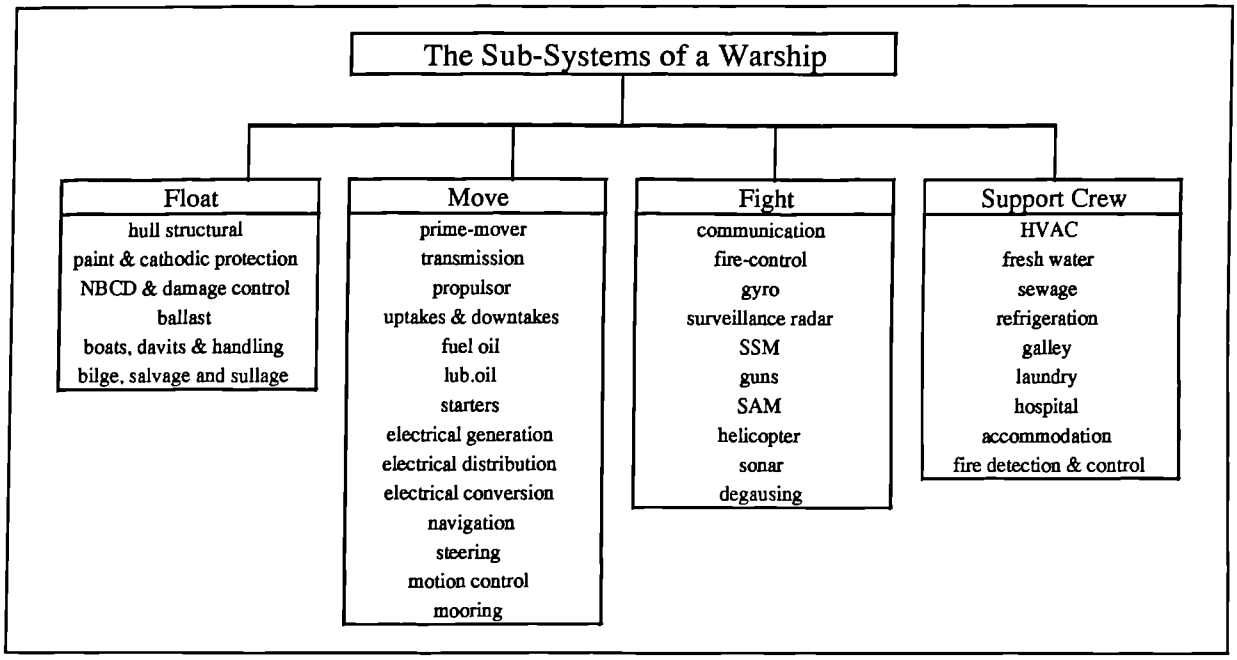


Figure I.1: The principal functional sub-systems of a warship, grouped according to the principal functional / operational requirements

In essence, a *schematic design-definition*, in which all sub-systems are represented in very simple, yet realistic functional and geometrical terms, represents the output of a warship's *pre-contract design-development phase.*

This definition reflects the words of French [French 85], when he states that "the products (outputs) of the conceptual design stages will be called 'schemes' (schema). By a scheme is meant an outline solution to a design problem, carried to a point where the means of performing each major function has been fixed, as have the spatial and structural relationships of the principal components".

Furthermore, this approach to developing a *schematic design-definition* shares some similarities with the "Building Block Approach" developed by Andrews *et al* [Andrews 88, 96, 97] and the work of Brown [Brown 86] with regard to warships.

By way of example, Figures I.2a-e illustrate a schematic design-definition where the emphasis is on establishing the functional architecture of a ship's engine room [Snaith 82].

Each module is represented in very simple, yet realistic terms. This type of representation defines the spatial location and full-scale geometry of each module in terms of a circumscribing space-envelope. These envelopes include:

(i) the outline geometry of each module plus the soft geometry which represents the access and withdrawal space which is needed for repair and maintenance

(ii) the precise geometry of input/output flanges and foundation connections

(iii) the associated non-graphic attributes of each module. E.g. make, part numbers, etc.

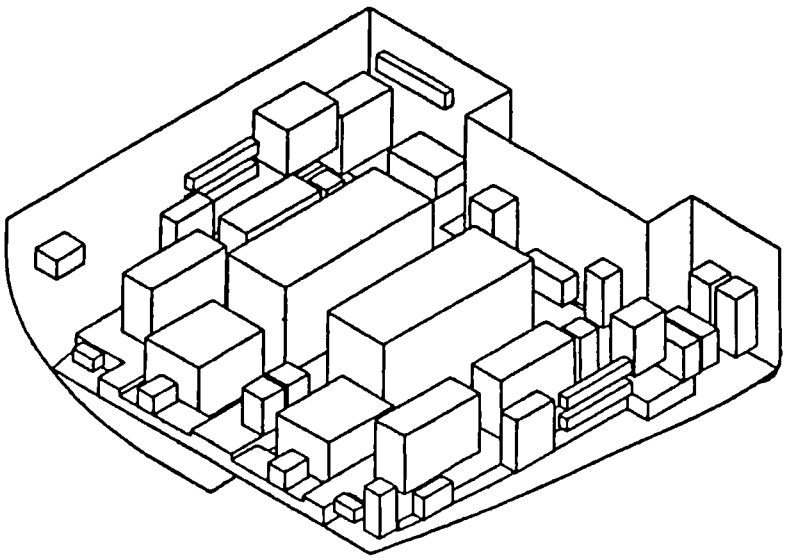


Figure I.2a: The schematic design-definition of a ship's engine room - the position of modules

Space-envelopes represent the spaces reserved for long distance connectives as well as space reserved for module seatings and foundations.

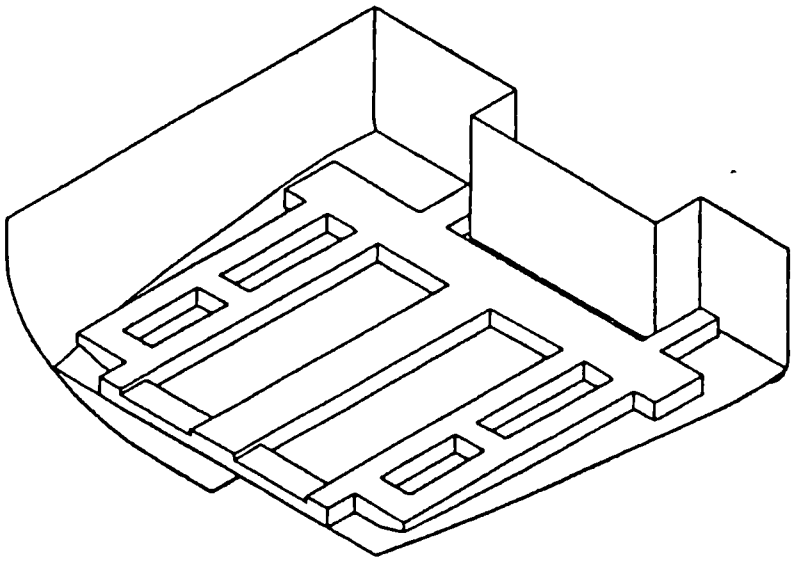


Figure I.2b: The schematic design-definition of a ship's engine room - under-floor reserved spaces

Space envelopes represent the spaces reserved for walkways and general transit space around modules to allow the crew access to all equipment for routine operation, repair and maintenance.

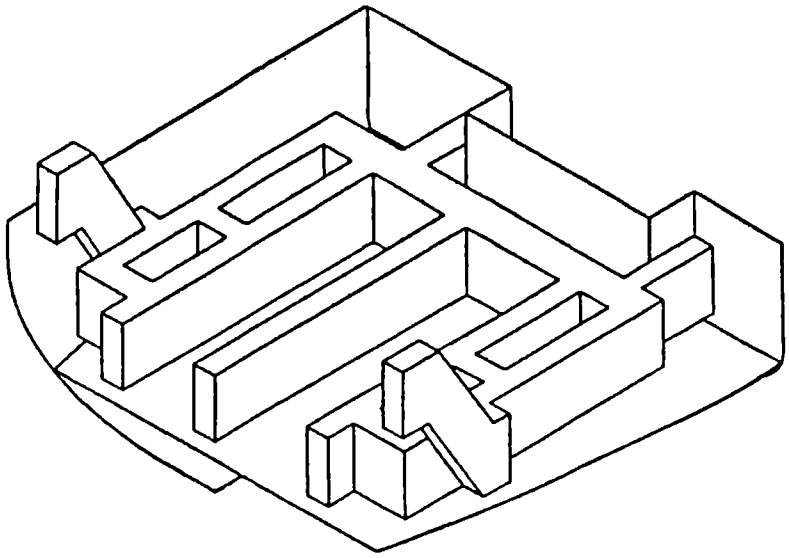


Figure I.2c: The schematic design-definition of a ship's engine room - general transit reserved spaces

Space envelopes represent the spaces reserved for the long distance connectives to be cited on the under-deck.

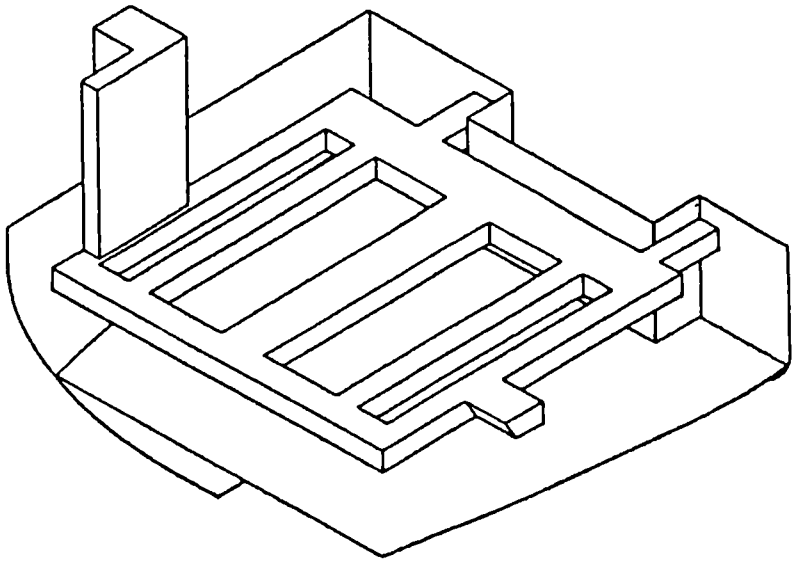


Figure I.2d: The schematic design-definition of a ship's engine room - under-deck reserved spaces.

That part of the diagrammatics that represents long-distance connectives. (Plan view)

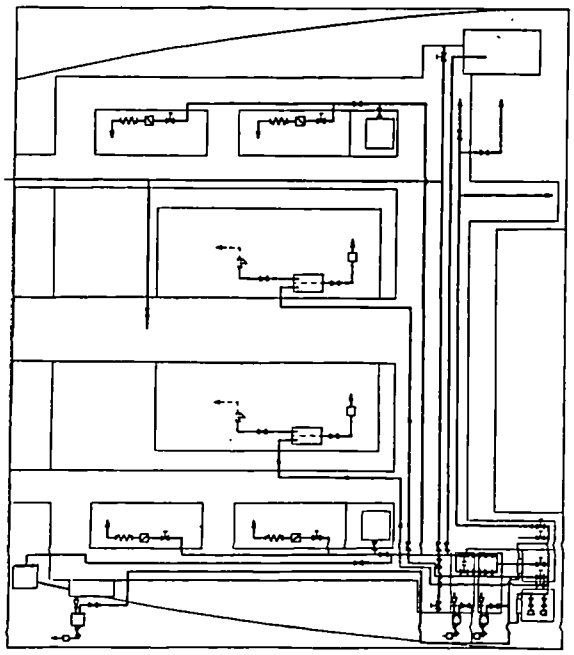


Figure I.2e: The schematic design-definition of a ship's engine room
- *that part of the diagrammatic that represents long-distance connectives.*

This basic process description has now been demonstrated to a large number of engineering companies and, based on comments and feedback, continues to be modified to reflect the views of industry. Whilst it has been explained with specific reference to a warship, it is considered equally applicable to the design-development of other complex engineered products.

II. Review of Contemporary Modelling Techniques

Based on a set of requirements, it is important for a decision-maker to be aware of the advantages and shortcomings of any given modelling technique. By understanding the individual capabilities of a set of techniques, a decision-maker is more likely to select that which is most appropriate by matching capabilities to requirements. The requirements listed below in Table II.1 are summarised from Section 3.5 where the requirements of a strategy for modelling, planning and scheduling a product's design-development phase are first introduced.

Number	Representational Requirements: *The model should be capable of representing*
1	a product design-work breakdown structure of design-development activities
2	the precedence relationships between activities
3	the data-dependencies between activities
4	the mutual inter-dependence of activities on each other for data
	Analytical Requirements: *The model should be*
5	normative
6	quantitative
7	dynamic

Table II.1: A summary of the requirements of a model of the design-development phase of a product

The origin of modelling systems and their life-cycle phases dates back to the mid 1970s. One of the earliest examples of such is the *directed graph* [Warfield 73] which, is based on a set of nodes inter-connected by arcs. In the context of the design-development phase of a product, the nodes represent design-development activities whilst the arcs represent directed data-flow between activities. Figure II.1 illustrates a directed graph.

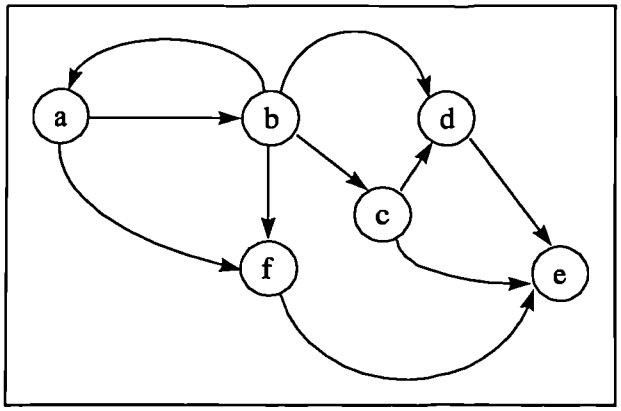


Figure II.1: A directed graph.

Whilst a directed graph represents activities and their data-dependencies, it does not explicitly represent precedence relationships between activities. (*See* Section 3.4 for an explanation of the difference between data-dependencies and precedence relationships). Furthermore, as the number of activities increase, the directed graph becomes cluttered and increasingly difficult to interpret. At the same time, a directed graph cannot satisfy any of the analytical requirements listed above.

DeMarco is credited with the development of the *structured analysis* methodology for creating data-flow models [DeMarco 79]. The *data-flow diagram* (*DFD*) represents data-flow between activities at varying levels of detail using a hierarchy. Using this more structured approach to creating a representation helps to ensure that it is easier to establish and interpret models of activities and their data-dependencies. However, as with the directed graph, data-flow diagrams do not totally capture precedence relationships between activities, nor do they satisfy any of the analytical requirements listed above.

Whilst variations and extensions of the *structured analysis* methodology have been proposed in the literature [Page-Jones 80, Ward 85, Hatley 87], it is the *structured analysis & design technique (SADT)* [Ross 77] and it's later incarnation, the *integrated computer aided manufacturing (ICAM) definition (IDEF)* language [Ross 85] that have been most widely used for the representation of systems and their life-cycle phases.

The IDEF language comprises a number of models. The most common, $IDEF_0$, is made up of (*i*) diagrams; (*ii*) explanatory text; and, (*iii*) a glossary; all cross-referenced to each other. In the diagram, which is the focus of the $IDEF_0$ model, a box, assigned an active verb phrase, represents a function. Each diagram represents between 2 and 6 *function boxes* described within a hierarchy. The function boxes are linked together by *inputs*, *outputs*, *controls*, and *mechanisms*.

Inputs (I) enter the function box from the left, are processed by the function, and exit the function box from the right as outputs (O). A control (C) enters the function box from the top and influences how the function is to be performed. A mechanism (M) enters the box from the bottom and is a tool or resource used to perform the function. Whilst Figure II.2 merely shows the logic of an $IDEF_0$ diagram, the complete modelling technique is detailed by Colquhoun *et al* [Colquhoun 93].

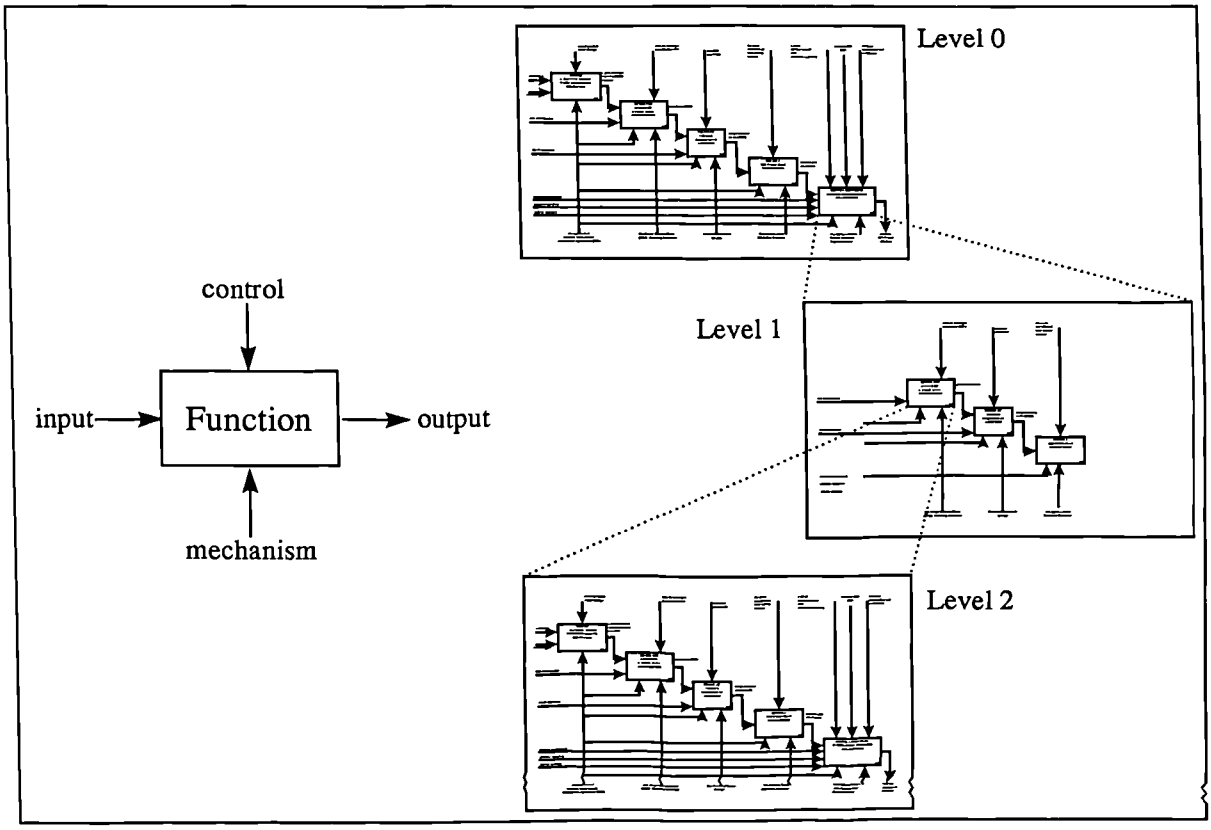


Figure II.2: The logic of an IDEF$_0$ diagram.

Additional IDEF models have been developed. IDEF$_1$ can be used for information analysis; IDEF$_2$ for dynamic analysis; and, IDEF$_3$ for process modelling. In terms of the representational requirements listed above, IDEF$_0$ can be used to model activities and their data-dependencies, but like those techniques mentioned previously, IDEF$_0$ does not adequately model precedence relationships between activities.

IDEF$_3$ overcomes the representational failings of IDEF$_0$ and, in fact, satisfies all of the representational requirements listed above. However, because of the mere qualitative nature of the IDEF modelling technique, it is difficult to apply mathematical analysis and optimisation techniques to an IDEF$_3$ model. In this respect, the IDEF modelling technique cannot satisfy any of the analytical requirements listed above.

Perhaps the most popular models currently used to represent the life-cycle phases of a product system are the traditional *activity network diagrams* which form the basis of planning analysis techniques such as *critical path analysis (CPA)* and the *programme evaluation and review technique (PERT)*. However, whilst an activity network diagram represents activities and their precedence relationships, it does not represent data-dependencies. Even modified techniques

based on network diagrams such as the *graphical evaluation and review technique (GERT)* [Moore 76] fail to meet all of the representational requirements listed above: whilst such techniques can represent iteration, it is based on a two-way precedence relationship rather than a mutual data inter-dependency. However, whilst these network diagram-based techniques fail some of the representational requirements, they each satisfy all of the analytical requirements listed above.

A more recent technique, developed specifically for modelling the design development phase of a product is the *design structure matrix (DSM) system* [Steward 81a]. This matrix-based model, in it's most basic form prior to analysis is known as the *precedence matrix* (*See* Figure II.3). Based on a product design-work breakdown structure, activities are listed, in *a sequence of execution*, down the left-hand side of the matrix and, in the same sequence, along the top of the matrix.

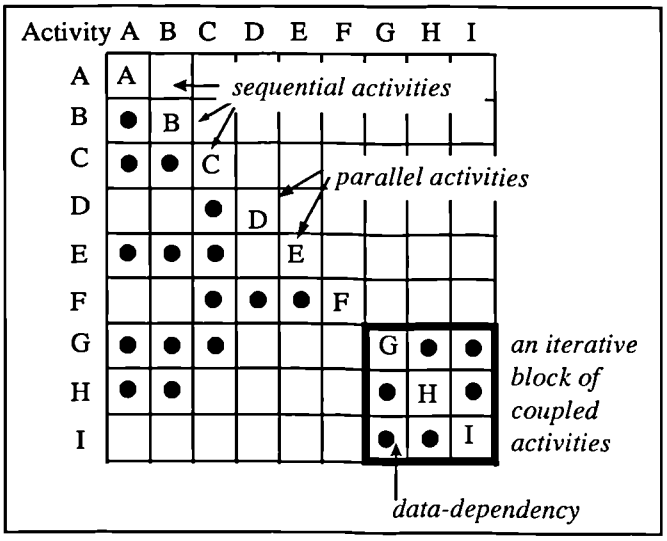


Figure II.3: A Precedence Matrix

The marks in the main body of the matrix represent data-dependencies between activities. *Reading across a row* reveals all of the activities whose data-output is depended upon to perform the activity corresponding to the row. For example, reading across row C in Figure II.3, it can be seen that Activity C depends on data-output from Activities A & B. *Reading down a column* reveals all of the activities which depend on data-output from the activity corresponding to the column. For example, reading down column C in Figure II.3, it can be seen that Activities D, E, F & G all depend on data-output from Activity C.

The relative positioning of the marks implies one of three types of *precedence relationship* between activities; (*i*) sequential; (*ii*) parallel; and, (*iii*) coupled.

Sequential activities

Consider Activities A, B and C in Figure II.3. Activity B depends on data from Activity A, as indicated by the *dependency-mark* in Row B, Column A. Since Activity B depends on data from Activity A, it cannot be started until Activity A has at least started. In the same way, Activity C depends on data from Activities A and B. Therefore Activities A, B and C can only be processed sequentially.

Parallel activities

Consider Activities D and E in Figure II.3. Activity E does not depend on data from Activity D, as indicated by the absence of a dependency-mark in Row E, Column D. Therefore Activities D and E can be processed in parallel, resource availability permitting.

Coupled activities

Consider Activities G, H and I in Figure II.3. All three activities mutually depend on each other for data. The dependency-marks above the leading-diagonal represent up-stream data flows. For example, Activity G depends on data from Activities H and I, however since the sequence of execution dictates that Activity G precedes Activities H and I, in the first instance, Activity G has to be processed using guesstimates of the actual data that will be derived subsequently through the processing of Activities H and I. In the same way, Activity H has to be processed using a guesstimate of the data that will be derived subsequently through the processing of Activity I. After all three activities have been processed for the first time, they may have to be repeated if the initial guesstimates, when compared against the derived data, are found to be insufficiently accurate. In summary, any mark above the leading-diagonal *implies* iteration.

In this respect, the DSM system satisfies all of the representational requirements listed above, but what about the analytical requirements? Early published research that makes use of the DSM system uses a binary representation of data-dependency [Steward 81b, Kusiak 90b]. In this case, dependency-marks in the matrix merely identify the existence of data-dependencies between activities. However, according to Gebala and Eppinger [Gebala 91], "...one shortcoming of the binary matrix representation is the assumption that all data-dependencies

are of equal importance. No attempt is made to differentiate between the amount and importance of data transfer within the matrix..." As they go on to say, "...it is reasonable to expect that certain dependencies will be stronger than others, or that certain data will be critical..."

For example, if an activity depends on data from another activity but that data is known to lie within predictable limits, then it may be possible to begin the dependant activity with a guesstimate of the required data. In such cases, the dependency between the two activities would be classed as weak. In the same way, if the data could not be guesstimated with any certainty but had little impact upon the dependant activity, then again, the dependency would be classed as weak. However, if the data could not be guesstimated with any certainty, and at the same time had a major impact upon the dependant activity, then the dependency would be classed as strong.

Therefore, before assigning a strength of data-dependence between any two activities, an understanding of the *impact* of data transfer, and the *degree of acceptable estimation* that can be associated with a given data transfer, must be considered. Eppinger *et al* [Eppinger 94] have developed a simple four-level dependency scheme which quantifies the dependence of data between activities. (*See* Table II.2)

High	Data is required to start the activity
Average	Data is required to finish the activity
Low	Data is required to check result compatibility
Zero	No data is required

Table II.2: Eppinger et al's four-level dependency scheme [Eppinger 94]

Rogers, as part of the design manager's aid for intelligent decomposition (DeMAID) system [Rogers 92, Rogers 96a], has developed an eight-level data-dependency scheme [Rogers 96b]. The eight levels are extremely strong, very strong, strong, nominal, weak, very weak, extremely weak, and of course, zero. The system user can supply these strengths, or alternatively they can be determined through sensitivity analysis [Bloebaum 92]. Austin *et al* [Austin 96], in their application of the DSM system to civil-engineering design-work, have developed an approach similar to Eppinger and have again developed a simple four-level dependency scheme. (*See* Table II.3)

Class A	It is essential to an activity that Class A data is available before it's commencement
Class B	It is not essential to an activity that Class B data is available before it's commencement, but it is preferable
Class C	It is not essential to an activity that Class C data is available before it's commencement
Zero	No data is required

Table II.3: Austin et al's four-level dependency scheme [Austin 96]

These dependency schemes have evolved through experience. Prior to the creation of the four-level dependency scheme of Table II.2, Krishnan and Eppinger [Krishnan 90] advocated the assignment of an *importance ratio* in terms of a percentage (0-100%); the larger the value, the stronger the dependence. However, when faced with an almost infinite choice of dependency strength, populating the matrix with dependencies becomes very subjective, such that different members of the PDO tend to populate the matrix with completely different dependency strengths.

The potential disparity and inaccuracies that occur as a result of using an infinite scale of dependency strengths can be overcome by adopting a more pragmatic approach to the problem by classifying data-dependencies into levels such as those represented in Tables II.2 and II.3. Some disparity will still result, however after the precedence matrix has been populated with dependencies by a select group of members from the PDO, all disparities can be resolved during group sessions.

As well as providing a better definition of the design-development phase, different strength data-dependencies empower all subsequent analysis of the matrix. As a result, a four-level dependency scheme, based upon those developed by Eppinger and Austin has been adopted as the basis of the modelling strategy developed as part of this research.

In order to carry out mathematical analysis using the matrix representation, the dependencies, "High", "Average" and "Low" require conversion into numeric values. The choice of numeric values for each of the dependencies requires careful consideration, since their final selection may have an impact on the overall results of any analysis. Sensitivity analysis is required in order to examine the influence of the choice of numeric values. The sensitivity and robustness of a chosen set of dependency values can be demonstrated by scaling all of the chosen values

by a constant factor as well as scaling one or two of the values by differing factors. Sensitivity analysis [Eppinger 94] results in the choice of dependency values, as detailed in Table II.4.

High (1.0)	It is essential to an activity that data is available before it's commencement
Average (0.5)	It is not essential to an activity that data is available before it's commencement but it would be preferable
Low (0.1)	It is not essential to an activity that data is available before it's commencement
Zero (0.0)	No data is required

Table II.4: The final choice of data-dependency values

The representation of data-dependencies as point values enables the matrix-model to be used as a basis for normative and quantitative analysis. This analysis tends to be focused on deriving near optimal sequences of activities (*See* Chapters 5 and 6).

Based on the foregoing text, it would appear that the DSM system is an almost ideal technique for modelling design-development since it satisfies almost all of the requirements listed above. However, whilst the DSM System satisfies the first six requirements, prior research [Smith 94] indicates that the dynamic capabilities of the DSM system are limited.

Having reviewed a number of modelling techniques, Table II.5 summarises the capabilities of each against the lists of requirements summarised in Table II.1 at the beginning of the appendix.

Requirements	directed graph	DFD	$IDEF_0$	$IDEF_3$	Network Diagramming			DSM
					PERT	CPM	GERT	
1 PWBS	✓	✓	✓	✓	✓	✓	✓	✓
2 Precedence relationships				✓	✓	✓	✓	✓
3 Data dependency	✓	✓	✓	✓				✓
4 Data interdependency	✓	✓	✓	✓				✓
5 Normative					✓	✓	✓	✓
6 Quantitative					✓	✓	✓	✓
7 Dynamic					✓	✓	✓	

Table II.5: A summary of the capabilities of a range of modelling techniques

In conclusion, the DSM system has been chosen as the principal modelling technique for the proposed strategy since it satisfies six of the seven requirements. However, because the DSM system has limited dynamic capabilities, the strategy also uses activity network diagrams as part of the last two functions that focus on time-based scheduling issues. It should be noted that activity network diagrams were selected rather than $IDEF_3$ because, whilst the latter also has dynamic capabilities, the former is more widely known and therefore better accepted by industry.

III. The GA-Local Search Results

The following tables present the full results of the tests described in Sub-Sections 4.6.1 and 4.6.2. (The results of Test 5 are detailed in the main body of the thesis in Sub-Section 4.6.2).

Test 1 Results

Max. Gens: 100

Of the 720 experiments, Overall Worst Fitness = 62,988 (Normlsd = 1)
Overall Best Fitness = 39,953 (Normlsd = 0)
Normalised Fitness =(Derived Fitness-Overall Best Fitness) / (Overall Worst-Best)

Population size: 100

Set Number	Gen gap	Pc	Pm	Fitness Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Best Fitness	Normlsd Best Fitness	Average Fitness	Normlsd Average Fitness
1	0.5	0.9	0.50	48,022	46,491	47,949	46,170	49,919	46,170	0.270	47,710	0.337
2			0.10	39,953	39,953	39,953	39,953	39,953	39,953	0.000	39,953	0.000
3			0.01	39,953	39,953	39,953	39,959	41,496	39,953	0.000	40,263	0.013
4			0.00	41,496	41,496	39,953	41,496	39,953	39,953	0.000	40,879	0.040
5		0.6	0.50	50,236	43,688	50,147	45,150	53,172	43,688	0.162	48,479	0.370
6			0.10	40,572	39,953	39,953	39,953	39,953	39,953	0.000	40,077	0.005
7			0.01	39,953	39,982	39,953	39,953	39,953	39,953	0.000	39,959	0.000
8			0.00	41,496	39,953	47,261	43,109	41,496	39,953	0.000	42,663	0.118
9		0.3	0.50	50,250	48,184	42,796	47,261	46,876	42,796	0.123	47,073	0.309
10			0.10	39,953	39,959	39,953	39,953	39,953	39,953	0.000	39,954	0.000
11			0.01	41,789	39,953	39,953	43,109	45,665	39,953	0.000	42,094	0.093
12			0.00	41,496	41,496	42,796	39,953	39,953	39,953	0.000	41,139	0.051
13		0.0	0.50	50,250	48,184	42,796	50,250	46,876	42,796	0.123	47,671	0.335
14			0.10	39,953	39,959	41,496	39,953	39,953	39,953	0.000	40,263	0.013
15			0 01	39,953	39,982	39,953	39,953	39,953	39,953	0.000	39,959	0.000
16			0.00	47,261	47,261	47,261	43,109	41,496	41,496	0.067	45,278	0.231
17	0.7	0.9	0 50	49,385	48,513	45,944	51,809	49,872	45,944	0.260	49,104	0.397
18			0.10	39,953	39,959	39,953	39,953	39,953	39,953	0.000	39,954	0.000
19			0 01	39,959	40,916	39,953	39,982	39,953	39,953	0.000	40,153	0.009
20			0 00	41,496	39,953	39,953	41,496	41,496	39,953	0.000	40,879	0.040
21		0.6	0.50	44,024	41,789	41,455	49,415	43,009	41,455	0.065	43,938	0.173
22			0.10	41,789	39,953	39,953	39,953	39,953	39,953	0.000	40,320	0.016
23			0.01	41,486	39,953	39,953	39,953	41,789	39,953	0.000	40,627	0.029
24			0 00	39,953	39,953	39,953	39,953	41,496	39,953	0.000	40,262	0.013
25		0.3	0 50	45,402	41,884	47,617	48,003	43,856	41,884	0.084	45,352	0.234
26			0.10	41,789	41,268	41,183	40,916	40,916	40,916	0.042	41,214	0.055
27			0.01	42,192	39,953	41,789	39,959	39,989	39,953	0.000	40,776	0.036
28			0.00	39,953	41,496	41,496	41,496	39,953	39,953	0 000	40,879	0.040
29		0.0	0.50	45,402	41,884	47,617	48,003	47,617	41,884	0.084	46,105	0.267
30			0.10	41,789	41,268	41,183	40,916	40,916	40,916	0.042	41,214	0.055
31			0.01	41,486	41,486	39,953	39,953	41,486	39,953	0.000	40,873	0.040
32			0.00	45,402	45,402	47,617	41,789	45,402	41,789	0.080	45,122	0.224
33	0.9	0.9	0 50	53,433	46,916	45,992	49,352	48,073	45,992	0 262	48,753	0.382
34			0.10	39,953	39,953	39,959	39,953	39,982	39,953	0.000	39,960	0.000
35			0.01	39,953	39,953	39,953	40,916	39,953	39,953	0.000	40,146	0.008
36			0.00	41,496	39,953	39,953	39,953	41,496	39,953	0.000	40,570	0.027
37		0.6	0.50	47,107	43,353	42,746	48,554	45,554	42,746	0.121	45,463	0.239
38			0 10	39,953	41,781	39,959	39,953	39,959	39,953	0.000	40,321	0.016
39			0.01	39,953	41,789	39,982	39,953	39,959	39,953	0.000	40,327	0.016
40			0.00	41,496	41,496	41,496	41,496	39,953	39,953	0.000	41,187	0.054
41		0.3	0 50	43,328	45,072	44,402	50,504	44,026	43,328	0.147	45,466	0.239
42			0.10	39,982	39,953	39,953	39,953	40,863	39,953	0.000	40,141	0.008
43			0.01	41,486	40,863	39,982	39,989	44,045	39,982	0.001	41,273	0.057
44			0.00	41,496	41,496	44,402	39,953	41,496	39,953	0.000	41,769	0.079
45		0.0	0.50	47,107	47,107	42,746	48,554	45,554	42,746	0.121	46,213	0 272
46			0.10	42,746	41,781	39,959	39,953	39,959	39,953	0.000	40,879	0.040
47			0.01	41,486	41,486	39,982	39,989	44,045	39,982	0.001	41,398	0.063
48			0.00	44,402	47,107	44,402	47,107	41,496	41,496	0.067	44,903	0.215

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after 100% Max Generations Fitness	100% Normlsd	75% Max Generations Fitness	75% Normlsd	50% Max Generations Fitness	50% Normlsd	25% Max Generations Fitness	25% Normlsd
2	1	39,953	39,953	0.000	39,953	0 000	39,953	0.000	39,953	0.000
27	1	42,192	39,953	0.000	39,953	0.000	39,953	0.000	39,953	0.000
25	1	45,402	39,953	0.000	39,953	0.000	39,953	0.000	39,953	0.000
33	5	48,073	39,953	0.000	39,953	0.000	39,953	0.000	39,953	0.000
33	1	53,433	39,953	0.000	39,953	0.000	39,953	0.000	39,953	0.000

Population size:	50

Set Number	*Gen gap*	*Pc*	*Pm*	*Fitness*					*Best Fitness*	*Normlsd Best Fitness*	*Average Fitness*	*Normlsd Average Fitness*
				Trial 1	*Trial 2*	*Trial 3*	*Trial 4*	*Trial 5*				
49	0.5	0.9	0.50	50,565	59,143	50,909	52,354	42,662	42,662	0.118	51,127	0.485
50			0.10	44,026	39,953	41,488	39,982	39,982	39,953	0.000	41,086	0.049
51			0.01	39,953	39,953	39,953	40,539	39,959	39,953	0.000	40,071	0.005
52			0.00	40,539	41,488	39,953	39,953	42,662	39,953	0.000	40,919	0.042
53		0.6	0.50	50,236	55,922	49,395	53,341	50,527	49,395	0.410	51,884	0.518
54			0.10	39,959	41,235	40,863	39,959	39,953	39,953	0.000	40,394	0.019
55			0.01	41,268	39,953	43,782	39,982	39,982	39,953	0.000	40,993	0.045
56			0.00	41,488	42,662	39,953	41,488	42,662	39,953	0.000	41,651	0.074
57		0.3	0.50	51,114	44,309	43,955	44,069	45,049	43,955	0.174	45,699	0.249
58			0.10	39,953	40,947	40,867	39,982	41,789	39,953	0.000	40,708	0.033
59			0.01	43,109	43,856	46,564	43,809	53,571	43,109	0.137	46,182	0.270
60			0.00	41,488	42,662	39,953	41,488	42,662	39,953	0.000	41,651	0.074
61		0.0	0.50	51,114	44,309	43,955	44,069	45,049	43,955	0.174	45,699	0.249
62			0.10	39,953	40,947	40,867	39,982	41,789	39,953	0.000	40,708	0.033
63			0.01	41,..68	39,953	43,782	39,982	39,982	39,953	0.000	40,993	0.045
64			0.00	49,395	42,662	44,069	44,069	42,662	42,662	0.118	44,571	0.200
65	0.7	0.9	0.50	47,227	43,930	52,877	46,491	57,004	43,930	0.173	49,506	0.415
66			0.10	39,953	41,235	41,789	39,953	39,959	39,953	0.000	40,578	0.027
67			0.01	41,886	40,916	42,199	43,708	41,789	40,916	0.042	42,100	0.093
68			0.00	41,886	39,953	41,886	41,886	39,959	39,953	0.000	41,114	0.050
69		0.6	0.50	48,769	45,631	50,761	51,467	45,232	45,232	0.229	48,372	0.365
70			0.10	39,959	39,953	41,884	41,789	39,982	39,953	0.000	40,713	0.033
71			0.01	43,061	45,665	40,863	42,093	42,219	40,863	0.040	42,780	0.123
72			0.00	39,953	39,953	41,866	41,488	39,959	39,953	0.000	40,644	0.030
73		0.3	0.50	51,627	49,704	46,494	45,560	45,113	45,113	0.224	47,700	0.336
74			0.10	40,947	39,953	39,959	41,488	39,959	39,953	0.000	40,461	0.022
75			0.01	41,235	40,867	39,989	43,817	39,982	39,982	0.001	41,178	0.053
76			0.00	40,867	41,235	43,817	41,488	39,982	39,982	0.001	41,478	0.066
77		0.0	0.50	48,769	45,631	50,761	51,467	45,232	45,232	0.229	48,372	0.365
78			0.10	40,947	39,953	39,959	41,488	39,959	39,953	0.000	40,461	0.022
79			0.01	41,235	40,867	39,989	43,817	39,982	39,982	0.001	41,178	0.053
80			0.00	41,488	42,662	43,817	45,631	42,662	41,488	0.067	43,252	0.143
81	0.9	0.9	0.50	54,746	50,715	48,376	53,540	50,523	48,376	0.366	51,580	0.505
82			0.10	39,953	39,989	43,072	39,982	39,953	39,953	0.000	40,590	0.028
83			0.01	44,026	39,959	42,363	39,953	44,025	39,953	0.000	42,065	0.092
84			0.00	39,959	41,488	39,959	39,953	39,953	39,953	0.000	40,262	0.013
85		0.6	0.50	49,284	48,303	43,147	46,721	42,596	42,596	0.115	46,010	0.263
86			0.10	41,268	41,751	39,989	39,959	43,072	39,959	0.000	41,207	0.054
87			0.01	44,977	39,953	39,959	45,974	46,073	39,953	0.000	43,387	0.149
88			0.00	39,959	39,959	41,488	41,488	41,488	39,959	0.000	40,876	0.040
89		0.3	0.50	52,507	43,003	45,079	47,922	48,516	43,003	0.132	47,405	0.324
90			0.10	40,947	39,953	42,701	42,277	45,974	39,953	0.000	42,370	0.105
91			0.01	54,709	41,789	39,959	42,823	43,782	39,959	0.000	44,612	0.202
92			0.00	39,959	41,488	39,959	39,953	39,953	39,953	0.000	40,262	0.013
93		0.0	0.50	49,284	48,303	43,147	46,721	42,596	42,596	0.115	46,010	0.263
94			0.10	40,947	39,953	42,701	42,277	45,974	39,953	0.000	42,370	0.105
95			0.01	54,709	41,789	39,959	42,823	43,782	39,959	0.000	44,612	0.202
96			0.00	45,631	45,631	41,488	42,277	41,488	41,488	0.067	43,303	0.145

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after 100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
51	1	39,953	39,953	0.000	39,953	0.000	39,953	0.000	39,953	0.000
83	3	42,363	39,953	0.000	39,953	0.000	39,953	0.000	39,953	0.000
57	5	45,049	39,953	0.000	39,953	0.000	39,953	0.000	39,953	0.000
73	2	49,704	39,953	0.000	39,953	0.000	39,953	0.000	39,953	0.000
49	2	59,143	39,953	0.000	39,953	0.000	39,953	0.000	45,665	0.248

Population size:	25

Set Number	Gen gap	Pc	Pm	Fitness					Best Fitness	Normlsd Best Fitness	Average Fitness	Normlsd Average Fitness
				Trial 1	Trial 2	Trial 3	Trial 4	Trial 5				
97	0.5	0.9	0.50	61,974	52,195	55,984	50,131	54,365	50,131	0.442	54,930	0.650
98			0.10	43,448	41,820	41,235	43,817	43,109	41,235	0.056	42,686	0.119
99			0.01	42,825	54,060	41,820	50,527	49,761	41,820	0.081	47,799	0.341
100			0.00	42,825	54,060	41,235	43,817	43,109	41,235	0.056	45,009	0.220
101		0.6	0.50	53,061	54,017	54,624	50,748	47,209	47,209	0.315	51,932	0.520
102			0.10	43,833	44,328	39,982	41,789	43,735	39,982	0.001	42,733	0.121
103			0.01	41,781	44,390	51,099	51,099	41,823	41,781	0.079	46,038	0.264
104			0.00	41,781	44,390	51,099	41,820	41,781	41,781	0.079	44,174	0.183
105		0.3	0.50	53,699	56,803	56,803	48,067	51,739	48,067	0.352	53,422	0.585
106			0.10	43,092	40,565	43,763	41,789	41,289	40,565	0.027	42,100	0.093
107			0.01	56,249	55,909	42,701	57,815	42,015	42,015	0.090	50,938	0.477
108			0.00	41,781	44,390	51,099	43,817	43,109	41,781	0.079	44,839	0.212
109		0.0	0.50	53,061	54,017	54,624	50,748	54,017	50,748	0.469	53,293	0.579
110			0.10	43,092	40,565	43,763	54,017	40,565	40,565	0.027	44,401	0.193
111			0.01	56,249	55,909	42,701	40,565	44,390	40,565	0.027	47,963	0.348
112			0.00	53,699	56,803	51,739	54,624	49,761	49,761	0.426	53,325	0.581
113	0.7	0.9	0.50	42,015	50,935	50,935	52,332	44,539	42,015	0.090	48,151	0.356
114			0.10	44,539	39,953	43,381	39,959	39,953	39,953	0.000	41,557	0.070
115			0.01	40,912	42,015	39,989	39,959	39,959	39,959	0.000	40,567	0.027
116			0.00	42,825	54,060	41,235	43,817	43,109	41,235	0.056	45,009	0.220
117		0.6	0.50	52,580	45,938	46,626	44,490	54,067	44,490	0.197	48,740	0.381
118			0.10	39,989	41,191	45,665	43,031	43,070	39,989	0.002	42,589	0.114
119			0.01	43,070	43,063	41,541	46,585	51,995	41,541	0.069	45,251	0.230
120			0.00	42,825	54,060	41,235	43,817	43,109	41,235	0.056	45,009	0.220
121		0.3	0.50	46,607	49,540	51,495	43,430	53,333	43,430	0.151	48,881	0.388
122			0.10	53,333	39,953	49,392	39,959	48,212	39,953	0.000	46,170	0.270
123			0.01	50,200	57,769	49,166	50,043	45,808	45,808	0.254	50,597	0.462
124			0.00	49,540	41,235	49,392	41,235	41,235	41,235	0.056	44,527	0.199
125		0.0	0.50	53,061	54,017	54,624	50,748	54,017	50,748	0.469	53,293	0.579
126			0.10	43,092	40,565	43,763	54,017	40,565	40,565	0.027	44,401	0.193
127			0.01	56,249	55,909	42,701	40,565	44,390	40,565	0.027	47,963	0.348
128			0.00	51,739	54,624	49,761	54,060	54,624	49,761	0.426	52,962	0.565
129	0.9	0.9	0.50	51,092	54,411	50,247	50,509	55,157	50,247	0.447	52,283	0.535
130			0.10	39,982	43,131	41,235	42,219	39,953	39,953	0.000	41,304	0.059
131			0.01	43,139	47,071	45,549	45,549	43,480	43,139	0.138	44,958	0.217
132			0.00	41,235	43,817	43,109	43,109	41,235	41,235	0.056	42,501	0.111
133		0.6	0.50	45,305	48,240	49,165	41,183	55,922	41,183	0.053	47,963	0.348
134			0.10	40,863	41,191	39,982	43,109	41,886	39,982	0.001	41,406	0.063
135			0.01	45,031	42,521	55,804	56,704	43,927	42,521	0.111	48,797	0.384
136			0.00	41,235	43,109	43,109	43,109	41,235	41,235	0.056	42,359	0.104
137		0.3	0.50	62,988	50,960	44,490	52,190	53,127	44,490	0.197	52,751	0.556
138			0.10	39,959	42,540	42,721	39,953	40,947	39,953	0.000	41,224	0.055
139			0.01	41,751	57,382	40,912	43,955	43,605	40,912	0.042	45,521	0.242
140			0.00	43,817	43,109	43,109	41,751	40,912	40,912	0.042	42,540	0.112
141		0.0	0.50	53,061	54,017	54,624	50,748	54,017	50,748	0.469	53,293	0.579
142			0.10	43,092	40,565	43,763	54,017	40,565	40,565	0.027	44,401	0.193
143			0.01	56,249	55,909	42,701	40,565	44,390	40,565	0.027	47,963	0.348
144			0.00	54,624	49,761	54,624	43,109	53,127	43,109	0.137	51,049	0.482

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after							
			100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
114	2	39,953	39,953	0.000	39,953	0.000	39,953	0.000	39,953	0.000
119	2	43,063	39,953	0.000	39,953	0.000	39,953	0.000	39,953	0.000
119	4	46,585	46,585	0.288	46,585	0.288	45,665	0.248	45,665	0.248
137	4	52,190	39,953	0.000	39,953	0.000	39,953	0.000	39,953	0.000
137	1	62,988	39,953	0.000	45,665	0.248	39,953	0.000	39,953	0.000

		Mean Normalised Best Fitness			Mean Result
		Population Size			
		100	50	25	
Generation Gap	0.5	0.047	0.071	0.163	0.093
	0.7	0.041	0.063	0.117	0.074
	0.9	0.045	0.050	0.113	0.069
Probabilty of Crossover	0.9	0.066	0.058	0.118	0.081
	0.6	0.029	0.066	0.085	0.060
	0.3	0.033	0.056	0.107	0.065
	0.0	0.049	0.064	0.213	0.109
Probabilty of Mutation	0.50	0.152	0.205	0.304	0.220
	0.10	0.007	0.000	0.014	0.007
	0.01	0.000	0.018	0.079	0.032
	0.00	0.018	0.021	0.127	0.055

Test 2 Results

Max. Gens:	150

Of the 720 experiments, Overall Worst Fitness = 816,961 (Normlsd = 1)
Overall Best Fitness = 372,611 (Normlsd = 0)
Normalised Fitness =(Derived Fitness-Overall Best Fitness) / (Overall Worst-Best)

Population size:	200

Set Number	*Gen gap*	*Pc*	*Pm*	*Fitness* Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	*Best Fitness*	*Normlsd Best Fitness*	*Average Fitness*	*Normlsd Average Fitness*
1	0.5	0.9	0.50	615,241	637,560	705,101	718,998	670,981	615,241	0.546	669,576	0.668
2			0.10	578,247	587,672	570,749	630,623	562,439	562,439	0.427	585,946	0.480
3			0.01	422,273	429,206	385,276	385,276	385,276	385,276	0.029	401,461	0.065
4			0.00	426,496	433,498	389,129	389,129	389,129	389,129	0.037	405,476	0.074
5		0.6	0.50	731,187	731,187	683,194	725,659	707,910	683,194	0.699	715,827	0.772
6			0.10	570,984	598,986	539,522	592,360	569,151	539,522	0.376	574,201	0.454
7			0.01	461,455	429,424	382,701	429,248	412,992	382,701	0.023	423,164	0.114
8			0.00	466,070	433,718	386,528	433,540	417,122	386,528	0.031	427,396	0.123
9		0.3	0.50	708,191	752,608	651,365	755,797	748,511	651,365	0.627	723,294	0.789
10			0.10	488,718	491,284	469,063	491,117	442,175	442,175	0.157	476,471	0.234
11			0.01	446,450	402,970	481,097	390,066	446,169	390,066	0.039	433,350	0.137
12			0.00	450,915	407,000	485,908	393,967	450,631	393,967	0.048	437,684	0.146
13		0.0	0.50	738,499	738,499	690,026	732,916	714,989	690,026	0.714	722,986	0.789
14			0.10	576,694	604,976	544,917	598,284	574,843	544,917	0.388	579,943	0.467
15			0.01	466,070	433,718	386,528	433,540	41 122	386,528	0.031	427,396	0.123
16			0.00	470,730	438,055	390,393	437,876	421,293	390,393	0.040	431,670	0.133
17	0.7	0.9	0.50	677,483	767,548	668,077	625,733	753,826	625,733	0 570	698,533	0.733
18			0.10	607,457	557,470	559,566	539,189	614,723	539,189	0.375	575,681	0.457
19			0 01	405,828	402,968	430,362	394,982	377,027	377,027	0.010	402,233	0.067
20			0.00	409,886	406,998	434,666	398,932	380,797	380,797	0.018	406,256	0.076
21		0.6	0 50	639,002	702,802	621,733	717,309	632,018	621,733	0.561	662,573	0.653
22			0.10	466,641	448,858	502,705	494,824	531,605	448,858	0.172	488,927	0.262
23			0.01	429,394	399,403	463,446	421,600	405,744	399,403	0.060	423,917	0.115
24			0.00	433,688	403,397	468,080	425,816	409,801	403,397	0.069	428,157	0.125
25		0.3	0 50	382,679	422,626	441,655	442,746	454,235	382,679	0.023	428,788	0.126
26			0.10	411,713	418,509	408,650	399,878	399,595	399,595	0.061	407,669	0.079
27			0 01	399,595	443,967	558,081	486,882	486,882	399,595	0.061	475,081	0.231
28			0.00	403,591	448,407	563,662	491,751	491,751	403,591	0.070	479,832	0.241
29		0.0	0.50	645,392	709,830	627,950	724,482	638,338	627,950	0.575	669,199	0 667
30			0.10	471,307	453,347	507,732	499,772	536,921	453,347	0.182	493,816	0.273
31			0.01	433,688	403,397	468,080	425,816	409,801	403,397	0.069	428,157	0.125
32			0.00	438,025	407,431	472,761	430,074	413,899	407,431	0.078	432,438	0.135
33	0.9	0.9	0.50	756,818	722,252	722,252	678,800	664,230	664,230	0.656	708,870	0.757
34			0.10	619,023	619,638	598,802	531,530	577,612	531,530	0.358	589,321	0.488
35			0.01	387,037	451,828	429,236	372,874	429,031	372,874	0.001	414,001	0.093
36			0.00	390,907	456,346	433,528	376,603	433,321	376,603	0 009	418,141	0.102
37		0.6	0 50	700,808	690,596	690,596	654,846	654,846	654,846	0.635	678,338	0.688
38			0.10	446,676	433,498	479,321	491,241	491,241	433,498	0.137	468,395	0.216
39			0.01	398,239	398,239	411,477	431,781	377,718	377,718	0 011	403,491	0.069
40			0 00	402,221	402,221	415,592	436,099	381,495	381,495	0.020	407,526	0.079
41		0.3	0.50	679,568	644,715	593,818	617,927	611,763	593,818	0.498	629,558	0.578
42			0.10	385,565	385,565	442,218	373,041	428,101	373,041	0.001	402,898	0.068
43			0.01	393,335	393,335	393,335	427,211	427,211	393,335	0 047	406,885	0 077
44			0.00	397,268	397,268	397,268	431,483	431,483	397,268	0.055	410,954	0.086
45		0.0	0.50	707,816	697,502	697,502	661,394	661,394	661,394	0.650	685,122	0.703
46			0.10	451,143	437,833	484,114	496,153	496,153	437,833	0.147	473,079	0.226
47			0.01	402,221	402,221	415,592	436,099	381,495	381,495	0.020	407,526	0.079
48			0.00	406,244	406,244	419,748	440,460	385,310	385,310	0.029	411,601	0.088

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after 100% Max Generations Fitness	Normlsd	75% Max Generations Fitness	Normlsd	50% Max Generations Fitness	Normlsd	25% Max Generations Fitness	Normlsd
35	4	372,874	372,611	0.000	409,043	0.082	372,611	0.000	372,611	0.000
10	3	469,063	372,611	0.000	372,611	0.000	372,611	0.000	372,611	0.000
6	1	570,984	372,611	0.000	372,611	0.000	372,611	0.000	372,611	0.000
1	5	670,981	372,611	0.000	372,611	0.000	372,611	0.000	372,611	0.000
17	2	767,548	372,611	0.000	372,611	0.000	372,611	0 000	372,611	0.000

Population size:	100

Set Number	Gen gap	Pc	Pm	Fitness					Best Fitness	Normlsd Best Fitness	Average Fitness	Normlsd Average Fitness
				Trial 1	Trial 2	Trial 3	Trial 4	Trial 5				
49	0.5	0.9	0.50	769,872	700,204	747,173	667,198	670,655	667,198	0.663	711,020	0.762
50			0.10	631,327	647,152	669,008	663,939	635,672	631,327	0.582	649,420	0.623
51			0.01	411,390	442,538	418,115	487,190	391,959	391,959	0.044	430,238	0.130
52			0.00	415,504	446,963	422,296	492,062	395,879	395,879	0.052	434,541	0.139
53		0.6	0.50	544,429	699,117	690,112	785,687	722,107	544,429	0.387	688,290	0.710
54			0.10	568,869	616,511	614,562	633,084	574,525	568,869	0.442	601,510	0.515
55			0.01	377,825	377,715	435,731	435,731	450,632	377,715	0.011	415,527	0.097
56			0.00	381,603	381,492	440,088	440,088	455,138	381,492	0.020	419,682	0.106
57		0.3	0.50	775,059	698,233	765,538	730,994	693,763	693,763	0.723	732,717	0.810
58			0.10	483,035	600,807	492,496	523,288	458,501	458,501	0.193	511,625	0.313
59			0.01	439,884	393,331	437,549	492,096	405,179	393,331	0.047	433,608	0.137
60			0.00	444,283	397,264	441,924	497,017	409,231	397,264	0.055	437,944	0.147
61		0.0	0.50	549,873	706,108	697,013	793,544	729,328	549,873	0.399	695,173	0.726
62			0.10	574,558	622,676	620,708	639,415	580,270	574,558	0.454	607,525	0.529
63			0.01	381,603	381,492	440,088	440,088	455,138	381,492	0.020	419,682	0.106
64			0.00	385,419	385,307	444,489	444,489	459,690	385,307	0.029	423,879	0.115
65	0.7	0.9	0.50	712,070	695,414	763,907	735,803	685,811	685,811	0.705	718,601	0.779
66			0.10	631,546	620,210	638,196	545,021	676,288	545,021	0.388	622,252	0.562
67			0.01	396,882	432,002	390,416	388,108	381,766	381,766	0.021	397,835	0.057
68			0 00	400,851	436,322	394,320	391,989	385,584	385,584	0.029	401,813	0.066
69		0.6	0.50	739,955	736,490	694,244	707,784	716,591	694,244	0.724	719,013	0.780
70			0.10	474,605	427,454	399,730	465,096	551,743	399,730	0.061	463,726	0.205
71			0 01	439,250	429,031	387,963	393,042	463,558	387,963	0 035	422,569	0.112
72			0 00	443,643	433,321	391,843	396,972	468,194	391,843	0.043	426,794	0.122
73		0.3	0.50	463,847	441,642	461,492	377,228	442,585	377,228	0.010	437,359	0.146
74			0.10	457,429	419,736	392,573	450,759	395,909	392,573	0.045	423,281	0.114
75			0 01	387,636	388,250	492,911	451,288	451,288	387,636	0 034	434,275	0.139
76			0 00	391,512	392,133	497,840	455,801	455,801	391,512	0 043	438,617	0.149
77		0.0	0.50	747,355	743,855	701,186	714,862	723,757	701,186	0.739	726,203	0.796
78			0.10	479,351	431,729	403,727	469,747	557,260	403,727	0.070	468,363	0.215
79			0.01	443,643	433,321	391,843	396,972	468,194	391,843	0.043	426,794	0.122
80			0.00	448,079	437,655	395,761	400,942	472,876	395,761	0 052	431,062	0.132
81	0.9	0.9	0.50	773,761	735,045	768,450	740,955	694,884	694,884	0.725	742,619	0.833
82			0.10	507,807	586,342	571,907	534,529	597,683	507,807	0.304	559,654	0.421
83			0.01	422,270	440,490	524,592	387,075	431,017	387,075	0.033	441,089	0.154
84			0.00	426,493	444,895	529,838	390,946	435,327	390,946	0.041	445,500	0.164
85		0.6	0.50	758_60	689,131	666,936	708,252	734,528	666,936	0.662	711,421	0.762
86			0.10	517,363	437,862	476,235	471,996	387,253	387,253	0.033	458,142	0.192
87			0.01	413,680	386,807	419,034	417,416	429,481	386,807	0.032	413,284	0.092
88			0.00	417,817	390,675	423,224	421,590	433,776	390,675	0.041	417,416	0.101
89		0.3	0.50	624,514	681,915	699,093	643,272	744,115	624,514	0.567	678,582	0.689
90			0.10	390,575	448,727	387,149	399,660	382,732	382,732	0.023	401,769	0.066
91			0.01	394,789	429,035	454,152	377,179	478,917	377,179	0.010	426,814	0.122
92			0 00	398,737	433,325	458,694	380,951	483,706	380,951	0.019	431,083	0.132
93		0.0	0.50	765,843	696,022	673,605	715,335	741,873	673,605	0.677	718,536	0.778
94			0.10	522,537	442,241	480,997	476,716	391,126	391,126	0.042	462,725	0.203
95			0.01	417,817	390,675	423,224	421,590	433,776	390,675	0.041	417,416	0.101
96			0.00	421,995	394,582	427,457	425,806	438,114	394,582	0.049	421,591	0.110

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after							
			100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
91	4	377,179	372,611	0.000	372,611	0.000	372,611	0.000	372,611	0.000
82	1	507,807	409,043	0.082	372,611	0.000	429,250	0.127	372,611	0.000
85	5	734,528	372,611	0.000	372,611	0.000	372,611	0.000	372,611	0.000
69	1	739,955	430,271	0.130	372,611	0.000	372,611	0.000	372,611	0.000
53	4	785,687	372,611	0.000	372,611	0.000	372,611	0.000	372,611	0.000

Population size:	50

Set Number	Gen gap	Pc	Pm	Fitness					Best Fitness	Normlsd Best Fitness	Average Fitness	Normlsd Average Fitness
				Trial 1	Trial 2	Trial 3	Trial 4	Trial 5				
97	0.5	0.9	0.50	765,752	774,844	693,178	739,584	733,160	693,178	0.721	741,304	0.830
98			0.10	513,056	626,388	626,845	660,220	629,370	513,056	0.316	611,176	0.537
99			0.01	457,503	485,971	532,802	422,132	440,923	422,132	0.111	467,866	0.214
100			0.00	462,078	490,831	538,130	426,353	445,332	426,353	0.121	472,545	0.225
101		0.6	0.50	710,344	717,072	757,307	690,418	727,879	690,418	0.715	720,604	0.783
102			0.10	600,162	576,929	579,750	587,855	508,927	508,927	0.307	570,725	0.446
103			0.01	382,249	498,400	453,175	540,050	423,149	382,249	0.022	459,405	0.195
104			0.00	386,071	503,384	457,707	545,451	427,380	386,071	0.030	463,999	0.206
105		0.3	0.50	653,419	738,382	737,532	746,712	779,409	653,419	0.632	731,091	0.807
106			0.10	507,899	462,809	497,663	412,275	426,722	412,275	0.089	461,474	0.200
107			0.01	426,908	465,540	454,190	379,780	429,283	379,780	0.016	431,140	0.132
108			0.00	431,177	470,195	458,732	383,578	433,576	383,578	0.025	435,452	0.141
109		0.0	0.50	717,447	724,243	764,880	697,322	735,158	697,322	0.731	727,810	0.799
110			0.10	606,164	582,698	585,548	593,734	514,016	514,016	0.318	576,432	0.459
111			0.01	386,071	503,384	457,707	545,451	427,380	386,071	0.030	463,999	0.206
112			0.00	389,932	508,418	462,284	550,905	431,654	389,932	0.039	468,639	0.216
113	0.7	0.9	0.50	816,961	635,523	780,089	762,518	746,371	635,523	0.592	748,292	0.845
114			0.10	564,312	595,799	597,531	605,943	601,083	564,312	0.431	592,934	0.496
115			0.01	385,387	442,607	459,197	427,388	448,587	385,387	0.029	432,633	0.135
116			0.00	389,241	447,033	463,789	431,662	453,073	389,241	0.037	436,960	0.145
117		0.6	0.50	718,161	704,592	711,662	689,252	688,804	688,804	0.712	702,494	0.742
118			0.10	500,553	486,134	475,724	534,715	547,142	475,724	0.232	508,854	0.307
119			0.01	478,977	422,356	450,072	429,484	475,402	422,356	0.112	451,258	0.177
120			0.00	483,767	426,580	454,573	433,779	480,156	426,580	0.121	455,771	0.187
121		0.3	0.50	541,365	478,696	393,157	540,158	494,322	393,157	0.046	489,540	0.263
122			0.10	460,187	431,943	468,674	413,734	387,714	387,714	0.034	432,450	0.135
123			0.01	449,368	462,717	439,026	457,461	408,771	408,771	0.081	443,469	0.159
124			0 00	453,862	467,344	443,416	462,036	412,859	412,859	0.091	447,903	0.169
125		0.0	0 50	7_5,343	711,638	718,779	696,145	695,692	695,692	0.727	709,519	0.758
126			0 10	505,559	490,995	480,481	540,062	552,613	480,481	0.243	513,942	0.318
127			0.01	483,767	426,580	454,573	433,779	480,156	426,580	0.121	455,771	0.187
128			0.00	488,604	430,845	459,118	438,117	484,958	430,845	0.131	460,328	0.197
129	0.9	0.9	0.50	786,784	788,377	770,427	761,311	640,046	640,046	0.602	749,389	0.848
130			0.10	6_3,302	596,520	655,387	586,637	626,908	586,637	0 482	617,751	0.552
131			0 01	459,446	443,387	445,370	457,082	517,240	443,387	0.159	464,505	0.207
132			0.00	464 040	447,821	449,824	461,653	522,412	447,821	0.169	469,150	0.217
133		0.6	0 50	773,379	727,862	784,883	762,433	764,756	727,862	0.799	762,663	0.878
134			0 10	447 046	536,685	611,827	481,418	413,427	413,427	0.092	498,081	0.282
135			0 01	461_07	488,573	588,912	588,912	430,794	430,794	0.131	511,680	0.313
136			0 00	465,819	493,459	594,801	594,801	435,102	435,102	0.141	516,796	0.324
137		0.3	0.50	638,430	778,786	699,716	747,341	735,705	638,430	0 598	719,996	0.782
138			0 10	460 611	418,934	388,620	470,835	453,385	388,620	0.036	438,477	0.148
139			0 01	477,687	439,279	556,336	466,319	418,398	418,398	0.103	471,604	0.223
140			0 00	48_,464	443,672	561,899	470,982	422,582	422,582	0.112	476,320	0.233
141		0.0	0 50	781,113	735,141	792,732	770,057	772,404	735,141	0 816	770,289	0.895
142			0.10	451,516	542,052	617,945	486,232	417,561	417,561	0.101	503,061	0.294
143			0.01	465,819	493,459	594,801	594,801	435,102	435,102	0.141	516,796	0.324
144			0 00	470,477	498,393	600,749	600,749	439,453	439,453	0.150	521,964	0.336

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after							
			100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
107	4	379,780	372,611	0.000	372,611	0.000	372,611	0.000	372,611	0.000
138	4	470,835	372,611	0.000	416,995	0.100	372,611	0.000	372,611	0.000
114	4	605,943	372,611	0.000	372,611	0.000	372,611	0.000	372,611	0.000
133	4	762,433	372,611	0.000	372,611	0.000	372,611	0.000	372,611	0.000
113	1	816,961	372,611	0.000	372,611	0 000	372,611	0.000	372,611	0 000

		Mean Normalised Best Fitness			Mean Result
		Population Size			
		200	100	50	
Generation Gap	0.5	0.263	0.258	0.264	0.262
	0.7	0.185	0.190	0.234	0.203
	0.9	0.205	0.206	0.290	0.233
Probabilty of Crossover	0.9	0.253	0.299	0.314	0.289
	0.6	0.233	0.208	0.284	0.242
	0.3	0.140	0.147	0.155	0.148
	0.0	0.244	0.218	0.296	0.252
Probabilty of Mutation	0.50	0.563	0.582	0.641	0.595
	0.10	0.232	0.220	0.223	0.225
	0.01	0.033	0.031	0.088	0.051
	0.00	0.042	0.039	0.097	0.060

Test 3 Results

Max. Gens: 300

Of the 720 experiments, Overall Worst Fitness = 6,953,810 (Normlsd = 1)
Overall Best Fitness = 3,434,500 (Normlsd = 0)
Normalised Fitness =(Derived Fitness-Overall Best Fitness) / (Overall Worst-Best)

Population size: 300

Set Number	*Gen gap*	*Pc*	*Pm*	*Fitness*					*Best Fitness*	*Normlsd Best Fitness*	*Average Fitness*	*Normlsd Average Fitness*
				Trial 1	*Trial 2*	*Trial 3*	*Trial 4*	*Trial 5*				
1	0.5	0.9	0.50	5,808,520	6,097,030	6,271,740	5,943,300	6,144,260	5,808,520	0.675	6,052,970	0.744
2			0.10	5,979,110	5,659,040	5,569,260	6,056,440	6,194,890	5,569,260	0.607	5,891,748	0.698
3			0.01	5,094,780	5,211,220	4,968,830	5,068,610	5,292,940	4,968,830	0.436	5,127,276	0.481
4			0.00	5,145,728	5,263,332	5,018,518	5,119,296	5,345,869	5,018,518	0.450	5,178,549	0.496
5		0.6	0.50	6,515,710	6,363,370	6,075,340	6,294,610	6,333,980	6,075,340	0.750	6,316,602	0.819
6			0.10	5,683,850	5,964,460	5,719,390	5,562,240	5,697,150	5,562,240	0.605	5,725,418	0.651
7			0.01	3,781,090	3,671,060	3,786,640	3,743,930	3,689,920	3,671,060	0.067	3,734,528	0.085
8			0.00	3,818,901	3,707,771	3,824,506	3,781,369	3,726,819	3,707,771	0.078	3,771,873	0.096
9		0.3	0.50	6,181,900	6,118,160	6,078,810	6,167,880	6,435,600	6,078,810	0.751	6,196,470	0.785
10			0.10	5,451,380	5,352,640	5,588,700	5,636,060	5,319,920	5,319,920	0.536	5,469,740	0.578
11			0.01	3,641,580	3,617,810	3,814,020	3,656,950	3,538,830	3,538,830	0.030	3,653,838	0.062
12			0.00	3,677,996	3,653,988	3,852,160	3,693,520	3,574,218	3,574,218	0.040	3,690,376	0.073
13		0.0	0.50	6,450,553	6,299,736	6,014,587	6,231,664	6,270,640	6,014,587	0.733	6,253,436	0.801
14			0.10	5,627,012	5,904,815	5,662,196	5,506,618	5,640,179	5,506,618	0.589	5,668,164	0.635
15			0.01	3,743,_79	3,634,349	3,748,774	3,706,491	3,653,021	3,634,349	0.057	3,697,183	0.075
16			0.00	3,780,712	3,670,693	3,786,261	3,743,556	3,689,551	3,670,693	0.067	3,734,155	0.085
17	0.7	0.9	0.50	6,094,220	6,154,110	5,863,220	6,277,500	6,122,220	5,863,220	0.690	6,102,254	0.758
18			0.10	5,653,070	5,667,180	5,794,940	5,562,780	5,572,540	5,562,780	0.605	5,650,102	0.630
19			0.01	3,900,630	3,635,960	4,326,830	4,377,150	4,226,630	3,635,960	0.057	4,093,440	0.187
20			0.00	3,939,636	3,672,3_0	4,370,098	4,420,922	4,268,896	3,672,320	0.068	4,134,374	0.199
21		0.6	0.50	6,2_0,310	6,360,390	6,_82,290	6,014,770	6,158,540	6,014,770	0.733	6,217,260	0.791
22			0.10	5,147,900	5,601,300	5,628,690	5,475,650	5,695,170	5,147,900	0.487	5,509,742	0.590
23			0.01	3,762,310	3,662,310	3,890,050	3,765,640	3,742,100	3,662,310	0.065	3,764,482	0.094
24			0.00	3,799,933	3,698,933	3,928,951	3,803,296	3,779,521	3,698,933	0.075	3,802,127	0.104
25		0.3	0.50	6,1_6,640	6,215,440	6,176,950	6,373,640	6,001,970	6,001,970	0.730	6,178,928	0.780
26			0.10	5,008,280	4,874,920	5,364,340	5,517,100	5,325,120	4,874,920	0.409	5,217,952	0.507
27			0.01	3,794,890	3,892,110	3,803,350	3,847,710	3,804,060	3,794,890	0.102	3,828,424	0.112
28			0.00	3,832,839	3,931,031	3,841,384	3,886,187	3,842,101	3,832,839	0.113	3,866,708	0.123
29		0.0	0.50	6,207,607	6,296,786	6,219,467	5,954,622	6,096,955	5,954,622	0.716	6,155,087	0.773
30			0.10	5,096,421	5,545,287	5,572,403	5,420,894	5,638,218	5,096,421	0.472	5,454,645	0.574
31			0.01	3,724,687	3,625,687	3,851,150	3,727,984	3,704,679	3,625,687	0.054	3,726,837	0.083
32			0.00	3,761,934	3,661,944	3,889,661	3,765,263	3,741,726	3,661,944	0.065	3,764,106	0.094
33	0.9	0.9	0.50	5,600,580	6,367,270	6,525,000	5,942,390	6,090,240	5,600,580	0.615	6,105,096	0.759
34			0.10	5,8_0,860	5,758,180	5,802,820	5,780,260	5,890,830	5,758,180	0.660	5,820,590	0.678
35			0.01	3,869,820	4,747,900	3,946,940	4,007,030	4,668,300	3,869,820	0.124	4,247,998	0.231
36			0.00	3,908,518	4,795,379	3,986,409	4,047,100	4,714,983	3,908,518	0.135	4,290,478	0.243
37		0.6	0.50	6,150,850	5,939,240	6,054,700	6,187,860	6,349,050	5,939,240	0.712	6,136,340	0.768
38			0.10	5,659,040	5,605,340	5,381,200	5,719,210	5,485,800	5,381,200	0.553	5,570,118	0.607
39			0.01	3,752,240	3,742,330	3,576,310	3,710,590	3,509,750	3,509,750	0.021	3,658,244	0.064
40			0.00	3,789,762	3,779,753	3,612,073	3,747,696	3,544,848	3,544,848	0.031	3,694,826	0.074
41		0.3	0.50	5,969,070	6,338,080	6,060,540	6,075,840	6,243,650	5,969,070	0.720	6,137,436	0.768
42			0.10	5,080,960	5,383,590	5,119,870	5,076,300	4,972,420	4,972,420	0.437	5,126,628	0.481
43			0.01	3,627,320	3,732,110	3,712,090	3,646,590	3,809,100	3,627,320	0.055	3,705,442	0.077
44			0.00	3,663,593	3,769,431	3,749,211	3,683,056	3,847,191	3,663,593	0.065	3,742,496	0.088
45		0.0	0.50	6,089,342	5,879,848	5,994,153	6,125,981	6,285,560	5,879,848	0.695	6,074,977	0.750
46			0.10	5,602,450	5,549,287	5,327,388	5,662,018	5,430,942	5,327,388	0.538	5,514,417	0.591
47			0.01	3,714,718	3,704,907	3,540,547	3,673,484	3,474,653	3,474,653	0.011	3,621,662	0.053
48			0.00	3,751,865	3,741,956	3,575,952	3,710,219	3,509,399	3,509,399	0.021	3,657,878	0.063

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after 100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
39	5	3,509,750	3,442,220	0.002	3,442,220	0.002	3,442,220	0.002	3,434,500	0.000
19	3	4,326,830	3,485,010	0.014	3,454,880	0.006	3,535,660	0.029	3,434,500	0.000
42	5	4,972,420	3,668,940	0.067	3,668,940	0.067	3,668,940	0.067	3,454,870	0.006
34	4	5,780,260	3,574,450	0.040	3,454,880	0.006	3,454,880	0.006	3,454,880	0.006
33	3	6,525,000	3,574,460	0.040	3,454,870	0.006	3,454,870	0.006	3,454,870	0.006

Population size:	150

Set Number	*Gen gap*	*Pc*	*Pm*	*Fitness*					*Best Fitness*	*Normlsd Best Fitness*	*Average Fitness*	*Normlsd Average Fitness*
				Trial 1	*Trial 2*	*Trial 3*	*Trial 4*	*Trial 5*				
49	0.5	0.9	0.50	6,605,460	6,442,220	6,213,300	6,832,140	6,017,740	6,017,740	0.734	6,422,172	0.849
50			0.10	6,112,300	6,078,800	5,451,160	6,199,280	5,736,250	5,451,160	0.573	5,915,558	0.705
51			0.01	4,683,760	5,342,110	5,357,820	4,982,380	5,045,650	4,683,760	0.355	5,082,344	0.468
52			0.00	4,730,598	5,395,531	5,411,398	5,032,204	5,096,107	4,730,598	0.368	5,133,167	0.483
53		0.6	0.50	6,438,750	6,834,680	5,891,160	6,557,040	6,794,610	5,891,160	0.698	6,503,248	0.872
54			0.10	6,110,910	5,720,450	5,688,760	5,914,350	5,916,230	5,688,760	0.641	5,870,140	0.692
55			0.01	3,610,010	3,750,130	3,721,500	3,793,530	3,768,440	3,610,010	0.050	3,728,722	0.084
56			0.00	3,646,110	3,787,631	3,758,715	3,831,465	3,806,124	3,646,110	0.060	3,766,009	0.094
57		0.3	0.50	6,160,640	6,385,230	6,251,230	6,264,420	6,468,320	6,160,640	0.775	6,305,968	0.816
58			0.10	5,620,620	5,270,550	5,377,110	5,482,680	5,718,320	5,270,550	0.522	5,493,856	0.585
59			0.01	3,788,270	3,782,130	3,847,120	3,928,520	3,829,320	3,782,130	0.099	3,835,072	0.114
60			0.00	3,826,153	3,819,951	3,885,591	3,967,805	3,867,613	3,819,951	0.110	3,873,423	0.125
61		0.0	0.50	6,374,363	6,766,333	5,832,248	6,491,470	6,726,664	5,832,248	0.681	6,438,216	0.853
62			0 10	6,049,801	5,663,246	5,631,872	5,855,207	5,857,068	5,631,872	0 624	5,811,439	0.675
63			0 01	3,573,910	3,712,629	3,684,285	3,755,595	3,730,756	3,573,910	0.040	3,691,435	0.073
64			0.00	3,609,649	3,749,755	3,721,128	3,793,151	3,768,063	3,609,649	0.050	3,728,349	0.083
65	0.7	0.9	0.50	6,375,840	6,133,000	6,480,590	6,116,250	6,418,220	6,116,250	0.762	6,304,780	0.816
66			0 10	5,934,790	5,800,070	5,954,920	5,806,610	6,068,020	5,800,070	0.672	5,912,882	0.704
67			0.01	3,742,890	3,796,190	3,731,900	3,756,520	3,722,010	3,722,010	0.082	3,749,902	0.090
68			0.00	3, 80,319	3,834,152	3,769,219	3,794,085	3,759,230	3,759,230	0.092	3,787,401	0.100
69		0.6	0.50	6,_76,030	6,004,780	6,423,370	5,806,340	6,477,940	5,806,340	0.674	6,197,692	0.785
70			0.10	5,613,070	5,846,830	5,660,070	5,896,260	5,763,010	5,263,010	0.520	5,655,848	0.631
71			0 01	3,741,000	3,662,980	3,655,910	3,705,230	4,000,810	3,655,910	0 063	3,753,186	0.091
72			0 00	3,778,410	3,699,610	3,692,469	3,742,282	4,040,818	3,692,469	0.073	3,790,718	0.101
73		0.3	0.50	6,_58,750	6,327,390	6,380,180	6,038,280	6,664,440	6,038,280	0.740	6,333,808	0.824
74			0.10	5,162,730	5,082,320	5,459,250	5,177,570	5,447,810	5,082,320	0.468	5,265,936	0.520
75			0.01	3,776,130	3,712,370	3,909,060	3,675,240	3,621,550	3,621,550	0.053	3,738,870	0.086
76			0.00	3,813,891	3,749,494	3,948,151	3,711,992	3,657,766	3,657,766	0.063	3,776,259	0.097
77		0.0	0 50	6,213,270	5,944,732	6,359,136	5,748,277	6,413,161	5,748,277	0.657	6,135,715	0.768
78			0.10	5,556,939	5,788,362	5,603,469	5,837,297	5,210,380	5,210,380	0.505	5,599,290	0.615
79			0 01	3,703,590	3,626,350	3,619,351	3,668,178	3,960,802	3,619,351	0.053	3,715,654	0.080
80			0 00	3,740,626	3,662,614	3,655,544	3,704,859	4,000,410	3,655,544	0.063	3,752,811	0.090
81	0.9	0.9	0 50	6,580,660	6,057,980	6,307,670	6,469,320	5,725,150	5,725,150	0.651	6,228,156	0.794
82			0 10	5,793,970	5,657,220	6,140,860	6,072,280	6,141,440	5,657,220	0.632	5,961,154	0.718
83			0 01	3,922,220	3,599,880	3,830,480	3,747,570	3,756,860	3,599,880	0.047	3,771,402	0.096
84			0 00	3,961,442	3,635,879	3,868,785	3,785,046	3,794,429	3,635,879	0.057	3,809,116	0.106
85		0.6	0 50	6,594,900	6,252,240	6,463,730	6,061,680	6,362,000	6,061,680	0.747	6,346,910	0.828
86			0.10	5,355,670	5,497,180	5,683,100	5,445,560	5,304,800	5,304,800	0.531	5,457,262	0.575
87			0.01	3,910,270	3,496,570	3,936,630	3,808,130	3,817,920	3,496,570	0.018	3,793,904	0.102
88			0 00	3,949,373	3,531,536	3,975,996	3,846,211	3,856,099	3,531,536	0 028	3,831,843	0.113
89		0.3	0 50	6,074,590	6,147,580	6,175,880	6,202,320	6,516,880	6,074,590	0.750	6,223,450	0.792
90			0 10	5,391,940	5,220,610	4,877,040	5,510,810	5,182,470	4,877,040	0 410	5,236,574	0.512
91			0 01	3,783,500	3,888,200	3,696,080	3,653,760	3,653,760	3,653,760	0.062	3,735,060	0.085
92			0.00	3,821,335	3,927,082	3,733,041	3,690,298	3,690,298	3,690,298	0.073	3,772,411	0.096
93		0.0	0.50	6,528,951	6,189,718	6,399,093	6,001,063	6,298,380	6,001,063	0.729	6,283,441	0.810
94			0.10	5,302,113	5,442,208	5,626,269	5,391,104	5,251,752	5,251,752	0.516	5,402,689	0.559
95			0.01	3,871,167	3,461,604	3,897,264	3,770,049	3,779,741	3,461,604	0.008	3,755,965	0.091
96			0 00	3,909,879	3,496,220	3,936,236	3,807,749	3,817,538	3,496,220	0.018	3,793,525	0.102

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after 100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
83	2	3,599,880	3,442,220	0.002	3,454,880	0.006	3,454,880	0.006	3,434,530	0.000
51	1	4,683,760	3,454,870	0.006	3,583,820	0.042	3,583,820	0.042	3,583,820	0.042
90	2	5,220,610	3,434,500	0.000	3,434,500	0.000	3,434,500	0.000	3,434,500	0.000
69	2	6,004,780	3,454,880	0.006	3,454,880	0.006	3,454,880	0.006	3,574,460	0.040
49	4	6,832,140	3,568,070	0.038	3,568,070	0.038	3,568,070	0 038	3,442,220	0.002

Population size:	75

Set Number	Gen gap	Pc	Pm	Fitness Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Best Fitness	Normlsd Best Fitness	Average Fitness	Normlsd Average Fitness
97	0.5	0.9	0.50	6,168,870	6,738,890	6,511,860	6,376,900	6,417,140	6,168,870	0.777	6,442,732	0.855
98			0.10	5,991,230	6,155,360	6,143,530	6,466,110	5,708,240	5,708,240	0.646	6,092,894	0.755
99			0.01	3,767,850	3,836,810	4,168,840	4,093,720	3,673,320	3,673,320	0.068	3,908,108	0.135
100			0.00	3,805,529	3,875,178	4,210,528	4,134,657	3,710,053	3,710,053	0.078	3,947,189	0.146
101		0.6	0.50	6,229,570	6,629,780	6,335,290	6,651,770	6,953,810	6,229,570	0.794	6,560,044	0.888
102			0.10	6,048,430	6,245,920	5,473,460	5,987,110	5,625,450	5,473,460	0.579	5,876,074	0.694
103			0.01	3,904,260	3,857,290	3,797,420	3,741,730	4,103,460	3,741,730	0.087	3,880,832	0.127
104			0.00	3,943,303	3,895,863	3,835,394	3,779,147	4,144,495	3,779,147	0.098	3,919,640	0.138
105		0.3	0.50	5,688,880	6,664,400	6,013,400	6,408,970	6,788,430	5,688,880	0.641	6,312,816	0.818
106			0.10	5,964,450	5,471,730	5,993,040	5,647,490	5,579,770	5,471,730	0.579	5,731,296	0.653
107			0.01	3,791,370	3,895,130	3,819,600	4,049,940	3,815,340	3,791,370	0.101	3,874,276	0.125
108			0 00	3,829,284	3,934,081	3,857,796	4,090,439	3,853,493	3,829,284	0.112	3,913,019	0.136
109		0.0	0.50	6,167,274	6,563,482	6,271,937	6,585,252	6,884,272	6,167,274	0.777	6,494,444	0.869
110			0.10	5,987,946	6,183,461	5,418,725	5,927,239	5,569,196	5,418,725	0.564	5,817,313	0.677
111			0.01	3,865,217	3,818,717	3,759,446	3,704,313	4,062,425	3,704,313	0.077	3,842,024	0.116
112			0.00	3,903,870	3,856,904	3,797,040	3,741,356	4,103,050	3,741,356	0.087	3,880,444	0.127
113	0.7	0.9	0.50	6,671,720	6,837,990	6,245,220	6,623,220	6,759,980	6,245,220	0.799	6,627,626	0.907
114			0.10	6,119,230	6,035,260	6,139,190	6,052,590	5,894,810	5,894,810	0.699	6,048,216	0.743
115			0.01	3,956,300	3,856,750	4,002,600	3,975,690	3,794,300	3,794,300	0.102	3,917,128	0.137
116			0.00	3,995,863	3,895,318	4,042,626	4,015,447	3,832,243	3,832,243	0.113	3,956,299	0.148
117		0.6	0.50	6,483,710	6,459,690	6,591,370	6,591,370	6,569,640	6,459,690	0.860	6,539,156	0.882
118			0.10	5,601,800	5,798,560	5,532,500	5,662,300	5,764,290	5,532,500	0.596	5,671,890	0.636
119			0.01	3,758,000	4,011,470	3,713,060	3,959,890	3,695,160	3,695,160	0.074	3,827,516	0.112
120			0.00	3,795,580	4,051,585	3,750,191	3,999,489	3,732,112	3,732,112	0.085	3,865,791	0.123
121		0.3	0.50	6,649,580	6,663,290	6,426,000	6,367,460	6,176,140	6,176,140	0.779	6,456,494	0.859
122			0.10	5,221,020	5,734,800	5,660,610	5,018,710	5,516,930	5,018,710	0.450	5,430,414	0.567
123			0 01	3,839,780	3,900,500	3,897,500	3,747,530	3,946,370	3,747,530	0.089	3,866,336	0.123
124			0 00	3,878,178	3,939,505	3,936,475	3,785,005	3,985,834	3,785,005	0.100	3,904,999	0.134
125		0.0	0.50	6,418,873	6,395,093	6,525,456	6,525,456	6,503,944	6,395,093	0.841	6,473,764	0.864
126			0.10	5,545,782	5,740,574	5,477,175	5,605,677	5,706,647	5,477,175	0.580	5,615,171	0.620
127			0 01	3,720,420	3,971,355	3,675,929	3,920,291	3,658,208	3,658,208	0.064	3,789,241	0.101
128			0 00	3,757,624	4,011,069	3,712,689	3,959,494	3,694,790	3,694,790	0.074	3,827,133	0.112
129	0.9	0.9	0.50	6,122,_80	6,505,550	6,356,750	6,487,060	5,978,700	5,978,700	0.723	6,290,068	0.811
130			0.10	5,716,440	6,177,980	5,674,450	5,873,360	5,873,360	5,674,450	0.636	5,863,118	0.690
131			0.01	3,623,180	3,867,630	3,952,370	3,906,830	3,678,330	3,623,180	0.054	3,805,668	0.105
132			0 00	3,659,412	3,906,306	3,991,894	3,945,898	3,715,113	3,659,412	0.064	3,843,725	0.116
133		0.6	0.50	6,704_80	6,255,120	6,662,850	6,703,190	6,751,540	6,255,120	0.801	6,615,396	0.904
134			0.10	6,067,580	5,813,100	5,739,500	5,868,780	5,647,300	5,647,300	0.629	5,827,252	0.680
135			0.01	3,894_80	3,885,300	3,907,900	3,884,090	3,900,240	3,884,090	0.128	3,894,362	0.131
136			0.00	3,933,223	3,924,153	3,946,979	3,922,931	3,939,242	3,922,931	0.139	3,933,306	0.142
137		0.3	0.50	5,767,360	6,362,650	6,728,610	6,519,710	6,493,790	5,767,360	0 663	6,374,424	0.835
138			0 10	5,263,890	5,035,680	5,742,750	5,552,950	5,195,360	5,035,680	0.455	5,358,126	0.547
139			0.01	3,790,730	4,055,190	3,907,810	4,008,340	3,790,180	3,790,180	0.101	3,910,450	0.135
140			0 00	3,828,637	4,095,742	3,946,888	4,048,423	3,828,082	3,828,082	0.112	3,949,555	0.146
141		0.0	0 50	6,637,237	6,192,569	6,596,222	6,636,158	6,684,025	6,192,569	0.784	6,549,242	0.885
142			0 10	6,006,904	5,754,969	5,682,105	5,810,092	5,590,827	5,590,827	0.613	5,768,979	0.663
143			0 01	3,855,337	3,846,447	3,868,821	3,845,249	3,861,238	3,845,249	0.117	3,855,418	0.120
144			0 00	3,893,891	3,884,911	3,907,509	3,883,702	3,899,850	3,883,702	0.128	3,893,973	0.131

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after 100% Max Generations Fitness	Normlsd	75% Max Generations Fitness	Normlsd	50% Max Generations Fitness	Normlsd	25% Max Generations Fitness	Normlsd
131	1	3,623,180	3,454,880	0.006	3,454,880	0.006	3,454,060	0.006	3,434,500	0.000
115	3	4,002,600	3,746,540	0.089	3,746,540	0.089	3,602,670	0.048	3,696,950	0.075
122	5	5,516,930	3,454,880	0.006	3,454,880	0.006	3,454,880	0.006	3,454,880	0.006
133	2	6,255,120	3,493,340	0.017	3,493,340	0.017	3,493,340	0.017	3,493,340	0.017
102	1	6,048,430	3,454,880	0.006	3,454,800	0.006	3,454,880	0.006	3,434,500	0.000

		Mean Normalised Best Fitness Population Size 300	150	75	Mean Result
Generation Gap	0.5	0.404	0.399	0.379	0.394
	0.7	0.340	0.346	0.394	0.360
	0.9	0.337	0.330	0.384	0.350
Probabilty of Crossover	0.9	0.427	0.419	0.397	0.414
	0.6	0.348	0.342	0.406	0.365
	0.3	0.332	0.344	0.348	0.341
	0.0	0.335	0.329	0.392	0.352
Probabilty of Mutation	0.50	0.710	0.717	0.770	0.732
	0.10	0.541	0.551	0.586	0.559
	0.01	0.090	0.077	0.088	0.085
	0.00	0.101	0.088	0 099	0.096

Test 4 Results

Max. Gens: 550

Of the 720 experiments, Overall Worst Fitness = 14,715,000 (Normlsd = 1)
Overall Best Fitness = 3,483,780 (Normlsd = 0)
Normalised Fitness =(Derived Fitness-Overall Best Fitness) / (Overall Worst-Best)

Population size: 500

Set Number	*Gen gap*	*Pc*	*Pm*	*Fitness*					*Best Fitness*	*Normlsd Best Fitness*	*Average Fitness*	*Normlsd Average Fitness*
				Trial 1	*Trial 2*	*Trial 3*	*Trial 4*	*Trial 5*				
1	0.5	0.9	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
2			0.10	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
3			0.01	13,276,100	13,045,400	12,987,600	13,226,100	13,153,500	12,987,600	0.846	13,137,740	0.860
4			0.00	13,408,861	13,175,854	13,117,476	13,358,361	13,285,035	13,117,476	0.858	13,269,117	0.871
5		0.6	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
6			0.10	14,684,500	14,343,100	14,715,000	14,715,000	14,075,200	14,075,200	0.943	14,506,560	0.981
7			0.01	4,037,700	3,603,280	3,851,030	3,615,720	3,570,030	3,570,030	0.008	3,735,552	0.022
8			0.00	4,078,077	3,639,313	3,889,540	3,651,877	3,605,730	3,605,730	0.011	3,772,908	0.026
9		0.3	0 50	14,715,000	14,715,000	14,298,400	14,715,000	14,715,000	14,298,400	0.963	14,631,680	0.993
10			0.10	11,900,300	11,054,000	11,394,600	10,558,800	10,532,800	10,532,800	0.628	11,088,100	0.677
11			0.01	4,424,180	3,871,930	3,992,740	4,749,160	4,063,190	3,871,930	0.035	4,220,240	0.066
12			0 00	4,468,422	3,910,649	4,032,667	4,796,652	4,103,822	3,910,649	0.038	4,262,442	0.069
13		0.0	0.50	14,567,850	14,567,850	14,567,850	14,567,850	14,567,850	14,567,850	0.987	14,567,850	0.987
14			0.10	14,537,655	14,199,669	14,567,850	14,567,850	13,934,448	13,934,448	0.931	14,361,494	0.969
15			0.01	3,997,3..3	3,567,247	3,812,520	3,579,563	3,534,330	3,534,330	0.005	3,698,196	0.019
16			0 00	4,037,296	3,60..9..0	3,850,645	3,615,358	3,569,673	3,569,673	0 008	3,735,178	0.022
17	0.7	0.9	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
18			0 10	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
19			0.01	7,961,0 0	6,634,050	7,511,310	3,953,720	5,764,190	3,953,720	0 042	6,364,868	0.257
20			0.00	8,040,681	6,700,391	7,586,423	3,993,257	5,821,832	3,993,257	0.045	6,428,517	0.262
21		0.6	0.50	14,715,000	13,850,100	14,715,000	14,715,000	14,715,000	13,850,100	0.923	14,542,020	0.985
22			0.10	12,479,200	11,338,400	12,080,200	12,849,600	11,717,800	11,338,400	0.699	12,093,040	0.767
23			0.01	3,640,110	3,666,480	3,716,850	3,674,670	3,910,850	3,640,110	0.014	3,721,792	0.021
24			0.00	3,676,511	3,703,145	3,754,019	3,711,417	3,949,959	3,676,511	0.017	3,759,010	0.025
25		0.3	0.50	14,715,000	14,715,000	14,502,900	14,715,000	14,715,000	14,502,900	0.981	14,672,580	0.996
26			0.10	5,907,150	5,734,480	4,211,090	3,708,520	5,079,790	3,708,520	0.020	4,928,206	0.129
27			0.01	3,767,120	4,043,650	3,740,210	3,570,640	3,668,640	3,570,640	0.008	3,758,052	0.024
28			0.00	3,804,791	4,084,087	3,777,612	3,606,346	3,705,326	3,606,346	0.011	3,795,633	0.028
29		0.0	0 50	14,567,850	13,711,599	14,567,850	14,567,850	14,567,850	13,711,599	0.911	14,396,600	0.972
30			0 10	12,354,408	11,225,016	11,959,398	12,721,104	11,600,622	11,225,016	0.689	11,972,110	0.756
31			0 01	3,603,709	3,629,815	3,679,682	3,637,923	3,871,742	3,603,709	0.011	3,684,574	0.018
32			0 00	3,639,746	3,666,113	3,716,478	3,674,303	3,910,459	3,639,746	0.014	3,721,420	0.021
33	0.9	0.9	0 50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
34			0.10	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
35			0.01	9,489,640	7,771,880	12,888,200	10,621,000	10,956,300	7,771,880	0.382	10,345,404	0.611
36			0.00	9,584,536	7,849,599	13,017,082	10,727,210	11,065,863	7,849,599	0.389	10,448,858	0.620
37		0.6	0.50	14,715,000	14,604,000	14,715,000	14,715,000	14,715,000	14,604,000	0.990	14,692,800	0.998
38			0.10	12,050,300	10,228,500	11,093,900	11,228,200	10,980,900	10,228,500	0.601	11,116,360	0.680
39			0.01	3,648,320	4,124,750	3,766,560	4,012,950	3,906,470	3,648,320	0.015	3,891,810	0.036
40			0.00	3,684,803	4,165,998	3,804,226	4,053,080	3,945,535	3,684,803	0.018	3,930,728	0.040
41		0.3	0.50	14,154,100	14,715,000	14,345,900	14,715,000	14,715,000	14,154,100	0.950	14,529,000	0.983
42			0.10	3,949,220	3,850,650	3,647,260	3,727,220	3,744,070	3,647,260	0 015	3,783,684	0.027
43			0 01	3,856,520	3,712,550	3,951,340	4,675,270	3,711,810	3,711,810	0.020	3,981,498	0.044
44			0.00	3,895,085	3,749,676	3,990,853	4,722,023	3,748,928	3,748,928	0.024	4,021,313	0.048
45		0.0	0.50	14,567,850	14,457,960	14,567,850	14,567,850	14,567,850	14,457,960	0.977	14,545,872	0.985
46			0.10	11,929,797	10,126,215	10,982,961	11,115,918	10,871,091	10,126,215	0.591	11,005,196	0.670
47			0.01	3,611,837	4,083,503	3,728,894	3,972,821	3,867,405	3,611,837	0.011	3,852,892	0.033
48			0.00	3,647,955	4,124,338	3,766,183	4,012,549	3,906,079	3,647,955	0.015	3,891,421	0.036

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after 100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
7	5	3,570,030	3,483,780	0.000	3,483,780	0.000	3,483,780	0.000	3,508,030	0.002
26	2	5,734,480	3,483,780	0.000	3,483,780	0.000	3,508,030	0.002	3,508,030	0.002
19	1	7,961,070	3,541,060	0.005	3,483,780	0.000	3,483,780	0.000	3,500,500	0.001
35	1	9,489,640	3,507,810	0.002	3,501,510	0.002	3,508,030	0.002	3,546,060	0.006
22	1	12,479,200	3,483,780	0.000	3,500,500	0.001	3,483,780	0.000	3,508,030	0.002

Population size:	250

Set Number	Gen gap	Pc	Pm	Fitness					Best Fitness	Normlsd Best Fitness	Average Fitness	Normlsd Average Fitness
				Trial 1	Trial 2	Trial 3	Trial 4	Trial 5				
49	0.5	0.9	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
50			0.10	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
51			0.01	9,365,270	11,951,200	10,185,400	10,255,700	11,675,500	9,365,270	0.524	10,686,614	0.641
52			0.00	9,458,923	12,070,712	10,287,254	10,358,257	11,792,255	9,458,923	0.532	10,793,480	0.651
53		0.6	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
54			0.10	14,715,000	13,880,100	14,186,000	14,715,000	14,715,000	13,880,100	0.926	14,442,220	0.976
55			0 01	3,612,600	4,112,270	4,013,660	3,782,790	3,826,690	3,612,600	0.011	3,869,602	0.034
56			0.00	3,648,726	4,153,393	4,053,797	3,820,618	3,864,957	3,648,726	0 015	3,908,298	0.038
57		0.3	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
58			0.10	10,552,000	11,518,300	11,128,700	10,612,000	9,886,200	9,886,200	0.570	10,739,440	0.646
59			0.01	3,806,690	4,260,050	3,766,880	3,864,050	3,926,460	3,766,880	0.025	3,924,826	0.039
60			0 00	3,844,757	4,302,651	3,804,549	3,902,691	3,965,725	3,804,549	0.029	3,964,074	0.043
61		0.0	0 50	14,567,850	14,567,850	14,567,850	14,567,850	14,567,850	14,567,850	0.987	14,567,850	0.987
62			0.10	14,567,850	13,741,299	14,044,140	14,567,850	14,567,850	13,741,299	0.913	14,297,798	0.963
63			0.01	3,576,474	4,071,147	3,973,523	3,744,962	3,788,423	3,576,474	0.008	3,830,906	0.031
64			0 00	3,612,239	4,111,859	4,013,259	3,782,412	3,826,307	3,612,239	0 011	3,869,215	0.034
65	0.7	0.9	0 50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
66			0 10	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
67			0 01	3,647,700	4,446,960	3,892,260	6,639,000	3,843,660	3,647,700	0.015	4,493,916	0.090
68			0 00	3,684,177	4,491,430	3,931,183	6,705,390	3,882,097	3,684,177	0.018	4,538,855	0.094
69		0.6	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
70			0.10	12,654,700	12,527,400	12,527,400	12,310,000	12,893,800	12,310,000	0.786	12,582,660	0.810
71			0 01	3,934,830	4,202,790	4,319,720	4,280,410	3 743,040	3,743,040	0.023	4,096,158	0.055
72			0 00	3,974,178	4,..44,818	4,362,917	4,323,214	3, 80,470	3,780,470	0 026	4,137,120	0.058
73		0.3	0 50	14,715,000	14,715,000	14,323,400	14,715,000	14,715,000	14,323,400	0 965	14,636,680	0.993
74			0 10	5,596,120	4,2..3,580	4,080,140	4,648,380	7,333,410	4,080,140	0 053	5,176,326	0.151
75			0 01	3,914,..60	3,996,440	3,887,050	3,777,210	4,193,980	3,777,210	0 026	3,953,788	0.042
76			0 00	3,953,403	4,036 404	3,925,921	3,814,982	4,235,920	3,814,982	0.029	3,993,326	0.045
77		0.0	0 50	14,56 ,850	14,567,850	14,567,850	14,567,850	14,567,850	14,567,850	0 987	14,567,850	0.987
78			0 10	1..528,153	12,40...126	12,402,126	12,186,900	12,764,862	12,186,900	0.775	12,456,833	0.799
79			0 01	3,895 482	4,160,762	4,276,523	4,237,606	3,705,610	3,705,610	0 020	4,055,196	0.051
80			0 00	3,934,437	4,20..370	4,319,288	4,279,982	3,742,666	3,742,666	0.023	4,095,748	0.054
81	0.9	0.9	0 50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
82			0 10	14 715 000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1 000	14,715,000	1.000
83			0 01	4,569,230	4,130,700	4,433,950	4,069,740	4,207,090	4,069,740	0 052	4,282,142	0.071
84			0 00	4,614,922	4,172,007	4,478,290	4,110,437	4,249,161	4,110,437	0 056	4,324,963	0.075
85		0.6	0 50	14,715 000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1 000	14,715,000	1.000
86			0 10	11,980,600	9,789,360	10,110,000	11,271,900	10,276,600	9,789,360	0 561	10,685,692	0.641
87			0 01	4,001,020	3, 61,480	3,957,860	3,799,770	4,105,330	3,761,480	0 025	3,925,092	0 039
88			0 00	4,041,030	3,799,095	3,997,439	3,837,768	4,146,383	3,799,095	0 028	3,964,343	0.043
89		0.3	0 50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
90			0 10	4,015,240	3,707,160	4,378,760	4,649,490	3,960,910	3,707,160	0 020	4,142,312	0.059
91			0 01	4,377,300	4,114,470	4,928,860	4,455,620	3,882,460	3,882,460	0 035	4,351,742	0.077
92			0 00	4,4..1 073	4,155,615	4,978,149	4,500,176	3,921,285	3,921,285	0 039	4,395,259	0 081
93		0.0	0 50	14,56 ,850	14,567,850	14,56 ,850	14,567,850	14,567,850	14,567,850	0 987	14,56 ,850	0.987
94			0 10	11,860,794	9,691,466	10,008,900	11,159,181	10,173,834	9,691,466	0.553	10,578,835	0.632
95			0 01	3,961,010	3,723,865	3,918,281	3,761,772	4,064,277	3,723,865	0.021	3,885,841	0.036
96			0 00	4,000 620	3,761,104	3,957,464	3,799,390	4,104,919	3,761,104	0 025	3,924,699	0 039

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after 100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
55	1	3,612,600	3,546,790	0.006	3,546,790	0.006	3,546,790	0.006	3,508,030	0.002
67	4	6,639,000	3,508,030	0.002	3,546,060	0.006	3,500,500	0.001	3,508,030	0.002
51	1	9,365,270	3,483,780	0 000	3,508,030	0.002	3,483,780	0.000	3,483,780	0.000
51	2	11,951,200	3,483,780	0.000	3,483,780	0.000	3,508,030	0.002	3,483,780	0.000
89	5	14,715,000	3,508,030	0.002	3,508,030	0 002	3,508,030	0 002	3,508,030	0 002

Population size:	125

Set Number	Gen gap	Pc	Pm	Fitness					Best Fitness	Normlsd Best Fitness	Average Fitness	Normlsd Average Fitness
				Trial 1	Trial 2	Trial 3	Trial 4	Trial 5				
97	0.5	0.9	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
98			0.10	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
99			0.01	6,482,930	7,261,970	5,267,740	4,519,180	4,020,080	4,020,080	0.048	5,510,380	0.180
100			0.00	6,547,759	7,334,590	5,320,417	4,564,372	4,060,281	4,060,281	0.051	5,565,484	0.185
101		0.6	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,040,500	14,040,500	0.940	14,580,100	0.988
102			0.10	14,715,000	13,952,000	13,829,800	12,952,500	14,715,000	12,952,500	0.843	14,032,860	0.939
103			0.01	4,038,700	3,984,750	3,797,930	4,226,150	3,969,300	3,797,930	0.028	4,003,366	0.046
104			0.00	4,079,087	4,024,598	3,835,909	4,268,412	4,008,993	3,835,909	0.031	4,043,400	0.050
105		0.3	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
106			0.10	9,967,990	10,261,100	11,245,700	9,714,200	10,282,900	9,714,200	0.555	10,294,378	0 606
107			0.01	4,182,850	4,680,720	4,492,040	4,844,010	4,689,480	4,182,850	0.062	4,577,820	0.097
108			0.00	4,224,679	4,727,527	4,536,960	4,892,450	4,736,375	4,224,679	0.066	4,623,598	0.101
109		0.0	0.50	14,567,850	14,567,850	14,567,850	14,567,850	13,900,095	13,900,095	0.927	14,434,299	0.975
110			0.10	14,567,850	13,812,480	13,691,502	12,822,975	14,567,850	12,822,975	0.832	13,892,531	0.927
111			0.01	3,998,313	3,944,903	3,759,951	4,183,889	3,929,607	3,759,951	0.025	3,963,332	0.043
112			0.00	4,038,296	3,984,352	3,797,550	4,225,727	3,968,903	3,797,550	0.028	4,002,966	0.046
113	0.7	0.9	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
114			0.10	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
115			0.01	5,509,590	3,991,120	4,470,160	4,240,550	4,374,790	3,991,120	0.045	4,517,242	0.092
116			0 00	5,564,686	4,031,031	4,514,862	4,282,956	4,418,538	4,031,031	0.049	4,562,414	0.096
117		0.6	0.50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
118			0.10	6,966,790	11,193,200	11,189,900	12,372,500	10,632,400	6,966,790	0.310	10,470,958	0.622
119			0.01	4,990,380	4,342,070	4,700,340	4,962,420	4,252,020	4,252,020	0.068	4,649,446	0.104
120			0.00	5,040,.84	4,385,491	4,747,343	5,012,044	4,294,540	4,294,540	0.072	4,695,940	0.108
121		0.3	0 50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
122			0.10	3,797,180	5,796,3"0	3,967,650	4,271,520	4,102,270	3,797,180	0 028	4,386,998	0.080
123			0.01	4,056,350	4,590,..00	4,342,840	5,101,840	4,377,350	4,056,350	0.051	4,493,716	0.090
124			0.00	4,096,914	4,636,102	4,386,268	5,152,858	4,421,124	4,096,914	0 055	4,538,653	0.094
125		0.0	0.50	14,567,850	14,567,850	14,567,850	14,567,850	14_67,850	14,567,850	0.987	14,567,850	0.987
126			0.10	6,897,122	11,081,268	11,078,001	12,248,775	10,526,076	6,897,122	0.304	10,366,248	0.613
127			0 01	4,940,476	4,298,649	4,653,337	4,912,796	4,209,500	4,209,500	0.065	4,602,952	0.100
128			0 00	4,989,881	4,341,636	4,699,870	4,961,924	4,251,595	4,251,595	0.068	4,648,981	0.104
129	0.9	0.9	0 50	14,715 000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
130			0.10	14,715,000	14,715,000	14,715,000	13,781,300	14,715,000	13,781,300	0.917	14,528,260	0.983
131			0 01	3,816,790	4,094,5_0	4,047,000	4,309,990	4,217,980	3,816,790	0.030	4,097,256	0.055
132			0 00	3,854,958	4,135,465	4,087,470	4,353,090	4,260,160	3,854,958	0 033	4,138,229	0.058
133		0.6	0 50	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	14,715,000	1.000	14,715,000	1.000
134			0 10	11,868,300	10,355,700	9,587,640	11,112,200	7,242,120	7,242,120	0.335	10,033,192	0.583
135			0.01	4,929 640	4,414,890	4,057,200	5,459,590	4,014,690	4,014,690	0.047	4,575,202	0 097
136			0 00	4,978,936	4,459,039	4,097,772	5,514,186	4,054,837	4,054,837	0.051	4,620,954	0.101
137		0.3	0 50	14 402,000	14,715,000	14,715,000	14,715,000	14,715,000	14,402,000	0 972	14,652,400	0.994
138			0 10	4,199,660	4,196,070	3,959,930	4,512,900	4,008,450	3,959,930	0.042	4,175,402	0.062
139			0 01	5,7-6,450	4,436,340	4,552,690	4,860,030	4,511,320	4,436,340	0 085	4,817,366	0.119
140			0 00	5,783,715	4,480, 03	4,598,217	4,908,630	4,556,433	4,480,703	0.089	4,865,540	0.123
141		0.0	0 50	14_ 67,850	14,567,850	14,567,850	14,567,850	14,567,850	14,567,850	0.987	14,567,850	0.987
142			0 10	11,749,617	10,252,143	9,491,764	11,001,078	7,169,699	7,169,699	0.328	9,932,860	0.574
143			0 01	4,880,344	4,370,741	4,016,628	5,404,994	3,974,543	3,974,543	0.044	4,529,450	0.093
144			0 00	4,929,147	4,414,449	4,056,794	5,459,044	4,014,289	4,014,289	0 047	4,574,744	0.097

Set Number	Trial Number	OSGA Derived Fitness @ Max. Gens	Local-Search procedure invoked after							
			100% Max Generations		75% Max Generations		50% Max Generations		25% Max Generations	
			Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd	Fitness	Normlsd
138	5	4,008,450	3,558,930	0.007	3,558,930	0.007	3,541,060	0.005	3,508,030	0.002
134	5	7,242,100	3,508,030	0.002	3,508,030	0.002	3,483,780	0.000	3,507,810	0.002
106	5	10,282,900	3,483,780	0.000	3,483,780	0.000	3,483,780	0.000	3,500,500	0.001
130	4	13,781,300	3,541,250	0.005	3,541,250	0.005	3,541,250	0.005	3,508,030	0.002
137	1	14,402,000	3,483,780	0.000	3,483,780	0.000	3,483,780	0 000	3,483,780	0 000

		Mean Normalised Best Fitness			Mean Result
		Population Size			
		500	250	125	
Generation Gap	0.5	0.579	0.534	0.465	0.526
	0.7	0.399	0.422	0.381	0.401
	0.9	0.437	0.400	0.375	0.404
Probabilty of Crossover	0.9	0.713	0.600	0.514	0.609
	0.6	0.437	0.450	0.394	0.427
	0.3	0.308	0.316	0.334	0.319
	0.0	0.429	0.443	0.387	0.419
Probabilty of Mutation	0.50	0.973	0.994	0.984	0.984
	0.10	0.676	0.680	0.541	0.632
	0.01	0.116	0.065	0.050	0.077
	0.00	0.121	0.069	0.053	0.081

IV. A Multiple Criteria Genetic Algorithm (MCGA)

Based on the algorithm illustrated in Figure 7.4 of Chapter 7, the following nine step procedure describes how the MCGA, incorporated within the *MCGA Project Scheduler*, derives near optimal schedules based on a set of varied and sometimes conflicting objectives.

Step 1: Create the initial population.

Sub-Section 7.5.2 describes how the *MCGA Project Scheduler* represents a schedule in terms of a string. Based on the input data contained in a *resource capabilities matrix* and *activity-description matrix*, an initial population of strings is randomly generated. This initial population should be spread over enough of the search space to represent as wide a variety of schedules as possible. The diversity so introduced prevents premature convergence to a local optimum. This population of randomly generated strings represents the first generation. The size of this population represented by the number of generated strings is defined by the user and, once set, the population size remains constant throughout the search.

Step 2: Build a schedule from each string using the Project Schedule Builder (PSB), and evaluate the population of strings on all pre-defined criteria.

Sub-Section 7.5.3 describes how the *Project Schedule Builder (PSB)* decodes a string and builds the corresponding schedule. Once decoded into a schedule, each string in the population can be evaluated on each of the pre-defined criteria to derive a fitness corresponding to each criterion. The output of this step is the current population of strings each linked to *a set* of fitnesses.

Step 3: Rank the population of strings using *Pareto sets* and *dominance*.

There are a number of methods for dealing with multiple criteria optimisation problems. The simplest way is to individually weight and then combine the criteria into a single objective function. However, whilst this approach has the advantage of simplicity, it is only viable if the criteria can be expressed in broadly commensurate terms. This is the case in the previously described multiple criteria ActRes problem where the mathematical functions used to evaluate iteration and concurrency are quite similar. However, the case does apply here because the mathematical expressions used to evaluate criteria such as elapsed time and utilisation are quite different. Consequently, the method for dealing with multiple criteria incorporated into the *MCGA Project Scheduler* uses *Pareto sets* and *dominance*.

Here, a two-criteria problem is used to illustrate the concepts of Pareto sets and dominance, however the concepts hold true for problems based on more than two criteria. When searching for optimal solutions in a *constrained search space*, there will be boundaries that distinguish the *feasible solution space* from those *infeasible regions* of the search space where constraints are violated. Figure IV.1 illustrates a two-criteria constrained search space divided in such a manner.

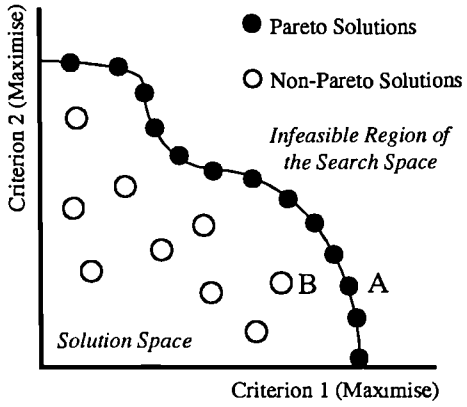


Figure IV.1: The Pareto front.

When faced with multiple criteria and multiple objectives, the aim of a search is to find the solution which optimises each of the criteria to the extent that, in order to improve the solution in terms of any one criterion, a trade-off, in terms of the degradation of one of the other criteria, is necessary (*See*, for example, Solution A in Figure IV.1). Stated another way, a solution which can be improved on in terms of any one criterion, without a degradation in another, does not represent an efficient solution (*See*, for example, Solution B in Figure IV.1); this is because there will always be another feasible solution that will be better.

Therefore, the aim of the search is to find those solutions, referred to as *Pareto solutions*, which lie on the section of the boundary (*Pareto front*) between the feasible solution space and the infeasible region where any improvement in one criteria will lead to a degradation in one or more of the other criteria. Ultimately, once these so-called Pareto solutions have been found they can be returned to the user and, based on preference, the most desirable solution can be chosen.

Whilst it is acknowledged that Pareto fronts exist, the search has no fore-knowledge about where in the search space these fronts are situated and, in fact, the aim of the search is to find

these fronts. Given a population of strings it would be desirable to rank the strings in terms of their *relative proximity* to the Pareto front. This is achieved using the concept known as *dominance*, which is defined as follows. *String* α is dominated by *String* β if every criteria associated with *String* β is equal or superior to the corresponding criteria associated with *String* α.

Based on dominance, *a rank* can be assigned to each string in the population by counting the other strings in the population, which are superior, or equal in terms of *all* criteria. By way of example, because the two criteria are to be maximised, the rank of 6, associated with the string represented by the shaded dot in Figure IV.2, is determined by counting those strings above and to the right of the two dotted lines.

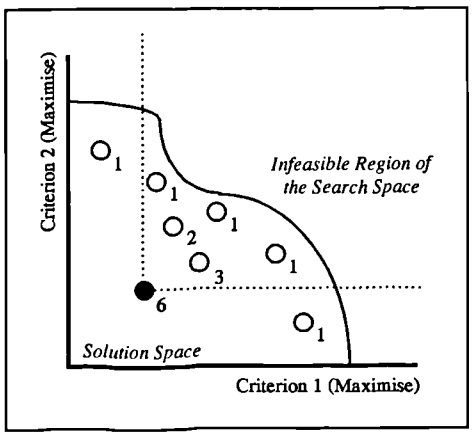


Figure IV.2: Dominance and ranking.

Step 4: Update the Pareto population

Strings with *Rank 1* are closest to the Pareto front and, as such, represent candidate *Pareto solutions*, which when grouped together, form a *Pareto population*. In order to ensure that Pareto solutions are never lost to genetic operation, the Pareto population, as distinct from the current population going on to receive genetic operation, is maintained separately and is not subject to any genetic operation. The Pareto population is updated every generation by adding any newly derived Rank 1 strings. Any duplicates or previous Rank 1 strings that are surpassed by the newly added Rank 1 strings are removed at every generation.

Step 5: Perform fitness conversion

Following Step 3, each string in the current population will have an associated rank between 1 (non-dominated) and population size (completely dominated), Rank 1 strings being the fittest.

Because selection assumes that fitter strings are represented by higher values, the rank of each string is converted to a fitness by subtracting the string's rank from the population size.

Step 6: Perform fitness sharing

In order to avoid deriving solutions that are bunched together in one region of the *Pareto front* (which becomes a *Pareto surface* when more than two criteria are considered) *fitness sharing* [Goldberg 87] is invoked. Fitness sharing acts to penalise strings in the current population which, based on their fitness, are bunched together in the same region of the search space. It is desirable to avoid the selection of such similar strings which, if not controlled, can lead to premature convergence to a local optima.

Fitness sharing is invoked for *each* solution in the current population. A hyper-cuboid of pre-defined size is defined in the search space with the solution point in question at it's centre. The procedure then counts all solutions which occupy the hyper-cuboid associated with this solution, and goes onto divide the fitness of the solution point in question by the count.

Whilst fitness sharing is invoked for each solution, it is specifically invoked in order to ensure that, Pareto solutions which are bunched together and, as such, only represent a small part of the optimal search space, have their fitness penalised. By penalising the fitness of bunched Pareto solutions, instead of each Pareto solution from the bunch having a high probability of surviving selection, on the law of averages, only one Pareto solution from the bunched group will survive selection.

The current population is now ready to go onto receive genetic operation with the aim of deriving fitter strings, which when re-evaluated will be found to represent superior solutions.

Step 7: Perform selection

As with the standard genetic algorithm, using the selection operator, strings begin the journey into the next generation based on their fitness. Fitter strings have a higher probability of contributing more offspring to the next generation. Pareto solutions are passed directly into the next generation's mating pool. Any spaces left in the mating pool are then filled with a random selection of strings from the Pareto population as well as strings selected from the current population using the biased roulette wheel routine first introduced in Chapter 4.

Step 8: Perform crossover

The aim of crossover is to recombine the strings in a meaningful and constructive way. In order to do this, strings in the mating pool are separated into the *resource choice section* and the *priority list sub-sections*. When crossover is carried out on a pair of strings each separated string section is only crossed with the corresponding section of the partner string. *Resource choice sections* are crossed with each other using a simple two point crossover whilst the *priority list sub-sections* are crossed using a modified version of the Enhanced Edge Crossover Operator [Grefensette 85]

Two-point crossover simply means that two random points are selected within the *resource choice section* and the alleles between these two points are swapped between two strings chosen at random. In the *priority list sub-sections* the two-point crossover operator cannot be used as it would introduce repeated activities. The Enhanced Edge Crossover Operator, which is slightly more complex, is used to ensure that activities are neither repeated nor lost during crossover. Once crossover has been applied, the resulting mating pool is subjected to mutation.

Step 9: Perform mutation

Mutation is a much simpler operator and is used to introduce small random changes into the population in order to maintain diversity. Again mutation is performed on separated sections of each string. In the *resource choice section*, if a gene is chosen, the allele is randomly reset to another legal resource choice. In the *priority list sub-sections* the allele associated with a gene selected for mutation will simply be swapped with the allele associated with another randomly selected gene in the same sub-section.

The population of strings resulting from mutation is then fed back into Step 2, and the process is repeated. The MCGA continues developing successive generations of even fitter Pareto populations until some pre-defined stop condition is met. At this point the Pareto population is returned as output. From the Pareto population the solution that most closely matches the decision-maker's preferences can then be selected and either analysed further or implemented accordingly.

References

[Ahuja 76] Ahuja, D., "Construction Performance Control by Networks", Wiley (New York), 1976.

[Akao 90] Akao, Y., "Quality Function Deployment: Integrating Customer Requirements into Product Design", Productivity Press, 1990.

[Alvarez-Valdez 89] Alvarez-Valdez, R., and Tamarit, J.M., "Heuristic Algorithms for Resource-Constrained Project Scheduling", in *"Advances in Project Scheduling"*, Slowinski, R., and Werglarz, J., (editors) Elsevier Science Publishers (Amsterdam), pp 113-134, 1989.

[Andrews 88] Andrews, D.J., A Comprehensive Methodology for The Design of Ship and Other Complex Systems, Transactions The Royal Society, Vol. 454, pp 187-211, 1988.

[Andrews 96] Andrews, D.J., Cudmore, A.C., Humble, P., and Wilson, D., "SUBCON - A New Approach to Submarine Concept Design", RINA Symposium on Naval Submarines (5): The Total Weapons System, Paper No. 19, 1996.

[Andrews 97] Andrews, D., and Dicks, C., "The Building Block Design Methodology Applied To Advanced Naval Ship Design", Sixth International Marine Design Conference (IMDC '97), pp 3-19, 1997.

[Ashley 95] Ashley, D., "A Spreadsheet Optimisation System for Library Staff Scheduling", Computer and Operations Research, Vol. 2, No. 6, pp 615-624, 1995.

[Asimow 62] Asimow, M., "Introduction to Design", Prentice Hall, 1962.

[Austin 89] Austin, S.A., Baldwin, A.N., Hassan, T., and Newton, A.J., "Techniques for the Management of Information Flow in Building Design, in *"Managing International Manufacturing"*, Ferdows, K., (editor), Elsevier Science Publishers, 1989.

[Austin 94] Austin, S.A., Baldwin, A.N., and Newton, A.J., "Manipulating the Flow of Design Information to Improve the Programming of Building Design", Construction Management and Economics, Vol. 12, pp 445-455, 1994.

[Austin 96] Austin, S., Baldwin, A., and Newton, A., "A Data Flow Model to Plan and Manage the Building Design Process", Journal of Engineering Design, Vol. 7, No. 1, pp 3-25, 1996.

[Austin 97] Austin, S., Baldwin, A., Huovila, P., and Koskela, L., "Application of the Design Structure Matrix to the Building Design Process", International Conference on Engineering Design (ICED 97), pp 217-220, 1997

[Bagchi 91] Bagchi, S., Uckun, S., and Miyabe, Y., "Exploring Problem-Specific Recombination Operators for Job-Shop Scheduling", Fourth International Conference on Genetic Algorithms, pp 10-17, 1991.

[Barclay 93] Barclay, I., Holroyd, P., and Taft, J, "A Sphenomorphic Model For The Management of The Complex Environment of Concurrent Engineering", International Conference on Engineering Management, Vol. 1, pp 72-76, 1993.

[Bechtold 91] Bechtold, S.E., Brusso, M.J., and Showalter, M.J., "A Comparative Evaluation of Labour Tour Scheduling Methods", Decision Sciences, Vol. 22, pp 683-699, 1991.

[Belhe 95] Belhe, U., and Kusiak A., "Resource Constrained Scheduling of Hierarchically Structured Design Activity Networks", Transactions of the IEEE on Engineering Management, Vol. EM-42, No. 2, pp 150-158, 1995.

[Bell 90] Bell, C.E., and Park, K., "Solving Resource Constrained Project Scheduling Problems by A* Search", Naval Research Logistics, Vol. 37, pp 61-84, 1990.

[Bell 91] Bell, C.E., and Han, J., "A New Heuristic Solution Method in Resource-Constrained Project Scheduling", Naval Research Logistics, Vol. 38, pp 315-331, 1991.

[Blazewicz 78] Blazewicz, J., "Complexity of Computer Scheduling Algorithms Under Resource Constraints", First Meeting of the AFCET-SMF on Applied Mathematics, pp 169-178, 1978.

[Blazewicz 88] Blazewicz, J., Finke, G., Haupt, R., and Schmidt, G., "New Trends in Machine Scheduling", European Journal of Operational Research, Vol. 37, pp 303-317, 1988.

[Bloebaum 92] Bloebaum, C.L., "An Intelligent Decomposition Approach for Coupled Engineering Systems", Transactions AIAA, Paper No. AIAA-92-4821-CP, pp1177-1187, 1992.

[Boctor 90] Boctor, F.F., "Some Efficient Multi-Heuristic Procedures for Resource-Constrained Project Scheduling", European Journal of Operational Research, Vol. 49, No. 6, pp 3-13, 1990.

[Boctor 96] Boctor, F.F., "Resource-Constrained Project Scheduling by Simulated Annealing", International Journal of Production Research, Vol. 34, No. 8, pp 2335-2351, August-1996.

[Boothroyd 88] Boothroyd, G., "The Importance of Product Design", American Machinist, Vol. 132, pp 54-57, 1988.

[Brand 64] Brand, J.D., Meyer, W.L., and Shaffer, L.R., "The Resource-Constrained Scheduling Problem in Construction", Civil Engineering

Studies Report No. 5, Dept. of Civil Engineering, University of Illinois, Urbana, 1964.

[Broadbent 80] Broadbent, G.H., "Design Methods - 13 Years After - A Review" in "*Design: Science: Method*" Jacques, R., and Powell, J.A., (editors), Westbury House, 1980.

[Brown 86] Brown, D.K., "Defining a Warship", Naval Engineers Journal, pp 35-47, March-1986.

[Bruns 85] Bruns, R., "Direct Chromosome Representation and Advanced Genetic Operators for Production Scheduling", Fifth International Conference on Genetic Algorithms, pp 352-359, 1985.

[Cheng 90] Cheng, T.C.E., and Sin, C.C.S., "A State Of The Art Review of Parallel Machine Scheduling Research", European Journal of Operational Research, Vol. 47, pp 271-292, 1990.

[Christensen 69] Christensen, J.H., and Rudd, D.F., "Advances in Chemical Engineering", Vol. 15, 1969.

[Christofides 87] Christofides, N., Alvarez-Valdes, R., and Tamarit, J.M., Project Scheduling with Resource Constraints: a Branch and Bound Approach, European Journal of Operational Research, Vol. 29, No. 3, pp 262-273, 1987.

[Churchman 68] Churchman, C.W., "The Systems Approach", Dell Publishing Co. Ltd, 1968.

[Clark 91] Clark, K., and Fujimoto, T., "Product Development Performance", HBS Press, 1991.

[Cleland 94] Cleland, G., "A Generic Approach to Spatial Engineering Incorporating a Knowledge-Based System", Ph.D. Thesis, University of Newcastle-upon-Tyne, UK, 1994.

[Colquhoun 93] Colquhoun, G.J., Baines, R.W., and Crossley, R., "A State of the Art Review Of $IDEF_0$", International Journal of Computers in Integrated Manufacturing, Vol. 6, pp 252-264, 1993.

[Crawford 92] Crawford, C.E., "The Hidden Costs of Accelerated Product Development", Journal of Product Innovation Management, Vol. 9, pp 188-199, 1992.

[Daniels 89] Daniels, R.L., "Resource Allocation and Multi-Project Scheduling", in "*Advances in Project Scheduling*", Slowinski, R., and Werglarz, J., (editors) Elsevier Science Publishers (Amsterdam), pp 87-113, 1989.

[Davis 73] Davis, E.W., "Project Scheduling Under Resource Constraints: A Historical Review and Categorisation of Procedures", Transactions AIIE, Vol. 5, pp 297-313, 1973.

[Davis 75] Davis, E.W., and Patterson, J.H., "A Comparison of Heuristic and Optimum Solutions in Resource-Constrained Project Scheduling", Management Science, Vol. 21, pp 944-955, 1975.

[Davis 89] Davis, L., "Adapting Operator Probabilities in Genetic Algorithms", Third International Conference on Genetic Algorithms, pp 61-69, 1989.

[Davis 91] Davis, L., (editor), "Handbook of Genetic Algorithms", Van Nostrand Reinhold, 1991.

[De Jong 75] De Jong, K.A., "An Analysis of the Behaviour of a Class of Genetic Adaptive Systems", Ph.D. Thesis, University of Michigan, USA, 1975.

[Dean 92] Dean, B.V., Denzler, D.R., and Watkins, J.J., "Multi-Project Staff Scheduling with Variable Resource Constraints", Transactions IEEE (Engineering Management), Vol. EM39, No. 1, pp 59-72, February-1992.

[DeMarco 79] DeMarco, T., "Structured Analysis and System Specification", Prentice Hall, Englewood Cliffs, New Jersey, 1979.

[Demeulemeester 92] Demeulemeester, E., Herroelen, W., "A Branch and Bound Procedure for the Multiple Resource-Constrained Project Scheduling Problem", Management Science, Vol. 38, No. 12, 1992.

[Deming 82] Deming, W.E., "Quality, Productivity and Competitive Position", MIT Press, 1982.

[Dixon 88] Dixon, J.R., and Duffy, M.R., "Quality is not Accidental - its Designed", The New York Times, June 26 1988.

[Doersch 77] Doersch, R.H., and Patterson, J.H., "Scheduling a Project to Maximise its Present Value: A Zero-One Programming Approach", Management Science, Vol. 23, pp 882-889, 1977.

[Drexl 91] Drexl, A., "Scheduling of Project Networks by Job Assignment", Management Science, Vol. 37, No. 12, pp 1590-1602, December-1991.

[Drucker 85] Drucker, P. F., "The Discipline of Innovation", Harvard Business Review, pp 67-72, May-1985

[Duffy 98] Duffy, A.H.B., Smith, J.S., and Duffy, S.M., "Design Reuse Research: A Computational Perspective", Proceedings of the Engineering Design Conference on Design Reuse, pp 43-56, 1998.

[Dumaine 89] Dumaine, B., "How Managers Can Succeed Through Speed", Fortune, Vol. 119, pp 54-59, 1989.

[Easa 89] Easa, S.M., "Resource Levelling in Construction by Optimisation", Journal of Construction Engineering Management, Vol. 115, pp302-316, 1989.

[Easton 93] Easton, F.F., and Mansour, N., "A Distributed Genetic Algorithm for Employee Staffing and Scheduling Problems", Fifth International Conference on Genetic Algorithms, pp 360-367, 1993.

[Eisenhardt 95] Eisenhardt, S., and Tabrizi, B.N., "Accelerating Adaptive Processes: Product Innovation in the Global Computer Industry", Administrative Science Quarterly, Vol. 40, pp 84-110, 1995.

[Elmaghraby 77] Elmaghraby, S.E., "Activity Networks: Project Planning and Control by Network Models", Wiley, New York, 1977.

[Eppinger 90] Eppinger, S.E., Whitney, D.E., Smith, R.P., and Gebala, D.A., "Organising the Tasks in Complex Design Projects", ASME Conference on Design Theory & Methodology (DTM '90), pp 39-46, 1990.

[Eppinger 94] Eppinger, S.D., Whitney, D.E., Smith, R.P., and Gebala, D.A., "A Model Based Method For Organising Tasks in Product Development", Research in Engineering Design, Vol. 6, pp 1-13, 1994.

[Evbuomwan 96] Evbuomwan, N., Sivaloganathan, S., and Jebb, A., "A survey of Design Philosophies, Models, Methods and Systems", Proceedings of the IMechE, Journal of Engineering Manufacture, Vol 210, No B6, pp 321-28, 1996.

[Flurscheim 77] Flurscheim, C., "Engineering Design Interfaces", Design Council Publications, 1977.

[French 85] French, M.J., "Conceptual Design for Engineers", Springer Verlag, 1985.

[Gebala 91] Gebala, D.A., and Eppinger, S.D., "Methods for Analysing Design Procedures", Transactions ASME (Design Theory & Methodology), DE-Vol. 31, pp 227-233, 1991.

[Glover 93] Glover, F., and Laguna, M., "Tabu Search", in "*Modern Heuristic Techniques for Combinatorial Problems*", Reeves, C.R., (editor), Blackwell Scientific Publications, Oxford, 1993.

[Goldberg 85] Goldberg, D.E., and Lingle, R., "Alleles, Loci, and the Travelling Salesman Problem", University of Alabama Working Paper: WP# AL 35486, 1985.

[Goldberg 87] Goldberg, D.E., and Richardson, J., "Genetic Algorithms with Sharing for Multi-Modal Function Optimisation", Second International Conference on Genetic Algorithms, pp 41-49, 1987.

[Goldberg 89] Goldberg, D.E., "Genetic Algorithms in Search, Optimisation and Machine Learning, Addison Wesley, 1989.

[Goldratt 84] Goldratt, E.M., and Cox, J., "The Goal: Excellence in Manufacturing", North River Press, 1984.

[Graham 79] Graham, R.L., Lawler, E.L., Lenstra, E.L., and Rinnooy Kan, A.H.G., "Optimisation and Approximation in Deterministic Sequencing and Scheduling Theory: A Survey", Annals of Discrete Mathematics, Vol. 5, pp 287-326, 1979.

[Grefensette 85] Grefensette, J., Gopal, R., and Rosmaita, B., "Genetic Algorithms for the Travelling Salesman Problem", First International Conference on Genetic Algorithms, pp 160-168, 1985.

[Hall 89] Hall, A.D., "Metasystems Methodology: A New Synthesis and Unification", Pergamon Press, 1989.

[Handfield 94] Handfield, R., "Effects of Concurrent Engineering on MTO Products, Transactions. IEEE (Engineering Management), Vol. EM41, No. 4, pp 384-393, November-1994.

[Harrison 92] Harrison, F.L., "Advanced Project Management: A Structured Approach", 3rd Edition, Gower, 1992

[Hassan 95] Austin, S.A., Baldwin, Thorpe, A., and Hassan, T., "Simulating the Construction Design Process by Discrete Event Simulation", International Conference on Engineering Design, ICED '95, pp 767-772, 1995.

[Hatley 87] Hatley, D.J., and Pirbhai, I.A., "Strategies For Real-Time System Specification", Dorset House, London, 1987.

[Hollins 93] Hollins, B., Hollins, G., and Hurst, D., "Design Management Processes that can be used by Practitioners", Ninth International Conference on Engineering Design, ICED '93, pp 656-663.

[Hubka 82] Hubka, V., "Principles of Engineering Design", Butterworth Scientific, 1982.

[Hyeon 93] Hyeon, H.J., Parsaei, H.R., and Sullivan, G.S., "Principles of Concurrent Engineering" pp 3-23, in "*Concurrent Engineering - Contemporary Issues and Modern Design Tools*", Parsaei, H.R., and Sullivan, G.S., (Editors), Chapman & Hall, 1993

[Icmeli 93] Icmeli, O., "Project Scheduling Problems: A Survey", International Journal of Operations & Production Management, Vol. 13, No. 11, pp 80-91, 1993.

[Jin 95] Jin, Y., Levitt, R.E., Christiansen, T.R., and Kunz, J.C., "The Virtual Design Team: Modelling Organisational Behaviour of Concurrent Design Teams", AIEDAM, Vol. 9, pp 14-5158.

[Jones 80] Jones, J.C., "Design Methods: Seeds of Human Futures", John Wiley & Sons, 1980

[Just 94] Just, M.R., and Murphy, J.P., "The Effect of Resource Constraints on Project Schedules", Transactions American Association of Cost Engineers, pp E 1.1 -1.6, 1994.

[Kanet 91] Kanet, J., and Sridharan, V., "PROGENITOR: A Genetic Algorithm for Production Scheduling", Wirtschaftsinformatik, Vol. 10, pp 332-336, 1991.

[Kehat 73] Kehat, E., and Shacham, M., "Chemical Process Simulation Programs (2)", Process Technology International, Vol. 18, No. 3, pp 115-118, 1973.

[Kelly 63] Kelly, J.E., "The Critical Path Method: Resource Planning and Scheduling", in "*Industrial Scheduling*", Muth, J.F., and Thompson, G.L., (editors), Prentice Hall, pp 347-365, 1963.

[Kirkpatrick 83] Kirkpatrick, S., Gelatt, C.D., and Vecchi, M.P., "Optimisation by Simulated Annealing", Science, Vol. 220, No. 4598, pp 671-680, May-1983.

[Kline 72] Kline, M.B., and Lifson, M.W., "Systems Engineering and its Application to The Design of an Engineering Curriculum - Part 1", Journal of Systems Engineering, Vol. 2, No. 1, pp 3-22, 1972.

[Krishnan 90] Krishnan, V., Eppinger, S.D., and Whitney, D.E., "Organising the Tasks in Complex Design Projects, Transactions ASME (Design Theory & Methodology), pp 39-46, 1990.

[Krishnan 94] Krishnan, V., Eppinger, S.D., and Whitney, D.E., "A Model-Based Framework to Overlap Product Development Activities", MIT Working Paper: WP # 3635-93-MS, December-1994.

[Kron 63] Kron, G., "Diakoptics: Piecewise Solution of Large Scale Systems of Equations", Ph.D. Thesis, University of Texas, Austin, 1963.

[Kusiak 87] Kusiak, A., and Chow, W.S., "An Algorithm for Cluster Identification", Transactions of the IEEE, Systems, Man and Cybernetics, Vol. SMC-17, No. 4, pp 696-699.

[Kusiak 90a] Kusiak, A., and Cheng, C.H., "A Branch and Bound Algorithm for Solving the Group Technology Problem", Annals of Operations Research, Vol. 26, No. 4, pp 415-431, 1990.

[Kusiak 90b] Kusiak, A., and Park, K., "Concurrent Engineering: Decomposition and Scheduling of Design Activities", International Journal of Production Research, Vol. 28, No. 10, pp 1883-1900, 1990.

[Kusiak 91] Kusiak, A., and Wang, J., "Concurrent Engineering: Simplification of the Design Process", in "*Computer Applications in Production and Engineering: Integration Aspects*", Doumeingts, G., Browne, J., and Tomljanovich, M., (editors), Elsevier Science Publishers, 1991.

[Kusiak 92] Kusiak, A., "Intelligent Design and Manufacturing", John Wiley and Sons, 1992.

[Kusiak 93] Kusiak, A., and Wang, J., "Decomposition of the Design Process", Journal of Mechanical Design, Vol. 115, pp 687-695, December-1993.

[Lamb 97] Lamb, T., "CE or Not CE -That Is The Question", SNAME Ship Production Symposium, April-1997.

[Larson 94] Larson, T.N., Kusiak, A., and Wang, J., "Reengineering of Design and Manufacturing Processes", Computers in Industrial Engineering, Vol. 26, No. 3, pp 521-536.

[Lawler 82] Lawler, E.L., Lenstra, J.K., and Rinnooy Kan, A.H.G., "Recent Developments in Deterministic Sequencing and Scheduling: A Survey", in "*Deterministic and Stochastic Scheduling*", Dempster, M.A., Lenstra, J.K., Rinnooy Kan, A.H.G., (editors), Reidel, pp 35-73, 1982.

[Ledet 70] Ledet, W.P., and Himmelbau, D.M., "Decomposition Procedures For The Solving of Large Scale Systems", Advances in Chemical Engineering, Vol. 8, 1970.

[Lee 96] Lee, J., and Kim, Y., "Search Heuristics for Resource Constrained Project Scheduling", Journal of the Operational Research Society, Vol. 47, No. 6, pp 678-689, 1996.

[Lewis 98] Lewis, W.P., and Cangshan, L., "The Timely Allocation of Resources in the Concurrent Design of New Products", Journal of Engineering Design, Vol. 9, No.3, pp 3-15, 1998.

[Li 92] Li, K.Y., and Willis, R.J., "An Iterative Scheduling Technique for Resource-Constrained Project Scheduling", European Journal of Operational Research, Vol. 56, No. 3, pp 370-379, February-1992.

[Maylor 96] Maylor, H., "Project Management", Pitman Publishing, 1996.

[Moore 76] Moore, L.J., and Clayton, E.R., "GERT Modelling and Simulation: Fundamentals and Applications", Petrocelli-Charter, New York, 1976.

[Milner 71] Milner, C.G., "Innovation in Shipbuilding", R&D Management, Vol. 2, No. 1, 1971.

[Nabrzyski 96] Nabrzyski, J., and Werglarz, J., "On an Expert System with Tabu Search for Multi-Objective Project Scheduling", Transactions ASME (Petroleum Division), Vol. 79, No. 7, pp 95-98, August-1996.

[Nakano 80] Nakano, R., and Yamada, T., "Conventional Genetic Algorithms for Job-Shop Problems", Fourth International Conference on Genetic Algorithms, pp 239-249, 1980.

[Newton 95] Newton, A.J., "The Planning and Management of Detailed Building Design", PhD Thesis, Loughborough University of Technology, 1995.

[Ozdamar 95] Ozdamar, L., and Ulusoy, G., "A Survey on the Resource-Constrained Project Scheduling Problem", Transactions IIE, Vol. 27, No. 5, pp 574-586, October-1995.

[Page-Jones 80] Page-Jones, M., "The Practical Guide to Structured Systems Design", Yourden Press, New York, 1980.

[Pahl 96] Pahl, G and Beitz, W., "Engineering Design: A Systematic Approach", (2nd Edition), Springer, 1996.

[Patterson 90] Patterson, J.H., Talbot, B.F., Slowinski, R., and Werglarz, J., "Computational Experience with a Backtracking Algorithm for Solving a General Class of Precedence and Resource-Constrained Scheduling Problems", European Journal of Operational Research, Vol. 49, pp 68-79, 1990.

[Phillips 77] Phillips, S., and Dessouky, M.I., "Solving The Time/Cost Trade-Off Problem Using the Minimal Cut Concept", Management Science, Vol. 24, No. 4, pp 393-400, 1977.

[Pinson 94] Pinson, E., Prins, C., and Rullier, F., "Using Tabu Search for Solving the Resource-Constrained Project Scheduling Problem", Fourth International Workshop on Project Management and Scheduling, pp 102-106, 1994.

[Pourbabai 94] Pourbabai, B., and Pecht, M., "Management of design activities in a concurrent engineering environment", International Journal of Production Research, Vol. 32, No. 4, pp 821-832, 1994.

[Prasad 98] Prasad, B., Wang, F., and Deng, J., "A concurrent Workflow Management Process for Integrated Product Development", Journal of Engineering Design, Vol. 9, No. 2, 1998.

[Pritsker 81] Pritsker, A.A.B., Watters, L.J., and Wolfe, P.M., "Multi-Project Scheduling With Limited Resources: A Zero-One Programming Approach", Management Science, Vol. 16, pp 265-273, 1981.

[Pugh 90] Pugh, S., "Total design - Integrated Methods for Successful Product Engineering", Addison Wesley, 1990.

[Ramat 97] Ramat, E., Venturini, G., Lente, C., and Slimane, M., "Solving The Multiple Resource Constrained Project Scheduling Problem With A Hybrid Genetic Algorithm", Seventh International Conference on Genetic Algorithms, pp 489-496, 1997.

[Rodenacker 70] Rodenacker, W.G., "Methodisches Konstruieren", Springer Verlag, 1970

[Rogers 92] Rogers, J.L., and Barthelemy, J.M., "Enhancements to the Design Managers Aide for Intelligent Decomposition (DeMAID)", Transactions AIAA, Paper No. AIAA-92-4809-CP, pp 932-941, 1992.

[Reeves 95] Reeves, C.R., "Genetic Algorithm for Flowshop Sequencing", Computers and Operations Research, Vol. 22, No. 1, pp 5-13, January-1995.

[Rogers 96a] Rogers, J.L., "DeMAID/GA: An Enhanced Design Managers Aid for Intelligent Decomposition", Transactions AIAA, Paper No. AIAA-96-4157-CP, pp 1497-1504, 1996.

[Rogers 96b] Rogers, J.L., McCulley, C.M., and Bloebaum, C.L., "Integrating a Genetic Algorithm into a Knowledge-Based System for Ordering Complex Design Processes", Fourth International Conference on Artificial Intelligence in Design, pp 119-133, 1996.

[Rosenstein 67] Rosenstein, A.B., "The Modern View of the Design Process", National Congress of the Society of Automotive Engineers, 1967.

[Ross 77] Ross, D.T., "Structured Analysis (SA): A Language For Communicating Ideas, Transactions IEEE (Software Engineering), Vol. SE-3, pp 16-24, 1977.

[Ross 85] Ross, D.T., "Applications and Extensions of SADT", IEEE Computer Magazine, pp 25-34, April-1985.

[Roy 85] Roy, R., and Wield, D (editors) "Product Design & Technological Innovation", Open University Press, 1985.

[Russell 70] Russell, A.H., "Cash Flows in Networks", Management Science, Vol. 16, pp 357-373, 1970.

[Salewski 97] Salewski, F., Schirmer, A., and Drexl, A., "Project Scheduling Under Resource and Mode Identity Constraints: Model, Complexity, Methods and Application", European Journal of Operational Research, Vol. 102, pp 88-110, 1997.

[Sargent 64] Sargent, R.W., and Westerberg, A.W., "Speed-Up in Chemical Engineering Design", Transactions Institution of Chemical Engineers, Vol. 42, pp T190-T197, 1964.

[Schaffer 89] Schaffer, J.D., Caruana, L.J., Eshelman, L.J., and Das, R., "A Study of Control Parameters Affecting Online Performance of GAs for Function Optimisation", Third International Conference on Genetic Algorithms, pp 51-60, 1989.

[Senge 90] Senge, P.M., "The Fifth Discipline: The Art and Practice of the Learning Organisation", Currency Doubleday, New York, 1990.

[Shaffer 65] Shaffer, L., Ritter, J., and Mayer, W., "The Critical Path Method", McGraw Hill, 1965.

[Shaffer 91] Shaffer, S., "A Rule-Based System for Automated Staff Scheduling, IEEE Conference on Systems, Man, and Cybernetics, Vol. 3, pp 1691-1696, 1991.

[Shina 94] Shina, S., "Concurrent Engineering in New Products and Processes", pp 1-15, in "*Successful Implementation of Concurrent Engineering Products and Processes*", Shina, S., (Editor), Van Nostrand Reinhold, 1994.

[Siemens 71] Siemens, N., "A Simple CPM Time/Cost Trade-Off Algorithm", Management Science, Vol. 17, No. 6, pp B354-362, 1971.

[Simon 70] Simon, H.A., "The sciences of The Artificial", (2nd Edition), MIT Press, 1970.

[Slowinski 81] Slowinski, R., "Multi-Objective Network Scheduling with Efficient Use of Renewable and Non-Renewable Resources", European Journal of Operational Research, Vol. 7, pp 265-273, 1981.

[Smith 91] Smith, P., and Reinersten, D.G., "Developing Products in Half the Time", Van Nostrand Reinhold, 1991.

[Smith 94] Smith, R.P., and Eppinger, S.D., "A Predictive Model of Sequential Iteration in Engineering Design", MIT Working Paper: WP # 3160-90-MS, April-1994.

[Smith 97] Smith, R.P., and Eppinger, S.D., "Identifying Controlling Features of Engineering Design Iteration", Management Science, Vol. 43, No. 3, March-1997 .

[Snaith 82] Snaith, G.R, and Kennedy, B., "The Architecture of Engine-Rooms: A Case Study", British Shipbuilders Internal Research Paper, 1982.

[Snaith 95] Snaith, G.R., "The Essential Role of Stable Processes", in "*Requirements and Assessments for Global Shipbuilding Competitiveness*", Storch, R.L., (editor), NSRP - SNAME Ship Production Committee - SP 4 Panel, 1995.

[Snaith 99] Snaith, G.R, and Scott, J.A., "Design/Production Integration", in "*Ship Design and Construction*" (3rd Edition), Lamb, T., (editor), SNAME, forthcoming

[Speranza 93] Speranza, M., and Vercellis, C., "Hierarchical Models for Multi-Project Planning and Scheduling", European Journal of Operational Research, Vol. 64, pp 312-325, 1993.

[Steward 81a] Steward, D.V., "Systems Analysis and Management", Petrocelli Books, 1981.

[Steward 81b] Steward, D.V., "The Design Structure Matrix - A Method for Managing the Design of Complex Systems", Transactions IEEE (Engineering Management), Vol. EM28, No. 3, pp 71-74, August-1981.

[Steward 91] Steward, D.V., "Planning and Managing the Design of Systems", Management of Engineering and Technology Conference, pp 189-193, 1991.

[Suh 90] Suh, N.P., "The Principles of Design", Oxford University Press, New York, 1990.

[Syan 94] Syan, C.S., "Introduction to Concurrent Engineering", pp 3-23, in "*Concurrent Engineering - Concepts, Implementation and Practice*", Syan, C.S., and Mennon, U. (Editors), Chapman & Hall, 1994.

[Syswerda 91a] Syswerda, G., "Schedule Optimisation Using Genetic Algorithms", in "*Handbook of Genetic Algorithms*", Davis, L., (editor), Van Nostrand Reinhold, 1991.

[Syswerda 91b] Syswerda, G., and Palmucci, J., "The Application of Genetic Algorithms to Resource-Constrained Scheduling", Fourth International Conference on Genetic Algorithms, pp 502-508, 1991.

[Takeuchi 86] Takeuchi, H., and Nonaka, N., "The New Product Development Game", Harvard Business Review, January-February-1986.

[Tersine 76] Tersine, R.J., and Riggs, W.E., "Models: Decision Tools for Management", Journal of Systems Management, Vol. 8, pp 30-34, October-1976.

[Todd 97a] Todd, D., and Sen, P., "A Multiple Criteria Genetic Algorithm for Containership Loading", Seventh International Conference on Genetic Algorithms, pp 674-681, 1997.

[Todd 97b] Todd, D., and Sen, P., "Multiple Criteria Scheduling Using Genetic Algorithms in a Shipyard Environment", International Conference on Computer Applications in Shipbuilding (ICCAS '97), pp 234-265, 1997.

[Todd 97c] Todd, D.S., "Multiple Criteria Genetic Algorithms in Engineering Design and Operation", Ph.D. Thesis, University of Newcastle upon Tyne, 1997.

[Tsubakitani 90] Tsubakitani, S., and Deckro, R.F., "A Heuristic For Multi-Project Scheduling with Limited Resources in the Housing Industry", European Journal of Operational Research, Vol. 49, pp 80-91, 1990.

[Tufekci 82] Tufekci, S., "A Flow Preserving Algorithm for the Time/Cost Trade-Off Problem", Transactions AIII, Vol. 2, No. 3, pp 109-113, June-1982.

[Tzafestas 93] Tzafestas, S., and Triantafyllakis, A., "Deterministic Scheduling in Computing and Manufacturing Systems: A Survey of Models", Mathematics and Computers in Simulation, Vol. 35, No. 5, pp 397-434, November-1993.

[Ulrich 95] Ulrich, K.T., and Eppinger, S.D., "Product Design and Development", McGraw Hill, 1995.

[Wallace 89] Wallace, D., "Better by Design", Manufacturing Engineer, pp 35-36, 1989.

[Ward 85] Ward, P.T., and Mellor, S.J., "Structured Development For Real-Time Systems", Yourden Press, New York, 1985.

[Warfield 73] Warfield, J.N., "Binary Matrices in System Modeling", Transactions IEEE (Systems, Man and Cybernetics), Vol. SMC3, No. 5, pp 441-449, September-1973.

[Weiss 92] Weiss, J.W., and Wysocki, R.K., "5-Phase Project Management: A Practical Planning & Implementation Guide", Addison-Wesley, 1992.

[Wiest 63] Wiest, J.D., "The Scheduling of Large Projects with Limited Resources", Ph.D. Thesis, Carnegie Institute of Technology, California, USA, 1963.

[Wiest 64] Wiest, J.D., "Some Properties of Schedules for Large Projects with Limited Resources", Operations Research, Vol. 12, pp 395-418, 1964.

[Wiest 65] Wiest, J.D., "A Heuristic Model for Scheduling Large Projects with Limited Resources", Management Science, Vol. 13, B359- B377, 1965.

[Wiest 77] Wiest, J.D., and Levy, F.K., "A Management Guide to PERT/CPM", Prentice Hall, New Jersey, 1977.

[Willis 85] Willis, R.J., "Critical Path Analysis and Resource Constrained Project Scheduling - Theory and Practice", European Journal of Operational Research, Vol. 21, pp 149-155, 1985.