Monte Carlo Simulation in the Marine Environment

A Thesis Submitted to Liverpool John Moores University for the Degree of Doctor of Philosophy

Andrew Donald Cunningham

Liverpool Logistics, Offshore and Marine (LOOM) Research Centre, School of Engineering, Technology and Maritime Operations

January 2011

PAGE/PAGES EXCLUDED UNDER INSTRUCTION FROM UNIVERSITY

Acknowledgments

This thesis marks the culmination of a three year research project which I started in January 2007 with the Liverpool Logistics, Offshore and Marine (LOOM) Research Centre at Liverpool John Moores University. During this period I have received the help and support of a great number of people. It gives me great pleasure to be able to convey my thanks and sincere gratitude to them in my acknowledgment.

The first person I would like to thank is my principal supervisor Professor Jin Wang, for his guidance and advice, both professionally and academically, but also for his patience and encouragement while completing my research. His strong commitment to research and his constant endeavour in academia has served as a great inspiration to me during the last three years. His feedback and critique have proved invaluable and his unwavering demand for research of the highest quality has been the driving force behind this research work.

I wish to extend my sincere thanks and gratitude to my co-supervisors, Dr David Allanson and Dr Alan Wall, who always extended me time and support throughout the research project. I would also like to convey my special thanks to my research colleagues at LOOM, without their friendship, advice and support, especially in the early stages of my research when the challenge of producing this body of work at times seemed insurmountable.

Special thanks are extended to Enrico Zio, Professor of Safety and Risk Analysis at the Polytechnic of Milan who has acted as my technical advisor during the research. Prof. Zio and his research group provided a warm welcome and invaluable support during the early stages of the research process. I would also like to thank Professor Wen Bin Wang of Salford University for his invaluable feedback, support and time on delay-time analysis.

Thanks are also extended to Maurice Palin a good friend and accomplished programmer.

Finally I wish to thank my family and partner for providing me with unconditional love and support, I simply could not have achieved this without them.

A. Cunningham

Abstract

When continued operation and function of a ship's systems are required, unforeseen system failure or breakdown can often have disastrous and costly consequences. Ship owners and operators require ships to be operating at full capacity as often as possible in order to remain profitable. This places the ship's engineering department under pressure to plan maintenance schedules which optimise reliability and minimise downtime. The industry as a whole has been slow to respond to the need for better maintenance planning, often relying on manufacturer's recommendations for the setting of service intervals and the replacement of parts.

Monte Carlo Methods have proved to be a powerful tool in the nuclear sector and for around three decades remained exclusive to that industry. In recent years researchers have realized the vast potential and flexibility contained in the methodology and its possible application in other fields. One of the problems which is inherent to Monte Carlo Methods is the handling of rare events. Often to remain statistically significant, variance reduction techniques need to be implemented. One of the principal methods is the use of forced simulation. In the marine industry, due to the high levels of salinity, this problem of rare events becomes less significant. It is also the case that often the process mediums used are of a much lower quality than in other industries. This all contributes to the failure probabilities being much higher, negating the need for any forced simulation.

The majority of current reliability and maintenance practice is based on time to first failure, or time between failures. Delay-time modelling is a concept which has been developed to be relevant in the operating culture of today's industry. Delay-time analysis provides engineers with a tool which can help to minimise downtime of a machine or plant item, based on an inspection period. Classical delay-time analysis is mathematically arduous and takes time, however the benefits of implementing the technique are well proven and recognised. Monte Carlo Methods lend themselves well to delay-time techniques and could offer an automated analysis tool which requires very little user input. The availability of

such a tool to marine engineers would allow for better inspection and maintenance scheduling based on minimising downtime.

This research work is evidence of the implementation of Monte Carlo Methods in the creation of a simulation based maintenance methodology in the marine environment. The Monte Carlo Methods have been used to provide a measure of the unreliability of a complex marine cooling system. The unreliability measure has been used to perform a delay-time analysis using Monte Carlo Methods to suggest an inspection regime based on minimising downtime. The complex Monte Carlo Method has been extended to give an indicator as to the optimum staff level based on system downtime and a staff cost.

Table of Contents

List of Figures

List of Tables

Abbreviations

MCM	Monte Carlo Methods
DTA	Delay-Time Analysis
RAMS	Reliability, Availability and Maintainability Studies
CM	Corrective Maintenance
PM	Preventive Maintenance
RCM	Reliability Centered Maintenance
TPM	Total Productive Maintenance
HT	High Temperature
LT	Low Temperature
PDF	Probability Density Function
CDF	Cumulative Distribution Function
FTA	Fault Tree Analysis
BRM	Boolean Representation Method
BRT	Boolean Representation Table
ST.	Cold Standby

Nomenclature

$f(x)$	Probability density function of a variable x
$F(x)$	Cumulative distribution function of a variable x
λ	Failure rate
t	Time
R	Random Variable
$E\langle R\rangle$	Mathematical expectation value of R
$VAR(R)$	Variance of R
$U_R(r)$	Cumulative distribution function of a uniform distribution
$u_R(r)$	Probability density function of a uniform distribution
X_0	Seed value
τ	Deterministic repair time
a, c	Constants

Chapter 1 – Introduction

Summary

This chapter will present the background to this research work and will discuss the problems that exist in the Marine Industry when conducting reliability and maintenance studies. The aims and objectives of the research work are stated which will address some of the problems outlined previously. The research methodology and scope of the thesis are highlighted.

1.1 Background of the Research

The first documented use of Monte Carlo Methods was by Buffon (1777) in his 'Needle dropping experiment', Laplace provided some valuable insights into Buffon's work and Lord Kelvin did some work in kinetic gas theory. The technique was essentially revived in 1945 by Fermi, von Neumann and Ulam in the development of the Manhattan project. It was certainly at this time that, owing to its nature, the technique acquired its name. The first Monte Carlo algorithm used to solve linear equations, conceived by von Neumann and Ulam, was published in a paper by Forsythe et al (1950) and other important contributions were made by Curtiss (1954) and Halton (1962, 1994).

Since then with the various developments in the computer industry and the amount of computer power now available, even in home PCs, Monte Carlo techniques have been taken to levels which were unimaginable in the 'infancy' of the Monte Carlo Method (MCM). Monte Carlo remains a very powerful tool in the nuclear sector and for around three decades it remained exclusive to that industry. In recent years researchers have realized the vast potential and flexibility contained in the methodology.

A great deal of work has been done using Monte Carlo Methods especially concerning Reliability, Availability and Maintainability (RAMS) problems (Dubi, 1986), (Labeau, 2000), (Marseguerra & Zio, 2002), (Zio, 1995), (Barata et al. 2002). Despite the technique proving its worth in the nuclear industry very little work has been done with Monte Carlo techniques in the marine industry on RAMS based problems, (Zio et al. 2004). Papers are available showing the application of Monte Carlo Simulations in other ways. Dowd (2006) uses Monte Carlo techniques for the prediction of marine ecological growth; Soares & Garbatov (1996) use Monte Carlo Methods to examine the reliability of a ship hull girder.

Mechanical failure has been a major cause of marine accidents in the past, examples include "Brear", "Savannah Express" and "MV Symphony" (MAIB, 1991-2005). The motor tanker Brear sailed for Quebec on the 3rd January 1993 having left Mongstad fully laden with a cargo of 84,700 tonnes of light crude oil. On the evening of the 4th January at 04.40 hrs the main engine failed followed by a failure of the auxiliary generator. This left the ship without electrical power and she was adrift just ten miles off the southern tip of Shetland eventually grounding at 11.19 hrs.

On the 19th July 2005, at 11.46, the Savannah Express, one of the largest container ships in the world at 94483 gross tonnage, collided with a linkspan at its berth in Southampton docks. The engine lost astern engine power shortly before she turned to come along side resulting in the collision. The damage to the ship was only minor; some paint damage to the bulbous bow, however the damage to the linkspan was considerable and it was unable to be used again until extensive repairs had been carried out.

At 20.15 on the 4th October 1999 the MV Symphony experienced a steering gear failure on board the ship; as a consequence the vessel collided with the central support of the Lambeth Bridge on the Thames, London. Although in this instance there were no injuries to either passengers or crew there was some damage to the wheelhouse of the symphony. The examples given are just a few instances where mechanical failure has led to the occurrence of undesirable events, namely collisions and groundings. Monte Carlo techniques have the power and flexibility, given appropriate information, to simulate all of the system failures shown above. Such techniques if implemented correctly could provide us with further information on system failure modes and also be used to predict maintenance periods which would optimise a components useful lifetime without jeopardising system integrity.

In order to demonstrate the viability of Monte Carlo techniques in the marine environment, case studies must be carried out, with generic models which demonstrate some of the information that the models can yield.

1.2 Research Aims and Objectives

The primary aim of this research work is to demonstrate how MCM can be applied in the marine industry. The intention is to show a number of ways, through the use of case studies, in which application of MCM can aid marine engineers in their understanding of marine systems. This will enable them to make better and more informed decisions relating to maintenance and reliability. The following three objectives each contribute to this primary purpose.

The first objective is to develop a methodology which will allow the unreliability of a complex marine system to be assessed. The methodology will provide a framework from which the unreliability of the system, as a whole, can be determined from the individual component failure rates. The modelling of the system will be achieved through simulation using MCM. It will show the effect of using equal, deterministic repair times for each component, as well as individual repair times for each component. Completion of this objective will be the first demonstration of the MCM and how it can be applied to the marine industry. It will serve as a basis for the development of further technical work developed through the thesis.

The second objective is to develop, through simulation, an advanced risk-based maintenance methodology for marine equipment. The methodology will provide a framework to optimise the inspection and maintenance activities of the system. The modelling technique used will be delay-time analysis (DTA). The application of the DTA will differ from the classical analytical approach, through the application of simulation techniques. The result of the optimisation will be to establish an optimum inspection interval based on minimising downtime. The methodology will consider perfect and imperfect inspection, as well as perfect and imperfect repair.

The third objective is to develop a decision making methodology based on cost and system downtime. The methodology will provide a framework which can be used to assess staff levels for a given marine system. Providing a cost benefit analysis, the methodology will indicate how making more staff members available for the maintenance of a particular marine system may not provide an associated reduction in system downtime.

The objectives are set out in order to fulfil the main aim of the research. Throughout the research project a number of case studies have been used to demonstrate each of the associated methodologies. A marine cooling system for a main engine has been presented as a case study to demonstrate each of the proposed approaches. The inclusion of this same marine system, a number of times, is intentional to demonstrate how MCM can be applied in a number of ways to offer large amounts of information to a marine engineer. This large amount of information and the inherent ability of simulation models to allow 'trial and error' permutations of system operating conditions will allow marine engineers to achieve more informed, improved decisions.

Where possible 'real life' data was used from the marine industry, sourced from the Offshore Reliability Data handbook (OREDA). In the absence of appropriate historical available data, expert judgement has been used.

1.3 Statement of the Problem

In the marine environment maintenance is often conducted in a preventive way. Maintenance and inspection scheduling is determined by manufacturer recommendations. The development of a simulation based maintenance methodology could offer a better solution for systems analysis in the marine environment. MCM are widely implemented in the design and analysis of systems in other industries. While a great deal of work has been done in the area of reliability, availability and maintainability studies in other sectors, very little work exists in the practical application of Monte Carlo Methods to marine systems. Research work in the marine environment is often hindered by a lack of historical data (Pillay & Wang 2003). As a consequence of this, more qualitative research is conducted. There is a need for some quantitative models which can provide information to enable better decision making (Wang 2006). A simulation based maintenance methodology can offer quantitative results on the unreliability of complex systems based on individual component failure rates. The simulation of marine systems allows for 'trial and error' runs to assess the impact of decisions before they are applied to real systems.

1.4 Research Methodology and Scope

The methodology of the research work is the development of a simulation based maintenance model. The research work uses indirect MCM to provide information on system unreliability. Integrating DTA and simulation provides an optimised maintenance and inspection regime based on minimising downtime. Finally the Monte Carlo model is extended to provide information as to optimum staff level to minimise system downtime and staff cost.

The scope of the research project was to develop a simulation based maintenance methodology. The purpose of the methodology is to provide information relating to the unreliability of a complex marine system, suggest an optimum inspection and maintenance regime to reduce downtime and provide information as to optimum staff levels to reduce system downtime and staff cost.

1.5 Structure of the Thesis

The thesis comprises seven chapters. Chapter 1 provided an introduction to the research work, stated the aims and objectives, outlined the problems associated with current maintenance strategies in the marine environment and outlined the methodology and scope of the research.

Chapter 2 documents a review of the current literature relating to MCM, maintenance and decision making methods. The extensive work presented based in other industries and the lack of work in the marine environment highlights the need for research in this area. Consideration is given to the analytic hierarchical process as a decision making tool and draws attention to the inability of the method to assess the possible impact of decisions on a system.

Chapter 3 serves as an extension to the literature review introducing the MCM. The chapter introduces the Monte Carlo modelling techniques through the use of a simple model with three components in parallel. The inclusion of this chapter prevents further repetition of the basic MCM which forms a fundamental part of all the models developed in the following technical chapters.

Chapter 4 is the first technical chapter. It presents the application of MCM to a complex marine system. Three different operating scenarios are considered, the first case is a system where all the components have equal deterministic repair times and the system is not returned to operation until all components are repaired. In the second case the component repair times are different and individual to each type of component. In the third case the components all have equal deterministic repair times but the system is returned to operation as soon as enough components are available for normal system function. All three cases are discussed and comparisons are made concerning the differences in system unreliability.

Chapter 5 documents the application of DTA using simulation. DTA provides optimum inspection intervals based on minimising downtime. The delay-time methodology is introduced and an estimation of the arrival rate of defects, λ, is made based on the unreliability data presented in Chapter 4. The DTA via simulation relaxes some of the assumptions made in classic DTA. This allows for analysis considering perfect inspections, imperfect inspections and imperfect inspections with imperfect repairs.

Chapter 6 takes the complex Monte Carlo model presented in Chapter 4 and extends it to give information regarding staff levels. The model takes a prescribed staff level which alters during a trial as failures occur and repair actions need addressing. At a time when a failure occurs which requires repair and the staff level is insufficient to support the repair action, the system essentially 'waits' to be repaired. This leads to an increase in system downtime. At a certain point the staff level will minimise the system downtime. A further increase in staff level will cause additional staff cost. The model aims to ascertain this staff level which minimises downtime and a cost model is presented.

Chapter 7 draws conclusions from the research work. The conclusions from the main technical work will be presented and the contribution to research will be outlined. Limitations of the research will be presented and suggestions for future work will be made which can expand upon this body of work.

Chapter 2 – Literature Review

Summary

This chapter will give an overview of maintenance concepts and their role within the marine industry, historically and at present. A comprehensive literature review will be presented considering all of the methodologies used in this thesis. The need and justification for further research in this field will also be presented.

2.1 Introduction

Before any research work can be carried out the current field of knowledge must be defined. This is achieved with an extensive literature review which, when carried out properly will ensure the novelty of the research. This chapter contains a review of past and present work completed using Monte Carlo Methods as well as several topics relevant to the research work. Presented first is a review of literature pertinent to MCM and some key contributions which have been made are outlined. A review of current research work concerning systems and reliability studies is presented. Second is a review of current maintenance methodology including preventive, corrective and condition based maintenance. The possible application of total productive maintenance in the marine environment is discussed. Finally a review of decision making methods is conducted. Analytic hierarchical process has been identified as a decision making tool widely used. The consideration of these three key areas serves to highlight the need for further research.

2.2 Monte Carlo Methods

The technology available to modern marine engineers in an 'average' engine room has, over recent years, become ever more complex. As engine control rooms become more automated, the critical systems of the ship's main engines and ancillaries are constantly monitored. Emphasis is often placed on preventive maintenance. In systems where continued operation and function of the systems is required, unforeseen system failure or breakdown can often have disastrous and costly consequences. Ship owners and operators require the ships to be operating at full capacity as often as possible in order to remain profitable. This places the ship's engineering department under pressure to plan maintenance schedules which optimise reliability and minimise downtime. MCM has proved to be a powerful tool in the nuclear sector and for around three decades remained exclusive to that industry. The MCM was originally developed during the Manhattan project. Nuclear shielding needed to be developed to control radiation from the radioactive material. This proved challenging due to the random nature intrinsic to the way in which neutrons are emitted from the nuclear source. MCM were developed to model the neutrons pathway through the shielding material; with each interaction with the atoms in the shield the neutron has a chance of being deflected or absorbed. Traditional MCM model the free flight path of the neutrons and examine the interactions and collisions. Goldfield & Dubi (1987) modified the method to consider reliability issues. In recent years researchers have realized the vast potential and flexibility contained in the methodology and its possible application in other fields.

Some of the first work in MCM was published concerning calculating the position, energy and flight directions of particles in different mediums (Cashwell & Everett 1959), (Gerbard & Spanier 1969), (Dubi 1986), (Lux & Koblinger 1990). The MCM was later extended to include safety analysis of engineering systems and plants (Lewis & Bohm 1984), (Wu & Lewins 1992), (Dubi 2000), (Marseguerra & Zio 1993), (Marseguerra et al. 1998), (Marseguerra & Zio 2001). Zio (1994) made a further contribution with a technical note on biasing the transition probabilities in direct Monte Carlo which leads to simplification of the sampling procedure and improved computational efficiency. Marseguerra et al. (1998) produced a paper on dynamic reliability via Monte Carlo Simulation and showed how MCM is capable of handling dynamic probabilistic safety assessment problems. Marseguerra et al. (2001) considered the application of biased MCM to unavailability analysis for systems with time dependant failure rates and found that biasing schemes based on uniform distributions provided a more even distribution of failures over the component lifetime. Barata et al. (2002) simulated a repairable multi-component deteriorating system for 'on condition' maintenance optimisation determining degradation thresholds for maintenance intervention. Labeau & Zio (2002) presented two separate approaches for simulation, a direct component based approach and an indirect system based approach. Comparisons of the two approaches were made with respect to computing time and variance.

A great deal of work has been done using Monte Carlo Methods especially concerning Reliability, Availability and Maintainability (RAMS) problems (Dubi 1986), (Labeau 2000), (Labeau & Zio 2002), (Marseguerra & Zio 2002), (Zio 1995), (Barata et al. 2002), (Bevilacqua et al. 2000), (Accumoli 1996), (Zio et al. 2007). Marine engineers and the marine industry in general have failed to realise how useful a tool this could be to the marine industry. Very little work has been done with Monte Carlo techniques in the marine industry on RAMS based problems (Zio et al. 2004). Papers have reported the application of Monte Carlo Simulations in other ways. Aalbers et al. (2001) have developed a software system for safer rig moves based on a MCM. Dowd (2006) used Monte Carlo techniques for the prediction of marine ecological growth; Guedes Soares & Garbatov (1996) use Monte Carlo to examine the reliability of a ship hull girder; Santos & Guedes Soares (2004) use Monte Carlo Simulation to predict damaged ship survivability.

2.3 Maintenance

Maintenance is a huge area of interest and research for engineers. A number of papers based on maintenance strategy and decision have been published (Barbera et al. 1996), (Qi et al. 1999), (Wang et al. 2000), (Wang & Majid 2000), (El-haram

& Horner 2002), (Emblemsvåg & Tonnig 2003), (Beebe 2003), (Backlund & Akersten 2003), (Wang & Hwang 2004). Maintenance costs form a significant part of the overall operating costs in ship operations (Mokashi et al. 2002). Pillay & Wang (2003) define maintenance as the combination of all technical and administrative actions, including supervision actions, intended to retain an entity in, or restore it to a state, in which it can perform a required function. The ISM Code states that all ship operators 'should establish and implement procedures to identify equipment and technical systems the sudden operational failure of which would result in hazardous situations' (ISM 2002). In meeting these requirements the company should ensure that:

- Inspections are held at appropriate intervals.
- Any non conformity is reported with its possible cause, if known.
- Appropriate corrective actions are taken.
- Records of these activities are maintained.

Soncini (1996) suggests that most ship owners understand the need of having good control over accounting and purchasing and are found to be at the same level as their land based counterparts; however the same cannot be said when it comes to maintenance and stock control. Pintelon et al. (1997) introduce the 'maintenance concept', defined as the set of various maintenance interventions (corrective, preventive, condition based, etc.) and the general structure in which these interventions are brought together. The total cost of maintenance is difficult to calculate due to the number of factors involved.

Maintenance activities fall into two main categories, namely corrective maintenance (CM) and preventive maintenance (PM). CM is performed when action is taken to restore a system to a working condition after the system has failed. A CM maintenance concept can often lead to high maintenance related costs for the following reasons (Tsang 1995):

- The high cost of restoring equipment to an operable condition under crisis situation.

- The secondary damage and safety/health hazards inflicted by the failure.
- The penalty associated with lost production.

Reliability-centred maintenance (RCM) is a process which, when implemented effectively can produce PM concepts and reduce these kinds of CM associated costs. Developed in the late 1960's by the US aviation industry (Nowlan & Heap 1978), RCM is a systematic approach used to optimise PM strategies (Ben-Daya 2000). Moubray (1997) defines RCM as 'a process used to determine what must be done to ensure that any physical asset continues to do whatever its users want it to do in its present operating context'. Smith (1993) lists the four predominant features of the RCM methodology as:

1. Preserve functions of a system.
2. Identify failure modes that can defeat the functions of the system leading to failure.
3. Prioritise function need.
4. Select only applicable and effective maintenance tasks to be completed.

Mokashi et al. (2002) suggest some possible problems with the application of RCM in the marine industry, specifically with application to ships. Furthermore it is suggested that total productive maintenance (TPM) could be a good facilitator for implementing RCM.

TPM and its development started in Japan in the 1970's where it significantly improved the effectiveness and profitability of several Japanese companies. Nakajima (1988) defines TPM as productive maintenance involving total participation. Rich (1999) gives the five main objectives of TPM relating to equipment maintenance as:

1. To maximise the overall effectiveness of equipment within the manufacturing system.

2. To establish a systematic and comprehensive approach to productive maintenance for the entire life cycle of the equipment from purchase to disposal.
3. To integrate and form alliances with other departments within the manufacturing system such that the implementation of maintenance routine can be streamlined and become more effective.
4. The development of a company-wide planning process which is both a top-down means of managing the business and a bottom-up process of improving the production by operator involvement.
5. The development of TPM is that facilitated through the development of natural and cross functional teams or small groups who are capable of working autonomously within the factory.

Although these objectives are outlined in consideration of manufacturing systems the ethos of TPM is certainly applicable within the marine environment. The two main features of TPM are equipment management and empowerment of employees. The correct management of equipment in the marine environment will ultimately affect the availability of the ship's systems. The second idea of empowerment of employees is a feature already well established in the marine sector. Ben-Daya (2000) outlines problems in the organizational line between maintenance, production and engineering leading to inefficiency and higher costs. Furthermore it is suggested that operators be trained to perform mechanical maintenance tasks and vice-versa, developing a relationship between operations and maintenance. On board a ship this operations-maintenance relationship already exists in the engineering department, given that engineers operate as well as maintain machinery. This could however be extended so that deck officers were also trained to be able to perform maintenance tasks. Mokashi et al. (2002) also present this idea of dual competency marine officers.

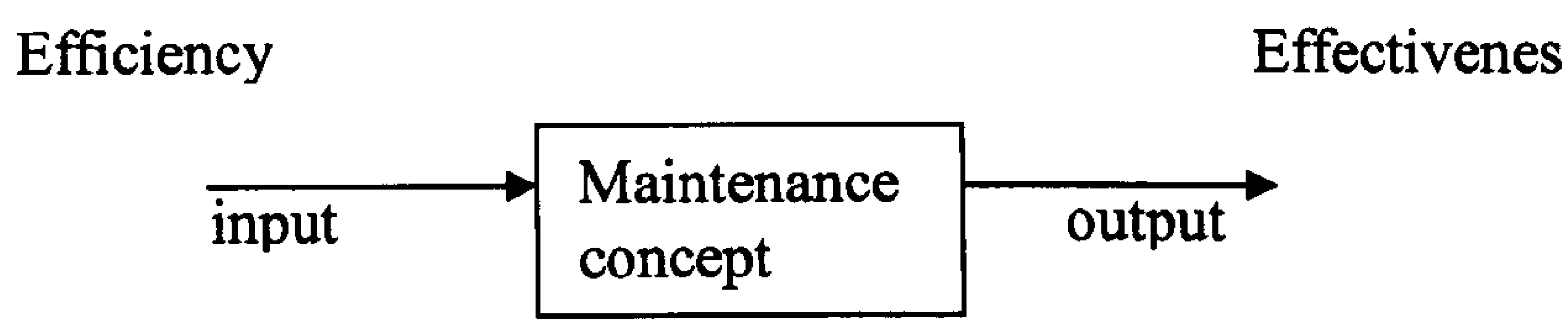

Fig 2.1 Optimum maintenance concept

Fig 2.1 shows the 'optimum' maintenance concept presented in Pintelon (1999). The efficiency of a concept is dependent upon the input. Ultimately all maintenance concepts are dependent upon appropriate information being available concerning equipment. To enable marine engineers to make educated informed decisions concerning maintenance decisions, methods must be developed which provide the marine engineer with information about unreliability, availability and downtime.

2.4 Decision Making Methods

Decisions and decision making forms a basic part of all interactions with the outside world. All people make a number of decisions everyday with little structured thought or consideration for consequence. Often decisions that we make as human beings are rationalised based on past experience. However there are often situations where decisions have to be made based on no past experience or prior knowledge. Decision making and particularly decision making in areas of little or no previous experience has become a mathematical science (Figuera et al. 2005), (Saaty 2008). Theses decision problems often represent complex multi criteria situations (Anderson et al. 2003), (Saaty 1980). The Analytical Hierarchy Process (AHP) is an example of a process which aids decision making. AHP is especially appropriate for complex decisions which involve the comparison of decision criteria that are difficult to quantify (Pillay & Wang 2003). It breaks down a decision problem using the following steps (Saaty 2008):

1. Define the problem and determine the kind of knowledge sought.

2. Structure the decision hierarchy from the top with the goal of the decision, then the objectives from a broad perspective, through the intermediate levels (criteria on which the subsequent elements depend) to the lowest level (which usually is a set of alternatives).
3. Construct a set of pairwise comparison matrices. Each element in an upper level is used to compare the elements in the level immediately below with respect to it.
4. Use the priorities obtained from the comparisons to weigh the priorities in the level immediately below. Do this for every element. Then for each element in the level below add its weighted values and obtain its overall or global priority. Continue this process of weighting and adding until the final priorities of the alternatives in the bottom most levels are obtained.

AHP has been applied in many areas, including the marine environment (Brown & Haugene 1998), (Lim et al. 2004), (Ugboma et al. 2006), risk and safety analysis (Sii et al. 2001), (Sii & Wang 2003), transportation systems & policies (Arslan & Khisty 2005), (Lambert et al. 2006), (Shang et al. 2004), (Berrittella et al. 2007), military applications (Cheng 1997), (Cheng et al. 1999), conflict resolution (Saaty 2007), pipeline feasibility studies (Dey & Gupta 2001), education (Drake 1998), (Grandzol 1998) and numerous others. Attempts have been made to create a reference text as a source of various examples of structured decision making using AHP (Saaty & Forman 1993), (Saaty & Ozdemir 2005).

2.5 Justification for the Research

The literature survey has shown that there are many maintenance techniques available to marine engineers. CM has been shown to lead to high maintenance related costs (Tsang 1995). As a result CM is not often seen in the modern ship's engine room. RCM could play a strong role in the marine industry, where maintenance is carried out predominantly to preserve function and failure modes are identified which can cause a system to fail. Mokashi et al. (2002) identify some problems associated with the application of RCM. TPM is suggested as a good facilitator of RCM. PM strategies implemented onboard the ship are often done so according to manufacturer's recommended service and overhaul intervals. Service and overhaul typically involves a system being taken offline completely, dismantled and inspected. O-rings, bearings and other 'consumables', within the piece of equipment are often replaced with no regard for condition. All the maintenance techniques shown require application of the method before the effect on the system can be ascertained. The optimum maintenance concept (Pinelton 1999) is introduced, in order for progress to be made toward the ideal more information is needed. All of the maintenance concepts outlined and currently available to marine engineers are not flexible and powerful enough to allow trial runs of 'what if' scenarios. A MC model can be developed for any piece of equipment, the main limitations being the ability of the analyst and the computer power available. However the length of simulation run-time considered appropriate is dependent upon each different situation and application. To a ship design office a three day run-time may be considered acceptable if the information obtained is of enough importance. Conversely if a model is needed to make real-time decisions then this could be considered inappropriate. Research in the marine industry is often impeded by a lack of data; MCM can be applied in such cases and used to produce quantitative estimates of the effects of various decisions/configurations on system reliability.

The use of MCM in the nuclear and chemical industry is widespread, some examples of which are discussed in this chapter. The simulation of an entire system, if conducted properly, offers a virtual model for engineers to work with. This has obvious advantages, if the model truly reflects the behaviour of the real system. Optimal maintenance scheduling can be decided upon by working with the simulation. MCM is powerful enough to deal with numerous sources of data, expert judgement and subjective data can be used to deal with any problems relating to a lack of historical data. Furthermore the simulation program could be linked to current failure databases, update itself as the system ages and more accurately reflect the system as more information becomes available. The MCM can in theory deal with any size of system. Despite all the advantages of this method, well proven in other industries, the volume of work done with MCM in the marine industry considering system analysis and system reliability is very small.

This thesis has addressed this problem by practically applying MCM to a complex marine system. Chapter 5 implements MCM integrated with delay-time methods to provide optimum inspection intervals based on minimising downtime.

AHP is a decision making method which is based on importance ranking of factors that are intrinsic to the decision making problem. Past experience and the effect of a decision in similar circumstance often have a bearing on future decisions. Saaty (2008) states that there are two possible ways to learn about anything. The first is to examine and study it in itself to the extent that it has various properties, synthesise the findings and draw conclusions from such observations about it. The second is to study that entity relative to other similar entities and relate it to them by making comparisons. This work and the models presented address the later, by modelling systems estimations of the systems various behaviours are made this are then related to the real system via comparison. This is seen in many areas of the marine industry, new ship designs are often, in the first instance, based on existing designs which serve a similar purpose. Chief Engineers on board a vessel will make decisions about system operation and maintenance often based on their experience. The qualification system within the marine industry, concerning ship's officers, is based both on technical knowledge and time of service to allow crew members to obtain the necessary experience. When making decisions regarding systems of which we have limited past experience and no similar systems exist, then the engineer faces a problem. Simulation can address this problem as simulation models can be used to provide quantitative estimates of the effects of decisions in terms of unreliability, system downtime and cost. Chapter 6 gives a practical example of how this can be achieved. The complex system presented in Chapter 4 is simulated and the effect of different staff levels on the system downtime is assessed. The results presented in terms of system downtime and costs suggest a staffing level which is sufficient to address maintenance tasks.

Chapter 3 – Monte Carlo Methods

Summary

This chapter serves as an extension to the literature review and the basis of MCM will be established. The theory presented forms the fundamental theoretical basis of all the technical chapters presented hereafter. In order to facilitate the demonstration of Monte Carlo Methods an example is presented consisting of three components in parallel.

3.1 Introduction

In order to demonstrate the viability of Monte Carlo techniques in the marine environment case studies must be carried out, with generic models which demonstrate some of the information that the models can yield. Obviously there are an infinite number of areas where the technique could be applied in the marine environment. The studies undertaken will concentrate on main engine failure and as such it is necessary to indicate all possible failures which could lead to this undesired event. British shipbuilders research department published work concerning the failure analysis of an entire ship, included was a fault tree analysis of a main engine failure and has been used as a basis for the work. The fault tree is shown in fig 3.1.

Fig 3.1 Fault tree analysis concerned with the failure of a ships main engine (British Shipbuilders 1990)

It can be seen from the fault tree that failure of the sea water pumps would certainly lead to a failure of the main engine. In light of this it was decided to model the cooling system and perform a Monte Carlo analysis in attempt to demonstrate the valuable contribution which the technique could have in the marine industry. The first part of the chapter looks at cooling systems on board in general so that a generic cooling system can be devised. Although the cooling system used in the analysis is not case specific the generic nature of it is such that it can be applied to most specific systems with minor modifications. The second part of the chapter reviews the methodology essential to the Monte Carlo techniques implemented. The final part will show the system which has been analysed, the results, discussion and conclusions.

3.2 Engine Cooling Systems – Brief Overview

Diesel engines generate a great deal of heat, only one third of which is converted to useful work (Calder 2007). The remaining two thirds must somehow be released to the environment so that the engine temperatures do not become dangerously high. Excessive temperatures can break down lubricating oils, causing engine seizure or cracking of the cylinder head. Just under half the heat is lost to the exhaust and just under half is lost to the cooling system while the rest of the heat is lost in various ways such as radiation from engine surfaces.

Cooling systems for the main engine on board a ship historically used saltwater. However the cooling water in such systems acted as an electrolyte. In a cooling water system many different metallic elements are used thus galvanic elements are formed and galvanic corrosion occurs. Sea water also contains various calcium salts and at high temperatures will quickly deposit lime on surfaces of the cooling water system. These lime deposits have a detrimental effect on the heat transfer in the system. Also, sea water contains a lot of air which is released upon heating causing corrosion. Due to these problems and the complexity of the appropriate solutions the marine industry looked for alternative solutions.

Now almost all ships have a fresh water cooling system and some even use distilled water. The fresh water is circulated through the diesel jacket and other parts of the engine and then the sea water is used to cool the fresh water. This means that the sea water is contained within an isolated system, its only interaction being through the seawater cooler. The fresh water system greatly reduced problems to do with corrosion when compared to sea water only systems. However fresh water systems still suffer from problems with corrosion; also the freshwater must be kept alkaline to prevent scale formation. There are two main types of cooling systems on board ships. The first type has a dedicated freshwater cooler and the second uses mixing to cool the freshwater via a thermostatic valve.

Cooling water pumps are either driven by the engine or driven by an electric motor from the 415v supply. The freshwater circulates through the engine cooling it down and is usually maintained at a temp of 73-82°C known as the High Temp (HT) system. The temperature of the HT cooling water is regulated in one of the following two ways depending on the system in use:

1. System with dedicated freshwater cooler – The HT water enters the cooler where it 'dumps' heat into the Low Temp (LT) freshwater system. The LT freshwater system is used to cool lubricating oils etc and is maintained at a lower temp than that of the HT. The LT water then enters a second cooler, known as the sea water cooler, where it 'dumps' heat into the sea water system. This means that the sea water is still implemented as an effective

cooling method but it is isolated so as not to cause corrosion problems. Fig 3.2 shows a simple schematic representation of the system.

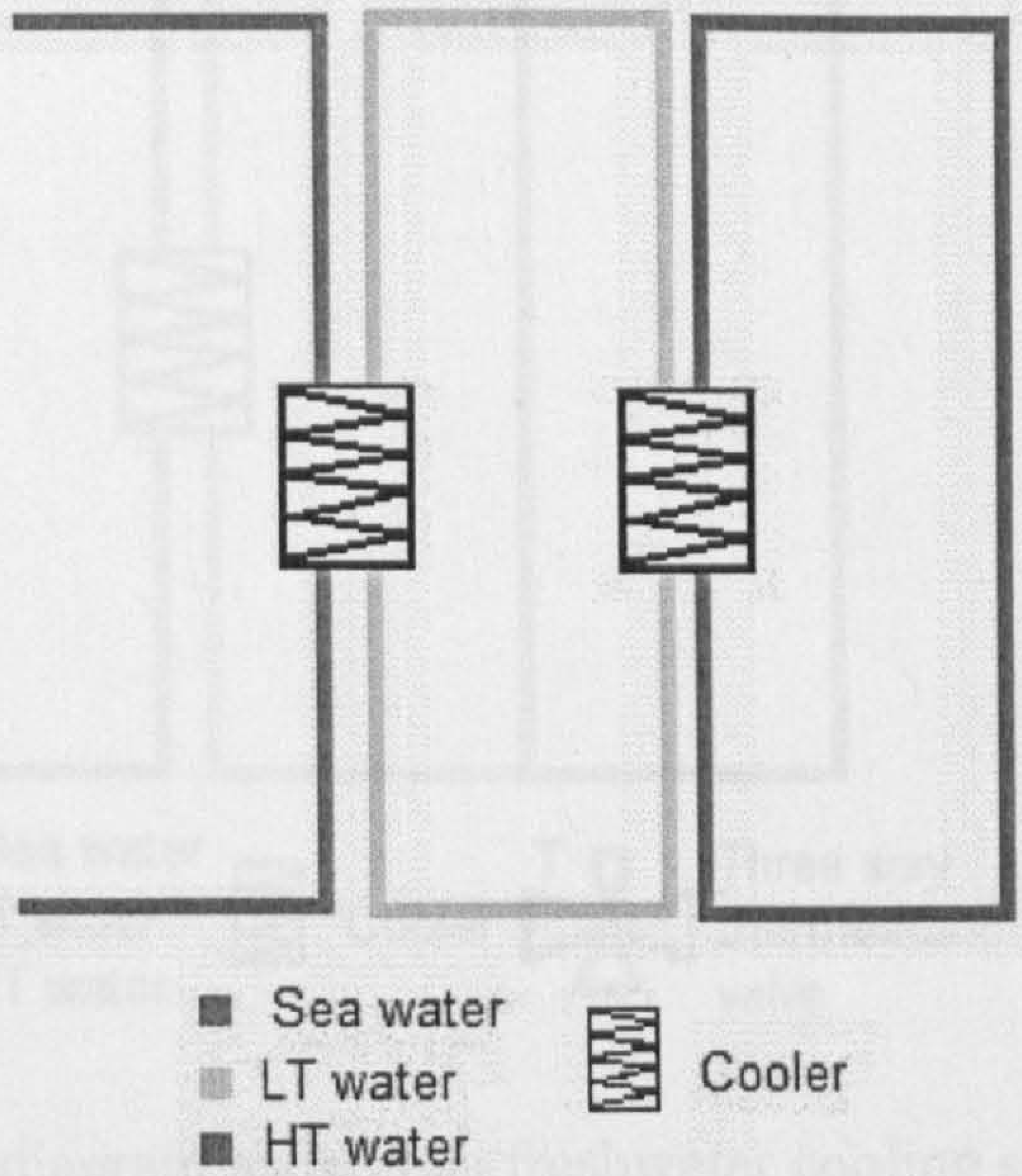

Fig 3.2 Schematic diagram showing a freshwater cooling system which uses a dedicated freshwater cooler

2. Mixing System – In the mixing system when the temperature of the HT reaches a certain level a thermostatic valve is opened, this allows water from the LT system to mix with that of the HT. Once the HT is sufficiently cooled the thermostatic valve closes. The LT water is now cooled through a dedicated sea water cooler. It should be noted that the mixing only ever occurs between the HT and LT which are both freshwater systems. Fig 3.3 shows a simple schematic diagram of the system.

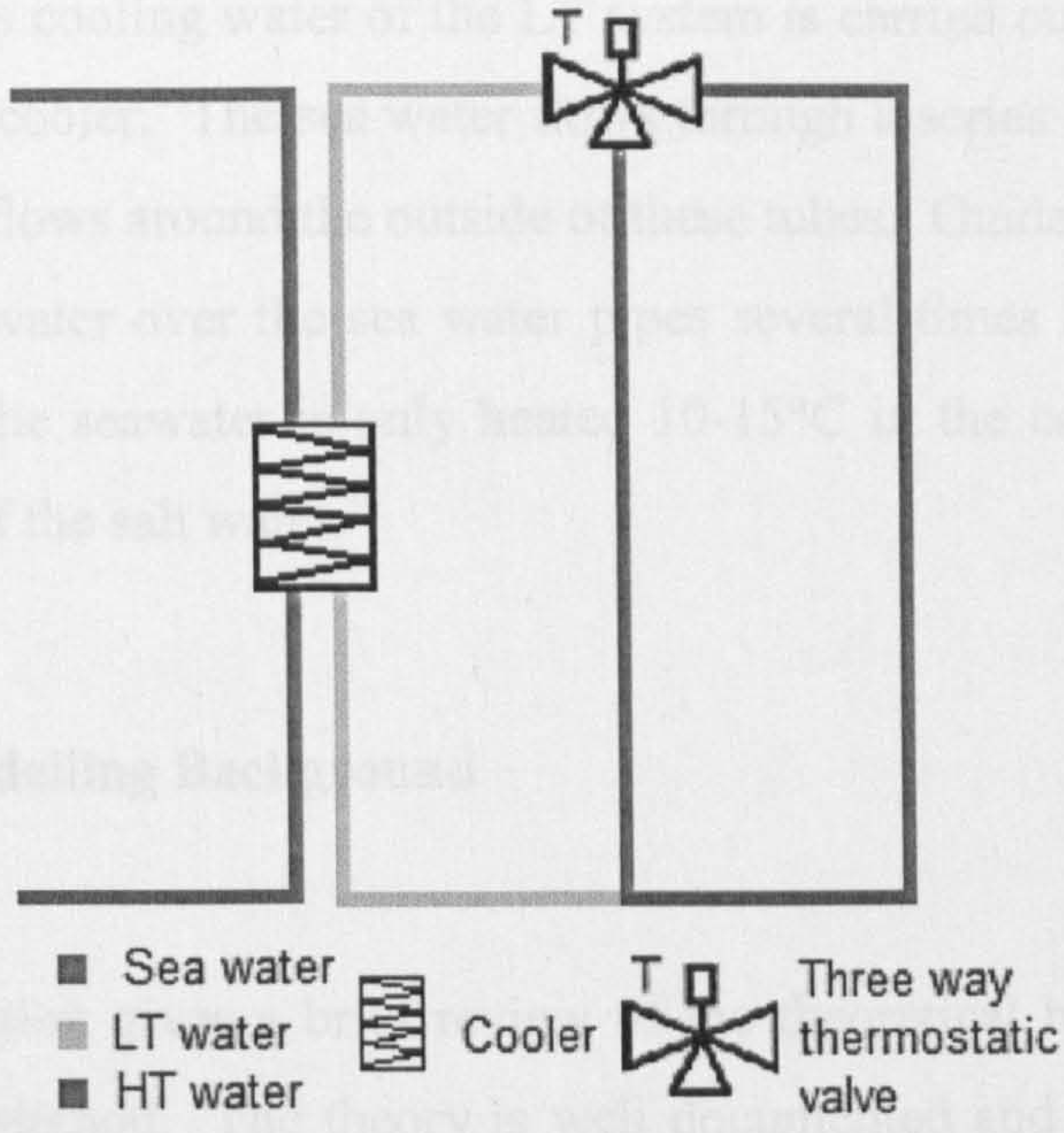

Fig 3.3 Schematic diagram showing a freshwater cooling system which uses a thermostatic valve to allow the mixing of the HT and LT circuits

The temperature of the cooling water is vital to the longevity and continued function of the engine. It is not as simple as keeping the water as cold as possible; water that is too cold can lead to thermal shocking in the engine materials and condensation of water and acids on the cylinder bores which can wash away the lubricating film. Also, if the cooling water is too hot then it would not remove an adequate amount of heat from the engine. This will increase wear rates and the formation of scale within the engine. The inlet and outlet temperature of the engine is monitored allowing for a fast response to changes in temperature.

All systems have a header tank which allows for expansion and is used to top up the cooling water levels; the header tank is manual fill only with a low level alarm. The cooling system can also contain a heater to be used to warm the engine through before starting. Central cooling systems are such that the fresh water cooling is fed through the main engine and all of the auxiliaries as well, e.g. generators. In this type of system the generators are normally running when the main engines are not providing the heat required to warm through an engine before starting it.

In both of the cases cooling water of the LT system is carried out by the sea water flowing through a cooler. The sea water flows through a series of tubes while the hotter fresh water flows around the outside of these tubes. Guide plates are used to force the cooling water over the sea water pipes several times and the process is regulated so that the seawater is only heated 10-15°C in the cooler reducing the corrosive effects of the salt water.

3.3 Review of Modelling Background

The following section gives a brief review of the theoretical background behind the Monte Carlo Method. The theory is well documented and can be found in a number of texts (Rubinstein 1981), (Manno 1999), (Dubi 2000), (Marseguerra & Zio 2002).

In mathematics, a probability density function represents a probability distribution in terms of integrals.

A probability distribution has density f if f is a non-negative Lebesgue-integrable function $\Re \to \Re$ such that the probability of the interval $[a,b]$ is given by:

$$\int_a^b f(x)dx \tag{3.1}$$

for any two numbers a and b. This implies that the total integral of f must be 1. Conversely, any non-negative lebesgue-integrable function with total integral 1 is the probability density of a suitably defined probability distribution.

The probability density function is also the derivative of its related cumulative distribution function, F(x) i.e.

$$f(x) = F'(x) = \frac{dF(x)}{dx} \tag{3.2}$$

The probability density function is related to the random variable rather than the interval. f(x) is non-negative since F(x) is non-decreasing. The probability density function is constrained by the normalisation condition, since F(∞)=1. Then:

$$\int_{-\infty}^{\infty} f(x)dx = 1 \tag{3.3}$$

If a small interval of Δx around a given point x_c is considered and the interval is small enough so that the variation of f(x) in it is negligible the probability that the random variable will be realised in that interval, $P(x \in \Delta x)$, can be estimated using the following equation:

$$P(x \in \Delta x) \cong f(x_c)\Delta x \tag{3.4}$$

3.3.1 The Exponential Distribution

Exponential distributions are used in situations where the analyst is unconcerned by the past of the system, Zio (2002) describes the distribution as being characterised by 'lack of memory'. Examples of exponential decays occur naturally in a number of different branches of physics, e.g. fall in amplitude of a harmonic vibration, the fall in voltage of a charged capacitor leaking through a high resistance and fall in activity of a radioactive decay (Manno 1999).

The probability density function of an exponential distribution has the form:

$$f(t)\begin{cases}\lambda e^{-\lambda t} & t \geq 0 \\ 0 & t < 0\end{cases} \tag{3.5}$$

Where λ is the failure rate equal to the reciprocal of the Mean Time Between Failure (MTBF).

The cumulative density function of an exponential distribution has the form:

$$F(t)\begin{cases}1-e^{-\lambda t} & x\geq 0\\ 0 & x<0\end{cases} \tag{3.6}$$

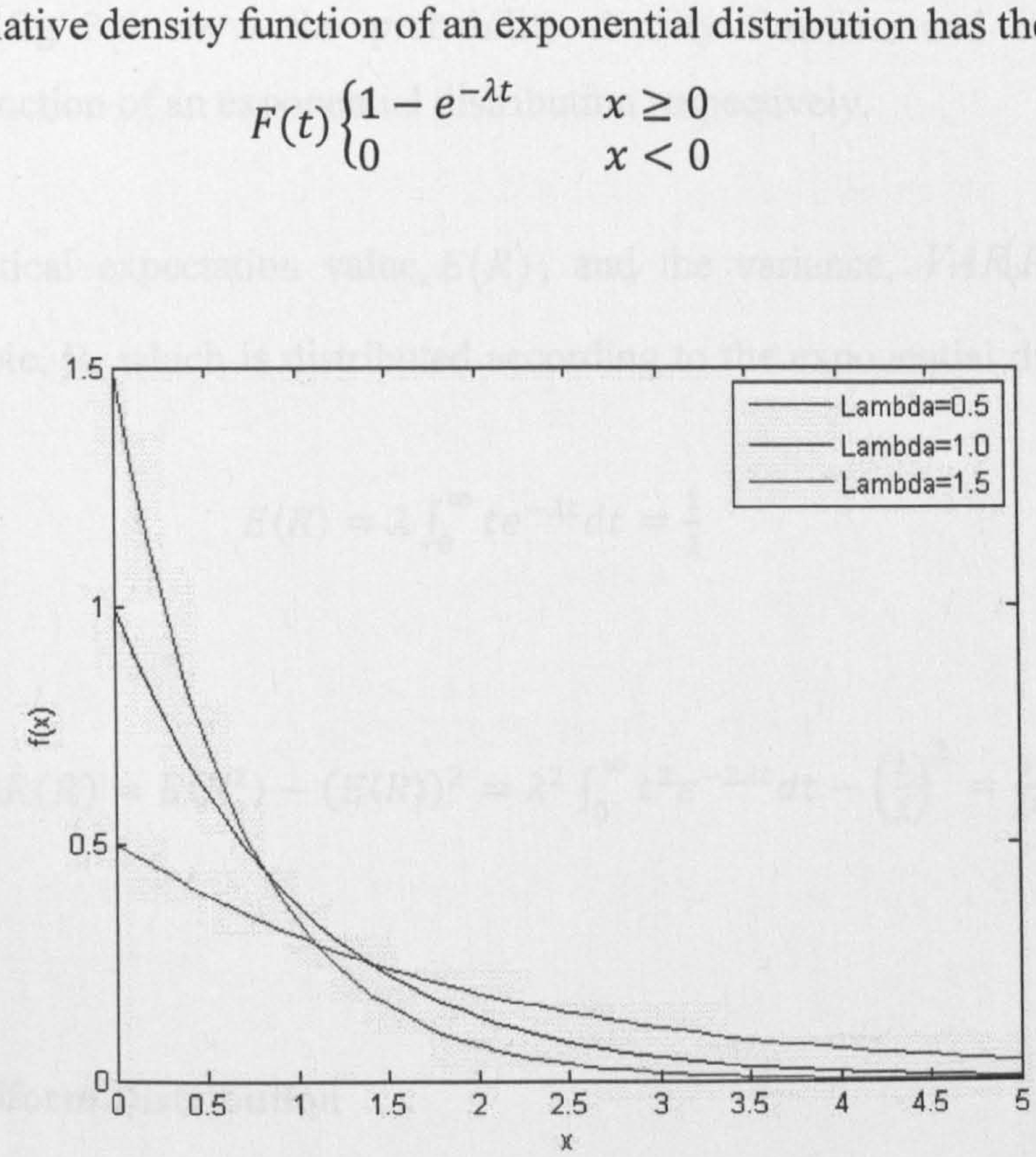

Fig 3.4 Diagram showing the probability density function of an exponential distribution

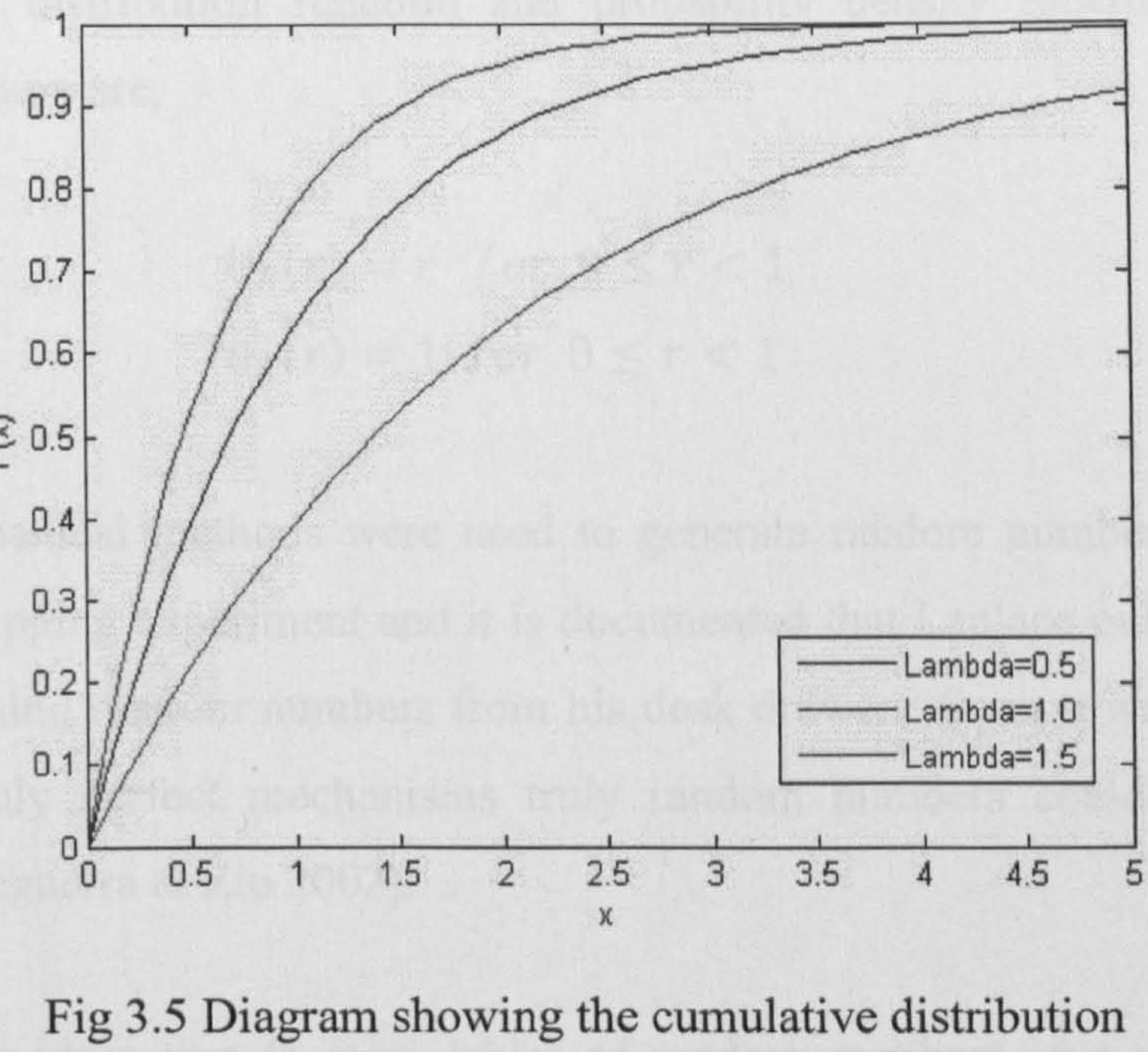

Fig 3.5 Diagram showing the cumulative distribution function of an exponential distribution

Fig 3.4 and fig 3.5 show the probability density function and cumulative distribution function of an exponential distribution respectively.

The mathematical expectation value, $E\langle R\rangle$, and the variance, $VAR(R)$, of the random variable, R, which is distributed according to the exponential distribution are given by

$$E\langle R\rangle = \lambda\int_0^\infty te^{-\lambda t}dt = \frac{1}{\lambda} \tag{3.7}$$

and

$$VAR(R) = E\langle R^2\rangle - (E\langle R\rangle)^2 = \lambda^2\int_0^\infty t^2 e^{-2\lambda t}dt - \left(\frac{1}{\lambda}\right)^2 = \frac{1}{\lambda^2} \tag{3.8}$$

3.3.2 The Uniform Distribution

Amongst all of the distributions available the uniform distribution allows us to obtain a random variable obeying any other distribution (McGrath, 1975).

The cumulative distribution function and probability density function of the uniform distribution are,

$$U_R(r) = r \quad for \;\; 0 \leq r < 1 \tag{3.9}$$

$$u_R(r) = 1 \quad for \;\; 0 \leq r < 1 \tag{3.10}$$

Originally mechanical methods were used to generate random numbers. Buffon used the pin dropping experiment and it is documented that Laplace pulled pieces of paper containing random numbers from his desk drawer. Soon it was realized that without truly perfect mechanisms truly random numbers could never be achieved (Marseguerra & Zio 2002).

One of the first ideas was to store tables of random numbers in a computer's memory to access and use, however this proved a very time consuming process and required a lot of memory at a time when memory was still very expensive.

In 1956 Von Neumann suggested that computers could be used to generate their own random numbers. He proposed that a function could be found, $g(\cdot)$, such that it generated the next random number in the sequence, there-by automating the whole process.

$$R_{k+1} = g(R_k) \tag{3.11}$$

where

$g(\cdot)$ represents an unknown function

Computers can now produce random numbers relatively quickly with the use of a congruential generator.

The congruential generator has the following form (Lehmer 1951):

$$R_{k+1} = (aR_k + c)\ mod\ m \tag{3.12}$$

The variables a,c and m must be defined. Also it is possible to define an initial or 'seed' value for the generator.

Rubinstein (1981) tells us that in order for the generator to produce suitable random numbers the following conditions should be met:

1. c is relatively prime to m, that is, c and m have no common divisor.
2. $a \equiv 1 (\mathrm{mod}\ g)$ for every prime factor g of m.
3. $a \equiv 1 (\mathrm{mod}\ 4)$ if m is a multiple of 4.

Suppose that we wished to set up a congruential generator with the following variables,

$X_0=2$	$a=5$
$c=1$	$m=16$

Once ran, the generator will produce the values shown in table 3.1.

Table 3.1 Congruential generator values

X	R_{k+1}
$x_0=2$	$R_0 = {}^{2}/_{16}$
$x_1=11$	$R_1 = {}^{11}/_{16}$
$x_2=8$	$R_2 = {}^{8}/_{16}$
↓	↓
$x_{15}=13$	$R_{15} = {}^{13}/_{16}$
$x_{16}=2$	

As can be seen in table 3.1 the sequence repeats itself on the sixteenth execution of the algorithm. This is in fact true for all the different combinations of numbers for the variables, a, c and m. The sequence is periodic with period m, for any given seed value. This means that the congruential generator produces a sequence which is in fact deterministic and is therefore not a random sequence at all. However if m is selected as large as possible, relative to the amount of samples taken then it is possible to achieve pseudo-random numbers. Pseudorandom sequences typically exhibit statistical randomness while being generated by an entirely deterministic causal process. Since most computers now utilise a decimal digit system it is possible to select $m = 10^{\beta}$ where ß denotes the word length of the particular computer, it is interesting to note that a Monte Carlo simulation which works perfectly on a computer can produce unexpected results on a second machine due to different word lengths. Since in all cases, irrespective of the word length, m will be a great deal larger than 1, the generator will always provide statistically random numbers distributed between $[0, 1)$. This is very important as these uniformally generated random numbers can be used to produce random numbers which will conform to all other distributions.

3.4 Inverse Transform Method

It can be shown, that in utilising the inverse transform method, the uniform distribution can be used to sample from any other given distribution for both continuous and discrete distributions (Rubenstein 1981), (Marseguerra & Zio 2002).

3.4.1 Continuous Distributions

By taking the inverse of the required cumulative distribution function, using a sample, R, from the uniform distribution the algorithm works backwards to obtain a value x, from the desired distribution.

$$P\{X \leq x\} = P\{F_X^{-1}(R) \leq x\}$$

$$\Leftrightarrow P\{X \leq x\} = P\{R \leq F_X(x)\} \tag{3.13}$$

However from (3.9), $P\{R \leq r\} = r$

$$\therefore P\{R \leq F_X(x)\} = F_X(x) \tag{3.14}$$

This is shown graphically in fig 3.6.

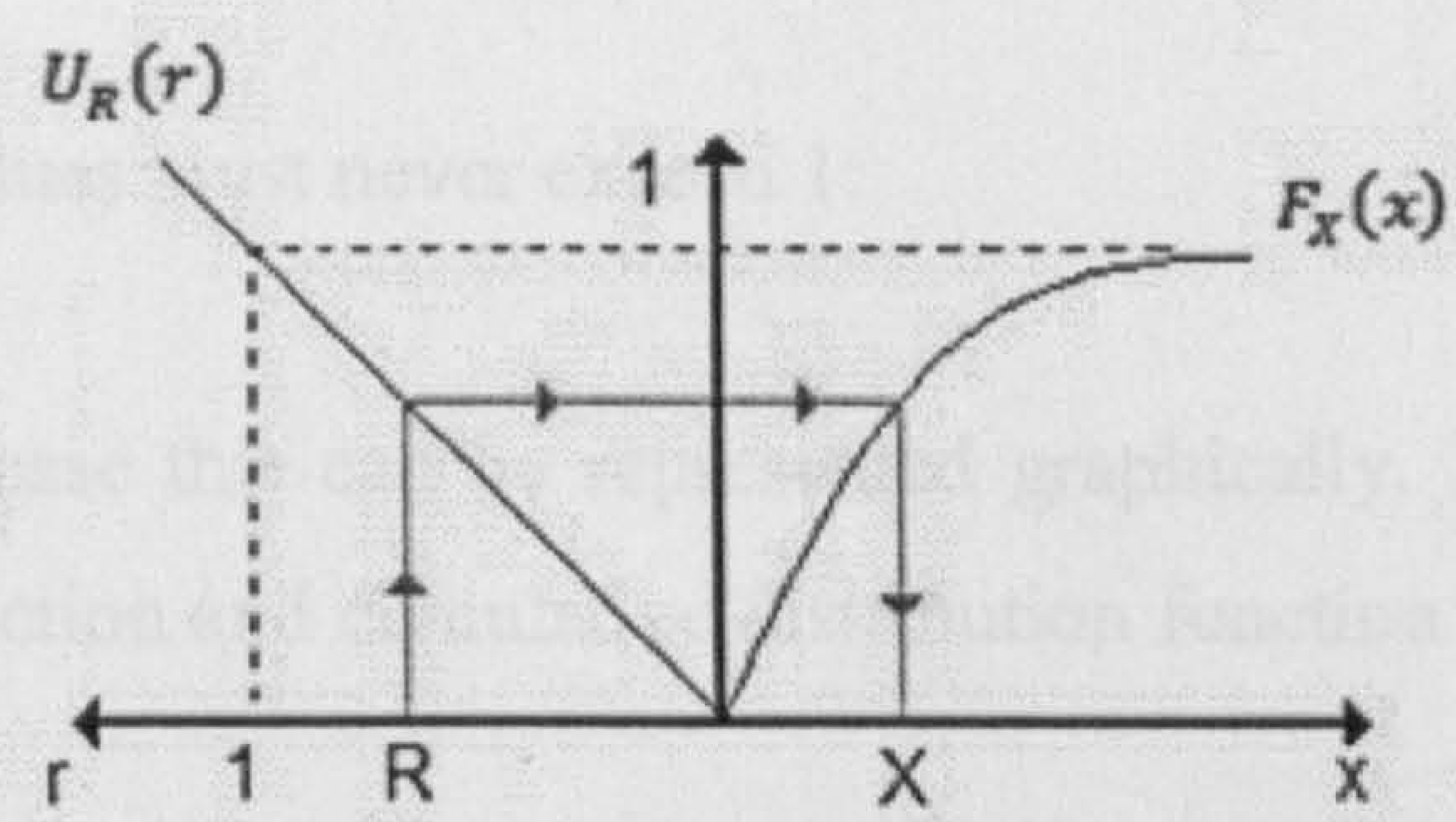

Fig 3.6 Graphical representation of the inverse transform sampling method for a continuous distribution

The inverse transform method can be summarised by the following steps.

- Identify the desired distribution, F_X, and find the inverse of its cumulative distribution function, F_X^{-1}.
- Sample a value from the uniform distribution, R.
- Calculate $X = F_X^{-1}(R)$ to give a sample realised from the desired distribution.

3.4.2 Discrete Distributions

Let X be a random variable which can only have the discrete values x_k, where k=0, 1, 2,….. , n with probabilities,

$$f_k = Pr\{X = x_k\} \geq 0 \tag{3.15}$$

$k = 0,1, \ldots.$

Ordering the sequence so that $x_{k-1} < x_k$ the cumulative distribution is

$$F_k = Pr\{X \leq x_k\} = \sum_{i=0}^{k} f_i = F_{k-1} + f_k \tag{3.16}$$

$k = 0,1, \ldots.$

where the normalisation condition of the cumulative distribution function now gives:

$$Lim_{k \to \infty} F_k = 1 \tag{3.17}$$

i.e. the sum of the k values must never exceed 1.

As in the continuous case this can be represented graphically. Fig 3.7 shows a probability density function and cumulative distribution function of a discrete data set.

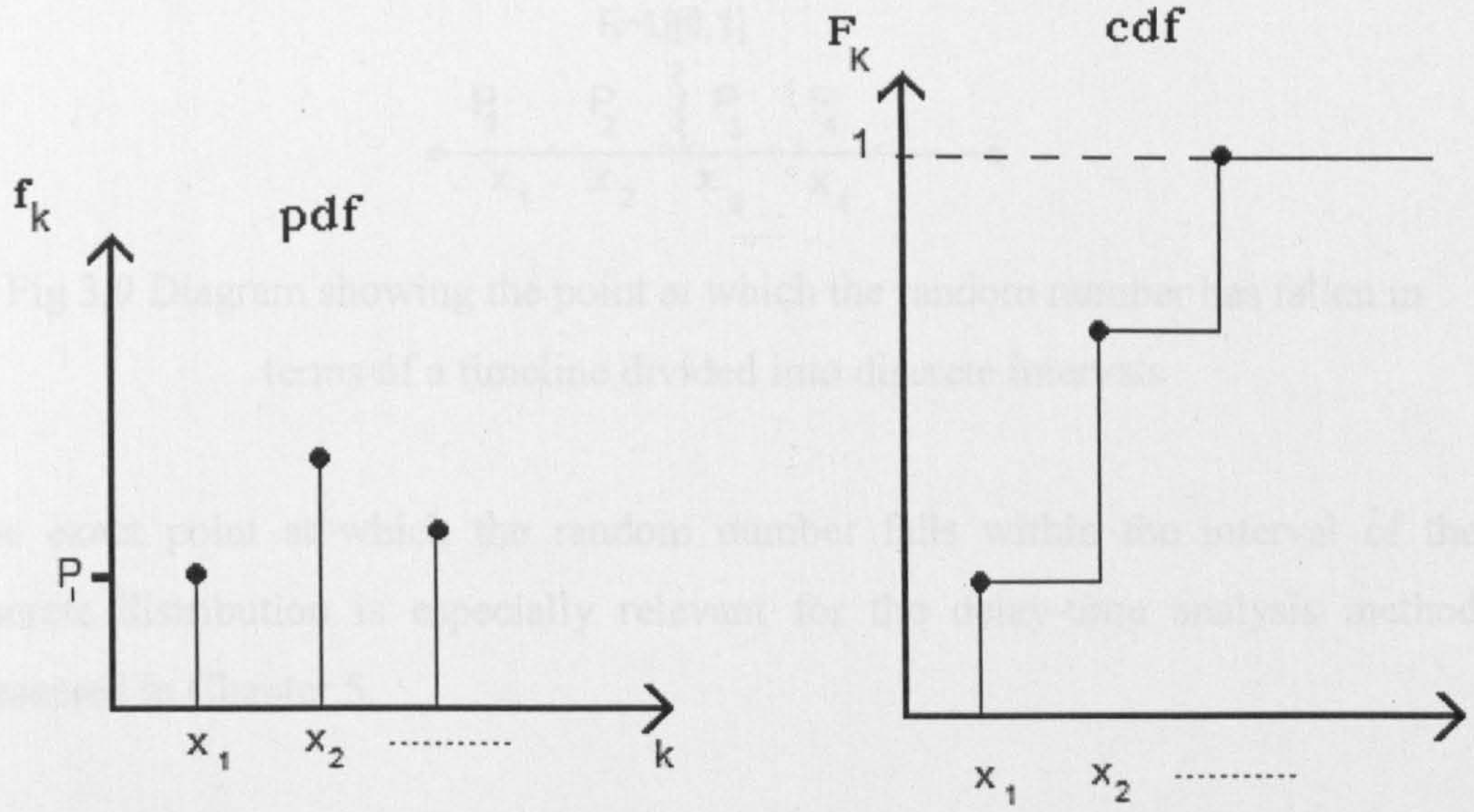

Fig 3.7 Probability density function (pdf) and cumulative distribution function (cdf) of a discrete distribution

Sampling is done in the same way as for the continuous case. However for discrete distributions R will fall in an interval $[F_{K-1}, F_K]$. This is shown in Fig 3.8.

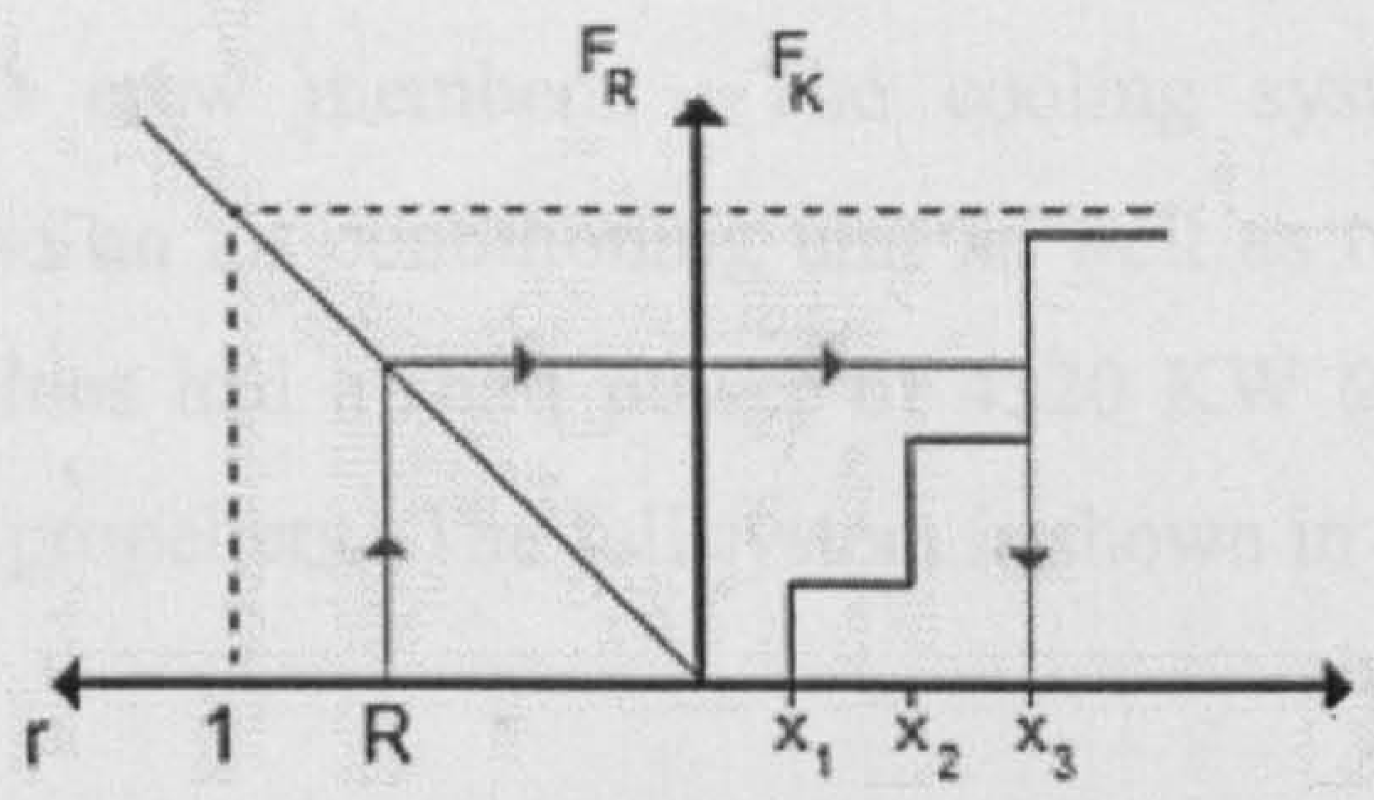

Fig 3.8 Graphical representation of the inverse transform sampling method for a discrete distribution

In certain cases it may be important to know how near or far from the end of an interval R falls. This is shown in fig 3.9.

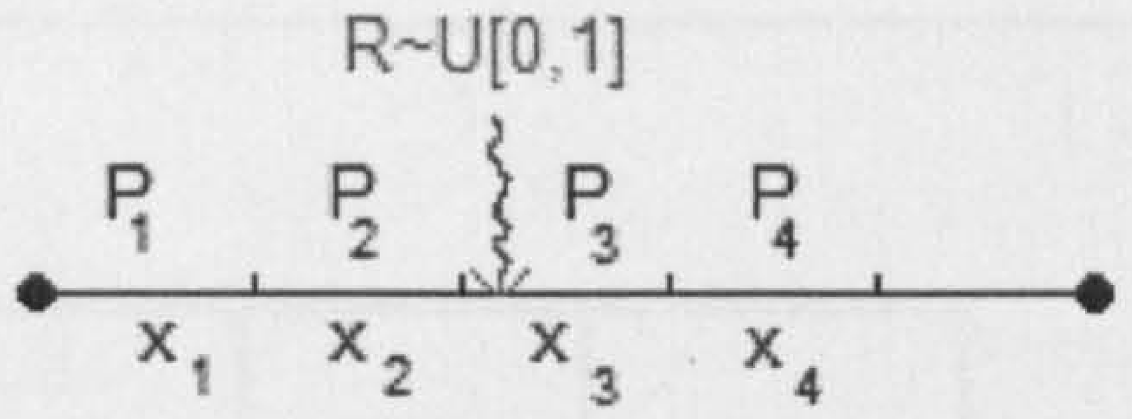

Fig 3.9 Diagram showing the point at which the random number has fallen in terms of a timeline divided into discrete intervals

The exact point at which the random number falls within the interval of the discrete distribution is especially relevant for the delay-time analysis method presented in Chapter 5.

3.5 Cooling System – Generic Model

In the following section a cooling system will be outlined for the purpose of the Monte Carlo analysis. The system is taken from the MV Hamnavoe a Ro-Ro passenger ferry on which the researcher served time during a cadetship. The ship had length overall of 112m and a beam of 18.60m. The passenger capacity was 600 passengers and 40 crew members. The cooling system serviced the lubricating oil pumps and an air conditioning unit as well as two MAK 9M32C main engines. The engines had a shaft power of 4320 KW @ 600 rpm which drove two variable pitch propellers. The full system is shown in fig 3.10.

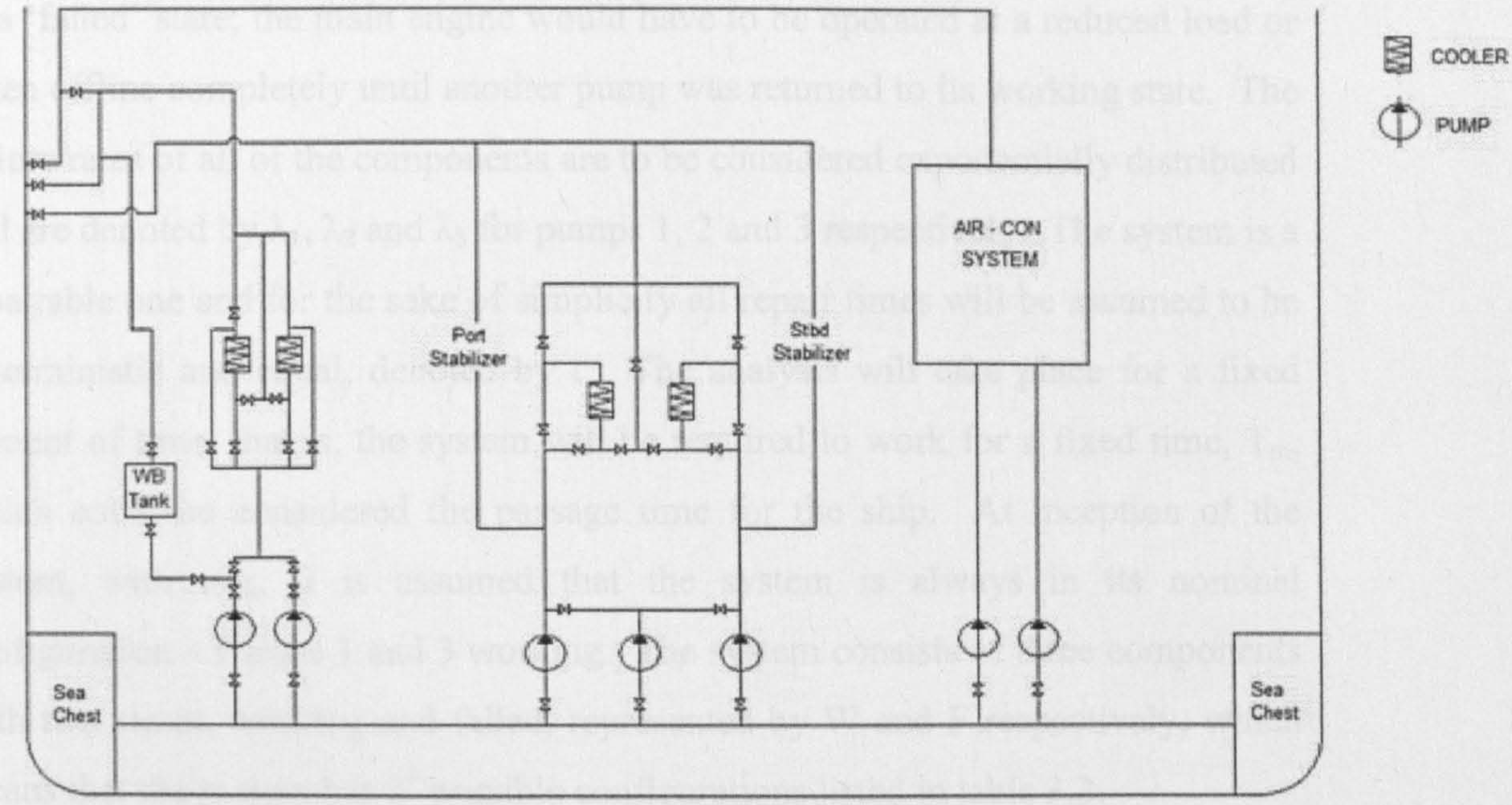

Fig 3.10 Schematic diagram showing the cooling system under consideration (Repeated in Fig 5.15)

For the model the system will be reduced to the three pumps which supply the coolers for the main engine. All valves are assumed to operate perfectly at all times. The system can now be represented by that shown in fig 3.11.

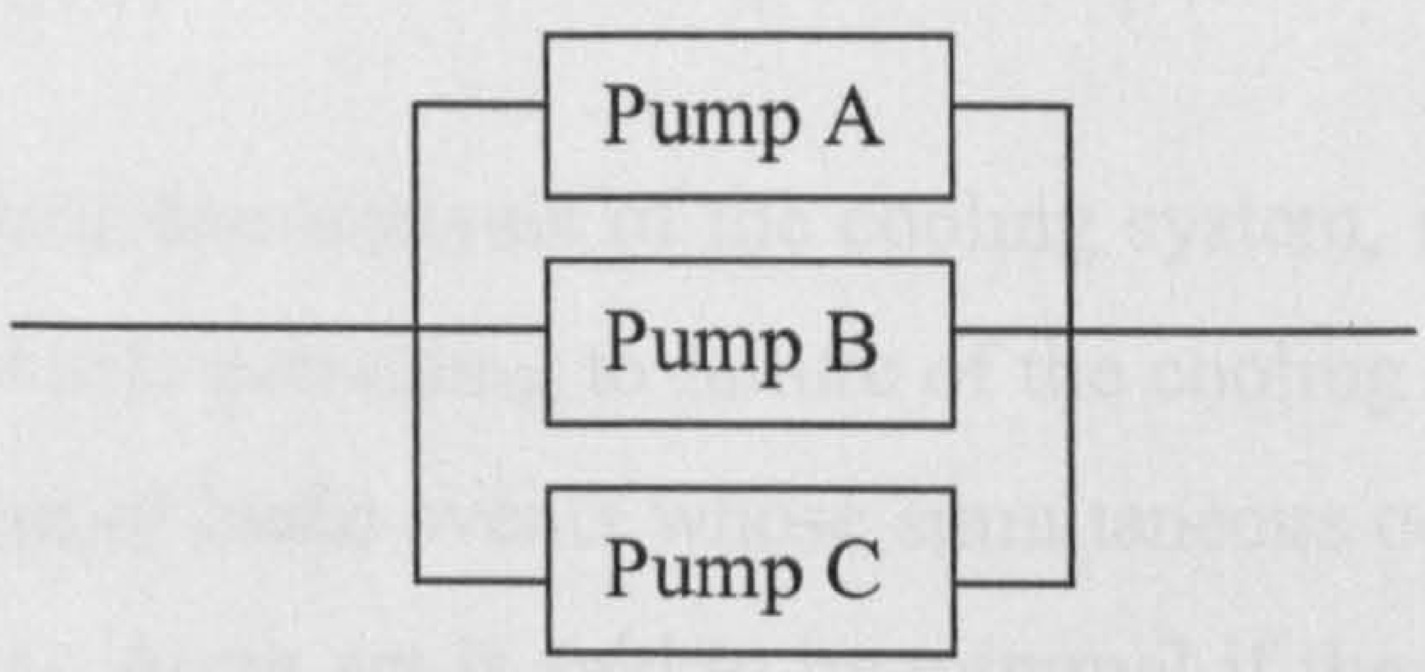

Fig 3.11 Diagram showing the simplified version of the cooling system

For ease of notation let pumps A, B and C be represented by 1, 2 and 3 respectively. The system is in its nominal configuration when pumps 1 and 3 are working and pump 2 is a 'cold standby' unit i.e. cannot fail while in standby. The main engine requires a flow rate such that a minimum of two pumps must be functioning for adequate cooling to take place. The cooling system will still operate on a single pump but for the purpose of this model it is assumed that when only a single pump is available for supply, that the system can be considered to be in a 'failed' state; the main engine would have to be operated at a reduced load or taken offline completely until another pump was returned to its working state. The failure rates of all of the components are to be considered exponentially distributed and are denoted by λ_1, λ_2 and λ_3 for pumps 1, 2 and 3 respectively. The system is a repairable one and for the sake of simplicity all repair times will be assumed to be deterministic and equal, denoted by t. The analysis will take place for a fixed amount of time, that is, the system will be required to work for a fixed time, T_m, which could be considered the passage time for the ship. At inception of the system, with $t=t_0$, it is assumed that the system is always in its nominal configuration – Pumps 1 and 3 working. The system consists of three components with two states, working and failed, represented by W and F respectively, which means that the system has 2^3 possible configurations listed in table 3.2.

Table 3.2 A table showing all the possible configurations for the system under consideration

B	1	2	3	4	5	6	7	8
Pump 1	W	W	F	F	W	W	F	F
Pump 2	W	F	W	F	F	W	W	F
Pump 3	W	F	W	W	W	F	F	F

Fig 3.12 shows a fault tree analysis of the cooling system, it can be seen that the system has three cut sets pertaining to failure of the cooling system. In FTA a cut set is defined as a set of basic events whose simultaneous occurrence ensures that the top event occurs. A cut set is said to be minimal if the set cannot be reduced without losing its status as a cut set.

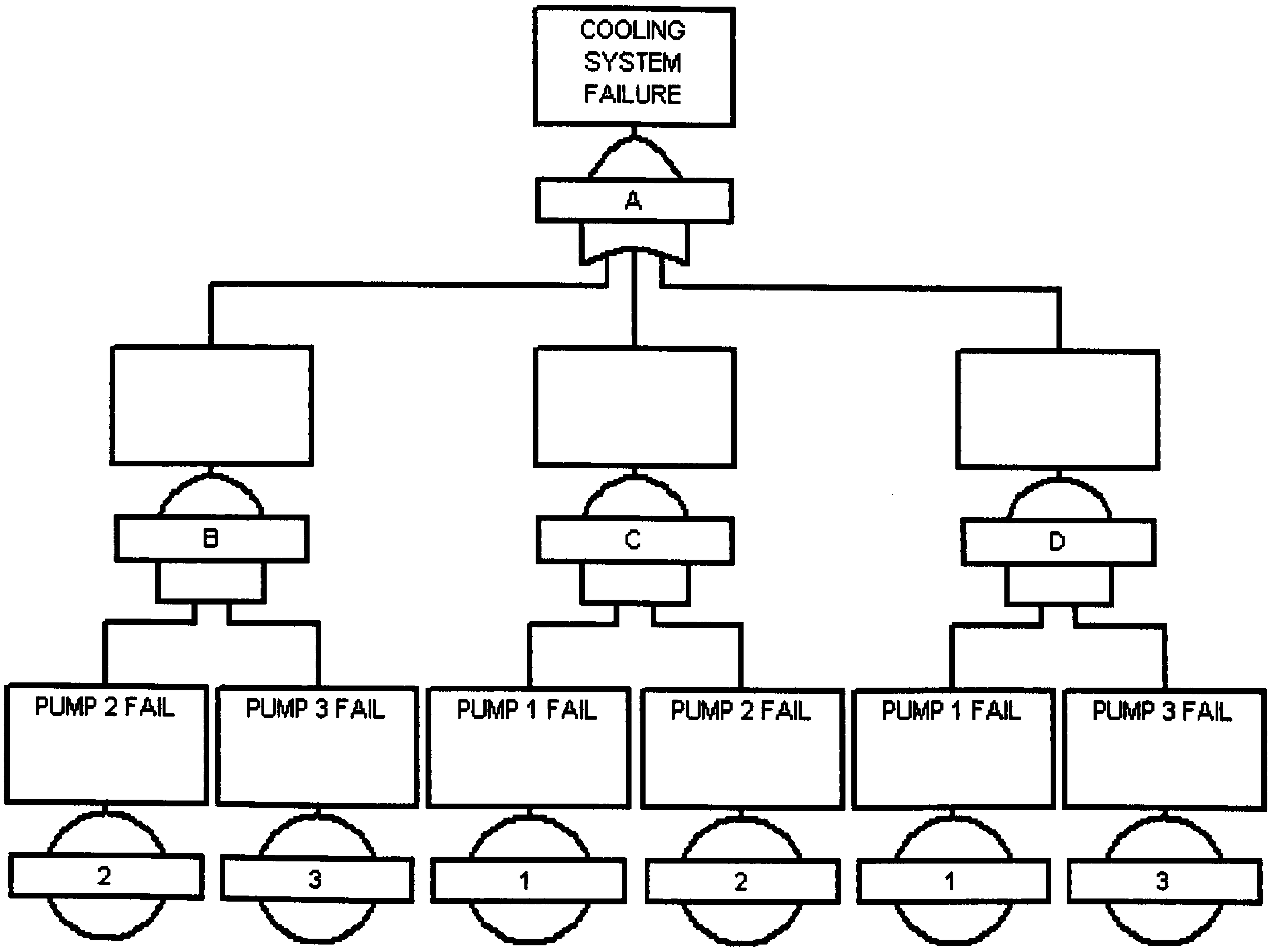

Fig 3.12 Fault tree for the cooling system

For the purpose of the following descriptions it is necessary to define the following. A system is a collection of $m \geq 1$ components and the system state can be represented by the vector **B**=(b_1,b_2,....,b_m) whose elements are the state indicators of the components. In the case of the system under consideration, the system state vector **B** will contain three elements **B**=(b_1,b_2,b_3) representing the pumps 1, 2 and 3 respectively. Furthermore the system state vector B can be represented by a single integer 1-8 corresponding to each permutation of working and failed states as shown in table 3.3. From the starting configuration, **B** = 1 = (W, STANDBY, W) the system has two possible transitions, namely the failure of pump 1 or pump 3 which will occur at a time t_1. The time is sampled using the inverse transform of the exponential distribution as shown below.

$$t_1 = t_0 - \frac{1}{\lambda^s} ln(1 - R_t) \qquad (3.18)$$

where R_t is the random variable in the equation sampled using the inverse transform method outlined previously. It should be noted that the failure rate, λ^s, is for the whole system and as such consists of the sum of all the failure rates of active components within the system. In the case of the first event the failure rate would be as follows:

$$\lambda_1^s = \lambda_1 + \lambda_3 \tag{3.19}$$

The system failure rates for the eight different possible configurations of the system are shown in table 3.3.

Table 3.3 Table showing the system failure rates for each of the possible system configurations

Integer	**B**	λ_B^s
1	(W,STANDBY,W)	$\lambda_1^s = \lambda_1 + \lambda_3$
2	(W,F,F)	$\lambda_2^s = \lambda_1$
3	(F,W,W)	$\lambda_3^s = \lambda_2 + \lambda_3$
4	(F,F,W)	$\lambda_4^s = \lambda_3$
5	(W,F,W)	$\lambda_5^s = \lambda_1 + \lambda_3$
6	(W,W,F)	$\lambda_6^s = \lambda_1 + \lambda_2$
7	(F,W,F)	$\lambda_7^s = \lambda_2$
8	(F,F,F)	$\lambda_8^s = 0$

The trial starts by sampling a random number $R_t \sim U(0,1]$. Once the time is sampled a test is run to ensure that the event is within the mission time i.e. $t<T_m$. If this condition, $t<T_m$, is satisfied then a system transition has occurred. At this point it is not known which of the components has failed and therefore the nature of the transition is unknown. In order to ascertain which component it is, a second random number, R_c, is sampled. A new discrete distribution composed of the probabilities of pumps 1 and 3 undergoing a transition out of their original states,

given that a transition has occurred at t_1, is created. The probabilities are as follows:

$$\frac{\lambda_1}{\lambda_1^s}, \frac{\lambda_3}{\lambda_1^s}$$

The inverse transform method can now be applied to the discrete distribution and used to indicate which of the components has undergone a transition. In the case shown in fig 3.13 it can be seen that pump 3 has undergone the transition and is now therefore in a failed state.

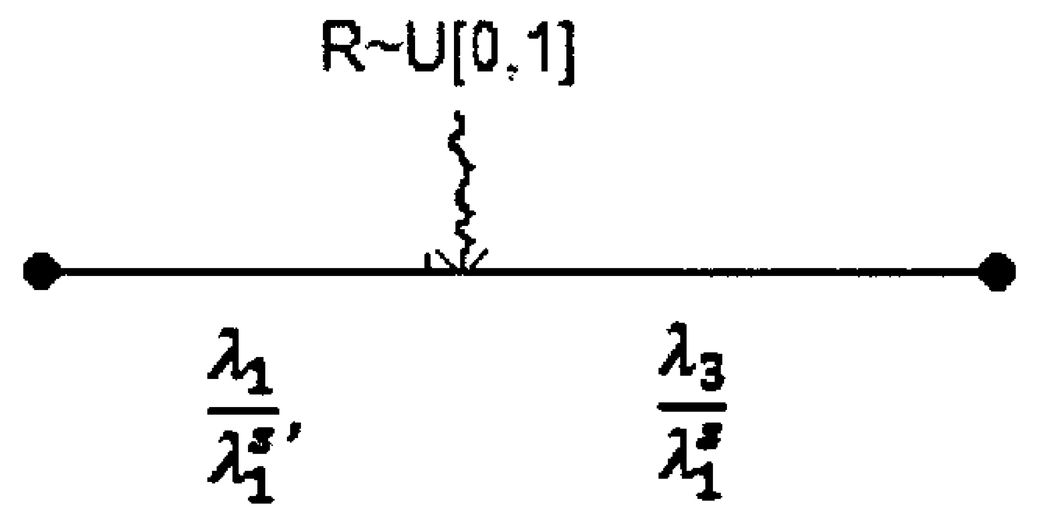

Fig 3.13 Diagram showing that the random number R_c has fallen within the interval pertaining to the failure of pump 3

In light of this transition the system state vector **B** now changes so that $\boldsymbol{B} = 6 = (W, W, F)$. Also the value T_0, sometimes referred to as the 'birth time' of the component, has been updated to t_1, the time of the last transition. The system is now sampling for failures of pumps 1 and 2. This is done using the inverse transform of the exponential distribution as shown below,

$$t_2 = t_1 - \frac{1}{\lambda^s} ln(1 - R_t) \tag{3.20}$$

The failure rate is for the whole system and as such consists of the sum of all the failure rates of active components within the system, in the case of this new state the failure rate would be as follows:

$$\lambda_1^s = \lambda_1 + \lambda_2 \tag{3.21}$$

The test is performed to check that the sampled time, t_2, is within the mission time and if a second transition of the system has occurred. The first term in equation (3.2), t_1, the time of the last transition, becomes important at this point; if the term had remained equal to 0 and had not been updated, then the test of whether or not the transition was within the mission time would not be relevant to our current system. In the first instance, equation (3.1), where $t_0=0$ then the test, $t<T_m$, involves the whole of the mission time. However, once the term is updated, $t_0=t_1$, to represent the time of the last failure then the test only considers the remaining mission time i.e. the shaded portion shown in fig 3.14. In this way the system is essentially 'moved on' within the time frame of the mission.

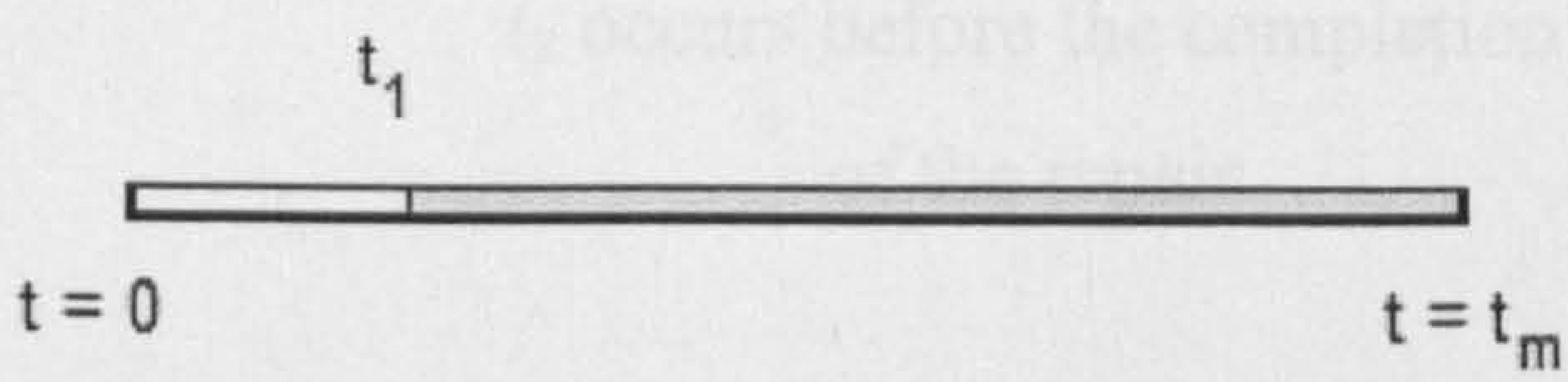

Fig 3.14 Time line of system transitions

Again the inverse transform method is applied to the appropriate discrete probabilities, given that a transition has occurred at t_2, in order to ascertain which of the components has undergone the transition. It should be noted that the sampling of the time of transition and which component has undergone the transition can be achieved in one step with a single random number called direct simulation (Marseguerra & Zio 2002). In order to fully demonstrate the techniques used the indirect simulation is preferred. After the last transition the system was left in state **B** = 6 = (W,W,F). For the purpose of this explanation it is assumed that the second transition was that of pump 1 and can now be considered in a failed state. Now, it is known that a transition has occurred and the component which has undergone transition has been found. There are only two possible system state transitions that can occur, namely $\boldsymbol{B} = 6 \rightarrow \boldsymbol{B} = 7$ or $\boldsymbol{B} = 6 \rightarrow \boldsymbol{B} = 3$. In the following each case will be considered in turn.

1. First consider the case, $\boldsymbol{B} = 6 \rightarrow \boldsymbol{B} = 7$. In this instance the failure of pump 3 has already occurred and a deterministic repair time is underway. In order for the system to be able to undergo this transition the deterministic repair of pump 3 must remain incomplete before the transition t_2 and subsequent failure of pump 1, as shown in fig 3.15.

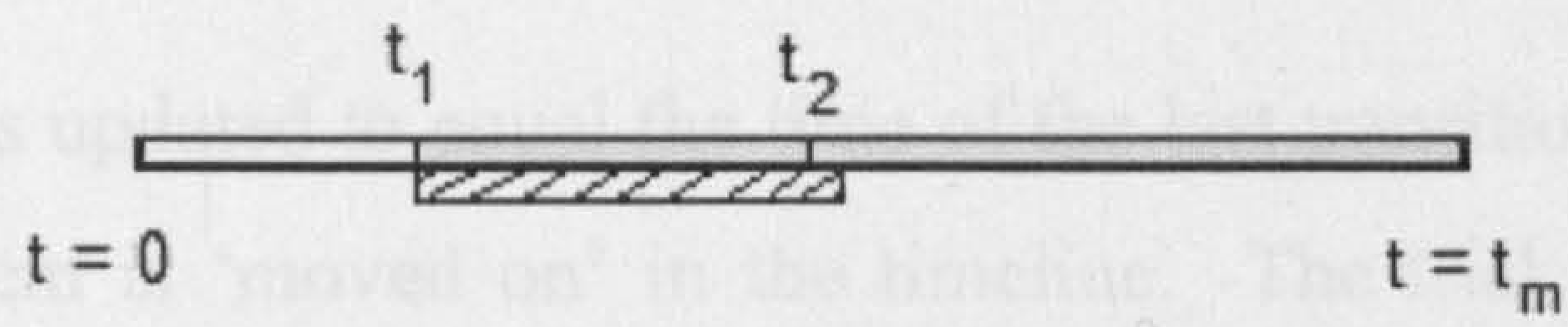

Fig 3.15 Time line including repair time, where

t_2 occurs before the completion

of the repair

As can be seen the transition occurs before the completion of repair and as a result the system enters a failed state, $\boldsymbol{B} = 7$, with one of the combinations satisfied, **B**=(F,W,F). Therefore a contribution of one is made to the counter of system failures. The system remains in this failed state until the repair of pump 3 is completed at which point it comes back online in state $\boldsymbol{B} = 3 = (F, W, W)$. The trial now carries on with the sampling of transitions relating to pumps 2 and 3.

2. Next consider the second case, $\boldsymbol{B} = 6 \rightarrow \boldsymbol{B} = 3$. In this instance the failure of pump 3 has already occurred and a deterministic repair time is underway. In this case however the transition of the system and the subsequent failure of pump 1, occur after the repair of pump 3 as shown in fig 3.16. The system now changes state, $\boldsymbol{B} = 6 \rightarrow \boldsymbol{B} = 3$.

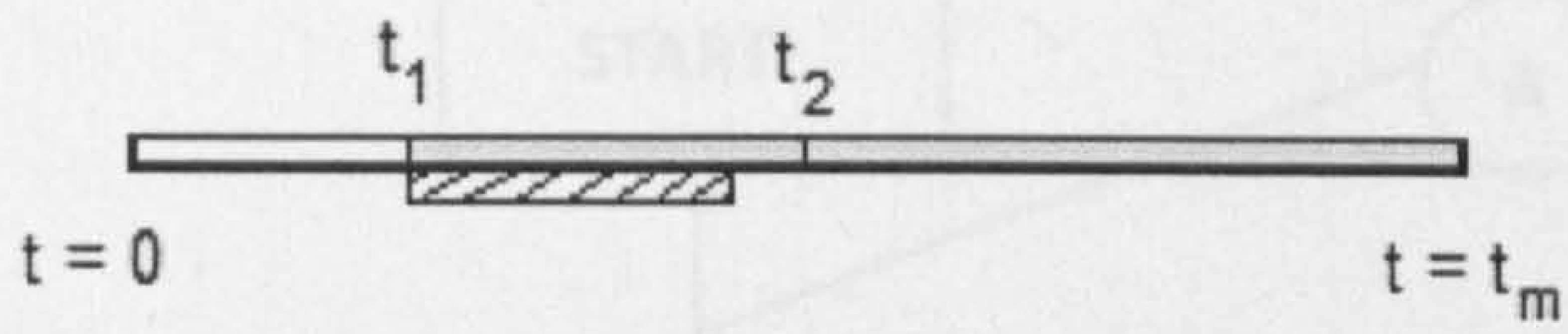

Fig 3.16 Time line including repair time, where
t_2 occurs after the completion of the repair

> The term t_1 is updated to equal the time of the last transition t_2 and in this way the system is 'moved on' in the timeline. The trial now carries on with the sampling of transitions relating to pumps 2 and 3.

In the same way as before the next transition in each of these two cases will be one of two possibilities depending on whether the repair of the other component is completed or not. It can be seen that each transition uses the same equation but utilises different failure rates, dictated by the state of the system at the time of sampling. The system will carry on like this until the completion of the mission time.

3.6 Code Generation

Before attempting to create the code for this simulation a number of flowcharts were produced to facilitate a better understanding of the system logic.

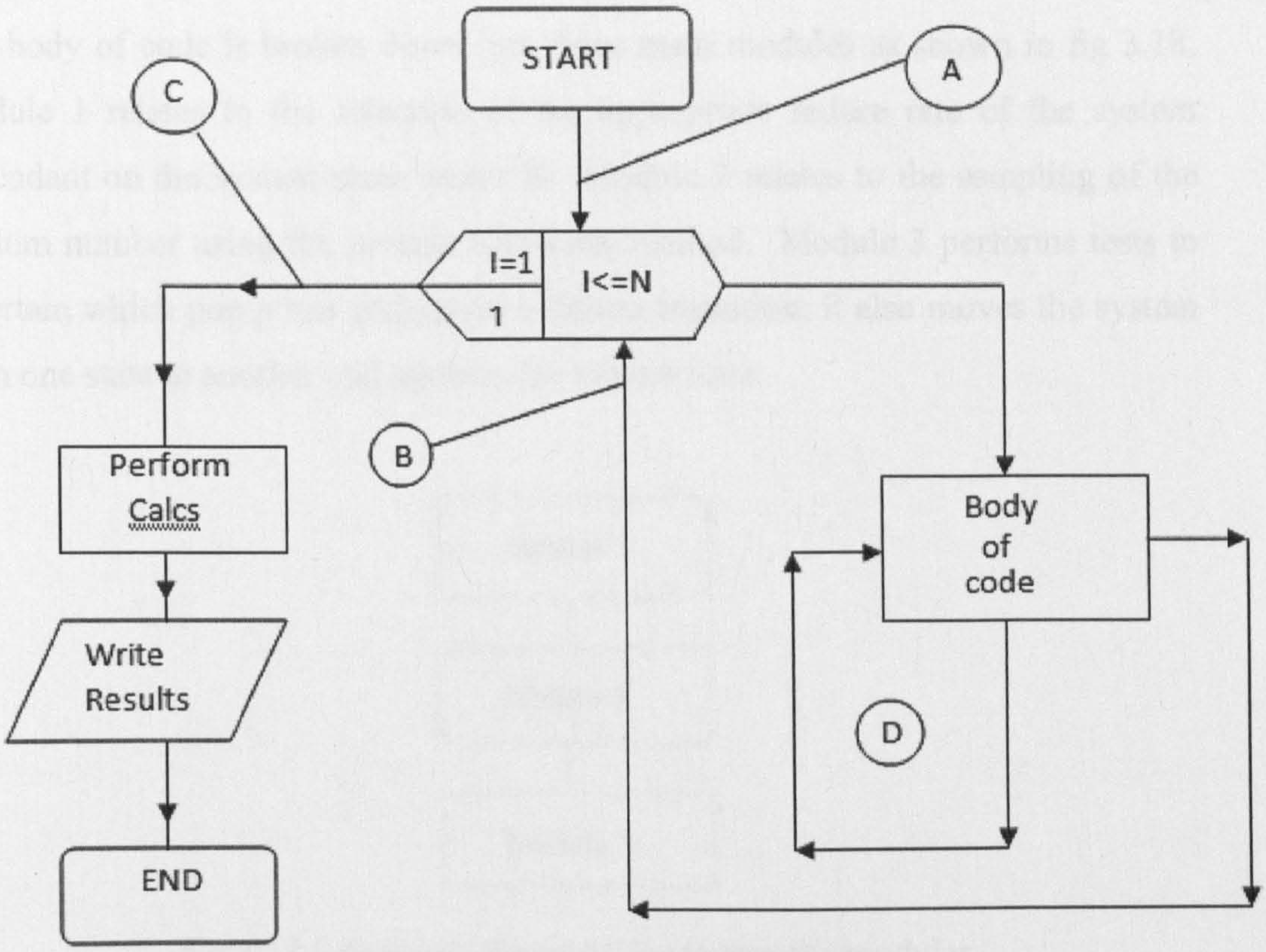

Fig 3.17 Flow chart showing the general structure of a Monte Carlo Simulation

Fig 3.17 shows a flow chart which could be considered as the generic model for all Monte Carlo codes. When the code is initialized it enters a counting loop, at A, where the integer N is predefined and I is a counter which is incremented by one every time the code re-enters at B, this continues until I=N at which point the code exits at C. N is set to the value of trials required in the simulation. While I<N then the body of code is executed. The loop structure at D, is an implied do loop within the code which executes indefinitely until a condition is met, this ensures that the simulation continues to sample for failures until there is no mission time left.

A flow chart is produced for the body of the code; all of this is carried out prior to any code being written. Due to the nature of the system, the flow chart for the body of code is complex. With this in mind, the flow chart is broken down even further into modules of code. The modules are essentially blocks of code which when put together in the correct way will produce the main body of the code.

The body of code is broken down into three main modules as shown in fig 3.18. Module 1 relates to the selection of the appropriate failure rate of the system dependant on the system state vector **B.** Module 2 relates to the sampling of the random number using the inverse transform method. Module 3 performs tests to ascertain which pump has undergone a failure transition; it also moves the system from one state to another and updates the system time.

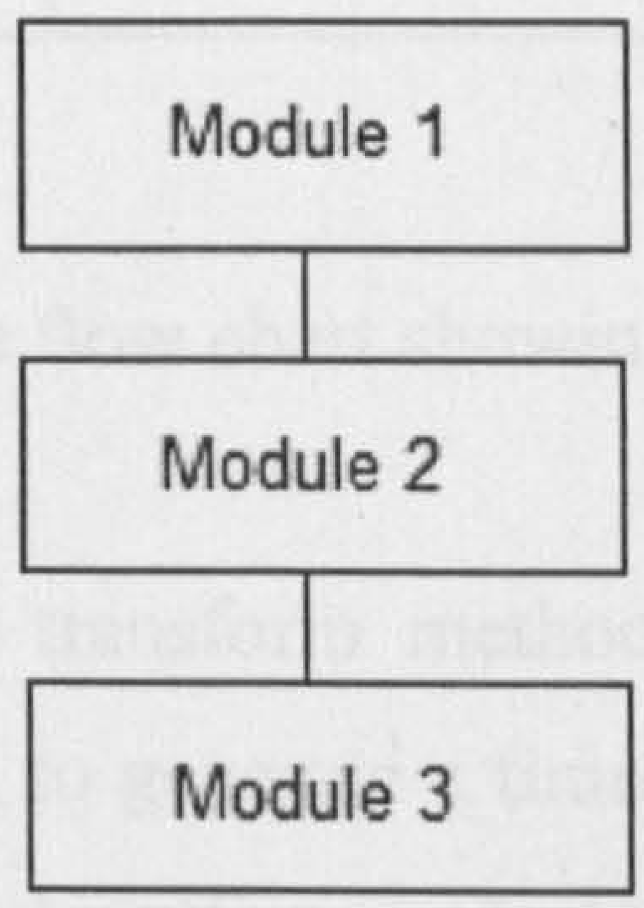

Fig 3.18 Schematic diagram illustrating the modular structure of the programming

Module 1 is essentially a single IF construct which tests the system state and alters the system failure rate accordingly. Throughout the program an integer, **B**, is used to define the system state outlined in table 3.2. According to the state the system enters, the system failure rate is then altered according to table 3.3. Fig 3.19 shows the IF construct, it should be noted that the test only encompasses the system states of **B** = 1, 3, 5 and 6. This is because the system can only exist in one of these four states at this point in a trial; it would be useless to allow a trial to test for failures when the system is in a failed state already.

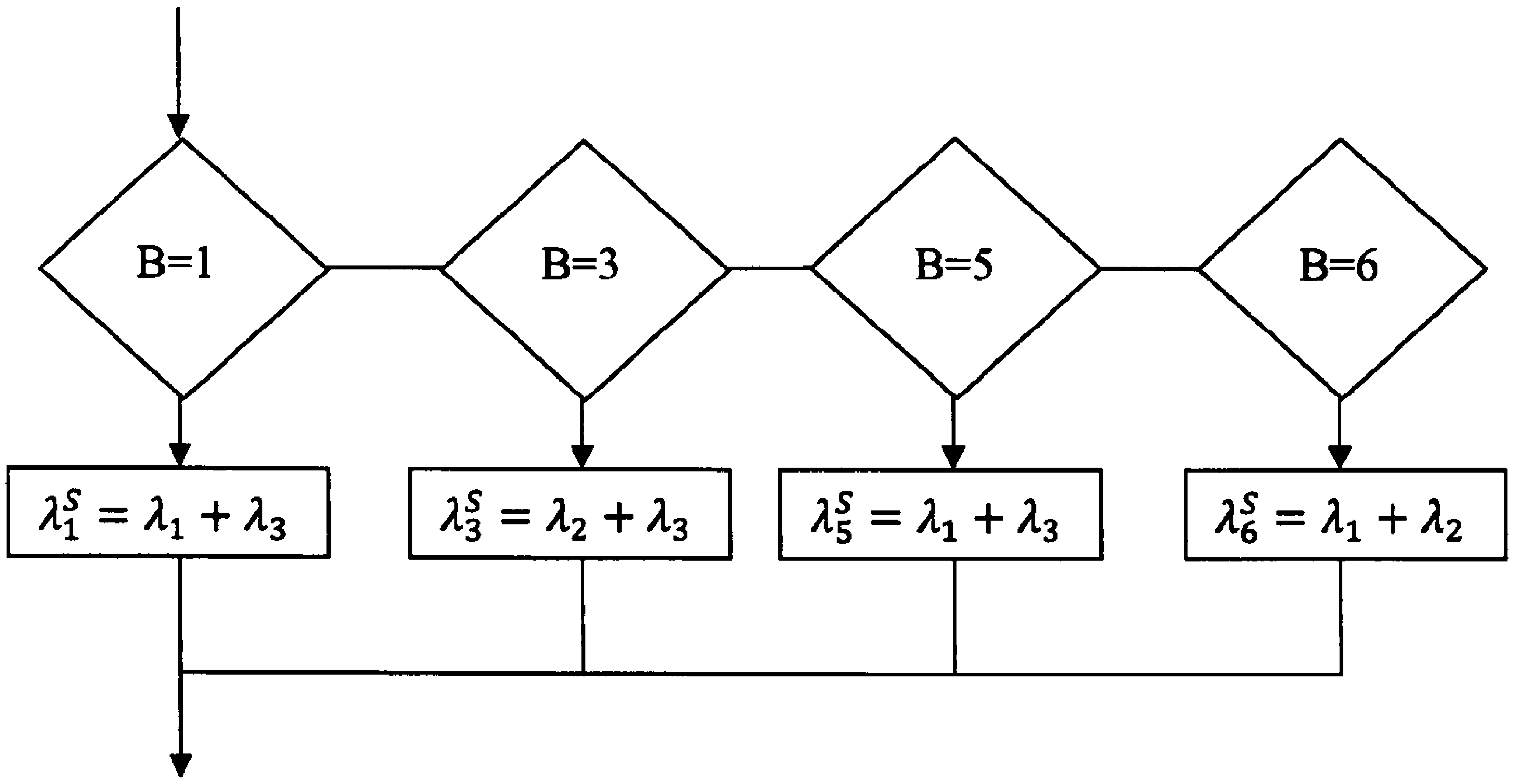

Fig 3.19 Programme flow chart showing the function of module 1

Module 2 uses the inverse transform method outlined previously to sample a uniform random number, R_t, to generate a time of system transition. It contains a conditional IF statement which performs a test which determines if the transition is within the mission time, if not the trial is aborted with no contributions to any failure counters made and can be considered a successful trial. It also samples a second random number, R_s, which is used in module 3 to determine which pump has undergone a transition. Fig 3.20 shows module 2.

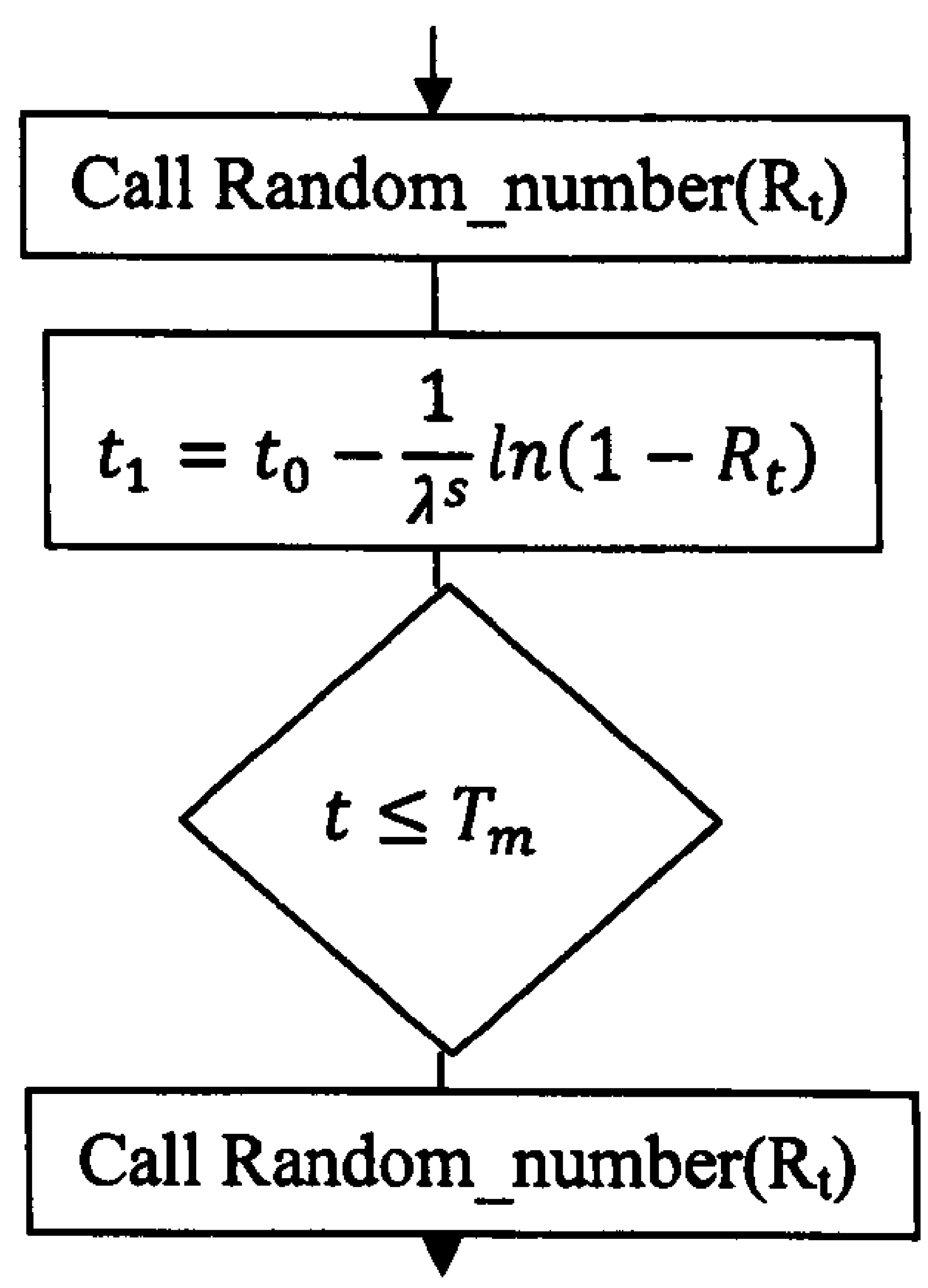

Fig 3.20 Programme flow chart showing the function of module 2

Module 3 is the most complex of the three modules. The basic functions of the module are:

- To learn which of the components has undergone a transition.
- Whether this occurs before the repair of the last pump to fail.
- To make appropriate contributions to the system failure counter if one of the failure combinations is satisfied.
- To update the system time within the analysis.

The module is built from a number of conditional IF statements at different levels within the program. The IF construct conditionally executes constructs, or statements, depending on the evaluation of a logical expression. Fig 3.21 is a representation of the module showing all eleven of the block IF constructs.

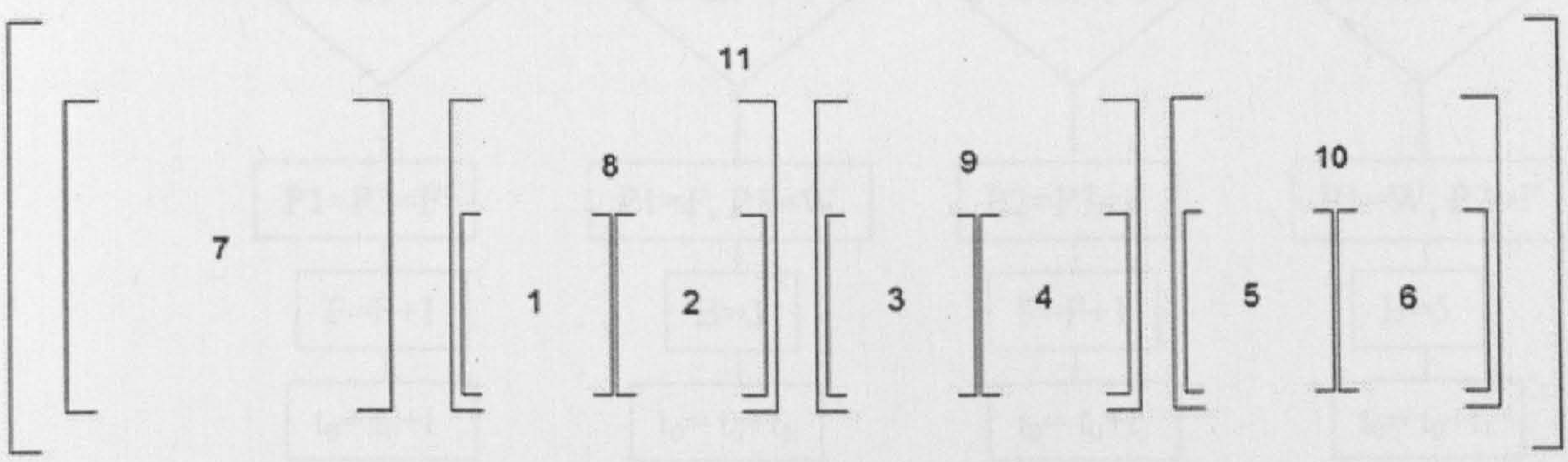

Fig 3.21 Graphic representation of the IF structure contained in module 3

In Fig 3.21 it can be seen that the whole module is contained within IF statement 11, this IF statement evaluates whether $t < T_m$ which is the primary condition that must be fulfilled. If $t > T_m$ then the failure occurs outside the mission time and is of no interest. The IF constructs numbered 7, 8, 9 and 10 use conditional statements to decide which of the active components has undergone a transition. Finally the IF constructs numbered 1, 2, 3, 4, 5 and 6 ascertain whether the transitions occur before or after the repair of the already failed component from a previous trial. Fig 3.22 shows the block IF construct number 10 in a more conventional flow chart format. IF statements 5 and 6, contained within 10, are included in the flow chart. It should be noted that IF statements 8, 9 and 10 are very similar and it is only the statement that differs, which re-assigns the system state vector **B** according to the transitions that have occurred during the most recent trial.

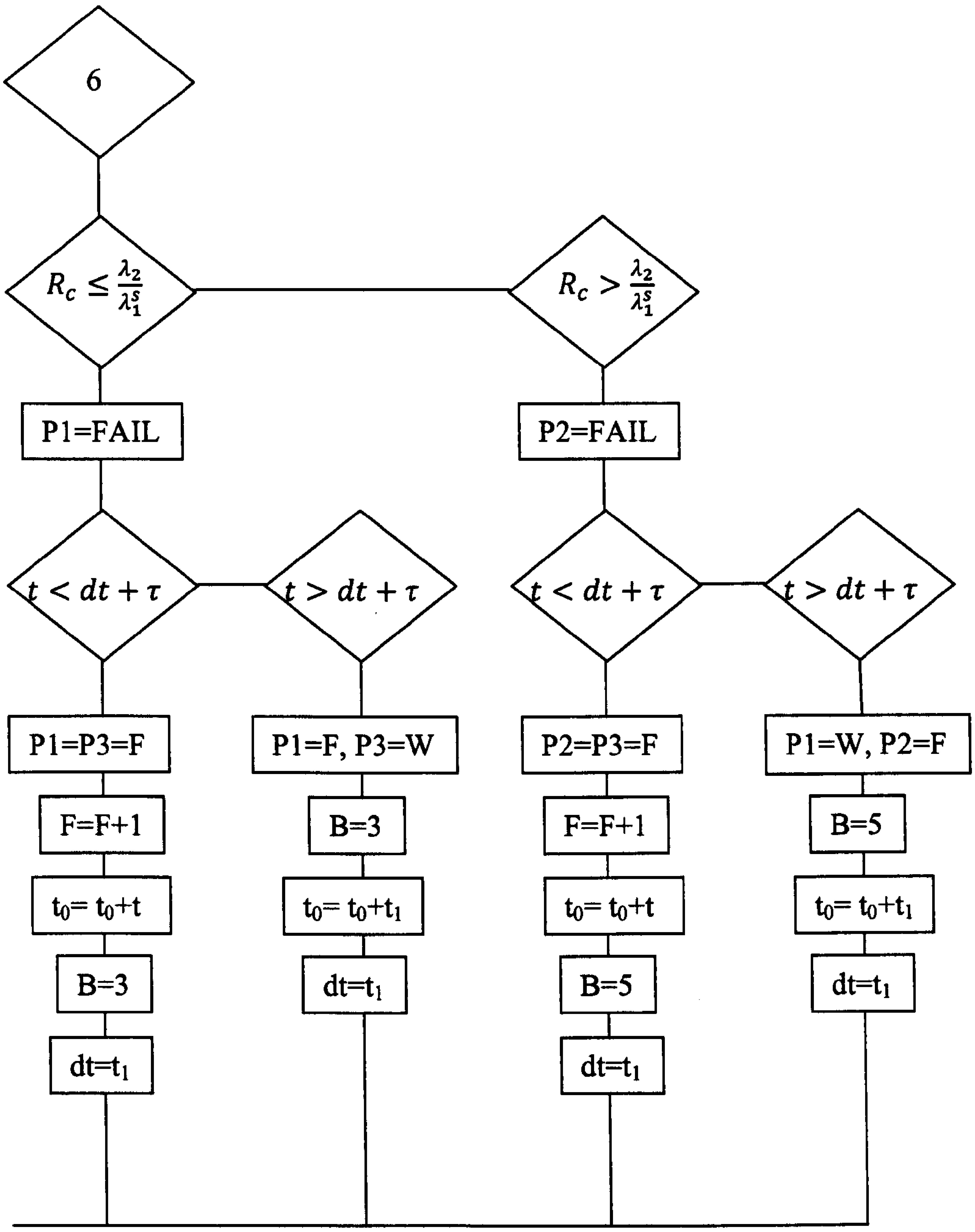

Fig 3.22 Program flowchart showing IF statements 10, 5 and 6

3.7 Results

The input values shown in table 3.4 were used for the parameters in the program. The values used for the failure rates are not based in real system observations however they are similar to failure rates used in previous studies on engineering systems (Cantoni et al. 2000), (Marseguerra & Zio 1993, 2000), (Marseguerra et al. 2002), (Marseguerra et al. 2005), (Zio 1995).

Table 3.4 Table showing the input values for the simulation program

Parameter	**Value**
N	10^7
T_m (hrs)	200
τ (hrs)	10
λ_1 (hrs^{-1})	10^{-4}
λ_2 (hrs^{-1})	10^{-5}
λ_3 (hrs^{-1})	10^{-6}

In order to provide a benchmark the system was modelled using FaultTree+ V6.0 to give an analytical solution to the problem, fig 3.23 shows the fault tree from the software package.

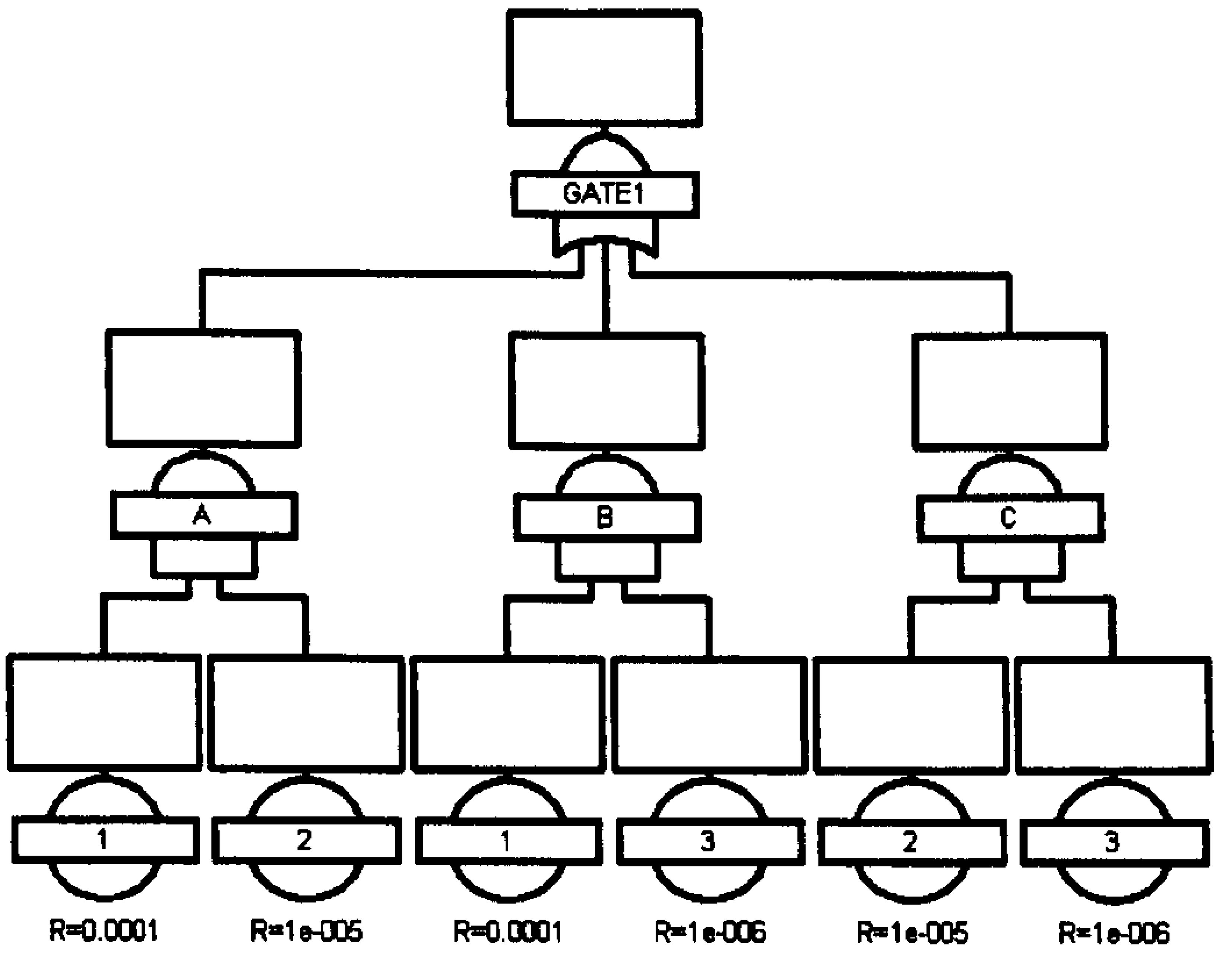

Fig 3.23 Fault tree modelled using FaultTree+ V6.0

The failure rates for the components were identical to those used in the Monte Carlo analysis and the system lifetime was set at 200 hours. The software asked for a repair rate, rather than a deterministic repair time, therefore the repair time was converted to a repair rate of 0.1. The analysis was completed using the software package and a screen shot showing the results is shown in fig 3.24.

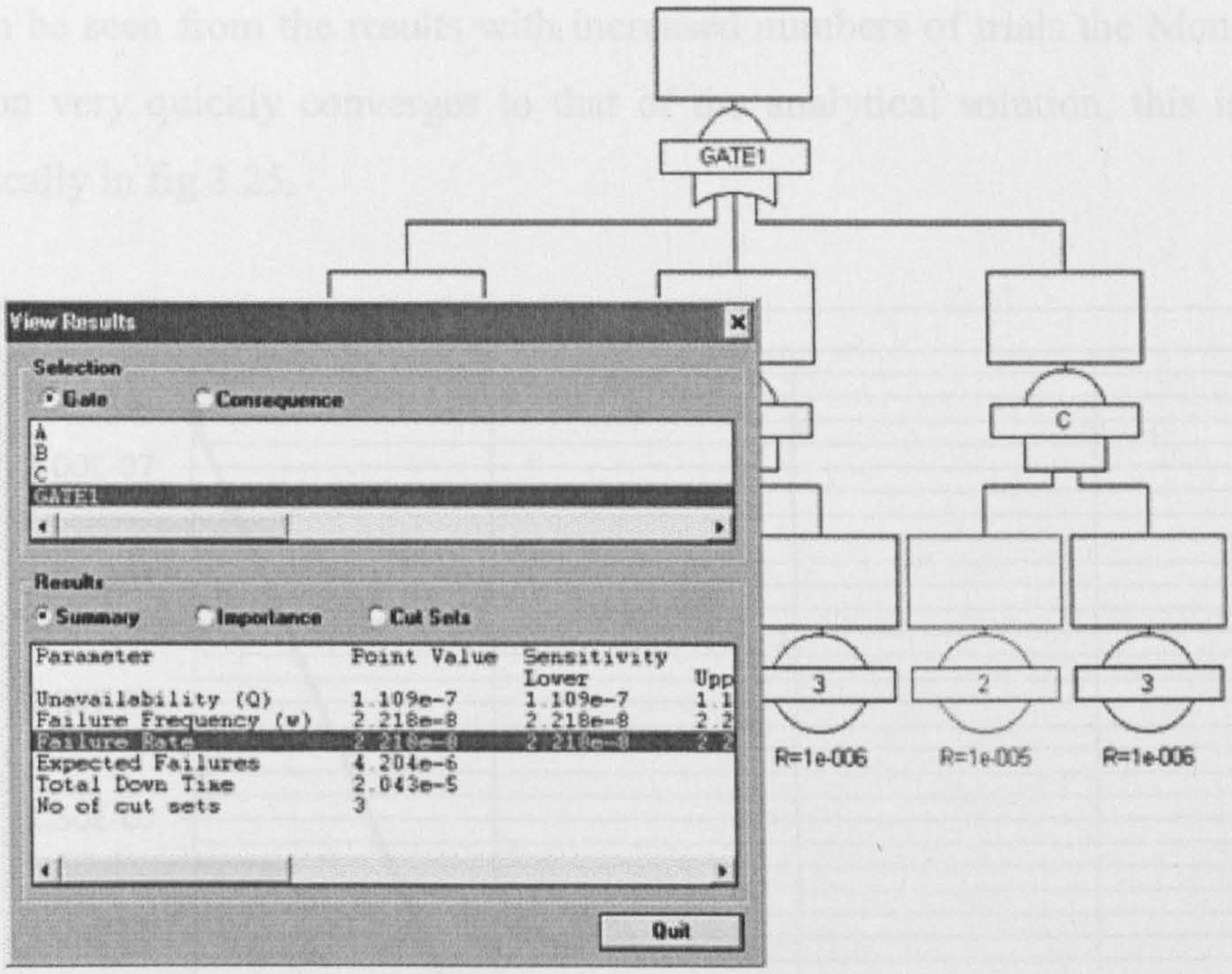

Fig 3.24 Screen shot showing the results from FaultTree+ V6.0

It can be seen that the result obtained using the fault tree software was 2.218×10^{-8}. When the simulation programme was run with $N=10^7$ a value of 2.4×10^{-6} was obtained. This result would appear to be incorrect as it represents a large error and would thus be unacceptable, however, due to the nature of Monte Carlo simulation the larger the number of trials the more accurate the results become. The simulation programme was run a number of times, each time the number of trials was increased to see if the answer would approach that calculated analytically. Table 3.5 shows the results of the trials.

Table 3.5 Table showing the results from the Monte Carlo trials

Trials	Failure Probability	No. of Failures
10^7	2.4×10^{-6}	24
10^8	2.39×10^{-7}	239
10^9	2.308×10^{-6}	2308
10^{10}	3.231×10^{-7}	3231
10^{11}	2.818×10^{-8}	2818
10^{12}	2.816×10^{-8}	28163
10^{13}	2.816×10^{-8}	281624
10^{14}	2.816×10^{-8}	2816132

As can be seen from the results with increased numbers of trials the Monte Carlo solution very quickly converges to that of the analytical solution, this is shown graphically in fig 3.25.

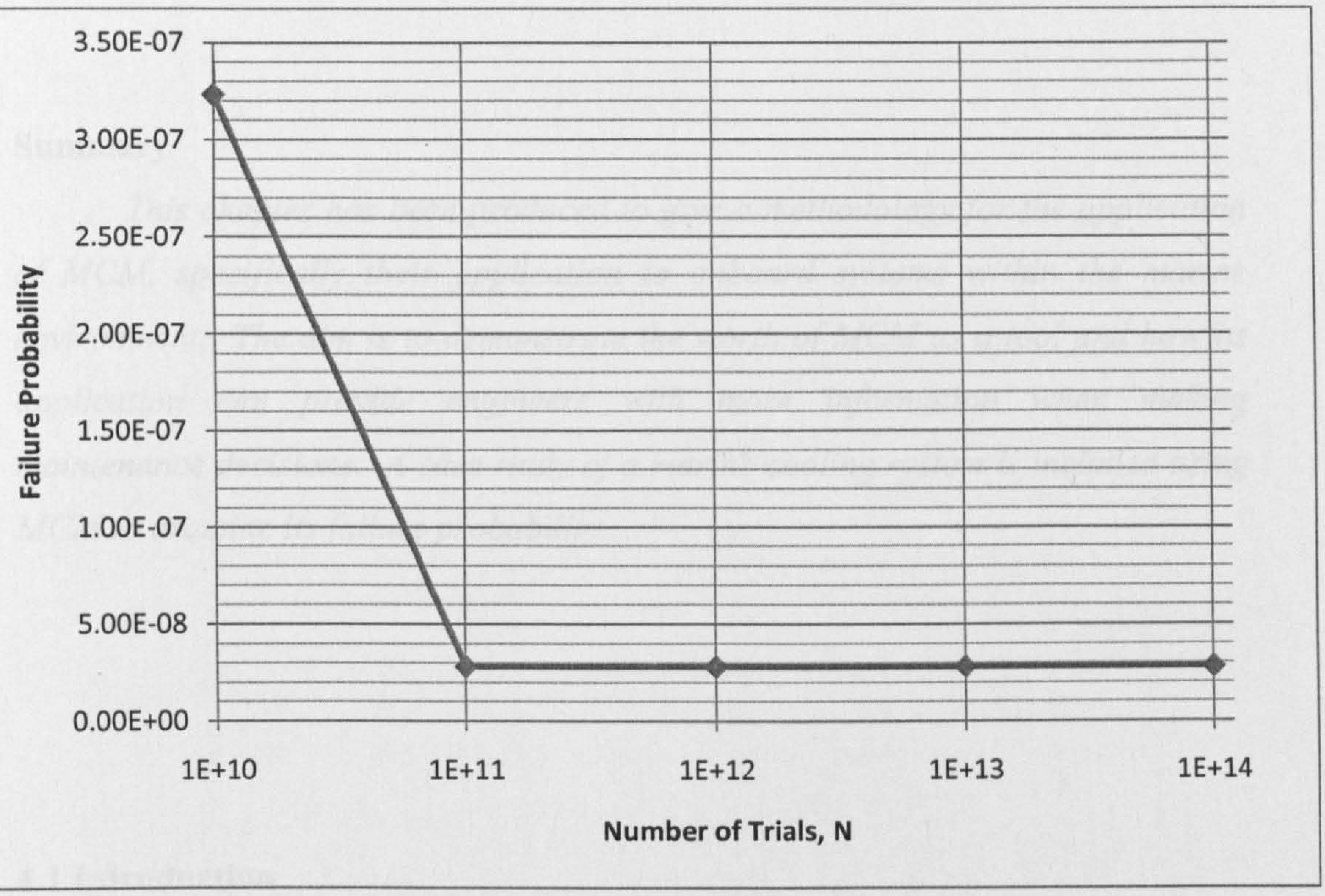

Fig 3.25 Graph showing the Monte Carlo trial results compared to the analytical solution

3.8 Conclusion

The chapter has demonstrated the basic concepts involved in the application of Monte Carlo Methods. The input data used for the model in the chapter was similar in magnitude to the examples presented in the literature. As will be seen in the following technical chapters it will be the case that failure rates in the marine industry and therefore the input failure parameters will be higher than those used in this analysis. However it serves as the basis for the forthcoming technical chapters, through which the method will be extended to encompass more complex models and incorporate data specific to the marine field.

Chapter 4 – Monte Carlo Simulation of a Complex System

Summary

This chapter has been produced to give a methodology for the application of MCM, specifically their application to onboard systems within the marine environment. The aim is to demonstrate the worth of MCM as a tool and how its application can provide engineers with more information when making maintenance decisions. A case study of a marine cooling system is included using MCM to examine its failure probability.

4.1 Introduction

MCM as a tool in the marine industry are well facilitated by company collaborations such as OREDA (OREDA 2002) which provide real data from the field. Such data is essential as the marine environment is unique in the extreme conditions in which systems have to function. It is often the case in Monte Carlo RAMS studies that biasing is required. Zio & Sansavini (2007) propose a biasing procedure which improves the efficiency of the unreliability estimate of complex multi-state network systems.

In the marine environment the system components experience higher rates of failure, due to the adverse operating conditions. In this chapter it will be shown that a Monte Carlo Reliability study can be developed for a complex marine system using real data where no biasing is required. It is assumed that the reader is familiar with the basic premise of the MCM outlined in the previous chapter, however a more in-depth explanation of the theory can be found in Marseguerra & Zio (2002) and Rubinstein (1981).

4.2 Background

4.2.1 Monte Carlo Methods – Brief Overview

MCM are implemented in this chapter as described in chapter 3 and by several sources (Rubinstein 1981), (Manno 1999), (Dubi 2000), (Marseguerra & Zio 2002).

The process of random number generation and the implementation of the inverse transform method for a continuous distribution is implemented as follows:

1. Generate a random number R_t, $U{\sim}[0,1)$.
2. Enter the random number, R_t, into the inverse transform of the desired distribution to give a value t.

$$t_1 = t_0 - \frac{1}{\lambda^s} ln(1 - R_t)$$

This value, t_1, represents the time of a system transition.

The inverse transform method can also be applied to discrete distributions (Marseguerra & Zio 2002) in a similar way to the continuous case. In this chapter the method is implemented to select which of the working components has undergone a transition at the time t. The continuous distribution is composed of the probabilities of the individual working components undergoing a transition. The interval in which a new random number, R_c $U{\sim}[0,1)$, falls indicates which component has changed state.

4.2.2 Fault Tree Analysis (FTA)

FTA is one of the most widely used technique for applying hazard identification and risk evaluation. If a number of events exist, which all interact to produce an output set of events it is often most practical to define the system using simple logical relationships at the various levels. This forms a rational structure which represents a model of the system. It is used throughout the marine, chemical and nuclear industries. When applied, the associated undesirable top event is often defined by experience i.e. a previous failure is recorded and logical steps are taken in reverse through the system until the basic events (faults) have been identified (Andrews & Moss 2002), (Lewis 1987), (Dhillon 1983). Fault tree analysis is especially useful in the fact that it can be applied in both a qualitative and quantitative way. In a qualitative sense it can provide information on how the top event may occur and what consequences may be caused; in a quantitative sense it can be solved to provide the occurrence probability of the top event.

In the marine industry the top event is usually an event which will lead to catastrophic damage or loss. This is, as the name would suggest, placed at the top of the fault tree and all the associated levels mapped out below. The pathways, or "cut sets", from the base events to the top event should represent all the possible paths the system can take to reach the top event. The simplification rules can be applied until a fault tree is achieved which is irreducible in form, that is to say that it can be simplified no further. The remaining pathways which form part of the irreducible fault tree are referred to as the minimum cut sets of the system. Specifically a cut set is said to be a minimum cut set if, when any basic event is removed from the set, the remaining events collectively are no longer a cut set (Kececioglu 1991).

Before a fault tree can be constructed for a system, the analyst must have an understanding of how the system functions. The system is usually outlined using a system flow diagram (Andrews & Moss 2002), (Zio 2007) to aid in this understanding. The first stage in the fault tree analysis must always be the selection of a top event and care must be taken to ensure that only components which contribute to incidence of the top event are considered in the further analysis. The FTA process is outlined as follows:

- Identification of top events.
- Representation of top event by means of a fault tree.
- Evaluation of the occurrence of probability of each top event.
- Determination of critical failure modes.

Once the fault tree has been constructed work must be done to simplify the fault tree in order to obtain the desired minimal cut sets which contribute to the undesired top event.

4.2.3 Boolean Representation Method (BRM)

BRM was introduced during the 1970's as an automatic fault tree construction method (Apostolakis et al. 1978), (Dixon 1964), (Fussel 1973), (Henley & Kumamoto 1992), (Powers & Tompkins 1974), (Salem 1977). An engineering system can be described in terms of input and output events and each event may have several states. Typical states can be 'high' or 'low', 'on' or 'off' and in the case of this system's analysis 'working' or 'failed'. BRM can be applied in a number of different stages working from component level up to system level. In this way it provides a logical 'bottom-up' analysis of a system aimed at reducing the possibility of failure state omissions. Given suitable information about the system under consideration a Boolean Representation Table (BRT) can be quickly produced. The BRT can be simplified until a final BRT is produced. This final BRT contains the possible system top events and the associated cut sets.

In order to simplify the BRT the absorption and merging rules are used. Tables 4.1 and 4.2 give examples of the application of the rules and are only valid where variables A and B have two possible states namely working, W, and failed, F. The symbol * is used to represent 'don't care' as is standard in BRM.

Table 4.1 Absorption

A	B	OUTPUT
W	*	W
W	W	W

→

A	B	OUTPUT
W	*	W

Table 4.2 Merging

A	B	OUTPUT
F	F	F
F	W	F

→

A	B	OUTPUT
F	*	F

The final BRT can often be the same as that obtained using the fault tree analysis. Although the BRM is not as diagrammatic as FTA it can often allow for a much less 'bulky' representation of failure modes.

4.3 Development of Methodology

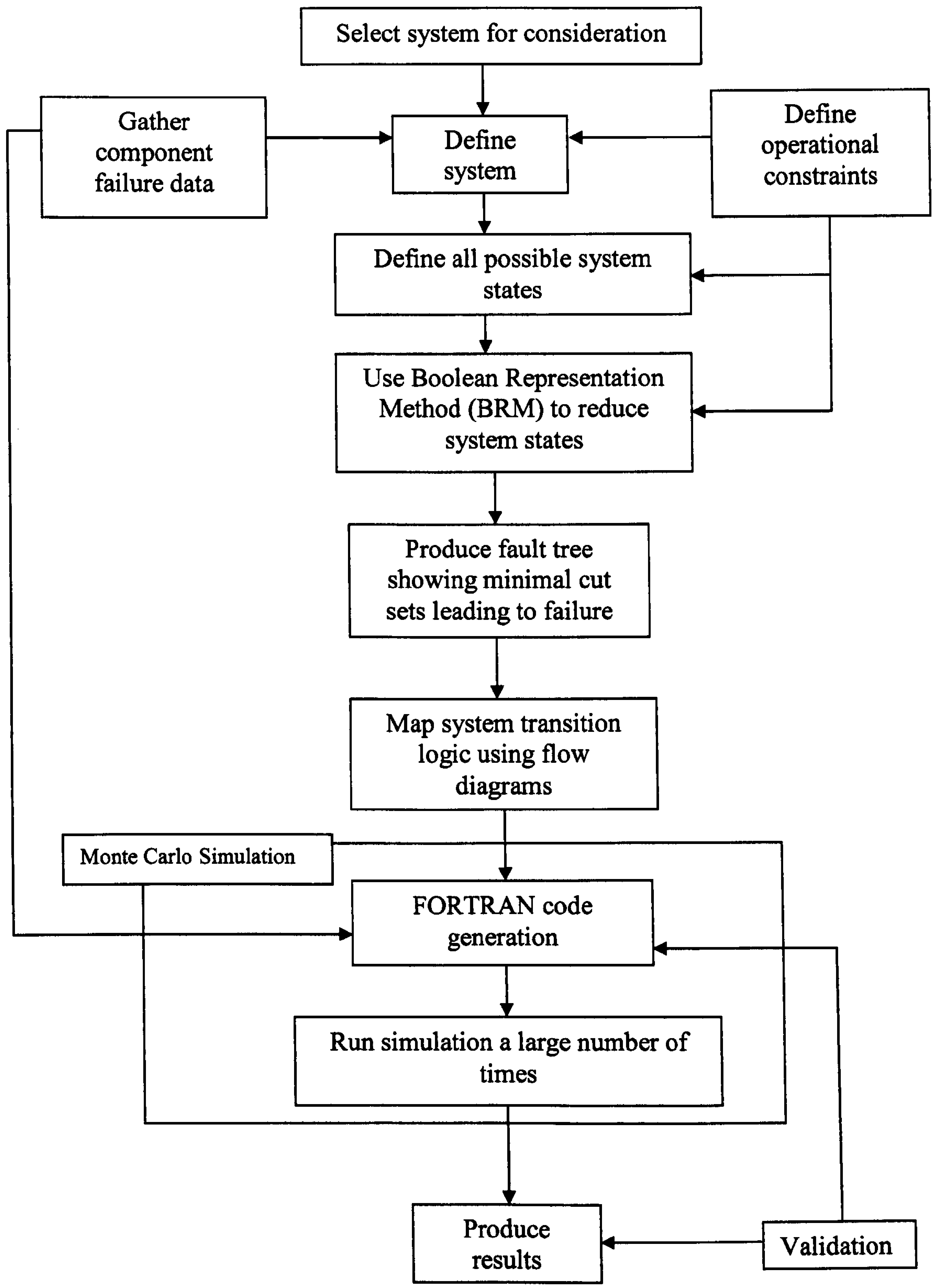

Fig 4.1 Proposed methodology for performing a Monte Carlo analysis

Fig 4.1 shows the methodology that has been developed. The diagram shows each of the stages in an analysis and illustrates the work that must be carried out before any of the analysis programs can be written. Understanding a system and its operational constraints is imperative as this can serve to significantly reduce the scope of the analysis. In the following, some important steps in the methodology will be outlined in greater detail. Where appropriate, simple examples will be given to aid the description.

4.3.1 Use Boolean Representation Method to Reduce System States

Consider the following simple system, shown in fig 4.2.

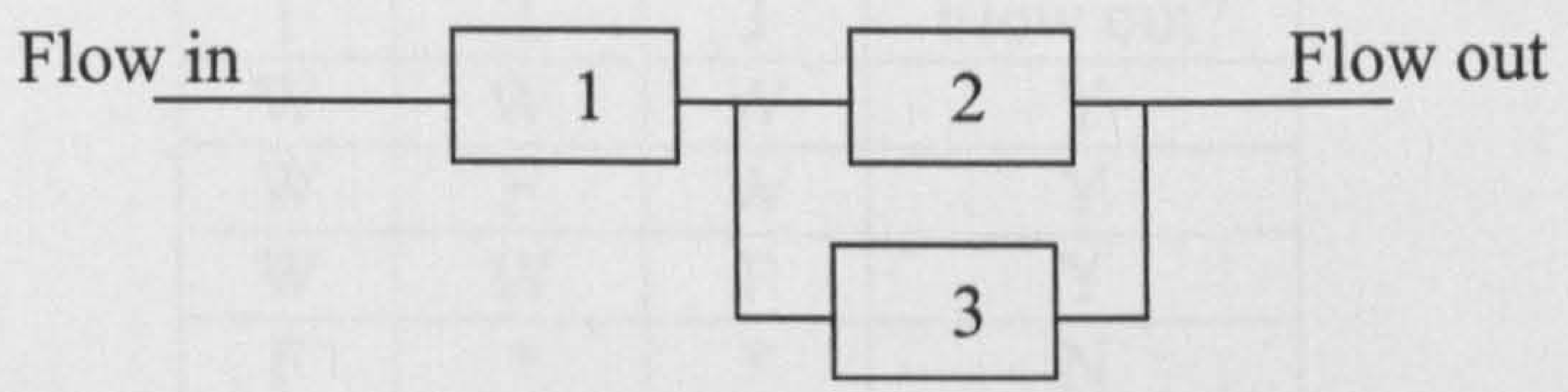

Fig 4.2 Diagram showing simple system for consideration

The first stage using the BRM method is to produce a table showing all of the system states. The system consists of three components, each of which has two states, namely working and failed. This gives rise to $2^3=8$ possible permutations. These are shown in table 4.3.

Table 4.3 Table showing all possible system states

1	2	3	Flow out?
W	W	W	Y
F	W	W	N
W	F	W	Y
W	W	F	Y
F	F	W	N
F	W	F	N
W	F	F	N
F	F	F	N

It can be seen that in each of the highlighted rows component 1 has failed, leading to 'NO' flow out. Now the merging rule can be applied, all four of the states shown can be replaced by the single state shown in table 4.4.

Table 4.4 Table showing the single state produced by application of the merging rule

1	2	3	Flow out?
F	*	*	N

In this way the BRM techniques can be applied until an irreducible table of system states is produced, shown in table 4.5.

Table 4.5 Irreducible table of system states

1	2	3	Flow out?
W	W	W	Y
W	F	W	Y
W	W	F	Y
F	*	*	N
*	F	F	N

4.3.2 Produce a Fault Tree Showing Minimal Cut Sets Leading to Failure

Once the system has been analysed using the FTA and BRM techniques, the analyst is left with the cut sets which lead to system failure and also the feasible system states. The fault tree for the system shown in fig. 4.2, is shown in fig. 4.3.

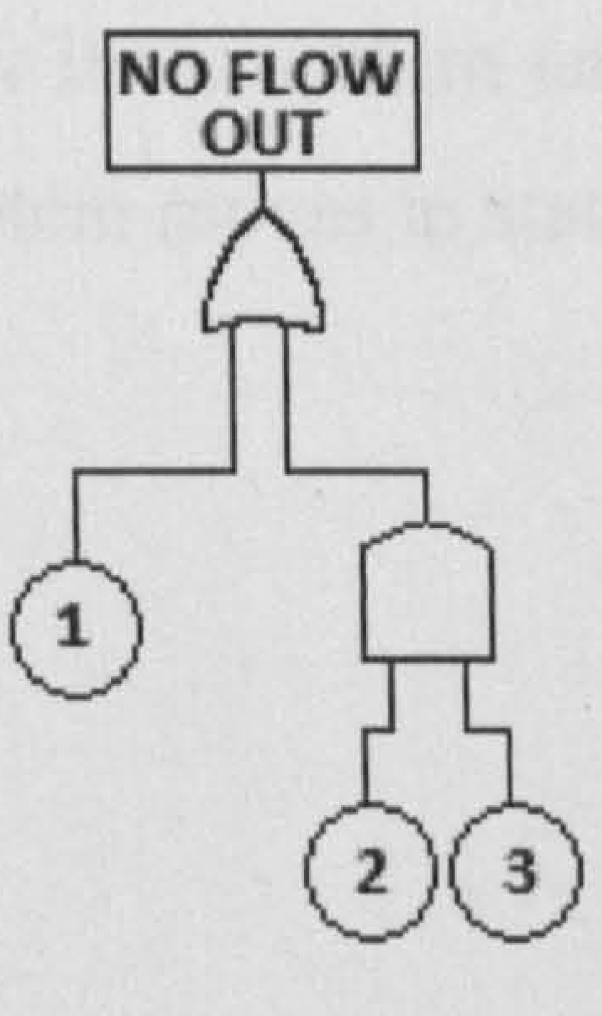

Fig 4.3 Fault tree showing minimal cut sets leading to the top event

4.3.3 Map System Transition Logic using Flow Diagrams

The most important stage is to map the system transition logic using flow diagrams. This process is greatly expedited by the introduction of a system state vector **B**=(b_1,b_2,....,b_m) whose elements are the state indicators of the components. If the system states shown in table 4.5 are considered, introduction of the system state vector **B** will result in table 4.6 being produced.

Table 4.6 Table showing the system states and their associated system state vector **B**

B	**b_1**	**b_2**	**b_3**	**Flow out?**
1	W	W	W	Y
2	W	F	W	Y
3	W	W	F	Y
4	F	*	*	N
5	*	F	F	N

The process involves starting at the nominal system state, i.e. **B**=1 and considering which components are vulnerable to failure. The analyst must consider the consequence of failure of each of these working components and how this changes the system state vector **B**. The transition of any individual component will lead to a change in the system state and hence the system state vector **B**. Fig 4.4 shows one of the flow charts produced to map the system transitional logic. It can be seen how an independent component transition leads to a different system state rather than that of any other independent component transition. For example when the system enters in state **B**=1 and component 1 undergoes a transition, the system leaves in state **B**=4. However if the system enters in state **B**=1 and component 3 undergoes a transition, the system moves to state **B**=3.

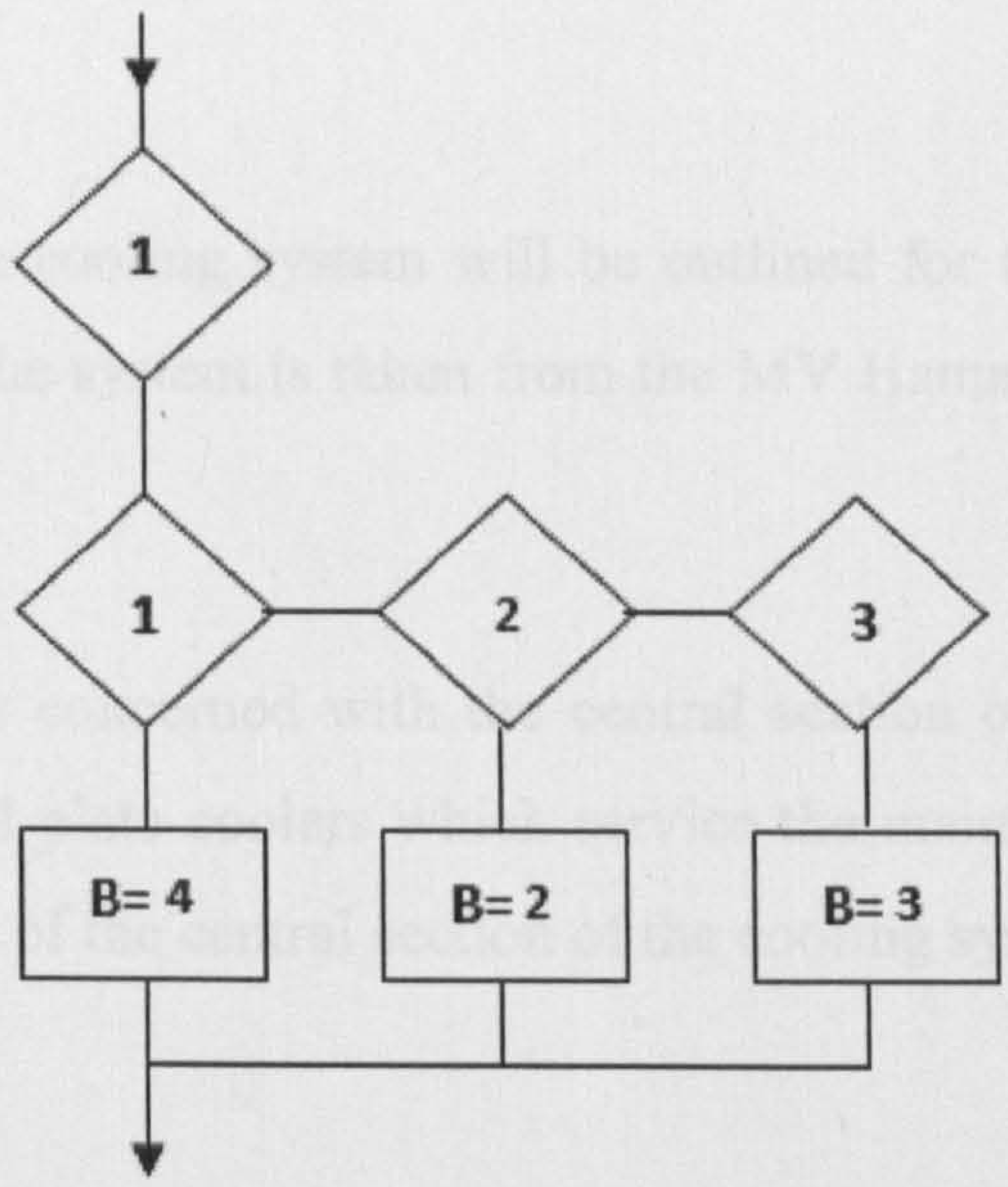

Fig 4.4 Flowchart mapping system transition logic

If conducted in a logical and rigorous manner the analyst is left with a number of flow diagrams which illustrate graphically how the system changes state with the failure of certain components in relation to the system state at the point of entry. The process of logically following the system transitions, through to an absorbing failure state, ensures that the analyst has a thorough understanding of the system and greatly facilitates code generation.

4.3.4 Code Generation

Code generation is now the process of converting the system transition flow diagrams into FORTRAN code using logical IF constructs. The code could be written in any language which allows the use of logical expressions and logical IF constructs. It should be noted that whereas production of an irreducible form of the system states is highly desirable, it is not essential. The production of the irreducible BRT means that the programmer can generate less code, as certain system states can be ignored. It is the definition of the operating constraints that decides which states are viable or not. The code will simply never enter into any states that the operational constraints do not allow.

4.4 Case Study

In the following a cooling system will be outlined for the purpose of the Monte Carlo analysis. The system is taken from the MV Hamnavoe, outlined previously in section 3.5.

This case study is concerned with the central section of the cooling system; the pumps, valves and plate coolers which service the main engine. Fig 4.5 shows a simplified version of the central section of the cooling system shown in Fig 3.10.

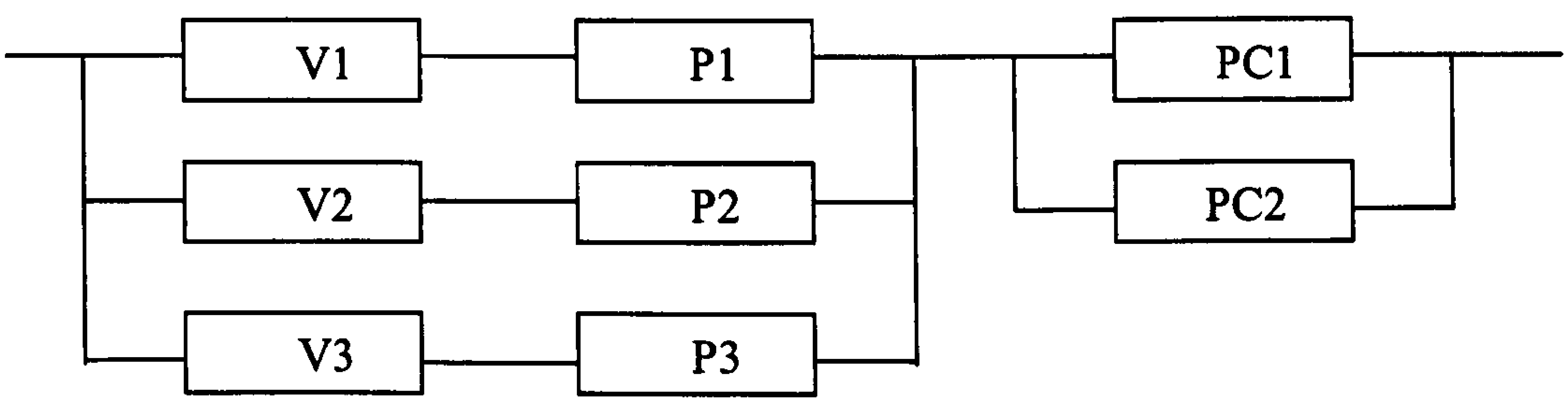

Fig 4.5 Diagram showing the system under consideration

V1, V2, V3, P1, P2, P3, PC1 and PC2 represent valve 1, valve 2, valve 3, pump 1, pump 2, pump 3, plate cooler 1 and plate cooler 2 respectively (highlighted in fig. 4.6). All the other components in the system are assumed to operate perfectly i.e. with no failure.

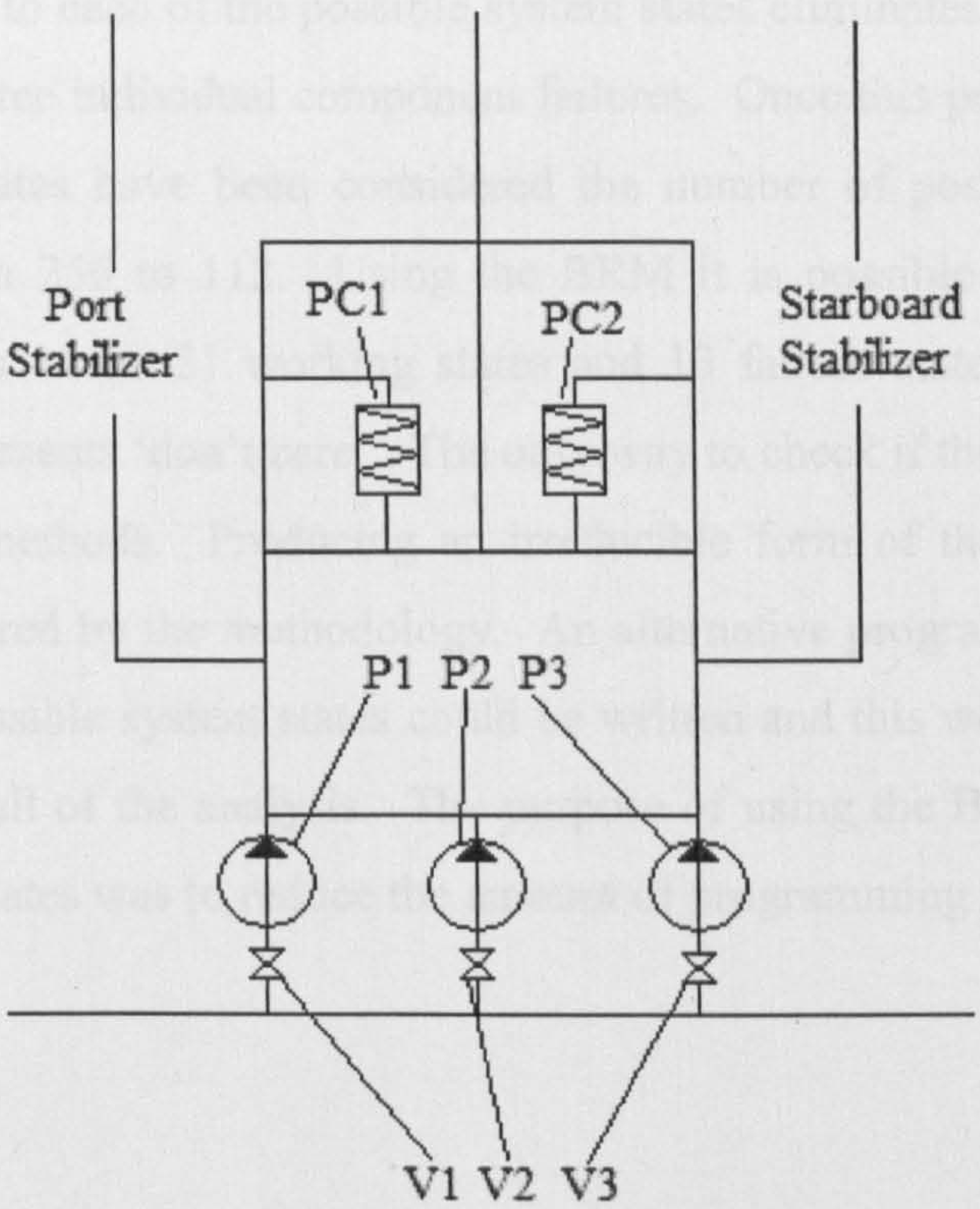

Fig 4.6 Diagram showing the components under consideration in the analysis

Each component has two modes of operation Working or Failed (W or F). The system constitutes eight individual components giving rise to 2^8=256 system states. This is a large number of states to be considered in a single analysis; operating requirements of the cooling system will serve to greatly reduce this number. The main engine requires a flow rate such that a minimum of two pumps must be functioning for adequate cooling to take place. The cooling system will still operate on a single pump. However for the purpose of this model it is assumed that when only a single pump is available for supply, the system can be considered to be in a 'failed' state and the main engine would have to be operated at a reduced load or taken offline completely until another pump was returned to its working state. This means that failure of all three pumps cannot occur as the system is considered failed after the failure of the first two pumps. For the same reason the failure of all three valves cannot occur. Failure of both the plate cooler units cannot occur for the same reasons; the system is considered failed when the first plate cooler fails. Once the system is in a failed state it is considered absorbing i.e. cannot leave this state, until the end of the repair time. Applying similar logic to each of the possible system states eliminates all system states with more than three individual component failures. Once this process is complete and all of the states have been considered the number of possible system states is reduced from 256 to 112. Using the BRM it is possible to further reduce the system states to the 21 working states and 13 failure states shown in table 4.7, where * represents 'don't care'. The only way to check if the table is irreducible is by manual methods. Producing an irreducible form of the system states is not strictly required by the methodology. An alternative program which encompasses all of the possible system states could be written and this would have no effect on the final result of the analysis. The purpose of using the BRM method to reduce the system states was to reduce the amount of programming required.

Table 4.7 Table showing the possible system states and failure states of the system

B	V1	V2	V3	P1	P2	P3	PC1	PC2
1	W	W	W	W	W	W	W	W
2	W	W	W	W	W	W	W	F
3	W	W	W	W	W	W	F	W
4	W	W	W	W	W	F	W	W
5	W	W	W	W	W	F	W	F
6	W	W	W	W	W	F	F	W
7	F	W	W	W	W	W	W	W
8	F	W	W	W	W	W	W	F
9	F	W	W	W	W	W	F	W
10	W	F	W	W	W	W	W	W
11	W	F	W	W	W	W	W	F
12	W	F	W	W	W	W	F	W
13	W	W	F	W	W	W	W	W
14	W	W	F	W	W	W	W	F
15	W	W	F	W	W	W	F	W
16	W	W	W	F	W	W	W	W
17	W	W	W	F	W	W	W	F
18	W	W	W	F	W	W	F	W
19	W	W	W	W	F	W	W	W
20	W	W	W	W	F	W	W	F
21	W	W	W	W	F	W	F	W
22	F	F	*	*	*	*	*	*
23	F	*	F	*	*	*	*	*
24	*	F	F	*	*	*	*	*
25	*	*	*	F	F	*	*	*
26	*	*	*	F	*	F	*	*
27	*	*	*	*	F	F	*	*
28	*	*	*	*	*	*	F	F
29	F	*	*	*	F	*	*	*
30	F	*	*	*	*	F	*	*
31	*	F	*	F	*	*	*	*
32	*	F	*	*	*	F	*	*
33	*	*	F	F	*	*	*	*
34	*	*	F	*	F	*	*	*

For the purpose of the descriptions of this case study it is necessary to define the following. A system is a collection of $m \geq 1$ components and the system state can be represented by the vector **B**=($b_1,b_2,....,b_m$) whose elements are the state indicators of the components. In the case of the system under consideration, the system state vector **B** will contain eight elements **B**=($b_1,b_2,b_3, b_4, b_5, b_6, b_7, b_8$) representing V1, V2, V3, P1, P2, P3, PC1 and PC2 respectively. In its nominal state with all components in a working state V2, P2 and PC2 are considered to be 'cold standby' (ST.) units which cannot fail while in standby. The system is required to function for a fixed amount of time, T_m, referred to as the mission time.

Three analyses will be conducted:

Case 1: The system is repairable, the repair rate will be deterministic, equal and fixed, denoted by t. In the event of a system failure all the failed components will be repaired before the system is put back online.

Case 2: The system is repairable. The repair times are deterministic but differ for each type of component, denoted by τ_v,τ_p and τ_{pc} representing the repair time of the valves, pumps and plate coolers respectively. In the event of system failure all the failed components will be repaired before the system is put back online.

Case 3: The system is repairable, the repair rate will be deterministic, equal and fixed, denoted by t. In the event of a system failure the system is returned to operation as soon as a minimum number of the failed components are repaired which allow normal operation.

The failure times of the system will be sampled using the inverse transform method based on the exponential distribution. The random number R_t will be generated using the intrinsic FORTRAN random number function. The failure rate λ^S represents a system failure rate and is equal to the sum of the failure rates of the working components in the system at the time of sampling. Table 4.8 shows the different system failure rates for the 21 working states obtained from table 4.7.

Table 4.8 Table showing the different system failure rates for the 21 working states of the system

Integer	B= $(V_1, V_2, V_3, P_1, P_2, P_3, PC_1, PC_2)$	λ_B^s
1	(W,ST.,W,W,ST.,W,W,ST.)	$\lambda_1^s = \lambda_{V1} + \lambda_{V3} + \lambda_{P1} + \lambda_{P3} + \lambda_{PC1}$
2	(W,ST.,W,W,ST.,W,W,F)	$\lambda_2^s = \lambda_{V1} + \lambda_{V3} + \lambda_{P1} + \lambda_{P3} + \lambda_{PC1}$
3	(W,ST.,W,W,ST.,W,F,W)	$\lambda_3^s = \lambda_{V1} + \lambda_{V3} + \lambda_{P1} + \lambda_{P3} + \lambda_{PC2}$
4	(W,W,ST.,W,W,F,W,ST.)	$\lambda_4^s = \lambda_{V1} + \lambda_{V2} + \lambda_{P1} + \lambda_{P2} + \lambda_{PC1}$
5	(W,W,ST.,W,W,F,W,F)	$\lambda_5^s = \lambda_{V1} + \lambda_{V2} + \lambda_{P1} + \lambda_{P2} + \lambda_{PC1}$
6	(W,W,ST.,W,W,F,F,W)	$\lambda_6^s = \lambda_{V1} + \lambda_{V2} + \lambda_{P1} + \lambda_{P2} + \lambda_{PC2}$
7	(F,W,W,ST.,W,W,W,ST.)	$\lambda_7^s = \lambda_{V2} + \lambda_{V3} + \lambda_{P2} + \lambda_{P3} + \lambda_{PC1}$
8	(F,W,W,ST.,W,W,W,F)	$\lambda_8^s = \lambda_{V2} + \lambda_{V3} + \lambda_{P2} + \lambda_{P3} + \lambda_{PC1}$
9	(F,W,W,ST.,W,W,F,W)	$\lambda_9^s = \lambda_{V2} + \lambda_{V3} + \lambda_{P2} + \lambda_{P3} + \lambda_{PC2}$
10	(W,F,W,W,ST.,W,W,ST.)	$\lambda_{10}^s = \lambda_{V1} + \lambda_{V3} + \lambda_{P1} + \lambda_{P3} + \lambda_{PC1}$
11	(W,F,W,W,ST.,W,W,F)	$\lambda_{11}^s = \lambda_{V1} + \lambda_{V3} + \lambda_{P1} + \lambda_{P3} + \lambda_{PC1}$
12	(W,F,W,W,ST.,W,F,W)	$\lambda_{12}^s = \lambda_{V1} + \lambda_{V3} + \lambda_{P1} + \lambda_{P3} + \lambda_{PC2}$
13	(W,W,F,W,W,ST.,W,ST.)	$\lambda_{13}^s = \lambda_{V1} + \lambda_{V2} + \lambda_{P1} + \lambda_{P2} + \lambda_{PC1}$
14	(W,W,F,W,W,ST.,W,F)	$\lambda_{14}^s = \lambda_{V1} + \lambda_{V2} + \lambda_{P1} + \lambda_{P2} + \lambda_{PC1}$
15	(W,W,F,W,W,ST.,F,W)	$\lambda_{15}^s = \lambda_{V1} + \lambda_{V2} + \lambda_{P1} + \lambda_{P2} + \lambda_{PC2}$
16	(ST.,W,W,F,W,W,W,ST.)	$\lambda_{16}^s = \lambda_{V2} + \lambda_{V3} + \lambda_{P2} + \lambda_{P3} + \lambda_{PC1}$
17	(ST.,W,W,F,W,W,W,F)	$\lambda_{17}^s = \lambda_{V2} + \lambda_{V3} + \lambda_{P2} + \lambda_{P3} + \lambda_{PC1}$
18	(ST.,W,W,F,W,W,F,W)	$\lambda_{18}^s = \lambda_{V2} + \lambda_{V3} + \lambda_{P2} + \lambda_{P3} + \lambda_{PC2}$
19	(W,ST.,W,W,F,W,W,ST.)	$\lambda_{19}^s = \lambda_{V1} + \lambda_{V3} + \lambda_{P1} + \lambda_{P3} + \lambda_{PC1}$
20	(W,ST.,W,W,F,W,W,F)	$\lambda_{20}^s = \lambda_{V1} + \lambda_{V3} + \lambda_{P1} + \lambda_{P3} + \lambda_{PC1}$
21	(W,ST.,W,W,F,W,F,W)	$\lambda_{21}^s = \lambda_{V1} + \lambda_{V3} + \lambda_{P1} + \lambda_{P3} + \lambda_{PC2}$

At the inception of each trial the system is always in the nominal configuration i.e. **B**=1. The failure times are sampled and the FORTRAN programme performs a series of 'tests'. The first of these tests is to ensure that the transition time is within the mission time of the system; if it is found to be outside of the mission time the programme performs no action and moves onto the next trial. When a transition occurs within the mission time the program has to ascertain the nature of the transition. This is achieved by creating a discrete distribution, between 0 and 1, containing the normalised failure rates for each of the working components. Fig 4.7 shows how the interval is divided into the normalised failure probabilities.

$$\frac{\lambda_{v1}}{\lambda_s} \quad \Big| \quad \frac{\lambda_{v3}}{\lambda_s} \quad \Big| \quad \frac{\lambda_{p1}}{\lambda_s} \quad \Big| \quad \frac{\lambda_{p3}}{\lambda_s} \quad \Big| \quad \frac{\lambda_{pc1}}{\lambda_s}$$

Fig 4.7 Diagram illustrating how the interval is divided into discrete probabilities pertaining to each working component, given that a transition has occurred

A second random number, uniformly distributed between 0 and 1, can now be generated using FORTRAN's intrinsic random number generator. The random number indicates which of the components has undergone a transition. The transition of any individual component will lead to a change in the system state and hence the system state vector **B**. Fig 4.8 shows one of the flow charts produced to map the system transitional logic. It can be seen how an independent component transition leads to a different system state rather than that of any other independent component transition. For example when the system enters in state **B**=1 and P1 undergoes a transition, the system leaves in state **B**=16. However if the system enters in state **B**=1 and P3 undergoes a transition, the system leaves in state **B**=4.

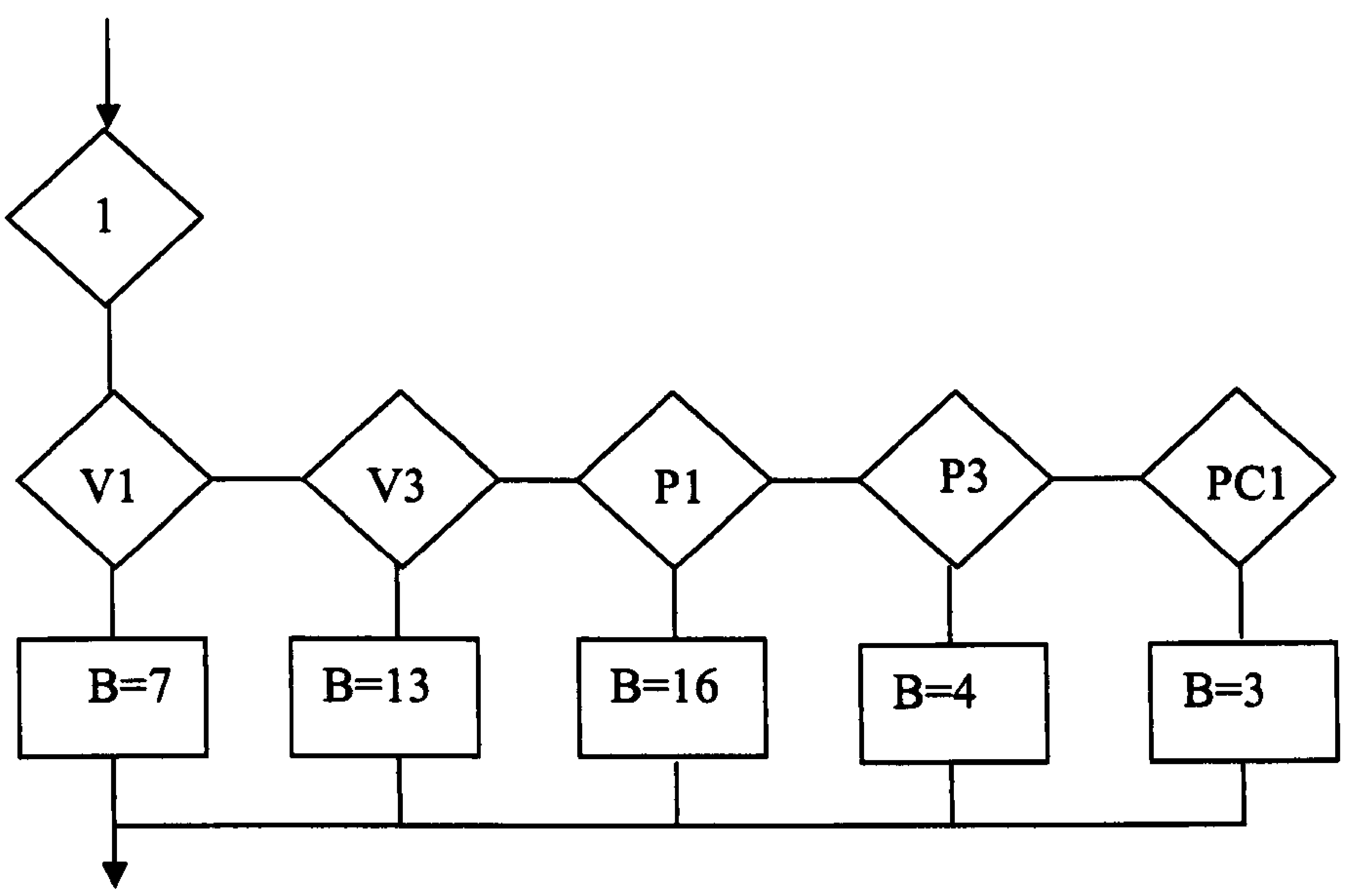

Fig 4.8 Flowchart showing how the system state changes according to which component undergoes a transition

Before the system could be modelled fully, flow diagrams for each of the working system states were generated. This facilitated a rigorous understanding of how individual component failures could alter the system state according to the state of the system at the point of entry.

Once all the flow diagrams had been completed, FORTRAN code was generated to represent the system and is shown in appendix 1. The FORTRAN program uses an integer value **B**, like that outlined above, to ascertain the state of the system. The system now undergoes a random walk through the various possible system states according to which component fails. Each component failure is recorded in its own failure counter. There is also a system failure counter which records when the system has failed. In the event of a system failure for each case the following happens:

Case 1: The system is moved on in the time frame of the analysis by the deterministic repair time, t, and only starts sampling for failures again once all the components are repaired and returned to a working state.

Case 2: The system is moved on in the time frame of the analysis by the longest deterministic repair time, τ_v,τ_p or τ_{pc}, and only starts sampling for failures again once all the components are repaired and returned to a working state.

Fig 4.9 shows a failure occurring at t_1. The next failure occurs at t_2 and the system fails. The system is not put back online until t_3 when all repair operations are completed as described in cases 1 and 2.

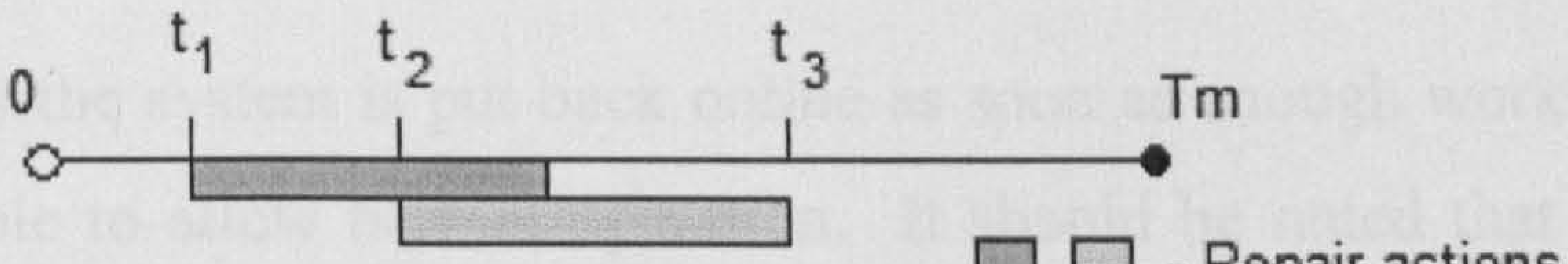

Fig 4.9 Timeline illustrating how all repair actions are completed before the system is put back online

Case 3: The system is moved on by an amount Δt which is determined according to which component failed first. This is where the system transition flow diagrams prove their worth. Analysis of the flow diagrams can allow the analyst to ascertain which component failed first according to the state it is in at present. Fig 4.10 shows the time interval of interest Δt.

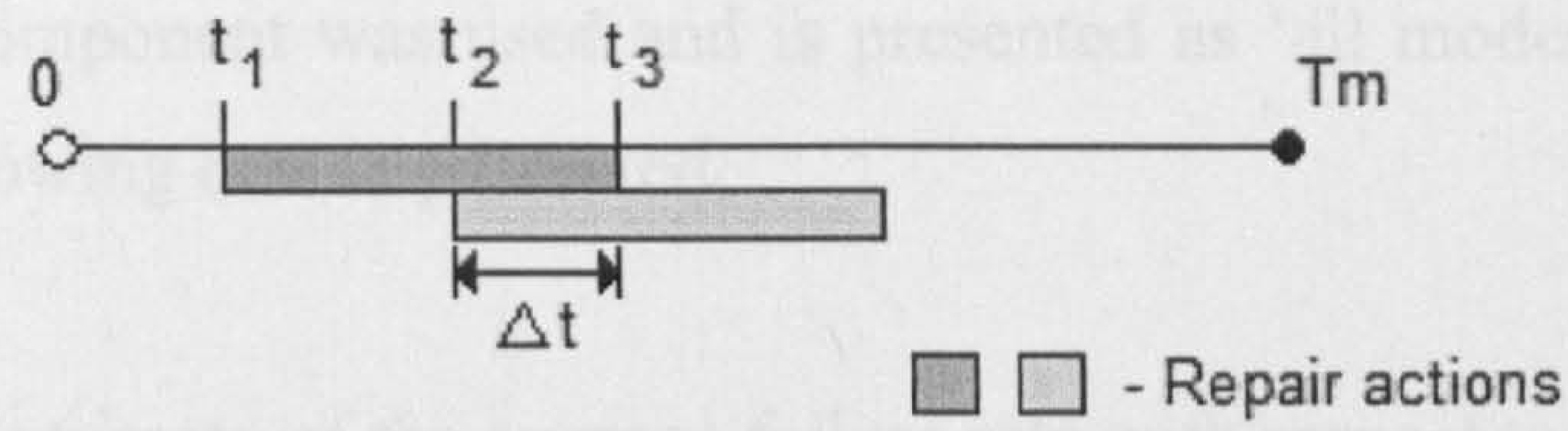

Fig 4.10 Timeline illustrating how the system is put back online as soon As enough components are repaired, with the time interval of Interest Δt

Consider the case where at time t_1 plate cooler 2 has failed. The next transition is at time t_2, the current time, where a second working component has failed. This leads to a system failure however as soon as the repair of plate cooler 2 is complete, the system can be put back online. The point at which the system is put back online is at t_3. The simulation must be advanced through the mission by the value Δt. The following equation gives the value of Δt:

$$\Delta t = (t_1 + \tau_*) - t_2$$

where

τ_*= repair time of the component which failed first.

In this way the system is put back online as soon as enough working components are available to allow normal operation. It should be noted that in this case the system never returns to the nominal state. The nominal state is that where all components are working but certain components are cold standby units. Also the system states are reduced as states which involve the failure of more than two components cannot exist.

4.5 Case Study Results

4.5.1 Case 1

The failure rates for the individual components in the program were taken from the OREDA handbook (OREDA 2002). For this study the accumulated number of failures for each component was used and is presented as 'all modes'. For each component the following data is presented.

- Mean – An estimate of the average failure rate with respect to the specified failure mode, obtained using the OREDA estimator. Further information about the OREDA estimator can be found in OREDA (2002).
- Lower, Upper.
- SD - A standard deviation indicating the variation between the multiple samples.
- n/τ – The total number of failures divided by the total time in service.
- Active repair hours – This shows the average calendar time (hours) required to repair and return the item to a repaired state. This is the time when actual repair work is being done. It does not include time factors such as shutdown, issue of work orders, waiting for spares, start up etc. The active repair hours are often shorter than the downtimes.
- Repair(man-hours) – The mean value is the average number of repair hours recorded to repair the failure and restore function. The Min and Max values represent the shortest and longest recorded repair times respectively.

Table 4.9 shows the failure data for centrifugal pumps in the marine industry, the data encompasses a population of 350 units over 59 offshore installations.

Table 4.9 OREDA failure data for centrifugal machinery pumps

Failure mode	No. of failures	Failure rate (per 10^6 hours)					Active repair hours	Repair (man-hours)		
		Lower	Mean	Upper	SD	n/τ		Min	Mean	Max
All modes	1949	172.90	1277.00	3233.73	1001.75	339.22	21.3	0.5	30.5	1025.0

Table 4.10 shows the failure data for ball valves, the data includes a population of 316 units over 18 offshore installations.

Table 4.10 OREDA failure data for ball valves

Failure mode	No. of failures	Failure rate (per 10^6 hours)					Active repair hours	Repair (man-hours)		
		Lower	Mean	Upper	SD	n/τ		Min	Mean	Max
All modes	328	8.80	43.70	100.54	29.47	35.18	6.4	1.0	10.7	113.0

Table 4.11 shows the failure data for plate heat exchangers, water→sea water, the data encompasses a population of 8 units over 3 offshore installations.

Table 4.11 OREDA failure data for plate heat exchangers

Failure mode	No. of failures	Failure rate (per 10^6 hours)					Active repair hours	Repair (man-hours)		
		Lower	Mean	Upper	SD	n/τ		Min	Mean	Max
All modes	9	0.27	39.75	137.51	50.71	35.25	13.7	6.0	29.0	109.0

OREDA also documented failure data which was specific to centrifugal pumps in cooling systems. This data only covered 18 units from 2 offshore installations and therefore the data for centrifugal pumps in general was chosen for the analysis. The mission time was based on the passage time from Liverpool→Shanghai→New York based on an average speed of 25 knots.

The passage time with a speed of 25 knots works out at 22 days and 4 hours. This gives a mission time of 532 hours. After a full mission time the components are treated as 'same as new'. A deterministic repair time of 10 hours is used as this represents the shortest mean repair time in the OREDA data. For the initial analysis table 4.12 shows the input values used.

Table 4.12 Input values for the Monte Carlo analysis

T_m (hrs)	532
$\boldsymbol{\lambda_{v1}}$**(hrs^{-1})**	4.37×10^{-5}
$\boldsymbol{\lambda_{v2}}$**(hrs^{-1})**	4.37×10^{-5}
$\boldsymbol{\lambda_{v3}}$**(hrs^{-1})**	4.37×10^{-5}
$\boldsymbol{\lambda_{p1}}$**(hrs^{-1})**	1.277×10^{-3}
$\boldsymbol{\lambda_{p2}}$**(hrs^{-1})**	1.277×10^{-3}
$\boldsymbol{\lambda_{p3}}$**(hrs^{-1})**	1.277×10^{-3}
$\boldsymbol{\lambda_{pc1}}$**(hrs^{-1})**	3.975×10^{-5}
$\boldsymbol{\lambda_{pc2}}$**(hrs^{-1})**	3.975×10^{-5}
t (hrs)	10

Tables 4.13 and 4.14 show the results of the analysis.

Table 4.13 Table showing the number of system and individual component failures for case 1

N	$1.00\times \mathbf{10^6}$	$1.00\times \mathbf{10^7}$	$1.00\times \mathbf{10^8}$	$1.00\times \mathbf{10^9}$
FSYS	19776	196367	1962569	19618928
FV1	5853	58511	584952	5849893
FV2	2658	26660	265967	1657886
FV3	5720	57296	573216	5729899
FP1	170739	1709502	17097265	170933348
FP2	77647	776209	7762847	77647017
FP3	167424	1674321	16745087	167455943
FPC1	2647	26540	264810	2649558
FPC2	38	396	3884	39322

It can be seen in table 4.13 that as the number of trials increases the number of failures also increases. Considering FV1, after 10^6 trials FV1 shows 5853 failures; when the number of trials is increased to 10^8 it now shows 584952 failures. The system failure rate FSYS follows the same trend, this is reflected in the failure probability shown in table 4.14. It can be seen that as the number of trials increases the failure probability becomes more accurate and converges on a result.

Table 4.14 Table showing the failure probability for case 1

N	Failure Probability
1.00×10^{6}	$\mathbf{1.98\times 10^{-2}}$
1.00×10^{7}	$\mathbf{1.96\times 10^{-2}}$
1.00×10^{8}	$\mathbf{1.96\times 10^{-2}}$
1.00×10^{9}	$\mathbf{1.96\times 10^{-2}}$

Fig 4.11 shows a graph of the results for case 1.

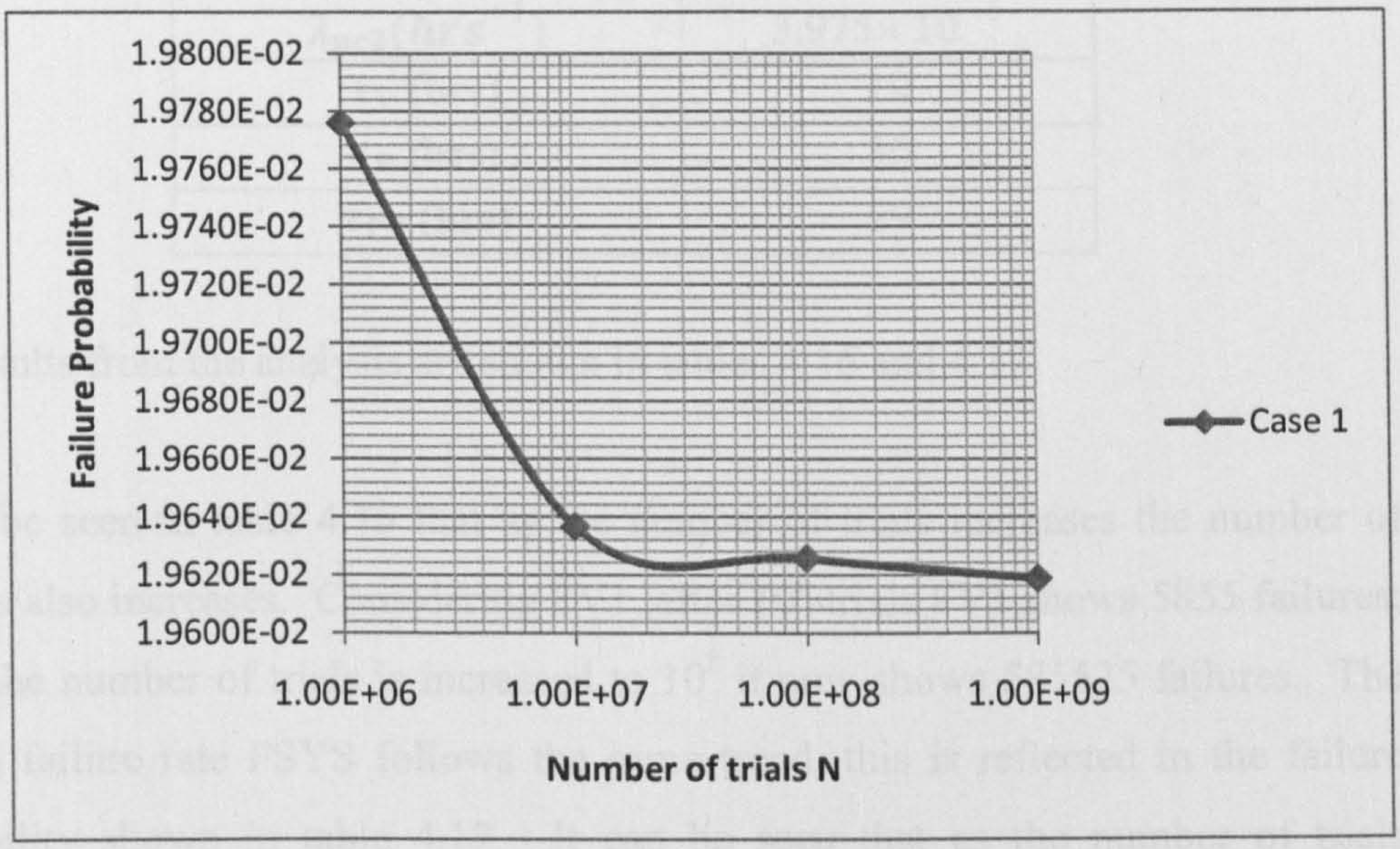

Fig 4.11 Graph showing the results of the analysis of case 1

4.5.2 Case 2

In this analysis the repair rates of the components were altered. Each type of component had an associated deterministic repair time, taken from the OREDA data for centrifugal pumps, ball valves and plate coolers shown in tables 4.9, 4.10 and 4.11 respectively. The input data for the second analysis is shown in table 4.15.

Table 4.15 Table showing the input values for case 2

T_m (hrs)	532
$\lambda_{v1}(hrs^{-1})$	4.37×10^{-5}
$\lambda_{v2}(hrs^{-1})$	4.37×10^{-5}
$\lambda_{v3}(hrs^{-1})$	4.37×10^{-5}
$\lambda_{p1}(hrs^{-1})$	1.277×10^{-3}
$\lambda_{p2}(hrs^{-1})$	1.277×10^{-3}
$\lambda_{p3}(hrs^{-1})$	1.277×10^{-3}
$\lambda_{pc1}(hrs^{-1})$	3.975×10^{-5}
$\lambda_{pc2}(hrs^{-1})$	3.975×10^{-5}
t_V (hrs)	10
t_P (hrs)	30
t_{PC} (hrs)	29

The results from the analysis are shown in tables 4.16 and 4.17.

It can be seen in table 4.16 that as the number of trials increases the number of failures also increases. Considering FV1, after 10^6 trials FV1 shows 5855 failures; when the number of trials is increased to 10^8 it now shows 585525 failures. The system failure rate FSYS follows the same trend, this is reflected in the failure probability shown in table 4.17. It can be seen that as the number of trials increases the failure probability becomes more accurate and converges on a result.

Table 4.16 Table showing the individual component and system failures for case 2

N	$1.00\times \mathbf{10^6}$	$1.00\times \mathbf{10^7}$	$1.00\times \mathbf{10^8}$	$1.00\times \mathbf{10^9}$
FSYS	56666	565021	5651882	56535012
FV1	5855	58537	585525	5854506
FV2	2606	26184	260968	2607879
FV3	5747	57484	575163	5750159
FP1	176997	1772428	17727017	177234998
FP2	76341	763130	7631098	76330190
FP3	174170	1741913	17421692	174221371
FPC1	3002	29851	297602	2977773
FPC2	38	419	1005452	17381592

Table 4.17 Table showing the failure probability for case 2

N	Failure Probability
1.00×10^{6}	$\mathbf{5.67\times 10^{-2}}$
1.00×10^{7}	$\mathbf{5.65\times 10^{-2}}$
1.00×10^{8}	$\mathbf{5.65\times 10^{-2}}$
1.00×10^{9}	$\mathbf{5.65\times 10^{-2}}$

Fig 4.12 shows a graph of the results for case 2.

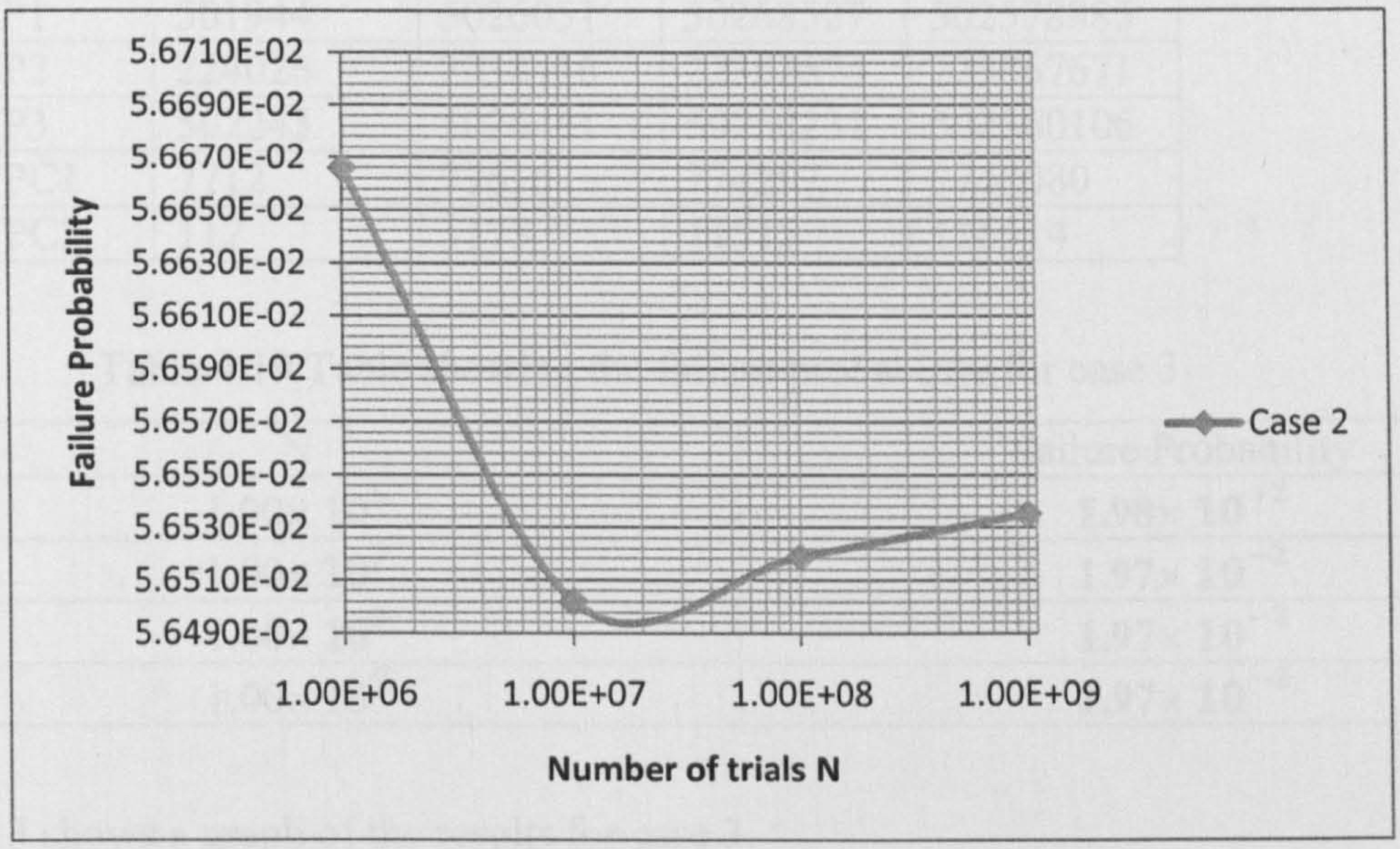

Fig 4.12 Graph showing the results of the analysis of case 2

4.5.3 Case 3

In case 3 the input values for the simulation are the same as those in case 1 shown in table 4.12. In this case the system is returned to operation as soon as enough working components are available. Tables 4.18 and 4.19 show the results for case 3.

It can be seen in table 4.18 that as the number of trials increase the number of failures also increase. Considering FV1, after 10^6 trials FV1 shows 17193 failures; when the number of trials is increased to 10^8 it now shows 1720046 failures. The system failure rate FSYS follows the same trend, this is reflected in the failure probability shown in table 4.19. It can be seen that as the number of trials increase the failure probability becomes more accurate and converges on a result.

Table 4.18 Table showing the individual component and system failures for case 3

N	1.00×10^6	1.00×10^7	1.00×10^8	1.00×10^9
FSYS	19837	197022	1968662	19678990
FV1	17193	172060	1720046	17200986
FV2	7636	76874	766380	7661648
FV3	17175	171962	1720299	17195361
FP1	501944	5026051	50268527	502578885
FP2	224026	2238416	22382674	223867671
FP3	502243	5024461	50254231	502560106
FPC1	7712	77616	774297	7748080
FPC2	112	1173	11515	116514

Table 4.19 Table showing the failure probability for case 3

N	Failure Probability
1.00×10^6	$\mathbf{1.98\times 10^{-2}}$
1.00×10^7	$\mathbf{1.97\times 10^{-2}}$
1.00×10^8	$\mathbf{1.97\times 10^{-2}}$
1.00×10^9	$\mathbf{1.97\times 10^{-2}}$

Fig 4.13 shows a graph of the results for case 3.

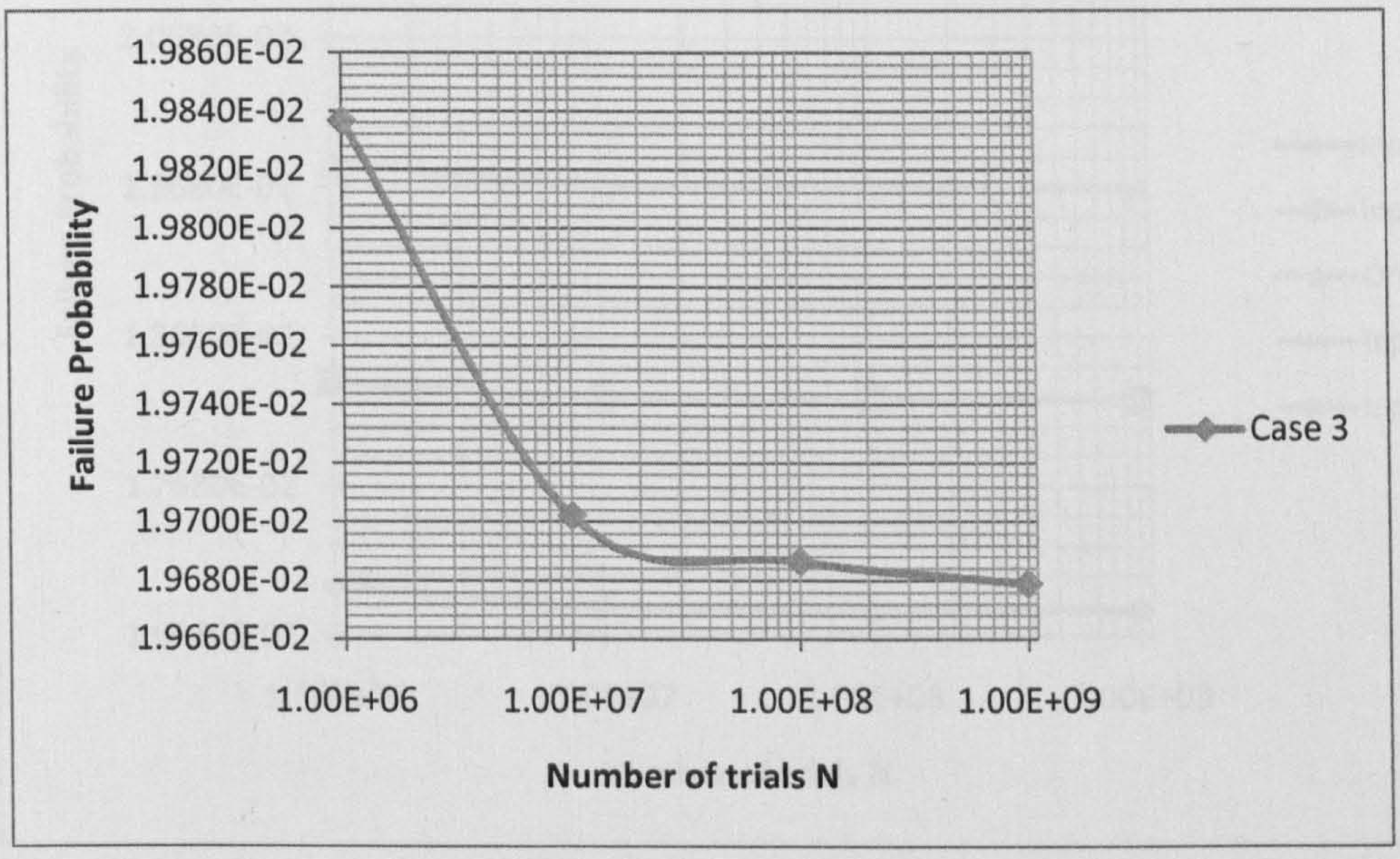

Fig 4.13 Graph showing the results of the analysis of case 3

4.5.4 Verification of Model

A sensitivity analysis can be used to partially verify a result or model. The European commission's guidelines for modelling prescribe sensitivity analysis as a tool to ensure the quality of the model (SEC(2005)791). The simplest form of a sensitivity analysis is one where the input variables are altered by varying degrees. In sensitivity analysis one looks at the effect of varying the inputs of a mathematical model on the output of the model itself. In the model presented the input variables will be altered up and down by various percentages. It is expected, if the model is functioning as it should, that an increase in the failure rates will result in an increased unreliability of the system. The basic code for the model is the same for each of the cases. It was decided, because of this, that partial validation of a single case would be sufficient. Case 3 was selected for the sensitivity analysis and each input component failure rate was varied by ±5% and ±10%. The results are shown in fig 4.14.

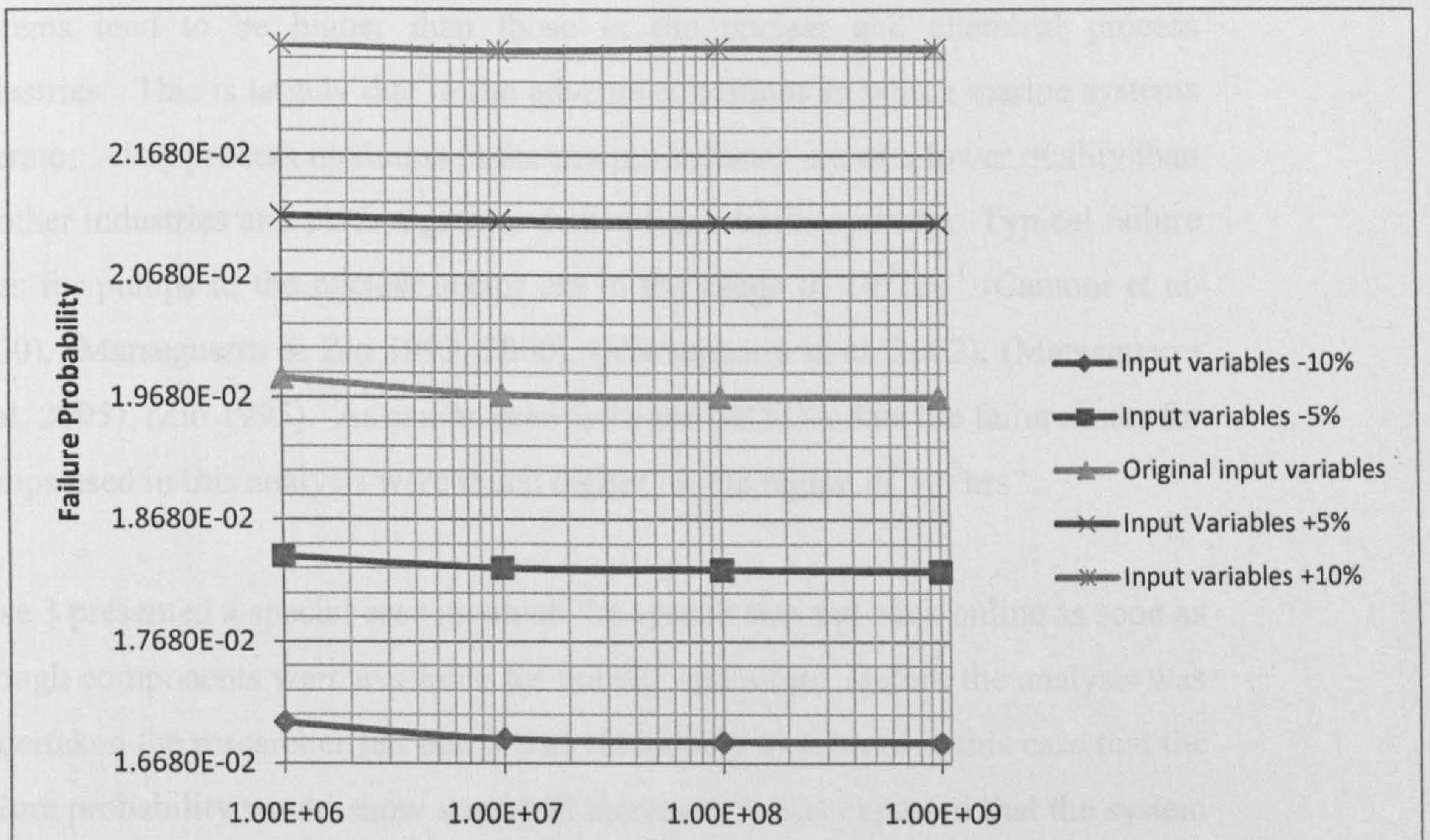

Fig 4.14 Graph showing the sensitivity analysis

4.6 Discussion

The sensitivity analysis partially verified the results; the resulting change in failure probability due to the change in the input failure rates is as would be expected. Reducing the input variables produced a reduction in the failure probability, conversely increasing the input variables had the effect of increasing the failure probability. This shows that the program functions as it should.

The greatest change in the failure probability occurred when the parameters of case 2 were applied. The failure probability was approximately three times higher than that of the other two cases. This was expected as in general the components experienced longer deterministic repair times. However a simple check was performed to ensure that the result was true. The check was done by changing the deterministic repair times of case 2 so that all were equal to 10 hours. The code was run for 10^9 trials and produced the same results as those documented for case 1. The simulation converged on a result by 10^9 trials. The failure rates of marine systems tend to be higher than those in the nuclear and chemical process industries. This is largely due to the adverse conditions in which marine systems operate. Also, process mediums in the marine industry are of a lower quality than in other industries and place a greater demand on marine systems. Typical failure rates for pumps in the nuclear sector are in the range of $10^{-5}hrs^{-1}$ (Cantoni et al. 2000), (Marseguerra & Zio 1993, 2000), (Marseguerra et al. 2002), (Marseguerra et al. 2005), (Zio 1995). As can be seen from the OREDA data the failure rates for pumps used in this analysis were much higher, in the region of $10^{-3}hrs^{-1}$.

Case 3 presented a special case in which the system was put back online as soon as enough components were available for normal operation. Before the analysis was undertaken the researcher felt that it was reasonable to assume in this case that the failure probability would show a marked increase. It was expected that the system would be more vulnerable to failure as at times the system would be operating without the security of cold standby units. The analysis proved this conjecture to be incorrect. The failure probability for case 1 was 1.96×10^{-2} and for case 3 was 1.97×10^{-2}. This showed only a small change in the failure probability. It should be noted that this case was unique in the fact that the repair rates were once again deterministic and equal. The simulation program does not allow for simultaneous failures of components. This means that even if three failures occurred it was always the first component to fail that was the first to be repaired. Often as soon as the first component was repaired it was put back online. If the individual component's repair times were not equal but varied according to component type, as in case 2, this would not always be the case. Consider a hypothetical case where the first failure involved the failure of one of the plate cooler units, with a deterministic repair time of 50 hrs. In the next two instances two valves fail, with deterministic repair time of 10 hrs, which takes the system offline. Then the repair of the valves would be completed before that of the plate cooler and the system put back online. An analysis where components had individual failure rates would involve considering whether the second failure is repaired before the end of the first. This would depend on the nature of the first and second failures.

The system under analysis was based on a system currently in operation on board a Ro-Ro passenger ferry. The researcher believes that the cooling systems are suitably generic that the model could be applied to most cooling systems in operation. It should be noted however that a Ro-Ro passenger ferry is likely to have a much shorter passage time and therefore a much shorter mission time. The shorter mission time means that more trials must be completed to produce reliable results.

4.7 Conclusion

The purpose of this chapter was to demonstrate how MCM can be utilised within the marine industry and, due to the nature of marine systems, how results can be produced without the need for biasing. The study highlighted the fact that this kind of simulation program could provide a useful tool for marine engineers on board ship. If a simulation program could be produced, which would allow the marine engineer to define a system and perform an analysis, then decisions concerning maintenance could be tested before any changes were made to the real life systems. Case 3 proved that complex systems do not always behave in a way that an analyst may logically expect. Putting the system back online does not have a significant effect on the failure probability but would possibly make the system more available to the marine engineers onboard. The beauty of simulation is that it allows for 'trial and error' runs without incurring any major cost or loss of system availability. At present it is common practice within the engineering department to perform service actions at designated intervals. Often parts are replaced without considering the premise that certain parts may not need replacing in the first place. If a marine engineer could gather suitable failure data about a piece of equipment over a period of time then simulation could be used to test new service intervals. This could ultimately save cost. Another area where simulation could play a huge part is the replacement of system units. In the analysis it was seen that the cold standby units recorded a significantly smaller number of failures. Could less reliable units be used? What effect would this have on the failure probability? Could less reliable units be acquired at lower capital cost? The answer of three such questions could be very useful in improving ship maintenance planning and operations.

The feasibility of MCM within the marine industry is ultimately reliant on the availability of failure data. OREDA provides a great source of information from the offshore industries. In order for this and similar research to progress, develop and become practicable, further efforts must be made by the marine industry as a whole, to collect and collate failure data. Finally, it should be remembered that the accuracy of the simulation is always dependant on the accuracy of the failure data.

Chapter 5 – Application of Delay-Time Analysis via Monte Carlo Simulation

Summary

This chapter has been written to give a methodology for the application of delay time analysis via simulation. The aim of this chapter is to demonstrate the efficacy and worth of delay-time analysis and how the application can provide engineers with more information when making maintenance decisions. A methodology has been developed and applied to two case studies.

5.1. Introduction

It has already been shown in the previous chapters that MCM can be applied in the marine environment to give information about system unreliability based on system failure rates. An analysis was also developed, which showed how development of MCM in the marine industry could allow engineers to run simulations of complex systems. Input variables and maintenance decision can be 'tested' within the simulation and the effects on system unreliability assessed. The purpose of this chapter is to show how MCM can also be used in other ways to facilitate maintenance decisions. Specifically DTA methods will be implemented using MCM to automate the process and produce results. DTA can be easily achieved through simulation methods.

5.2. Background

5.2.1 The Delay-Time Concept

The majority of current reliability and maintenance practice is based on time to first failure, or time between failures. Christer (1999) published a review considering the developments in DTA, stating that 'maintenance concepts based on RCM or TPM are prescriptive and often lack scientific concept, testing, verification or validation'. Delay-time modelling is a concept which has been developed to be relevant in the operating culture of today's industry (Christer 1999). DTA provides engineers with a tool which can help to minimise downtime, D(T) of a machine or plant item, based on an inspection period, T. The delay-time concept bifurcates the failure process as shown in fig 5.1.

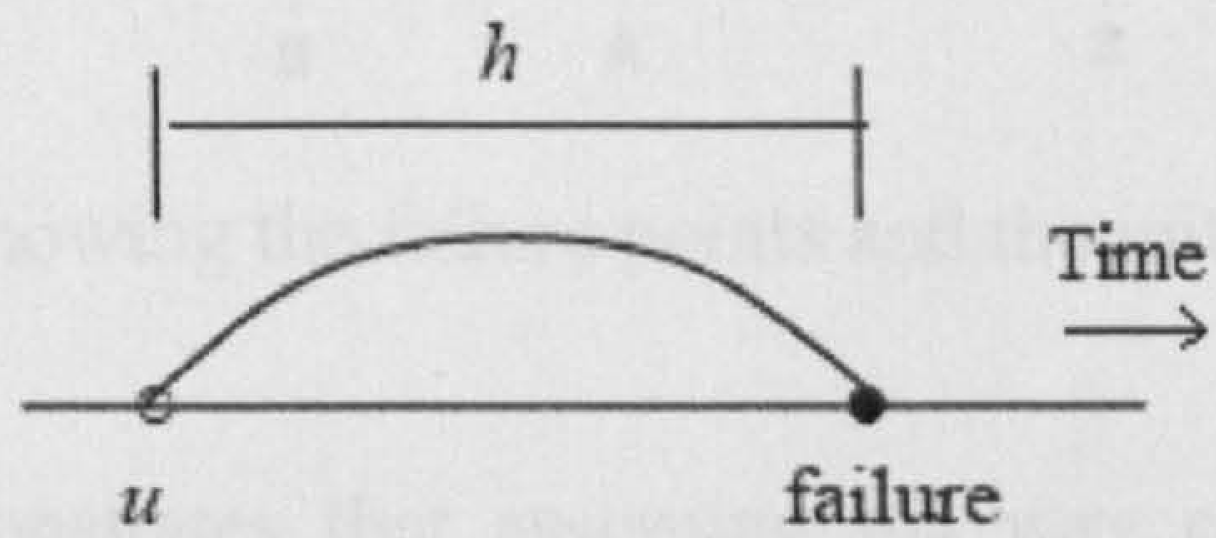

Fig 5.1 Diagram showing the delay-time concept (Christer 1999)

DTA is based on the idea of all failures having an individual 'tell-tale' sign. This is represented in fig 5.1 by the point, u, on the time line. The point u is called the initial point and is the point from which normal inspection activity could highlight the defect. If unattended the component will go on to fail at point u+h; where h is the time to failure of the component from point u, here-in referred to as the delay-time. If an inspection is scheduled to take place in the time period (u,u+h), then the failure could be discovered and arrested before it leads to full failure. If this initial point, u, exists for a number of failure conditions, then the delay-time represents a window in which failure could be prevented. To fully understand the benefit of the delay-time concept, consider the following example presented in Christer (1999).

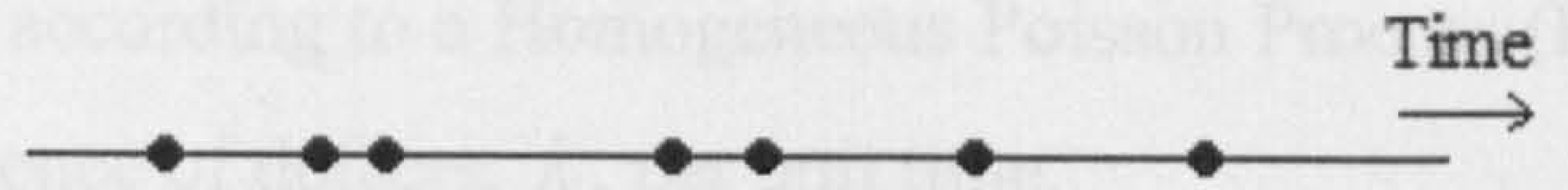

Fig 5.2 Diagram showing the failure points (Christer 1999)

Consider fig 5.3 incorporating the same failure point pattern as fig 5.2 along with the initial points associated with each failure arising under a breakdown system. Had an inspection taken place at point (A), one defect could have been identified and the seven failures reduced to six. Likewise had inspection taken place at points (B) and point (A), 4 defects could have been identified and the seven failures now reduced to three.

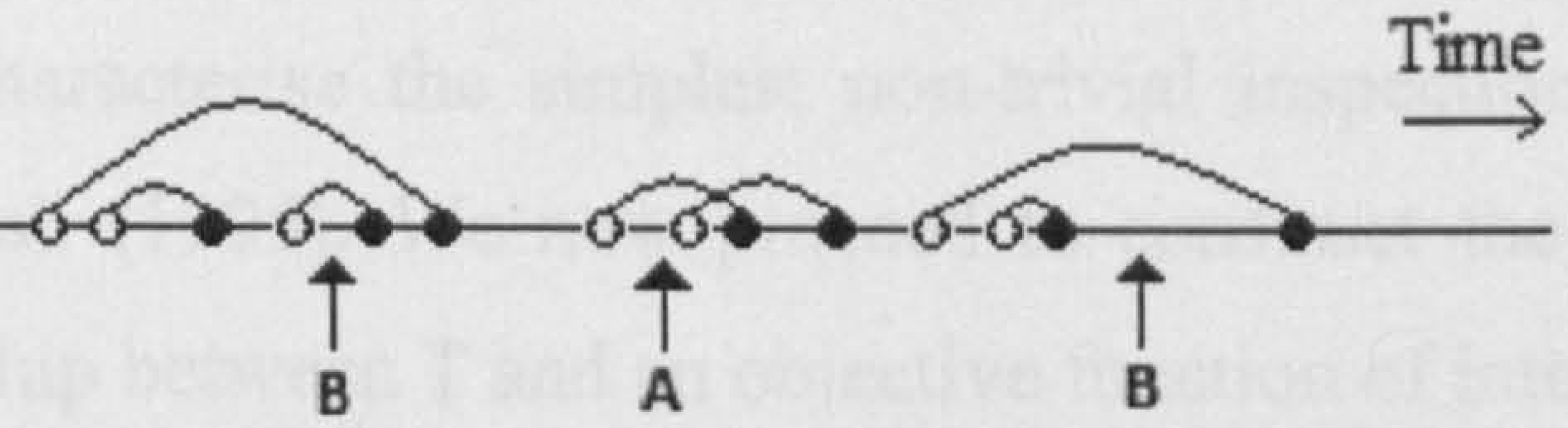

Fig 5.3 Diagram showing the failure points and the initial points (Christer 1999)

This example demonstrates that assuming the way can be modelled in which defects arrive, referred to as the arrival rate of defects k_f, and their delay-time h, the DTA concept can be applied to understand the relationship between inspection frequency and system failures (Christer 1995).

Here we present briefly the simplest delay time model used in the literature. We asssume there is a complex plant, or multi-component plant which has a large number of components with many failure modes, and the correction of one defect or failure has nominal impact in the steady state upon the overall plant failure characteristics. Considering the following basic complex plant maintenance modelling scenario where:

1. An inspection takes place every T time units, costs c_s units and requires d_s time units, where $d_s \ll T$.
2. Inspections are perfect in that all (and only) defects present are identified.
3. Defects identified are repaired during the inspection period.

4. Defects arise according to a Homogeneous Poisson Process (HPP) with the rate of occurrence of defects, λ, per unit time.
5. The delay time, H, of a random defect is described by a pdf. $f(h)$, cdf. $F(h)$, and is independent of the initial point U.
6. Failure will be repaired immediately at an average cost c_f and downtime d_f.
7. The plant has operated sufficiently long since new to be considered effectively in a steady state.
8. Defects and failures only arise whilst plant is operating.

These assumptions characterise the simplest non-trivial inspection maintenance problem, Christer et al. (1995). We now proceed to construct the mathematical model of the relationship between T and an objective function of interest.

From assumptions 1-4, it is obvious that the number of system failures is identical and independent over each inspection interval, and we can simply study the behaviour of such a failure process over one interval, say the first interval $[0,T)$. Suppose for now that we take the expected downtime per unit time, $D(T)$, as a measure of our objective function, the relationship between T and $D(T)$ can be established directly by using the renewal reward theorem, Ross (1981), as

$$D(T) = \lim_{t\to\infty} \frac{E(\text{Downtime over t})}{t} = \frac{d_f E[(N_f(T)] + d_s}{T + d_s} \tag{5.1}$$

where $E[N_f(T)]$ is the expected number of failures within [0,T). Clearly if $E[N_f(T)]$ is available, $D(T)$ can be readily calculated. It is shown that $E[N_f(T)]$ is given by:

$$E[N_f(T)] = \int_0^T \lambda F(t)dt \tag{5.2}$$

5.2.2 Monte Carlo Methods – Brief Overview

Monte Carlo Methods are implemented in several sources (Rubinstein 1981), (Manno 1999), (Dubi 2000), (Marseguerra & Zio 2002). Uniform random numbers are used to generate random system failure times from a continuous distribution via the inverse transform method (Rubinstein 1981).

5.2.2.1 The Weibull Distribution

The Weibull distribution is often used in the field of life failure analysis; it can mimic other distributions such as the normal or exponential. The Weibull distribution is chosen for this analysis as it is the only distribution flexible enough to represent the infant mortality, steady state, and wear-out periods associated with a component lifetime. An understanding of the failure rate can provide insight as to the types of failures occurring. It is a continuous distribution and its probability density function has the form:

$$f(x)=\begin{cases}\dfrac{k}{\lambda}\left(\dfrac{x}{\lambda}\right)^{k-1} e^{-\left(\frac{x}{\lambda}\right)^{k}} & x \geq 0 \\ 0 & x < 0\end{cases} \tag{5.3}$$

where:

$k > 0$ is the shape parameter

$\lambda > 0$ is the scale parameter

The cumulative distribution function has the form:

$$F(x)=\begin{cases}1-e^{\left(\frac{x}{\lambda}\right)^{-k}} & x \geq 0 \\ 0 & x < 0\end{cases} \tag{5.4}$$

When k=3.4, the Weibull distribution appears similar to the normal distribution. When k=1 the Weibull distribution reduces to the exponential distribution.

The expected value and variance of the random variable R, which is distributed according to the Weibull distribution, are given below:

$$E\langle R\rangle = \lambda\Gamma\left(1+\frac{1}{k}\right) \tag{5.5}$$

and

$$VAR(R) = \lambda^2\left[\Gamma\left(1+\frac{2}{k}\right)-\Gamma^2\left(1+\frac{1}{k}\right)\right] \tag{5.6}$$

where Γ = the gamma function

5.3. Development of Methodology

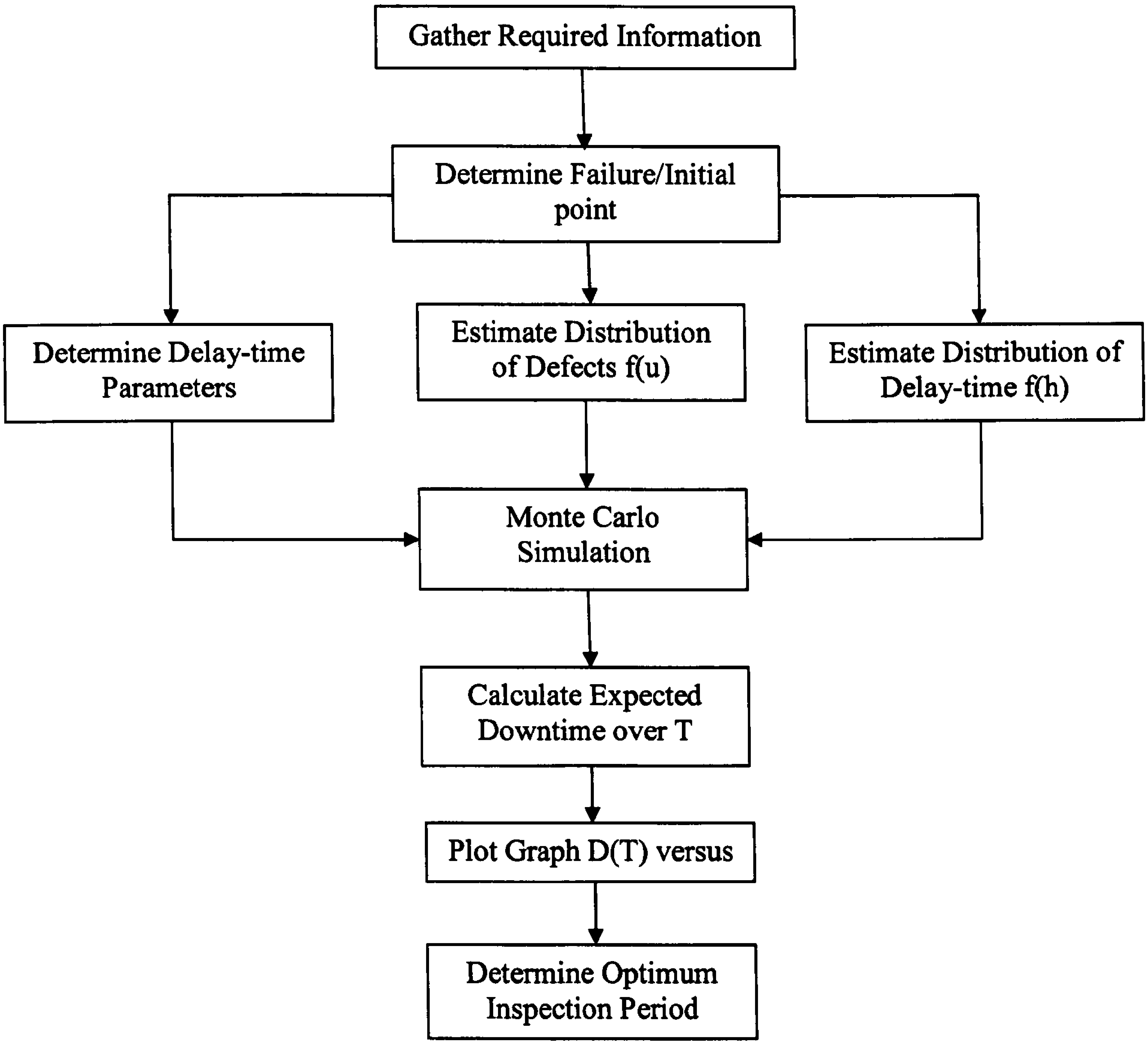

Fig 5.4 Proposed methodology for performing a delay-time analysis via Monte Carlo Simulation

Fig 5.4 shows the methodology that has been developed for the analysis. The diagram illustrates the various steps contained within the analysis and shows the information required before an analysis can be conducted. In the following, some important steps in the methodology will be explained and expanded upon. Where appropriate, examples will be given to aid the description.

5.3.1 Monte Carlo Simulation

Delay-time simulation involves the consideration of a number of defects and associated delay-times within a given time line. It is assumed that in order for a breakdown to occur, there exists a defect, u, which is a pre-curser to failure. Each u value has an associated delay-time, h, that represents a time window, in which, if normal inspection activity occurs the defect could be recognised and the systems transition into a failed state prevented. Simulation of the delay-time involves consideration of the system over a mission time, T_m. T_m should be sufficiently long such that downtime due to breakdown and inspection can be considered negligible. The process involves the estimation of a suitable distribution of defects, f(u) and a suitable distribution of delay-times, f(h). The program can be described in the following steps.

1. Generate a value, U_1, which represents a time of defect, where f(u) is the probability density function of the defect time.
2. Generate an associated delay-time, h_1, which represents the opportunity window in which inspection could arrest a developing failure, where f(h) is the probability density function of the delay-time.
3. Perform a test to see if the defect is found at the time of inspection.
4. Generate the next defect time, u_2, from the point u_1 and an associated delay-time h_2.
5. Repeat step 3.

The process outlined above is repeated until the cumulative value, CU, is greater than the mission time T_m where,

$$CU = u_1 + u_2 + \ldots + u_n \tag{5.7}$$

Fig 5.5 shows the generation of a number of u values within the mission time.

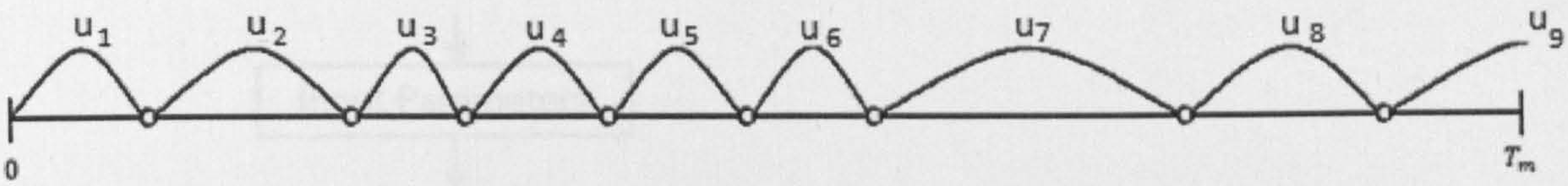

Fig 5.5 Diagram showing the generation of u values within the mission time

Fig 5.6 shows the generation and addition of the related h values.

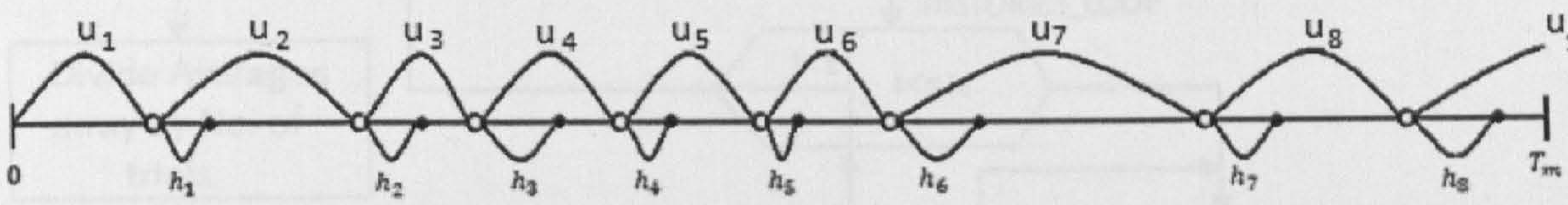

Fig 5.6 Diagram showing the generation and addition of h values within the mission time

Fig 5.7 shows the form of the program used to conduct the simulation in the form of a flowchart.

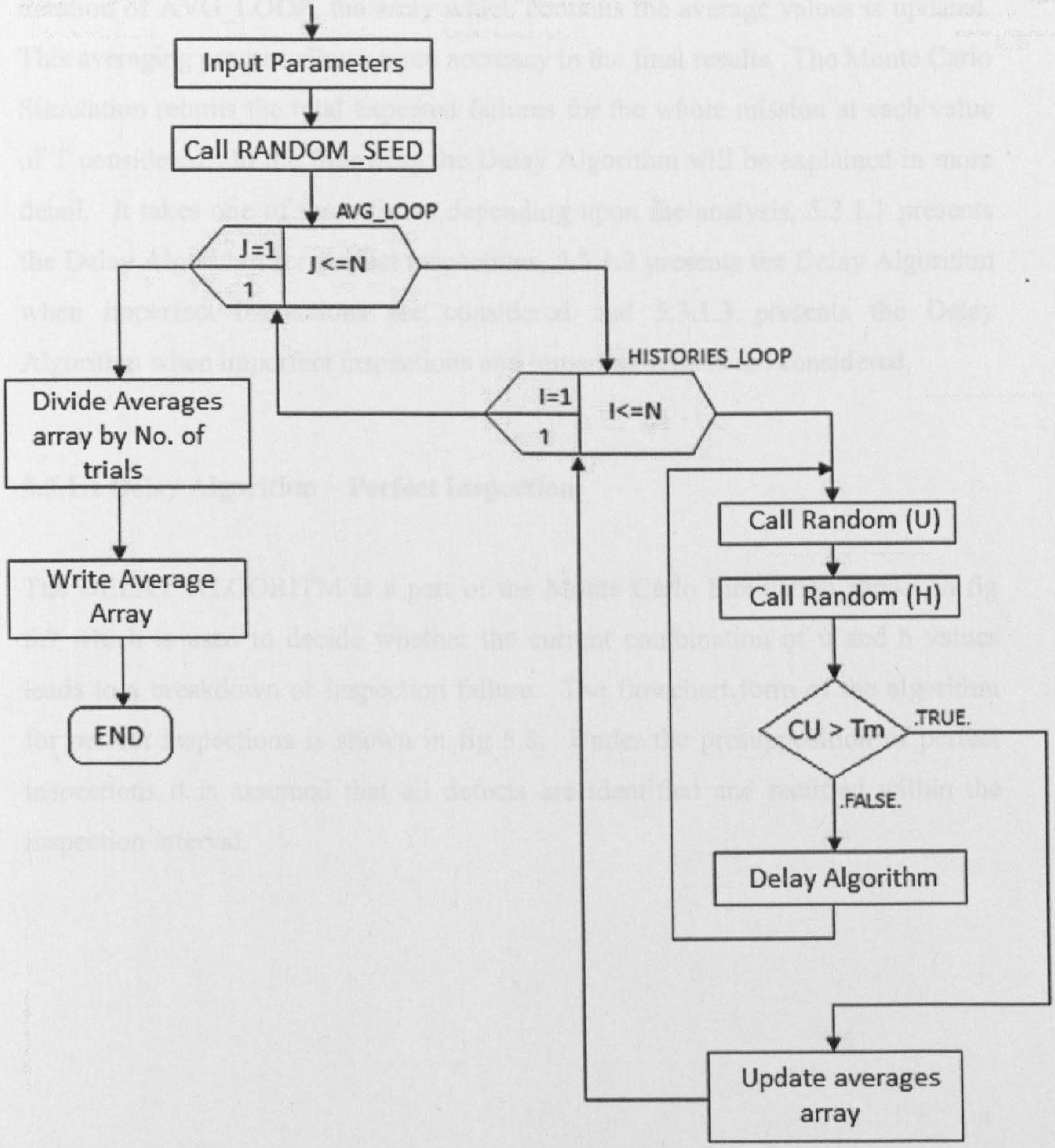

Fig 5.7 Flowchart representation of the Monte Carlo Simulation code

The full FORTRAN code can be found in appendix 2. The code shown in fig 5.7 starts by taking input parameters, defined by the user and uses the Call RANDOM_SEED function to randomise all seed values. The code then progresses in to the counting loop AVG_LOOP and calls a random value of u and h within a counting loop, HISTORIES_LOOP. DELAY ALGORITM then performs the test which decides if this particular combination of u and h leads to a failure or a breakdown. AVG_LOOP is a second counting loop which repeats the process a set number of times, N, for a given value of T. At the end of each iteration of AVG_LOOP, the array which contains the average values is updated. This averaging process allows more accuracy in the final results. The Monte Carlo Simulation returns the total expected failures for the whole mission at each value of T considered. In the following the Delay Algorithm will be explained in more detail. It takes one of three forms depending upon the analysis, 5.3.1.1 presents the Delay Algorithm for perfect inspections, 5.3.1.2 presents the Delay Algorithm when imperfect inspections are considered and 5.3.1.3 presents the Delay Algorithm when imperfect inspections and imperfect repairs are considered.

5.3.1.1 Delay Algorithm – Perfect Inspection

The DELAY ALGORITM is a part of the Monte Carlo Simulation shown in fig 5.7 which is used to decide whether the current combination of u and h values leads to a breakdown or inspection failure. The flowchart form of the algorithm for perfect inspections is shown in fig 5.8. Under the presupposition of perfect inspections it is assumed that all defects are identified and rectified within the inspection interval.

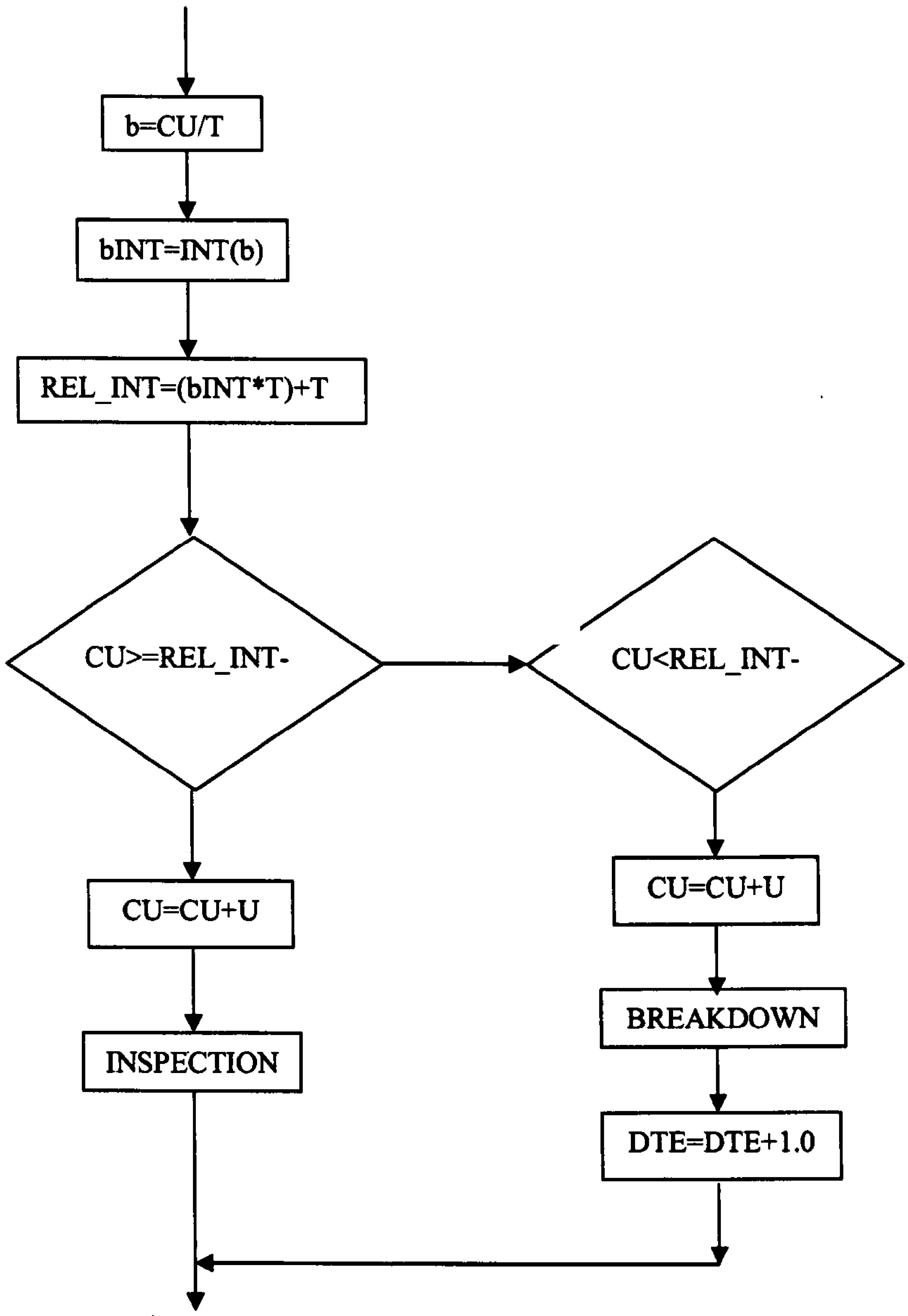

Fig 5.8 Flowchart representing the Delay Algorithm for perfect inspections

To fully explain how the algorithm works a simple example is considered. The Monte Carlo Simulation is run for a single trial, when T=2 and the random values of u and h are generated as 13 and 0.7 respectively. The delay-time algorithm works using the cumulative value of u, however this is the first iteration of the code and thus the cumulative value CU and u are equal. The value CU is divided by T to examine how many inspections can occur, giving the exact value b. In this example when CU=13 hours and T=2 hours, b=6.5 inspections. This is shown in fig 5.9.

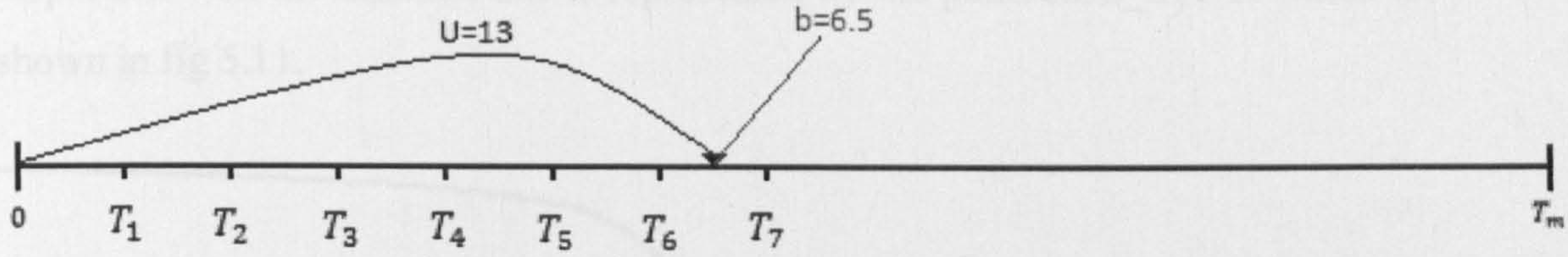

Fig 5.9 Figure showing the point b where b=CU/T

In order to be able to perform a test to see if an inspection or breakdown occurs whole values of T are required. The algorithm uses an intrinsic FORTRAN function INT(b) to achieve this. If b is of type real and | b | >= 1, INT(b) is the integer whose magnitude is the largest integer that does not exceed the magnitude of b and whose sign is the same as the sign of b. When the example is considered, b=6.5, INT(b) returns the value 6. In the flowchart shown in fig 5.8 bINT=6. It is now known that the defect, u, lies between the sixth and seventh inspection interval. In DTA it is always the time at the upper bound of the relevant interval which is of interest. From the lower bound of the interval the upper bound is simple to calculate. Fig 5.10 shows the interval of interest, bINT and REL_INT on the timeline.

$$\begin{aligned} REL\_INT &= (bINT \times T) + T \\ &= (6 \times 2) + 2 \\ &= 14 \end{aligned}$$

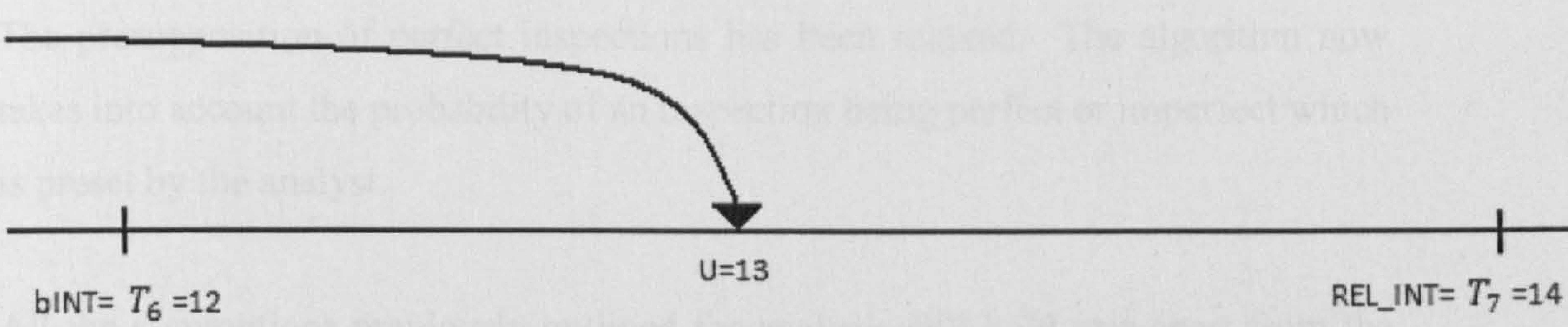

Fig 5.10 Diagram showing bINT and REL_INT on the timeline

The next part of the algorithm is where the test is performed to see if the delay-time is sufficient such that the defect will be recognised and repaired at the next inspection. On the timeline this is represented by the point REL_INT-H which is shown in fig 5.11.

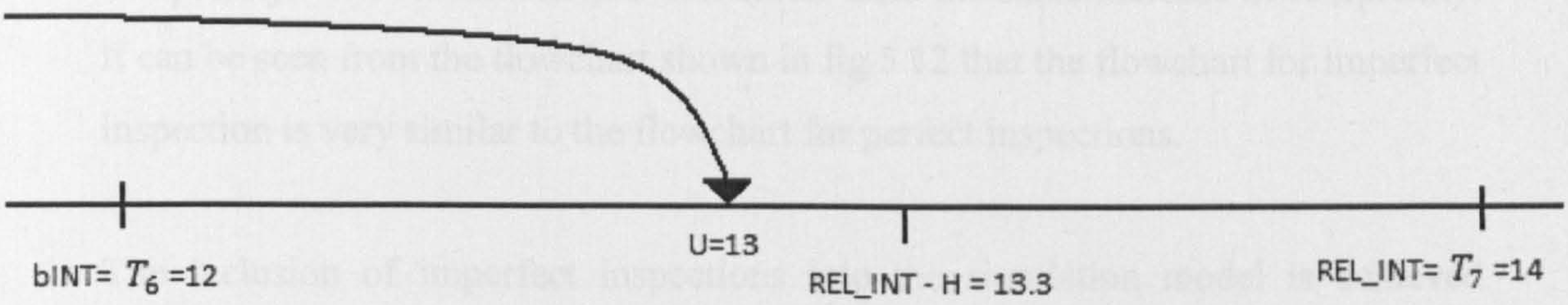

Fig 5.11 Diagram showing the point REL_INT-H

The algorithm performs the calculation,

$$REL\_INT - H = 14 - 0.7 = 13.3$$

The test is performed to see if $CU \geq (REL\_INT - H)$ or $CU < (REL\_INT - H)$, if the first condition is found to be true an inspection occurs, if the latter is found to be true a breakdown occurs. In the case of breakdown the counter DTE, is increased by one. In both cases of inspection and failure the present u value is added to the cumulative value CU.

5.3.1.2 Delay Algorithm – Imperfect Inspection

The flowchart form of the algorithm for imperfect inspections is shown in fig 5.12. The presupposition of perfect inspections has been relaxed. The algorithm now takes into account the probability of an inspection being perfect or imperfect which is preset by the analyst.

All the assumptions previously outlined for analysis still hold true apart from the assumption of perfect inspection. In the case of imperfect inspection it is assumed that at the point of inspection there is a probability, r, that a defect present will be identified. Conversely there is a probability, $1 - r$, that a defect will go unnoticed at inspection and will continue to develop into a full breakdown. Christer (1999) demonstrates how the analytic model can be extended to include imperfect inspections. It should be noted that imperfect inspection when using analytical methods is achieved at the cost of a significant increase in mathematical complexity. The simulation does not suffer from the same increase in complexity. It can be seen from the flowchart shown in fig 5.12 that the flowchart for imperfect inspection is very similar to the flowchart for perfect inspections.

The inclusion of imperfect inspections into the simulation model is achieved through the introduction of a discrete distribution which represents the probability of perfect and imperfect inspections. The distribution is made of two distinct intervals, $(0,1-r)$ and $(1-r,1)$. The random number, RI, is called where $RI = U{\sim}(0,1]$ and a test is performed to examine in which interval RI falls. This test decides whether a defect is recognised and repaired at inspection or unnoticed and left to develop into a breakdown failure. For the analysis in the following case studies inspections are considered imperfect 10% of the time. Woods (1984) suggest that in emergency situations this incorrect inspection rate could be as high as 60 %. The value of 10% in light of this can be considered appropriate as the inspections do not take place under emergency conditions.

5.3.1.3 Delay Algorithm – Imperfect Inspections, Imperfect Repair

Imperfect repair involves the consideration of delay-time analysis with a non-homogeneous defect arrival rate, k_f. The assumption that k_f is constant is a reasonable assumption for most systems that have been running for a sufficiently long period to be considered mature. Imperfect repair, first considered by Brown & Proschan (1983), can be closely linked to models considering 'minimal repair at failure' (Barlow & Proschan 1965), (Blumenthal et al. 1976). Further study and extension of the Brown & Proschan model was conducted by Whitaker & Samaniego (1989). Baker & Wang (1993) consider delay-time analysis where the assumption of constant k_f is relaxed. The model considers the effect of component age on the arrival rate of defects and the consequence of inspection activity and its possible hazardous or beneficial effect on the lifetime of a component.

The model developed in this work considers the effect of minimal repair after an inspection action. It is still assumed that in the case of a breakdown failure the repair of components is perfect and the system is put back online in a 'good as new state'. After a breakdown repair the system is put back online with the original steady-state arrival rate of defects, k_f. To examine the effect of non steady-state conditions it is assumed that when a defect is identified at inspection and the defect subsequently repaired, this repair action is non-perfect. This non-perfect repair action has the effect of increasing the arrival rate of defects by 20%. The flowchart form of the algorithm for imperfect inspections with imperfect repair is shown in fig 5.13.

5.3.2 Calculate Expected Downtime over T

The Monte Carlo Simulation outlined provides the total expected number of failures over a given mission time. Equation (5.5) for downtime per unit time requires the expected value of failures over T. In order to achieve this, results given by the simulation have to be divided by N. N is equal to the total number of inspections, T, possible within the given mission time, T_m, i.e. $N = \frac{T_m}{T}$.

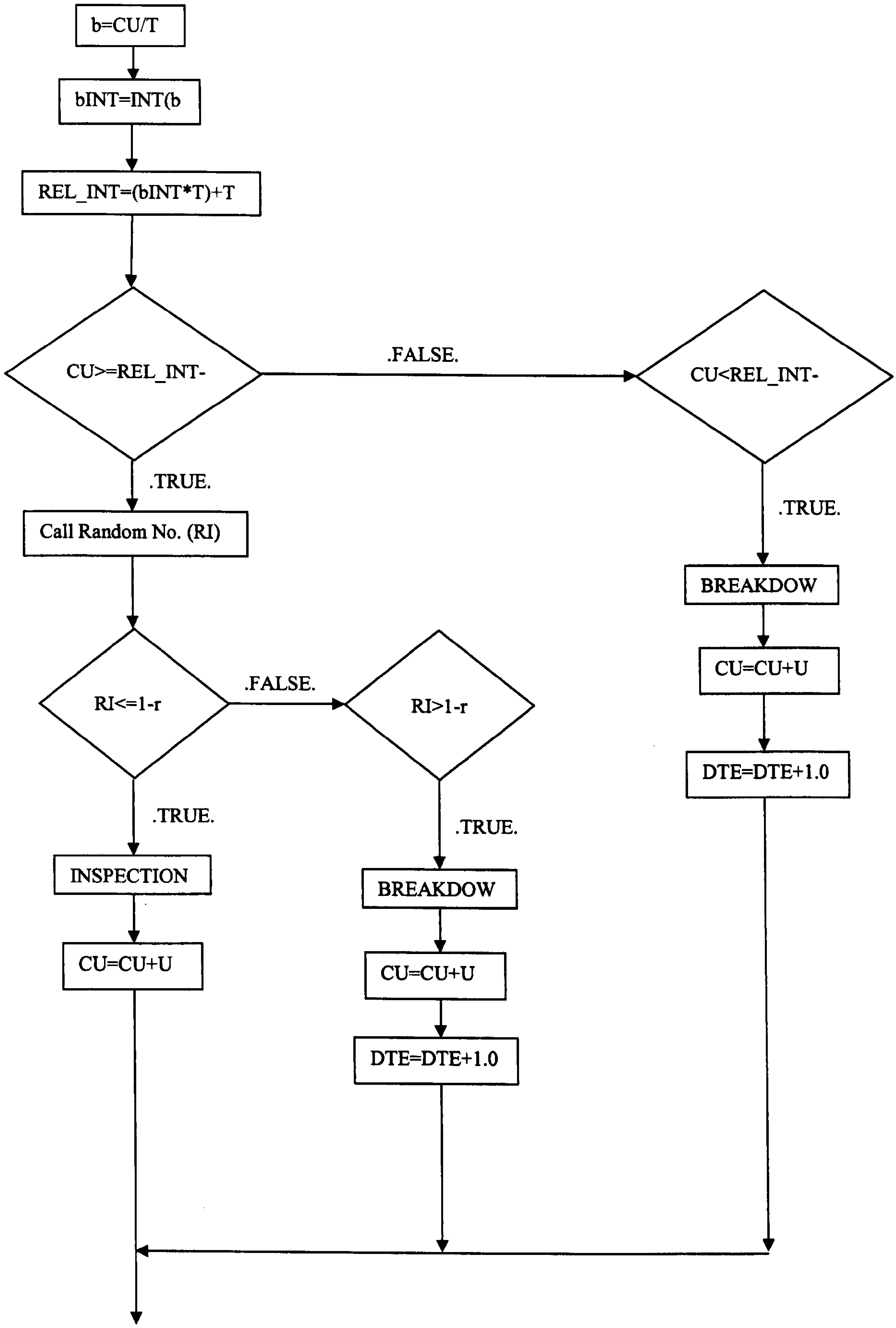

Fig 5.12 Flowchart representing the Delay Algorithm for imperfect inspections

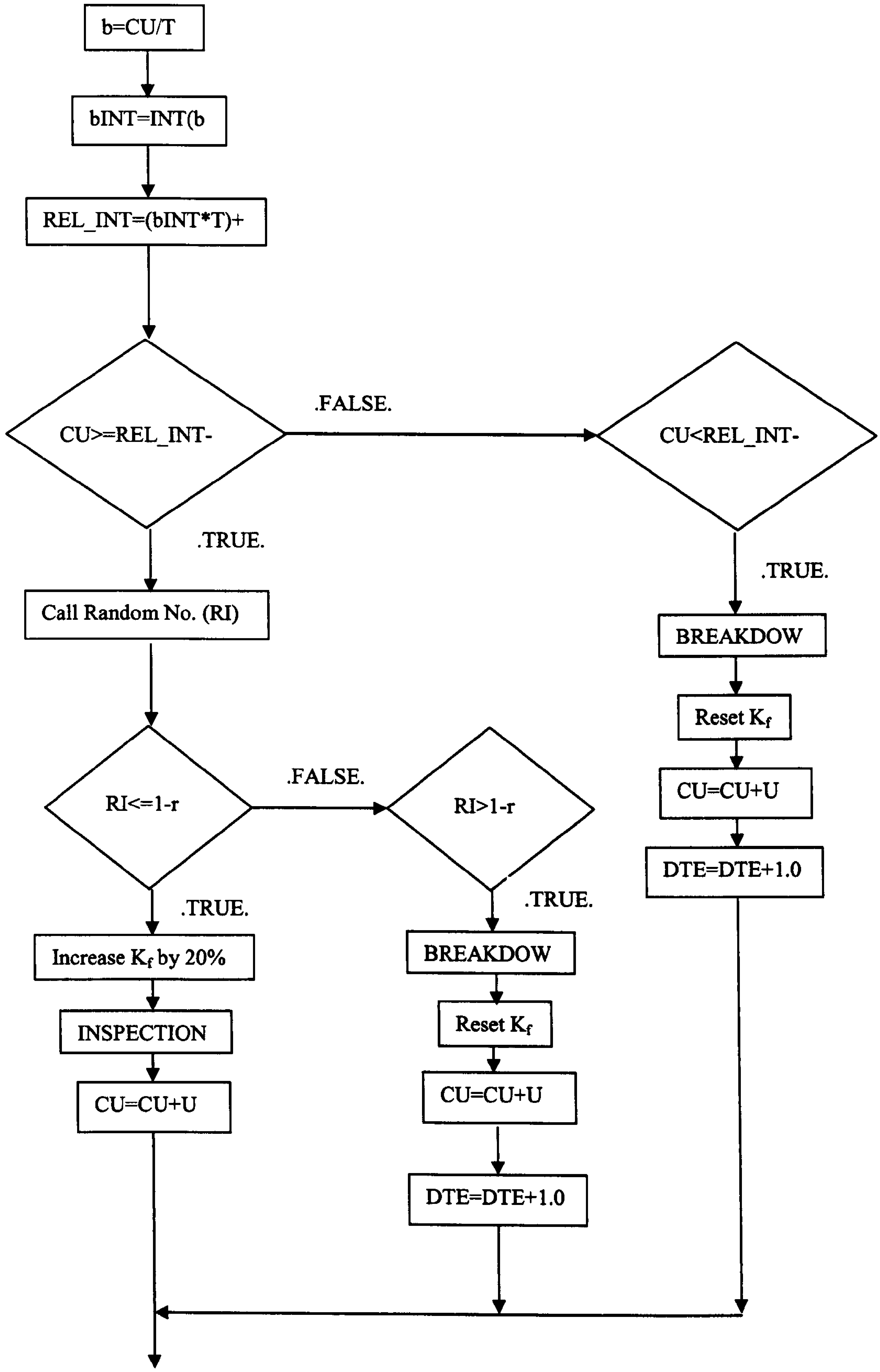

Fig 5.13 Flowchart representing the Delay Algorithm for imperfect inspections and imperfect inspection repairs

5.4. Case Studies

With the intent of demonstrating the method for DTA via simulation two case studies are presented. In the first the data for the case study was taken from an existing journal paper (Pillay et al. 2001). In the second case study a new model is presented based on a centrifugal pump, where repair data is based on OREDA data (OREDA 2002) and expert judgement.

5.4.1 Fishing Vessel Case Study

The delay-time model is based on the operation of a main hydraulic winch operating system on board a fishing vessel. The vessel has length overall of 60m and gross tonnage of 1266. Fig 5.14 shows a schematic of the main hydraulic piping system.

The data for the analysis, is taken directly from the existing journal paper (Pillay et al. 2001) and is shown in table 5.1.

Table 5.1 Table showing the input parameters for the analysis

Inspection Downtime	**ds**	0.25 hrs
Breakdown Downtime	**db**	108 hrs
Arrival rate of defects	**kf**	0.0223 hrs^{-1}

The downtime for breakdown repair takes into account any delays caused while waiting for spares to be sent to the vessel.

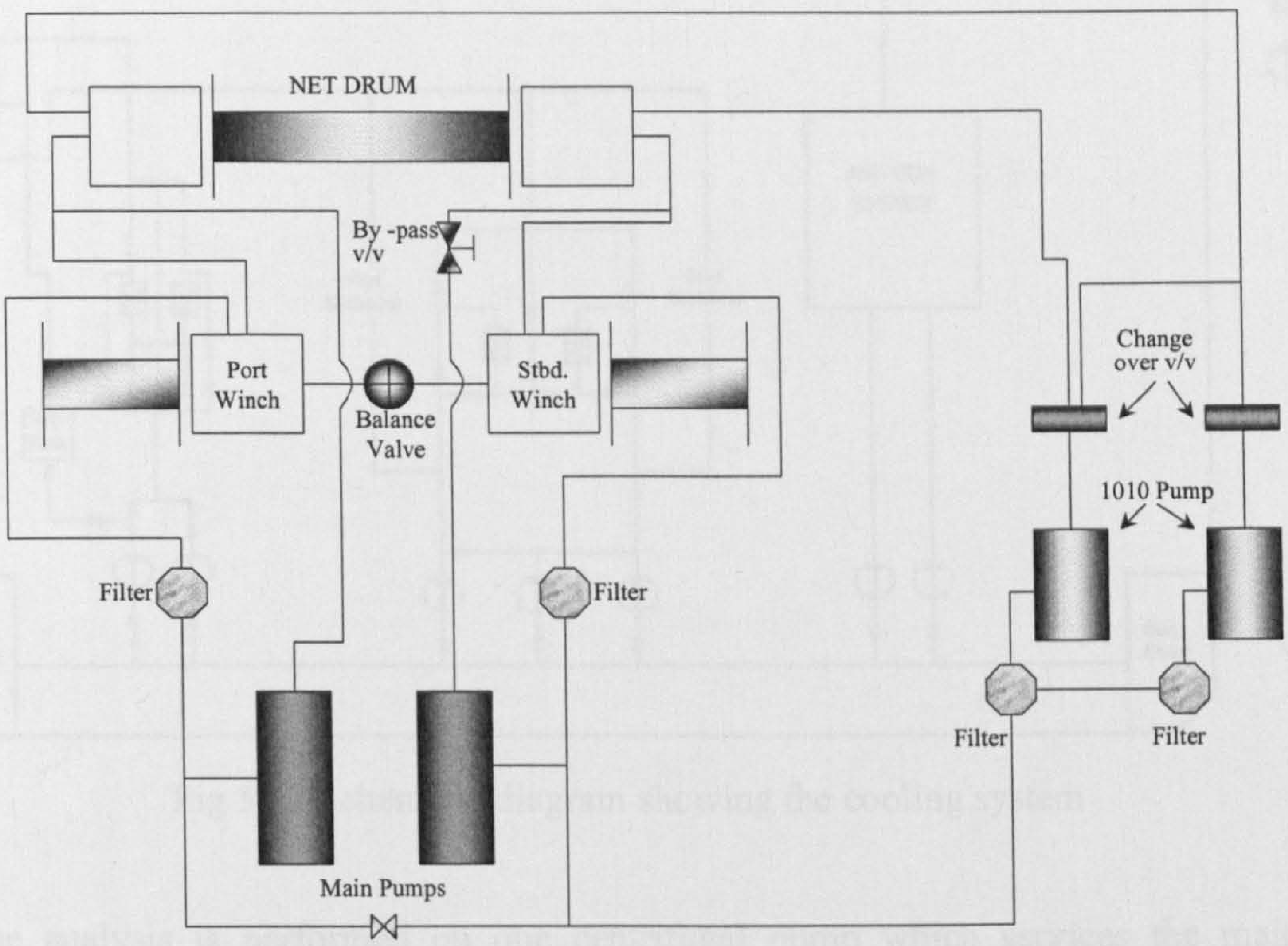

Fig 5.14 Hydraulic winch operating system

5.4.2 Cooling System – Centrifugal Pump

The system is taken from the MV Hamnavoe, a Ro-Ro passenger ferry on which the researcher served time during a cadetship and is the same system presented in the previous chapters. The full system is shown in fig 5.15.

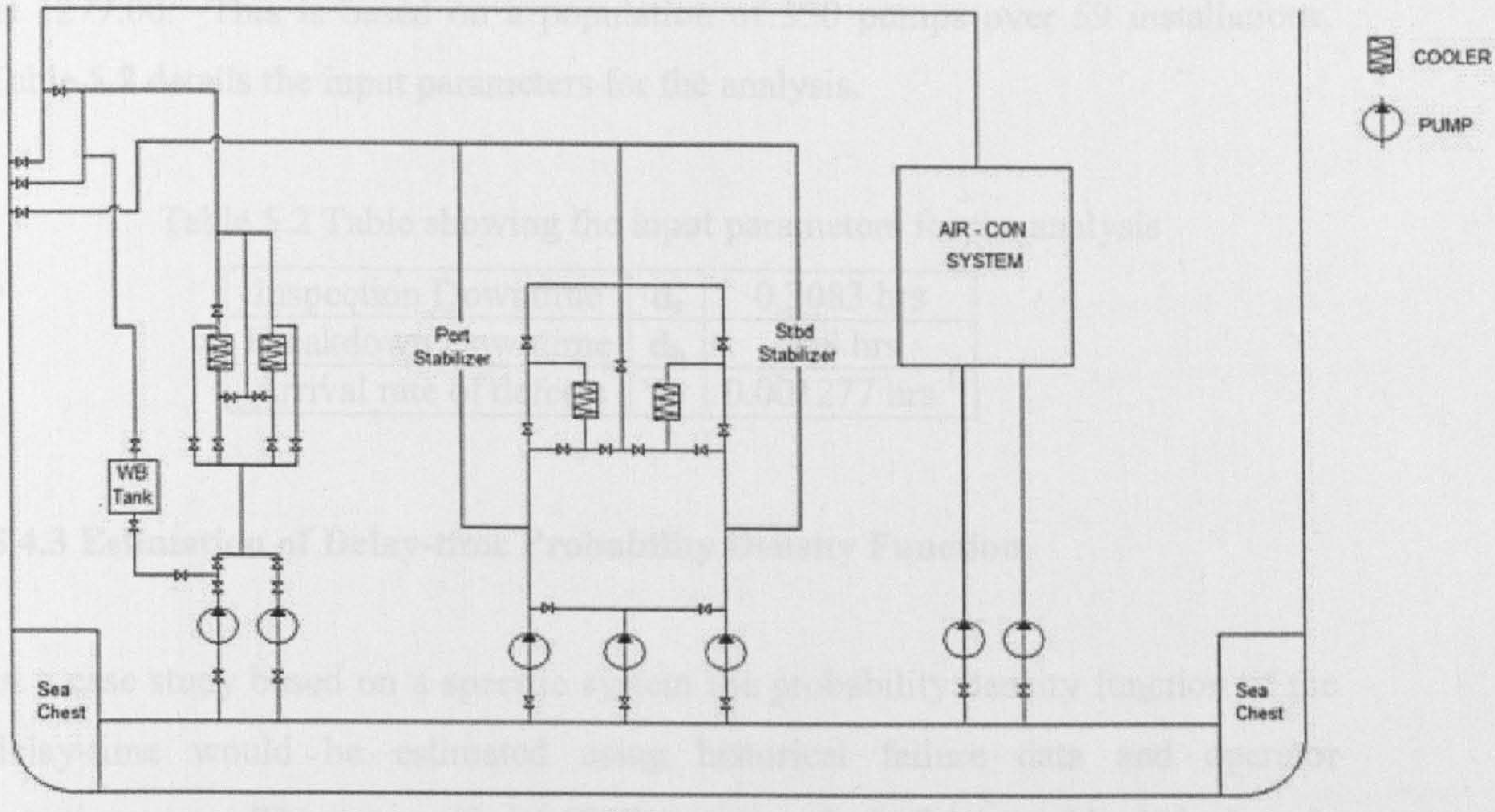

Fig 5.15 Schematic diagram showing the cooling system

The analysis is performed on one centrifugal pump which services the main cooling system. To carry out the analysis a number of input variables were required. The downtime due to breakdown was taken from OREDA 2002 and set equal to 168 hours or 7 days which allows for any logistical delay in spare part procurement. For the downtime due to inspection the expert opinion of Mr Ramin Riahi was used. A detailed description of Mr Riahi's industrial experience and academic qualifications is listed in appendix 4. Daily inspection of the centrifugal pump involves visual inspection of suction and discharge pressure, audio inspection for any abnormal noise and electrical inspection of the current being drawn by the electric motor. Mr Riahi suggested that this daily inspection on average would take 10-15 minutes. In light of this the downtime due to inspection was taken as 12.5 minutes or 0.2083 hours. When considering the arrival rate of defects it is argued that the failure rate of a system and the arrival rate of defects are intrinsically linked. In order for this to be true the component or system would have to be operated under a breakdown maintenance policy. OREDA data is not presented for systems operating under a breakdown maintenance regime. However for the purpose of the analysis it is assumed that the OREDA failure data for a centrifugal pump and the arrival rate of defects are equivalent. OREDA gives the failure rate per 10^6 hours for a centrifugal pump, in all modes of failure, as 1277.00. This is based on a population of 350 pumps over 59 installations. Table 5.2 details the input parameters for the analysis.

Table 5.2 Table showing the input parameters for the analysis

Inspection Downtime	d_s	0.2083 hrs
Breakdown Downtime	d_b	168 hrs
Arrival rate of defects	k_f	0.001277 hrs^{-1}

5.4.3 Estimation of Delay-time Probability Density Function

In a case study based on a specific system the probability density function of the delay-time would be estimated using historical failure data and operator questionnaires. This process in itself takes a great deal of time and logistical work. The purpose of this work was to demonstrate the simulation method of DTA, therefore the analysis was performed using a number of different Weibull distributions for the delay-time and may not represent accurately the true distributions of the delay-times for the real life systems. Table 5.3 shows the shape and scale parameters used for the different analyses. A number of shape and scale parameters are used to give an idea of their effect on the analysis.

Table 5.3 Table showing the shape and scale parameters of the Weibull distributions used in the analysis

k	Λ
10	5
8	6
3	10
2	20

5.4.4 Initial Modelling Assumptions

When performing the analysis for the case study the following modelling assumptions were made.

- Inspections take place at regular intervals of T hours and each inspection is identical.

- The arrival rate of defects is constant and distributed according to an exponential probability density function.
- Failures are repaired instantaneously and the system is returned to a 'good as new' state.
- The mission time is set to 10 years and is sufficiently large that downtime due to breakdown and inspection during the analysis can be considered negligible.
- Inspections are perfect in that any defect present will be identified and the failure arrested within the inspection period.

5.5. Case study Results

5.5.1 Fishing Vessel – Perfect Inspections

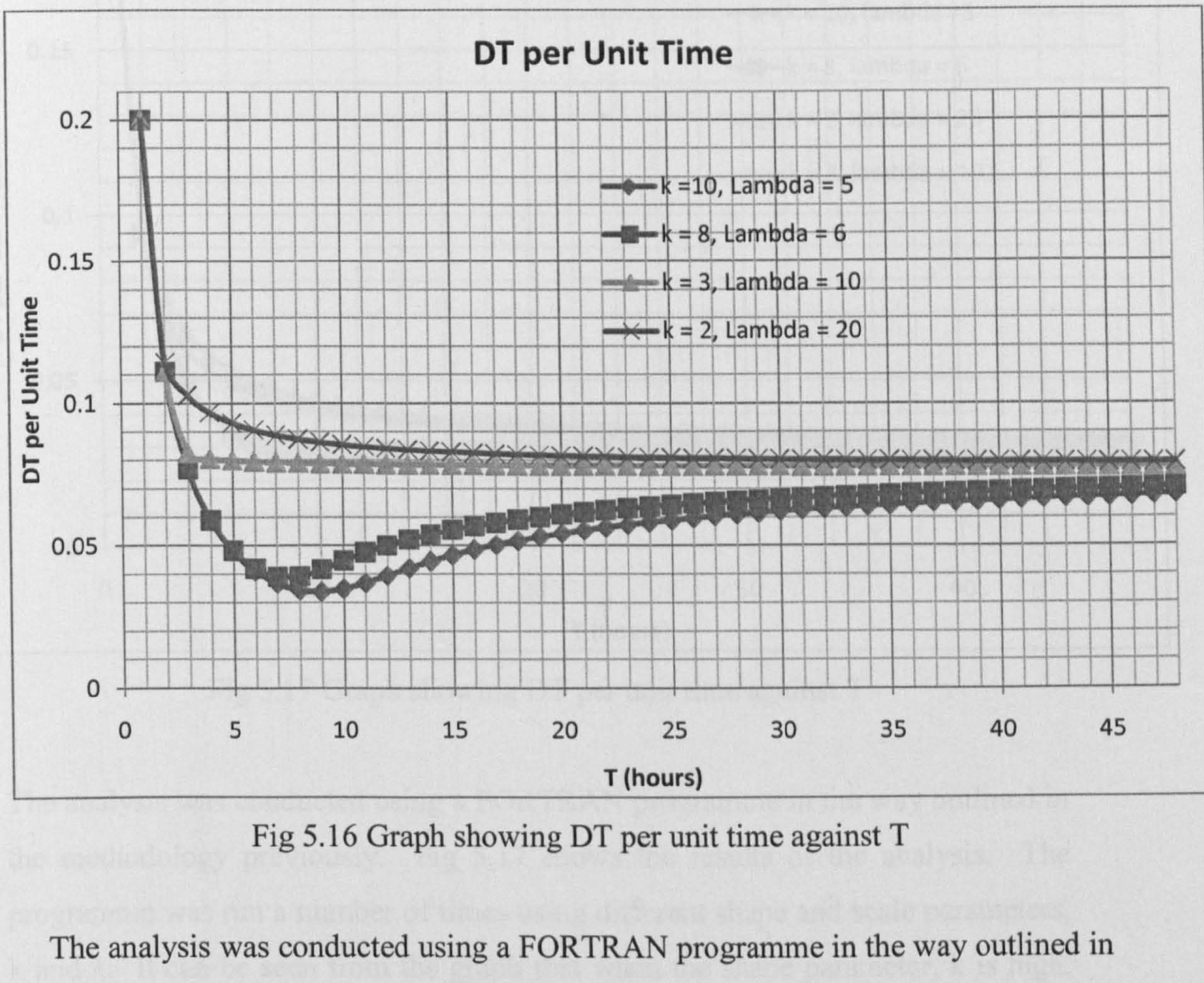

Fig 5.16 Graph showing DT per unit time against T

The analysis was conducted using a FORTRAN programme in the way outlined in the methodology previously. Fig 5.16 shows the results of the analysis. The programme was run a number of times using different shape and scale parameters, k and λ. It can be seen from the graph that when the shape parameter, k is high, then the analysis produced the best results. When k=10 and λ =5, DT per unit time was minimised at T=9 hours to give a DT per unit time of 0.034 hours. When k=8 and λ=6, DT per unit time was minimised at T=7 hours to give a DT per unit time of 0.04 hours. When k=3 and λ=10, no definitive minimum point was established. Also when k=2 and λ=20, no definitive minimum point was established. If the results where k=10 and k=8 are considered then an optimum inspection of 9 and 7 hours would be recommended respectively.

5.5.2 Cooling System - Centrifugal Pump – Perfect Inspections

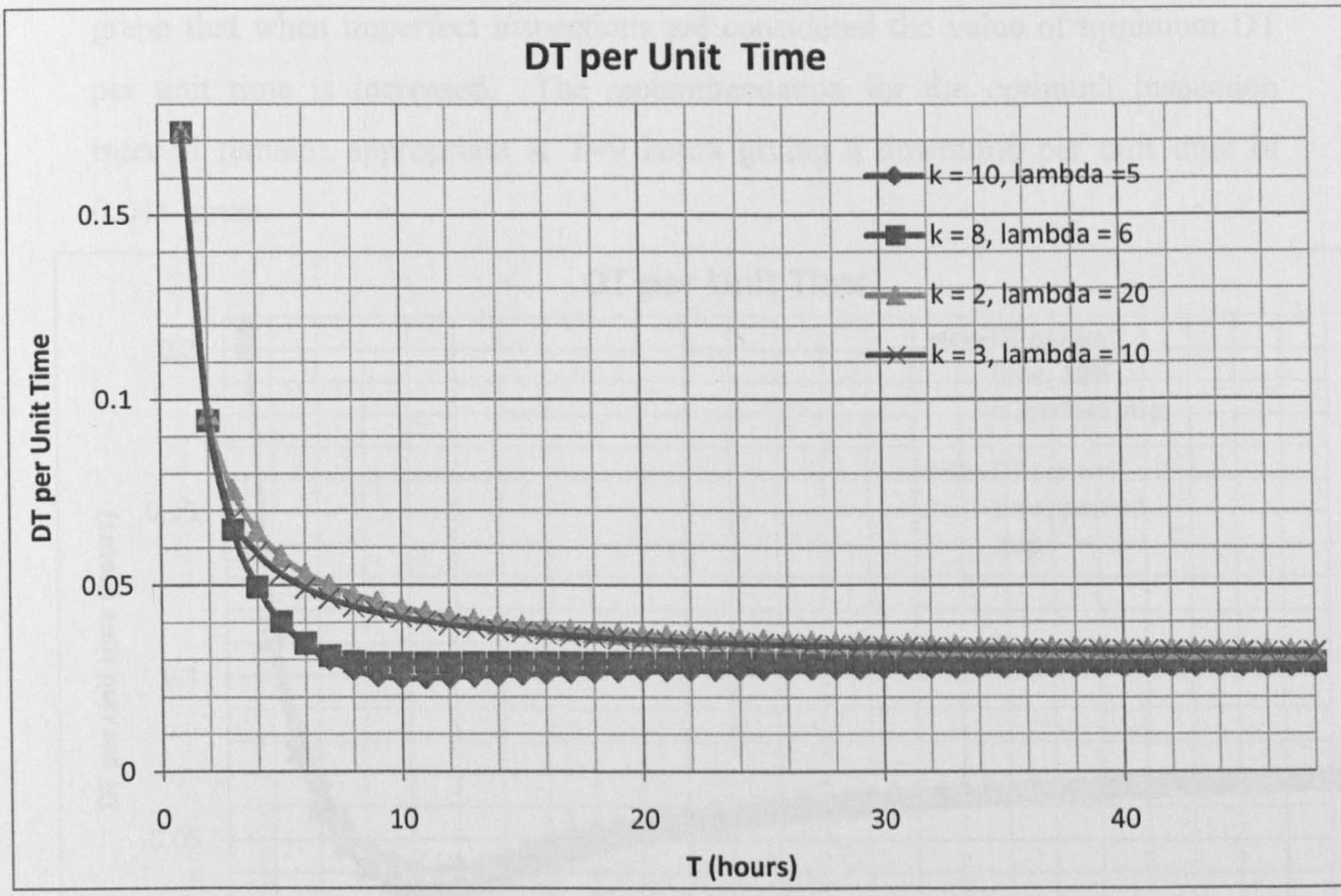

Fig 5.17 Graph showing DT per unit time against T

The analysis was conducted using a FORTRAN programme in the way outlined in the methodology previously. Fig 5.17 shows the results of the analysis. The programme was run a number of times using different shape and scale parameters, k and λ. It can be seen from the graph that when the shape parameter, k is high, then the analysis produced the best results. When k=10 and λ =5, DT per unit time was minimised at T=11 hours to give a DT per unit time of 0.025 hours. For all other values of K and λ considered, no definitive minimum point was established. From the results where k=10 and λ=5, an optimum inspection of 11 hours would be recommended.

5.5.3 Fishing Vessel – Imperfect Inspections

The analysis was conducted using a FORTRAN programme in the way outlined in the methodology previously. Fig 5.18 shows the results of the analysis. The programme was run using shape and scale parameters, k=10 and λ=5, which produced the most definitive result for perfect inspection. It can be seen from the graph that when imperfect inspections are considered the value of minimum DT per unit time is increased. The recommendation for the optimum inspection interval remains appropriate at T=9 hours giving a downtime per unit time of 0.041 hours.

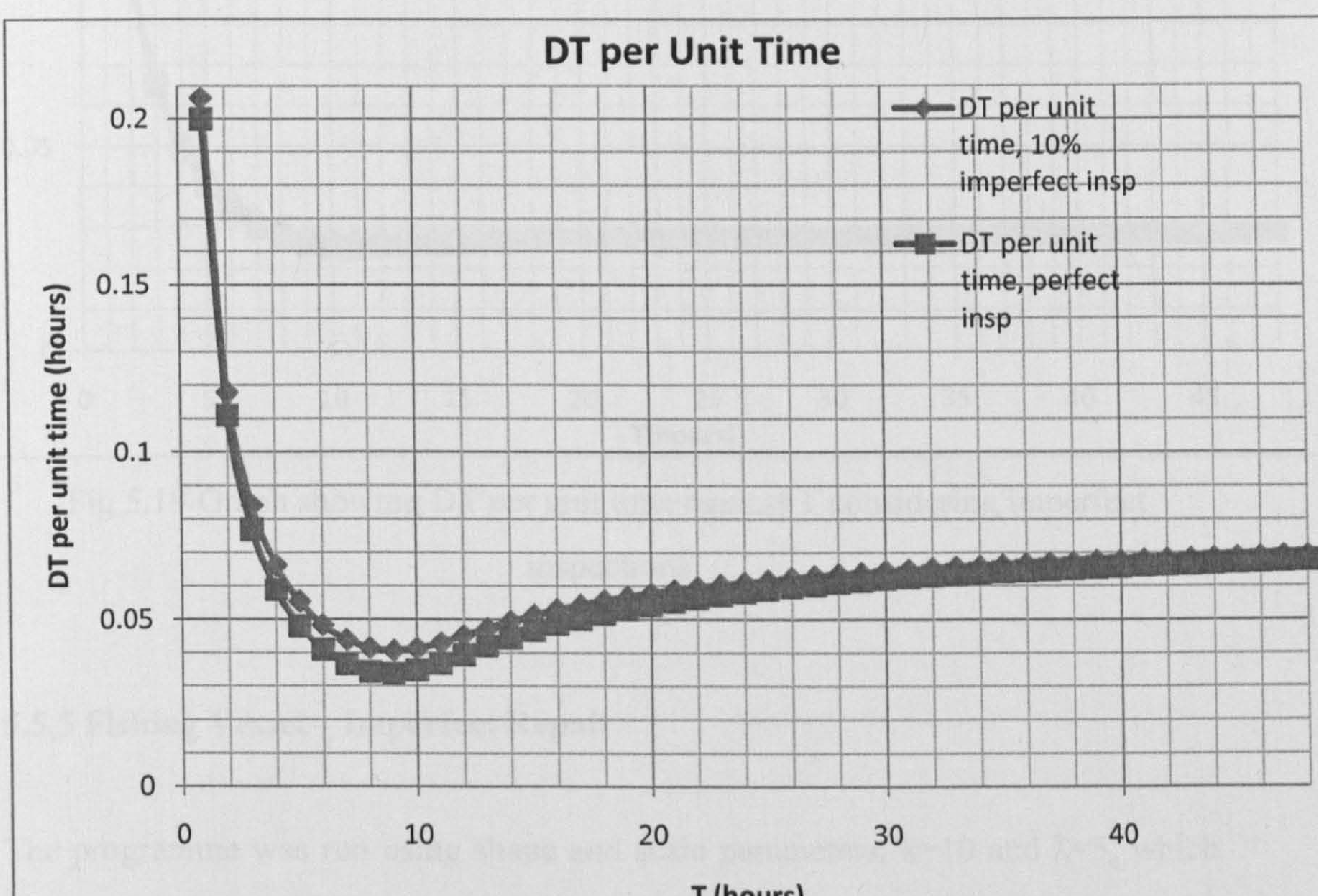

Fig 5.18 Graph showing DT per unit time against T considering imperfect inspections

5.5.4 Cooling System – Centrifugal Pump – Imperfect Inspections

The programme was run using shape and scale parameters, k=10 and λ=5, which produced the most definitive result for perfect inspection. Fig 5.19 shows the results of the analysis. It can be seen from the graph that when imperfect inspections are considered as in the first case study the value of minimum DT per unit time is increased. The recommendation for the optimum inspection interval remains appropriate at T=11 hours giving a downtime per unit time of 0.027 hours.

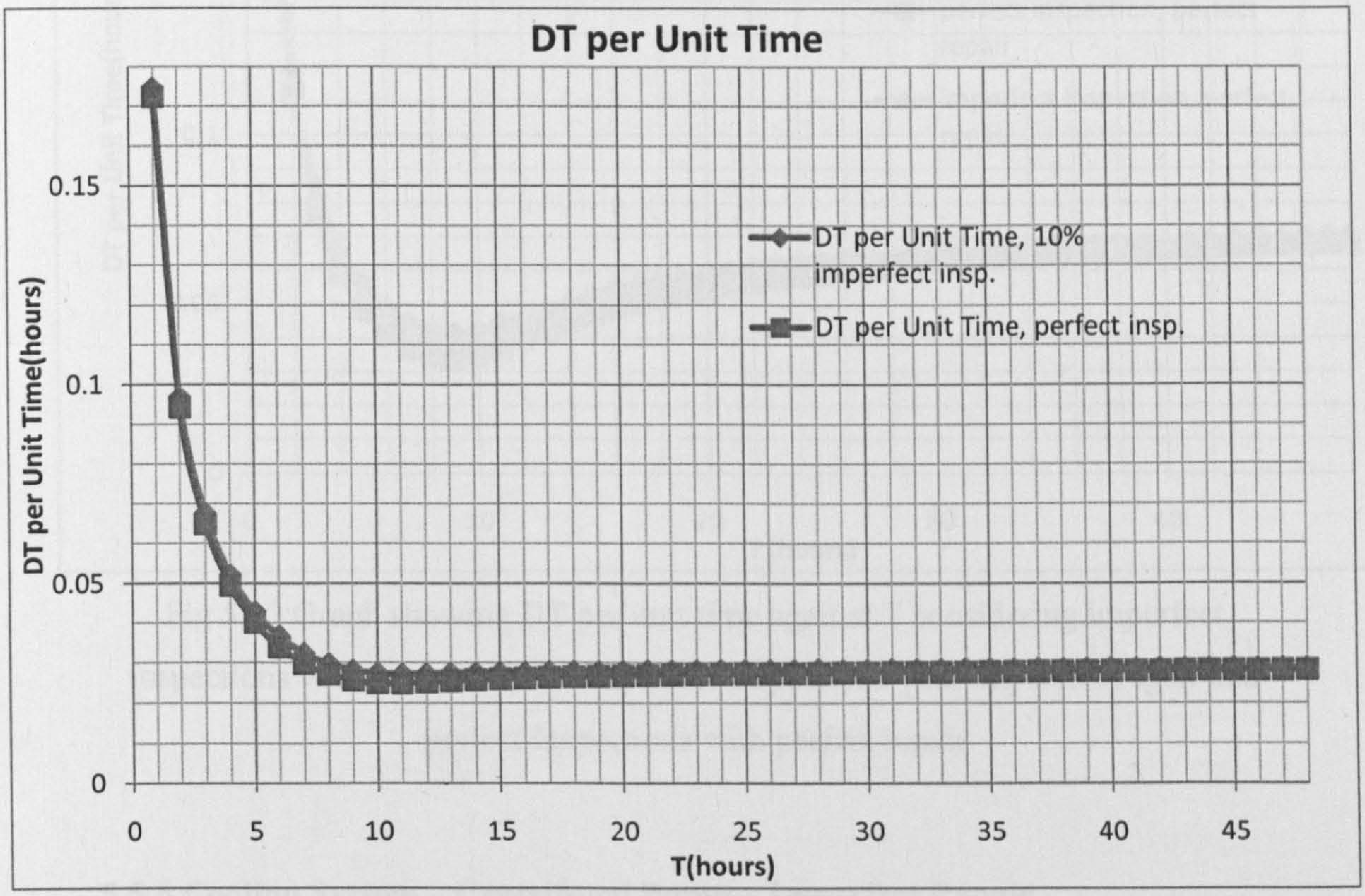

Fig 5.19 Graph showing DT per unit time against T considering imperfect inspections

5.5.5 Fishing Vessel – Imperfect Repair

The programme was run using shape and scale parameters, k=10 and λ=5, which produced the most definitive result for perfect inspection. Fig 5.20 shows the results of the analysis. It can be seen from the graph that the consideration of imperfect repair has a similar effect on the downtime per unit time achieved as imperfect inspection did previously. The recommendation for the optimum inspection interval remains appropriate at T=9 hours giving a downtime per unit time of 0.042 hours.

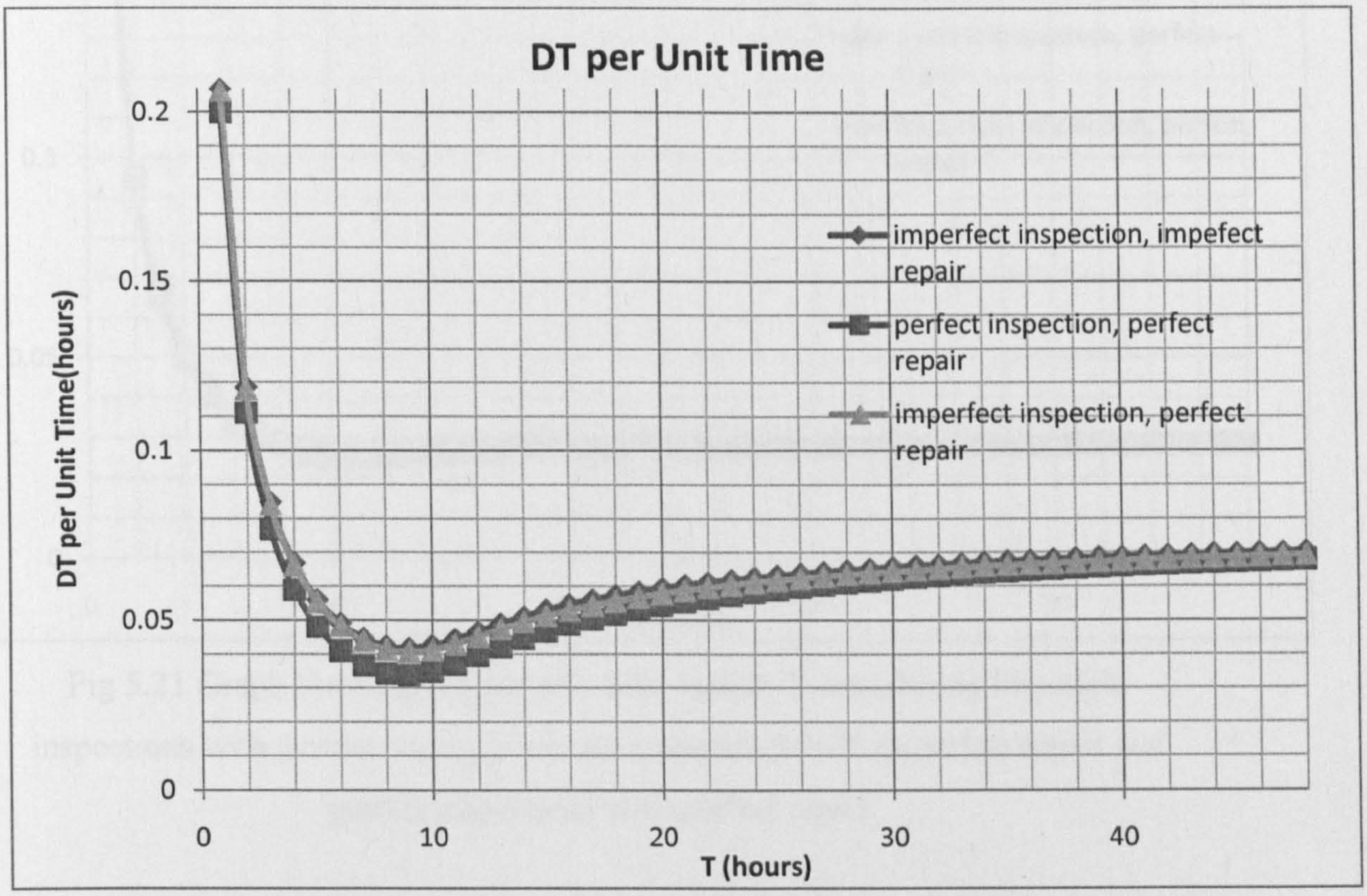

Fig 5.20 Graph showing DT per unit time against T considering imperfect inspections with perfect repair, imperfect inspections with imperfect repair and perfect inspections with perfect repair

5.5.6 Cooling System – Centrifugal Pump – Imperfect Repair

The programme was run using shape and scale parameters, k=10 and λ=5, which produced the most definitive result for perfect inspection. Fig 5.21 shows the results of the analysis. It can be seen from the graph that the consideration of imperfect repair has a similar effect on the downtime per unit time achieved as imperfect inspection did previously. The recommendation for the optimum inspection interval remains appropriate at T=11 hours giving a downtime per unit time of 0.028 hours.

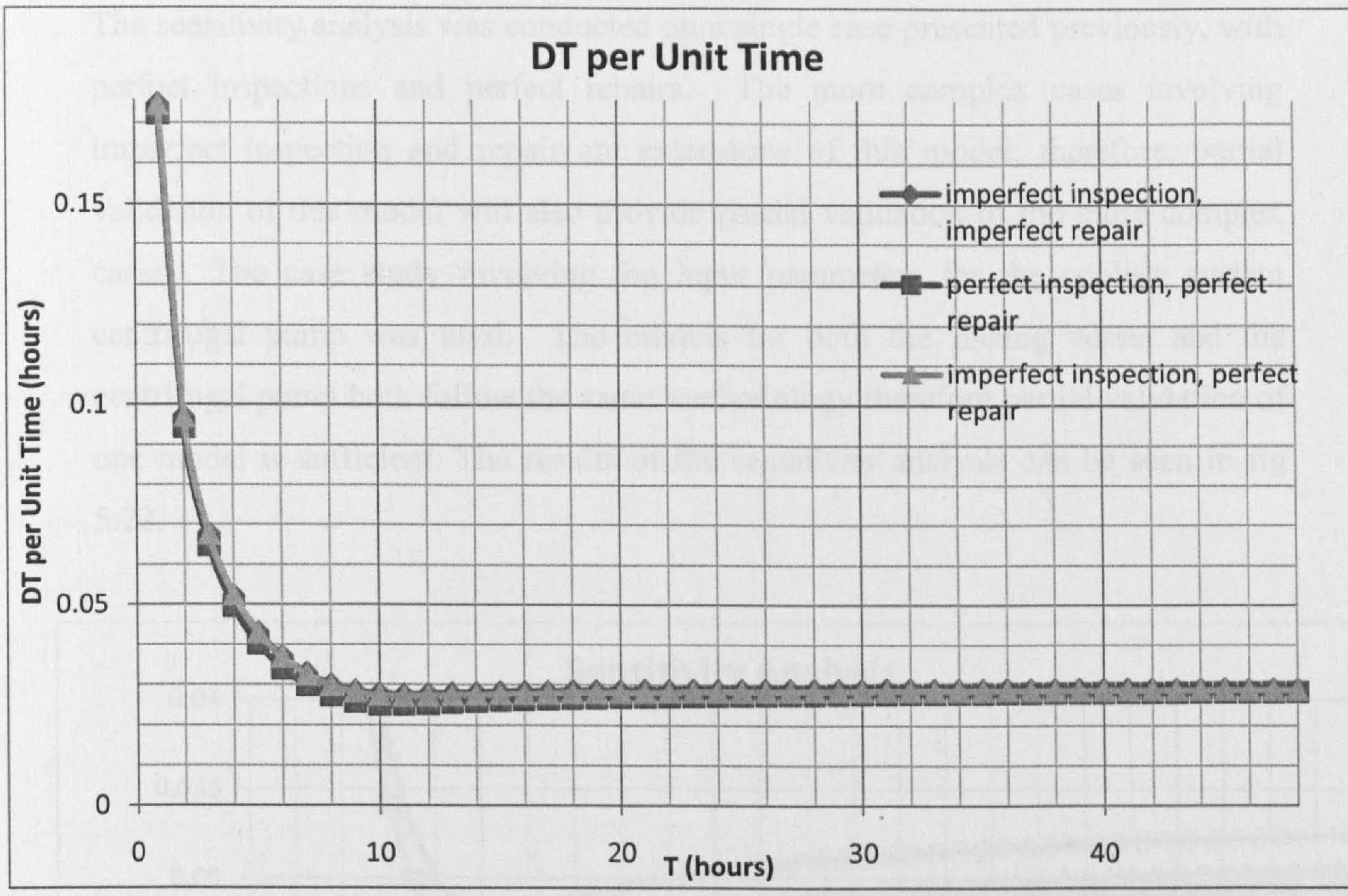

Fig 5.21 Graph showing DT per unit time against T considering imperfect inspections with perfect repair, imperfect inspections with imperfect repair and perfect inspections with perfect repair

5.6. Sensitivity Analysis

A sensitivity analysis provides a way of partially validating a model. For this model three axioms are detailed and must be satisfied before the sensitivity analysis can be considered complete.

1. An increase in the arrival rate of defects should result in a proportional increase in the DT per unit time.
2. Further increase in the arrival rate of defects should reflect a consistent increase in the DT per unit time.
3. An increase in more than one input parameter should result in a larger increase in DT per unit time than that caused by an increase in a single input parameter.

The sensitivity analysis was conducted on a single case presented previously, with perfect inspections and perfect repairs. The more complex cases involving imperfect inspection and repair are extensions of this model; therefore, partial validation of this model will also provide partial validation of the more complex cases. The case study involving the input parameters for the cooling system centrifugal pump was used. The models for both the fishing vessel and the centrifugal pump both follow the same methodology therefore partial validation of one model is sufficient. The results of the sensitivity analysis can be seen in fig 5.22.

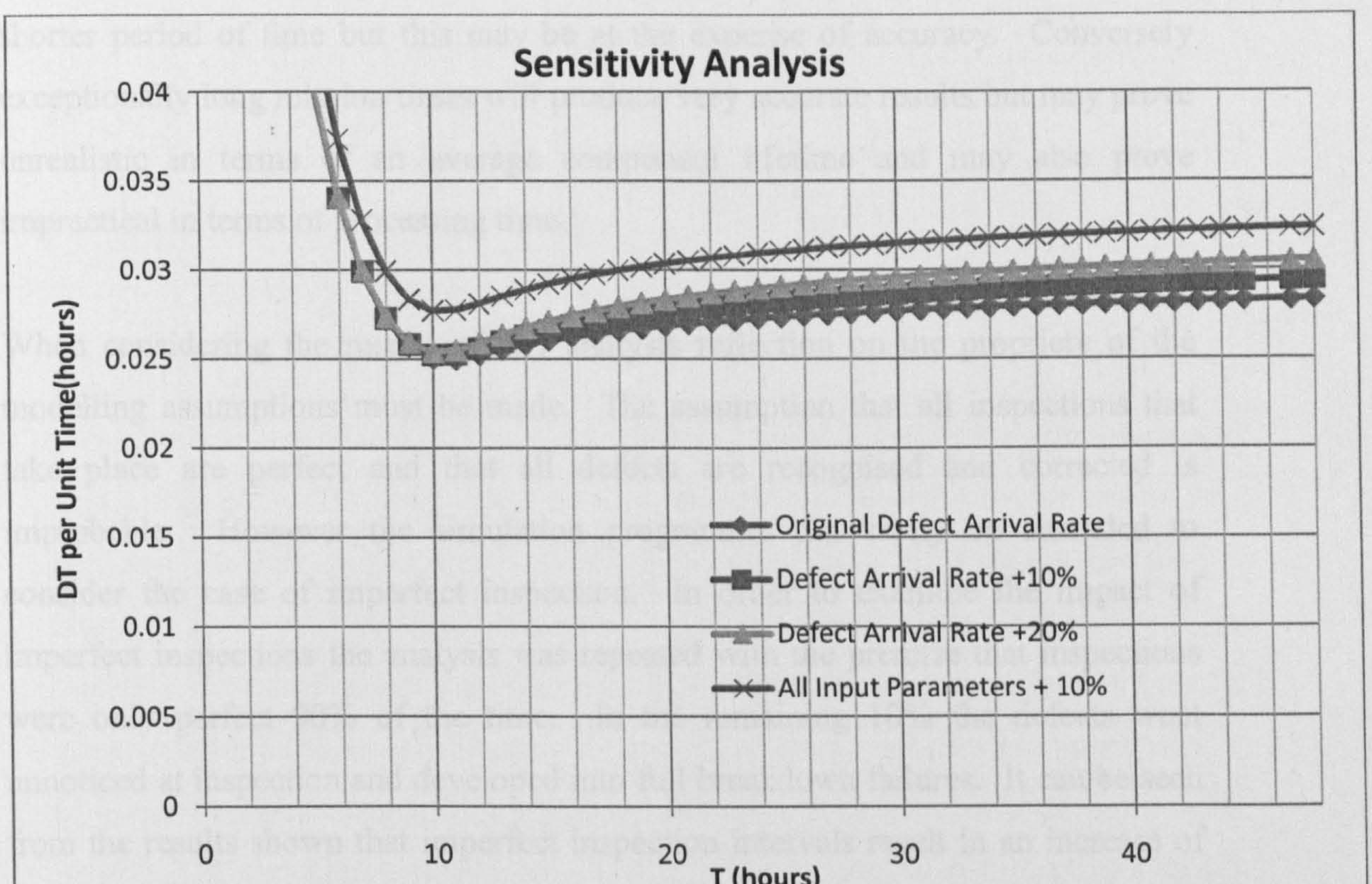

Fig 5.22 Graph showing the results of the sensitivity analysis

It can be seen from the results shown in fig 5.22 that when the arrival rate of defects is increased the DT per unit time also increases. Furthermore when the arrival rate of defects is further increased the DT per unit time increases again by a proportional amount. When all of the input parameters are increased the DT per unit time is increased by a greater magnitude, when compared to alteration of a single input parameter i.e. arrival rate of defects. These results satisfy the axioms outlined previously, thus giving partial validation to the model.

5.7. Discussion

The analysis programme can be easily altered to consider a different set of equipment, with different input parameters. The only limitation to the simulation method is the ability of the programmer to generate random numbers distributed to different distributions. The method gains accuracy when the mission time is set at larger values. This is often at the expense of time to compute simulation results. As computers increase in both speed and processing power this will become less of a problem, however the analyst should always give careful consideration to the suitability of the mission time. Short mission times will produce more results in a shorter period of time but this may be at the expense of accuracy. Conversely exceptionally long mission times will produce very accurate results but may prove unrealistic in terms of an average component lifetime and may also prove impractical in terms of processing time.

When considering the results of any analysis reflection on the propriety of the modelling assumptions must be made. The assumption that all inspections that take place are perfect and that all defects are recognised and corrected is improbable. However the simulation programme can easily be amended to consider the case of imperfect inspection. In order to examine the impact of imperfect inspections the analysis was repeated with the premise that inspections were only perfect 90% of the time. In the remaining 10% the defects went unnoticed at inspection and developed into full breakdown failures. It can be seen from the results shown that imperfect inspection intervals result in an increase of the DT per unit time. The value of 10% was assumed and it has been shown in the literature review that studies have been undertaken which show under certain conditions human error can be as high as 60%. The analysis was re-run to investigate the sensitivity of the model to alterations in the imperfect inspection rate shown in fig 5.23. The DT per unit time increases as the amount of imperfect inspections increase, reducing the amount of imperfect inspections reduces the DT per unit time. The optimal inspection interval remains unchanged. Further increase or reduction in the amount of imperfect inspections has a similar affect of 'shifting' the curve vertically away from or towards the perfect case.

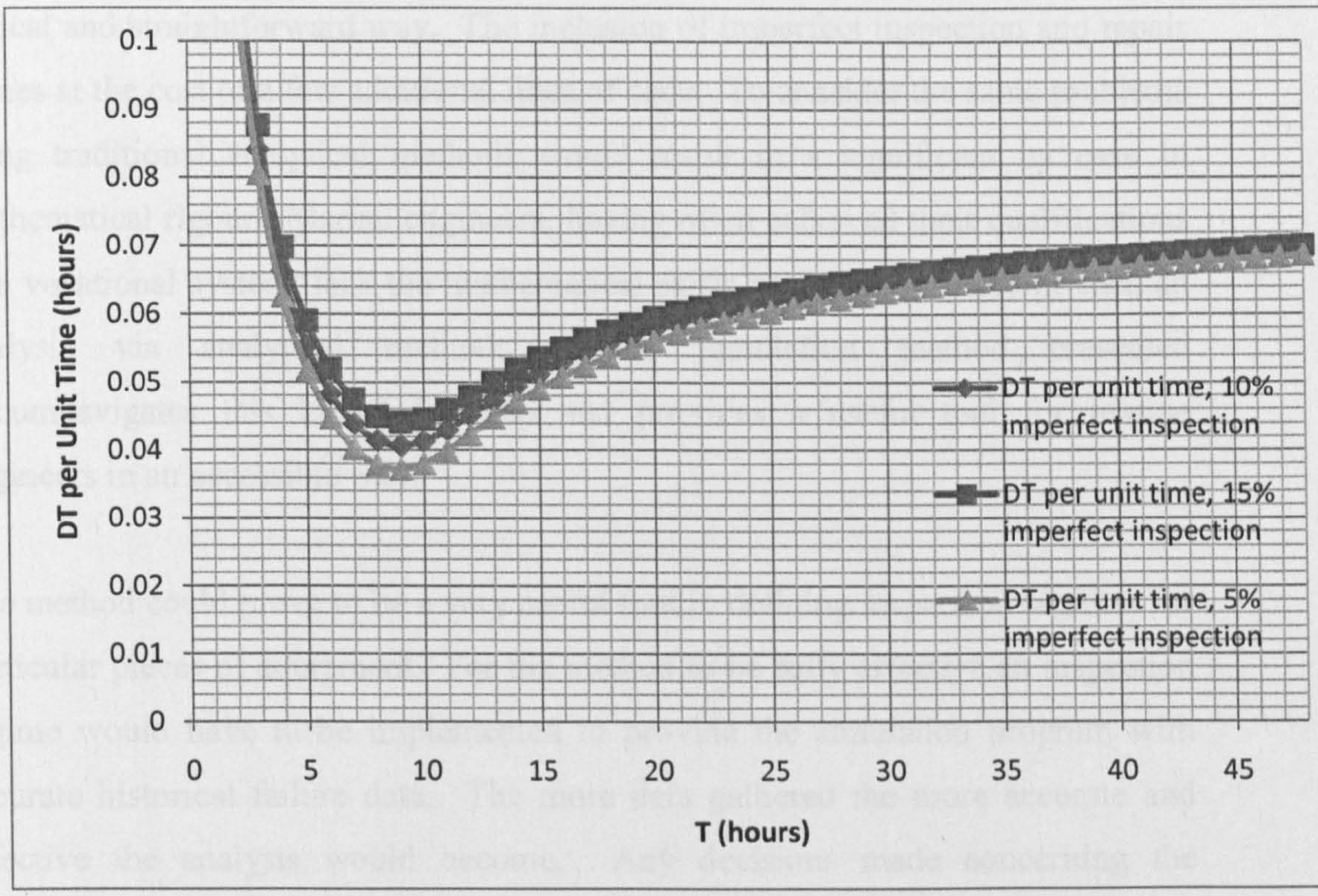

Fig 5.23 Graph showing the sensitivity of the model to different percentages of imperfect inspections

The assumption that the system is returned to 'good as new' after inspection and repair is also one that seems unrealistic. This may not prove to be the case in real life, systems may be put back into service in a degraded state after inspection or repair. This is ultimately dependant upon the experience and skill of the maintenance personnel and the quality of the replacement parts. To examine the effect of imperfect repair, the analysis was repeated with the assumption that after an inspection and subsequent corrective action the system is put back online with an arrival rate of defects increased by 20%. It can be seen from the results that this increases the level of downtime per unit time achieved. The optimal inspection periods remain unchanged.

The results of both models which concerned imperfect inspection and imperfect repair are logical. If inspections are imperfect then there is an increased chance for system breakdown, this is reflected in the increase of downtime per unit time. In the case of imperfect repair the arrival rate of defects increases, this leads to more defects and results in an increase of the downtime per unit time. The strength of the MCM of DTA is the method's ability to deal with different situations in a logical and straightforward way. The inclusion of imperfect inspection and repair comes at the cost of a few additional lines of code. To consider the same problems using traditional analytical methods would result in a significant increase in mathematical rigour. Marine engineers, having often achieved their qualifications in a vocational system, lack the mathematical skills necessary to perform such an analysis via analytical methods. The simulation method presented circumnavigates this knowledge gap and provides a useful tool for marine engineers in an accessible way.

The method could prove to be a very useful tool in defining inspection regimes for particular pieces of equipment. For the method to be fully effective an inspection regime would have to be implemented to provide the simulation program with accurate historical failure data. The more data gathered the more accurate and effective the analysis would become. Any decisions made concerning the maintenance regime onboard will ultimately be decided by the owner/operator of the vessel. The decision to implement DTA will depend upon existing operating and maintenance culture onboard.

5.8. Conclusion

The purpose of this work was to demonstrate an alternative method for DTA other than traditional analytical methods. Previous research work reported on DTA is often arduous in terms of the mathematical models presented. It has demonstrated the benefit of the method but the esoteric nature of the mathematical models, has often prevented engineers in industry from implementing the method. The intention of this researcher was to present a methodology which achieved the same results in a way which was more accessible to a wider range of engineers. Based on the evidence of the results presented the methodology outlined for performing the analysis will provide optimal inspection periods for a given set of data. This work also demonstrates the power and flexibility contained within the MCM to consider a number of different models and methods. A need is also identified for ship owners/operators to invest more time into the collation of failure data specific to their vessel. Different vessels operating in different areas and conditions will display different failure characteristics. The collection of failure data and its use in the analysis of systems with respect to reliability and appropriate maintenance scheduling could only prove beneficial to ship operators.

There is certainly huge scope for further work especially when the simplifying assumptions are considered. In the models presented two of these assumptions were relaxed. The more interesting of the two is the assumption that defect arrival rate is constant. In the analysis the arrival rate was changed as a result of different inspection and breakdown actions, however the arrival rate always obeyed the same distribution. Further work could be done to examine the effect of changing the distribution of the arrival rate of defects throughout the analysis. There is also scope for work considering the age of components and the effect of component age on defect arrival rate.

Chapter 6 – Monte Carlo Simulation to Facilitate Decision Making

Summary

This chapter presents a Monte Carlo Simulation which provides data concerning the effect of different staff levels on system downtime. A methodology is presented and applied to a case study. The aim is to provide the analyst with knowledge as to the point at which an increase in staff level may not necessarily offer a reduction in system downtime.

6.1 Introduction

Throughout this thesis it has been demonstrated how MCM can be used to provide information on system unreliability and maintenance scheduling. However decisions will ultimately need to be made regarding any system as to an appropriate level of staff to conduct maintenance and operation tasks. Chapter 5 presented a methodology which gave a clear indication as to an appropriate inspection regime to aid any decision making in this regard. Decision making and decision making methods are a huge area of ongoing research in the engineering sector. Models are often based on qualitative expert judgements as to the best course of action. The Analytical Hierarchical Process (AHP) is an example of a process which aids decision making. It breaks down a decision problem into a number of simple sub-problems, then by pair-wise comparisons it gives an indication of an optimal solution. AHP relies on human judgement and preference of one option over another, it is widely recognised and validated in a number of studies (Saaty 2008). However if the Monte Carlo model can be extended to give definite information pertaining to a certain decision then there are occasions where quantitative information is preferable over qualitative judgements. This chapter presents one such situation where a quantitative idea of a decisions impact, normally in terms of cost, is preferable.

6.2. Development of the Methodology

Fig 6.1 shows the methodology that has been developed. The diagram shows each of the stages in an analysis and illustrates the work that must be carried out before the analysis program can be written. The section outlines important steps in the methodology, where appropriate simple examples have been given.

6.2.1 Fully Define the System

In order to fully define the system knowledge of all its possible working states must be known. To achieve this there are a number of 'sub-steps' in the methodology, for the purpose of brevity and to avoid repetition these have been omitted from the methodology. For completeness each will be briefly discussed.

6.2.1.1 Use Boolean Representation Method to Reduce System States

The BRM can be used to reduce the system states. BRM takes a table of all possible system states and reduces them to a smaller number of working and failure states. In this way BRM techniques can be applied until an irreducible table of system states is produced.

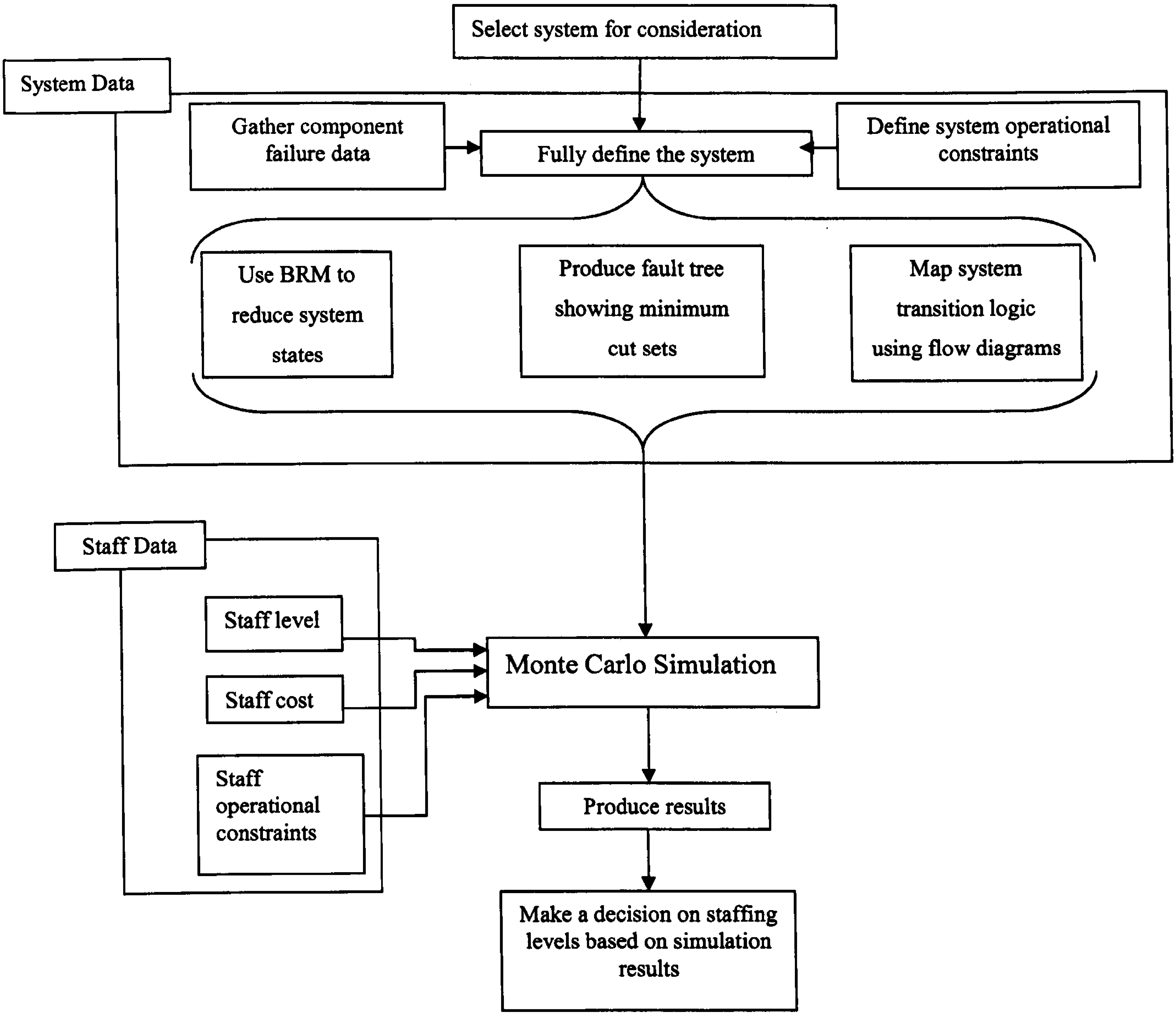

Fig 6.1 Proposed methodology for performing the Monte Carlo Simulation

6.2.1.2 Produce a Fault Tree Showing Minimum Cut Sets Leading to Failure

FTA identifies all the possible causes of a specified undesired top event. FTA is a structured top-down deductive graphical analysis tool. In FTA a cut set is defined as a set of basic events whose simultaneous occurrence ensures that the top event occurs. A cut set is said to minimal if the set cannot be reduced without losing its status as a cut set. The top event will therefore occur if all the basic events in a minimal cut set occur at the same time. Once the system has been analysed using the FTA and BRM techniques the analyst is left with the cut sets that lead to system failure and also the feasible system states.

6.2.1.3 Map System Transition Logic using flow diagrams

The transition of the system from one state to another is of great importance during the analysis. In order to fully understand the system and how the state of individual components affects the higher level system state, the full system must be mapped using system flow diagrams. If done in a thorough and rigorous way it provides a complete and full understanding of how the system behaves. This is done with the introduction of a system state vector $\mathbf{B} = (b_1, b_2, \ldots\ldots, b_m)$, whose elements are the state indicators of the components. For an in depth explanation of how this is implemented and achieved the reader is referred to Chapter 4.

6.2.1.4 Gather Component Failure Data

In the absence of failure data specific to a ship or component other sources of information must be used. It is the accuracy of this data which will ultimately determine the accuracy of the analysis. Expert judgement can be used in the complete absence of data however this can introduce some uncertainty into the results. There are sources of information produced by company collaborations. An excellent source of failure data in the offshore industry is the Offshore Reliability Data Handbook (OREDA 2002).

6.2.1.5 Define System Operational Constraints

Often it is the case in certain systems that certain operation constraints determine what is considered a failed state. It may be that in order for a cooling system to operate correctly at least two centrifugal pumps are required to provide enough fluid to ensure adequate cooling. This kind of operational constraint means that the system is considered to be failed when only one pump is available on demand. Consideration of constraints such as the one outlined can significantly reduce the number of system states which need consideration in the analysis.

6.2.2 Staff Data

Staff data will be independent to the vessel or operating company. Operational data concerning staff levels may also affect the analysis; these can also be independent to a vessel or operating company. Staff costs and staff levels will also be company specific.

6.2.2.1 Staff Level

Staff levels are independent to a particular type of vessel and usually dictated by the IMO safe manning level guidelines in accordance with regulation VIII/2 of the 1978 STCW convention. For the purpose of this study manning levels will be completely variable.

6.2.2.2 Staff Costs

Staff cost are individual to each ship and operator. The information concerning the cost level of a member of staff is not bounded by a particular ship or operator.

In this way the simulation program can be applied across a broad range of ships and operators determined by the analyst.

6.2.3.3 Staff Operational Constraints

Operational constraints concerning the staff levels are independent to a ship. It may be that in a certain application it is a requirement that one member of staff is required to be in the engine control room at all times. This impacts the analysis as no further repairs can take place once this minimum level is met. In this way the simulation is tailored to fit a number of different operational scenarios.

6.2.4 Monte Carlo Simulation – Code Generation

Code generation is now the process of converting the system transition flow diagrams into FORTRAN code using logical IF constructs. The code could be written in any language which allows the use of logical expressions and logical IF constructs. It should be noted that whereas production of an irreducible form of the system states is highly desirable, it is not essential. The production of the irreducible table means that the programmer can generate less code, as certain system states can be ignored. It is the definition of the operating constraints that decides which states are viable or not. The code will simply never enter into any states that the operational constraints do not allow.

6.3. Case Study

In the following a cooling system will be outlined for the purpose of the Monte Carlo analysis. The system is taken from the MV Hamnavoe, a Ro-Ro passenger ferry and has already been presented in fig 3.10.

This case study is concerned with the central section of the cooling system: the pumps, valves and plate coolers which service the main engine. Fig 6.2 shows a simplified version of the central section of the cooling system.

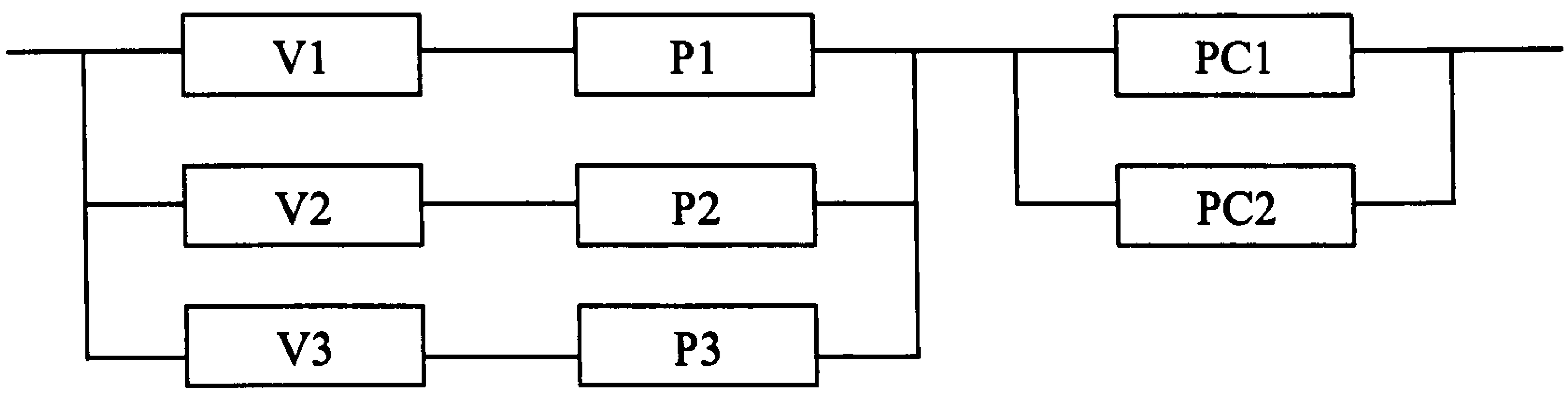

Fig 6.2 Diagram showing the system under consideration

V1, V2, V3, P1, P2, P3, PC1 and PC2 represent valve 1, valve 2, valve 3, pump 1, pump 2, pump 3, plate cooler 1 and plate cooler 2 respectively. All the other components in the system are assumed to operate perfectly i.e. with no failure.

Each component has two modes of operation Working or Failed (W or F). The system constitutes eight individual components giving rise to 2^8=256 system states. Using the BRM it is possible to reduce the system states to the 21 working states and 13 failure states shown in table 6.1, where * represents 'don't care'. The only way to check if the table is irreducible is by manual methods. Producing an irreducible form of the system states is not strictly required by the methodology. Table 6.1 is reproduced from table 4.7.

Table 6.1 Table showing the possible system states and failure states of the system

B	V1	V2	V3	P1	P2	P3	PC1	PC2
1	W	W	W	W	W	W	W	W
2	W	W	W	W	W	W	W	F
3	W	W	W	W	W	W	F	W
4	W	W	W	W	W	F	W	W
5	W	W	W	W	W	F	W	F
6	W	W	W	W	W	F	F	W
7	F	W	W	W	W	W	W	W
8	F	W	W	W	W	W	W	F
9	F	W	W	W	W	W	F	W
10	W	F	W	W	W	W	W	W
11	W	F	W	W	W	W	W	F
12	W	F	W	W	W	W	F	W
13	W	W	F	W	W	W	W	W
14	W	W	F	W	W	W	W	F
15	W	W	F	W	W	W	F	W
16	W	W	W	F	W	W	W	W
17	W	W	W	F	W	W	W	F
18	W	W	W	F	W	W	F	W
19	W	W	W	W	F	W	W	W
20	W	W	W	W	F	W	W	F
21	W	W	W	W	F	W	F	W
22	F	F	*	*	*	*	*	*
23	F	*	F	*	*	*	*	*
24	*	F	F	*	*	*	*	*
25	*	*	*	F	F	*	*	*
26	*	*	*	F	*	F	*	*
27	*	*	*	*	F	F	*	*
28	*	*	*	*	*	*	F	F
29	F	*	*	*	F	*	*	*
30	F	*	*	*	*	F	*	*
31	*	F	*	F	*	*	*	*
32	*	F	*	*	*	F	*	*
33	*	*	F	F	*	*	*	*
34	*	*	F	*	F	*	*	*

For the purpose of the descriptions of this case study it is necessary to define the following. A system is a collection of $m \geq 1$ components and the system state can be represented by the vector $\mathbf{B} = (b_1, b_2, \ldots, b_m)$ whose elements are the state indicators of the components. In the case of the system under consideration, the system state vector **B** will contain eight elements $\mathbf{B} = (b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8)$ representing V1, V2, V3, P1, P2, P3, PC1 and PC2 respectively. In the nominal state with all components in a working state V2, P2 and PC2 are considered to be 'cold standby' (ST.) units i.e. cannot fail while in standby. The system is required to function for a fixed amount of time, T_m, referred to as the mission time. In the event that a transition leads to system failure the system is not put back online until the repair of the last component to fail is completed. Also as part of the operating requirements of the ship it is required that there always be one member of engine room staff in the engine control room, this effectively reduces the number of staff available for repairs by one.

6.3.1 Calculating System Downtime and the Effect of Different Numbers of Staff

During a single mission the number of staff available will have a significant effect on the system downtime. In the work presented in the previous chapters it has been assumed that there is an infinite number of staff. Once a component experiences failure the repair action begins. The purpose of this chapter is to present a case study in which a staff level is determined before the Monte Carlo trials begin.

Fig 6.3 shows the effect that different staff levels have on the accumulated system downtime. The points T_1, T_2 and T_3 represent individual transition times of the system caused by the failure of a component. The first timeline shows the effect of an infinite number of staff, all of the repair actions start immediately at the respective transition times and the system downtime is equal to one deterministic repair time. In the second timeline shown in more detail in fig 6.4, the system downtime increases as there is a 'waiting' time between the time of transition and the start of the repair action.

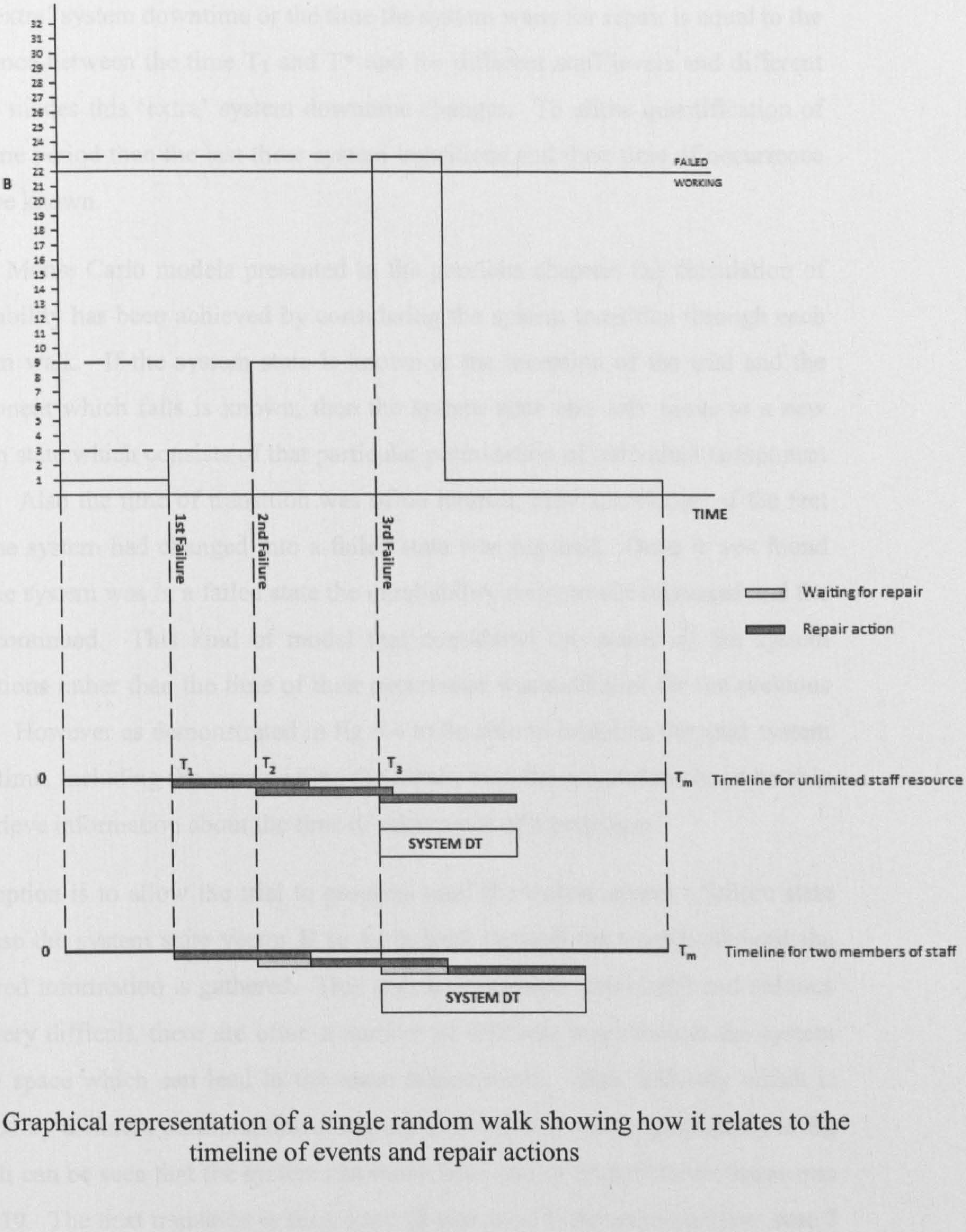

Fig 6.3 Graphical representation of a single random walk showing how it relates to the timeline of events and repair actions

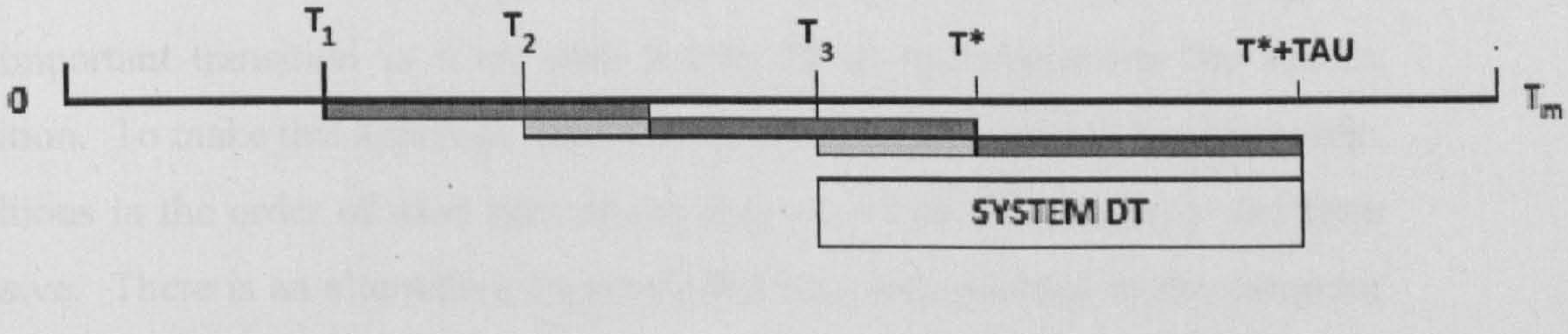

Fig 6.4 Timeline for two members of staff

This 'extra' system downtime or the time the system waits for repair is equal to the difference between the time T_3 and T^* and for different staff levels and different failure modes this 'extra' system downtime changes. To allow quantification of this time period then the last three system transitions and their time of occurrence must be known.

In the Monte Carlo models presented in the previous chapters the calculation of unreliability has been achieved by considering the system transition through each random walk. If the system state is known at the inception of the trial and the component which fails is known, then the system state can only move to a new system state which consists of that particular permutation of individual component states. Also the time of transition was of no interest, only knowledge of the fact that the system had changed into a failed state was required. Once it was found that the system was in a failed state the unreliability counter was increased and the trial continued. This kind of model that considered the nature of the system transitions rather than the time of their occurrence was sufficient for the previous work. However as demonstrated in fig 6.4 to be able to calculate the total system downtime, including the time waiting for repair, then the programme must be able to retrieve information about the time of occurrence of a transition.

One option is to allow the trial to progress until the system enters a failure state and use the system state vector B to work back through the transitions until the required information is gathered. This approach is rather convoluted and arduous and very difficult, there are often a number of different ways though the system phase space which can lead to the same failure mode. This difficulty which is created by different permutations of system transitions is shown graphically in fig 6.5. It can be seen that the system can move from one of nine different states into state **19**. The next transition is from state **19** into state **7**, the transition into state **7** could also have nine different possibilities, only one of which is shown in fig 6.7. The important transition is from state **9** into **23** as this represents the failure transition. To make this approach viable the program would have to log all system transitions in the order of their occurrence, this would be both memory and time intensive. There is an alternative approach that was implemented in this program which will be presented in the next section.

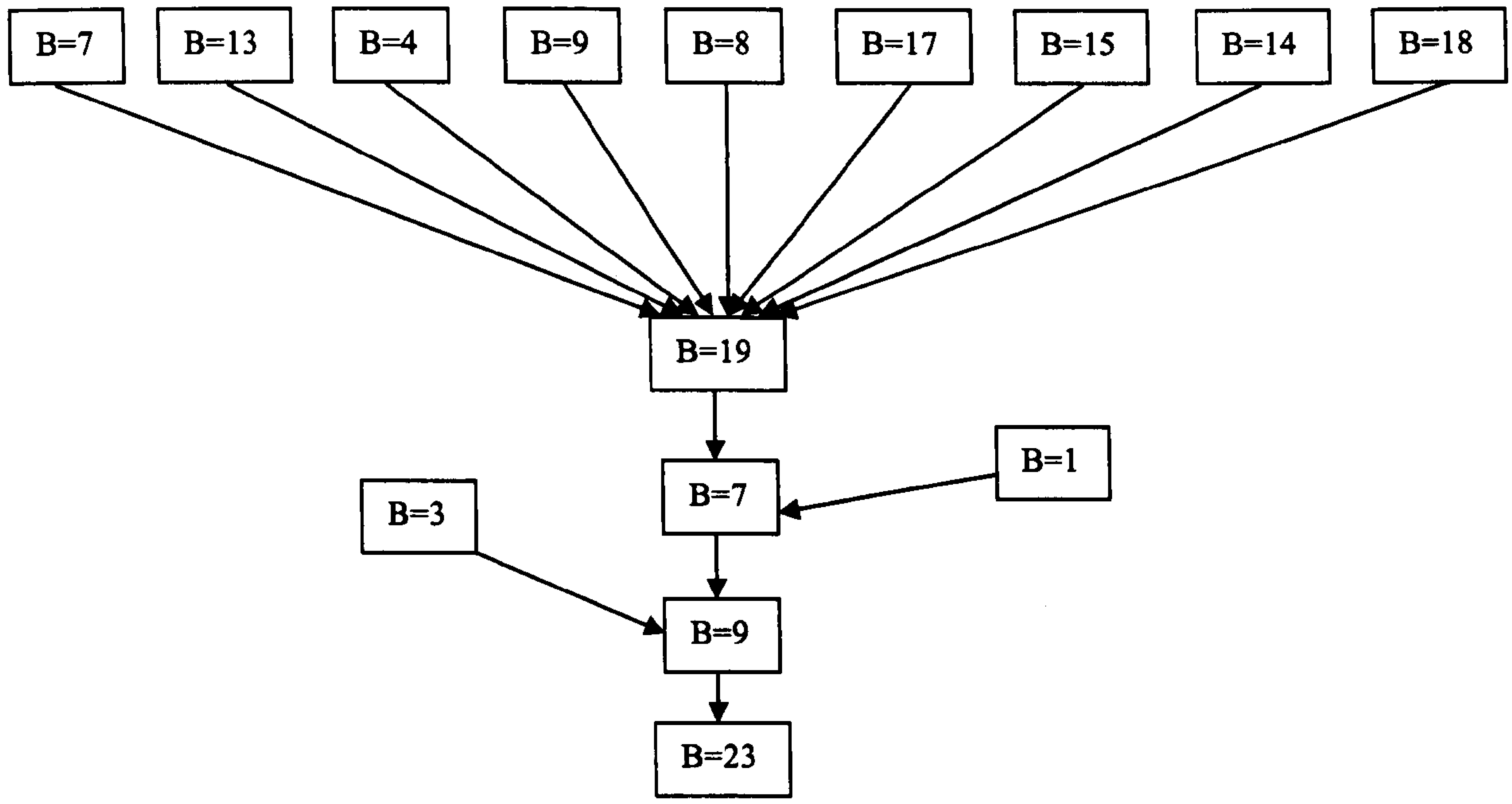

Fig 6.5 Graphical representation of the different state transitions which can lead to a certain failure state

6.3.2 Circular Buffer Algorithm (CBA)

Upon further examination of the system transition logic and the operating constraints it becomes apparent that in the system presented there are only two types of failure transition, a two transition system failure and a three transition system failure. Two transition system failures occur when two components fail in sequence leading to a system failure state. Three transition system failures occur when three components fail in sequence leading to a system failure state. It now becomes apparent that in order to be able to compute the system downtime, including the times when the system waits to undergo repair, only the last three times of system transition, at most, are required.

The CBA is created in a subroutine which the main program can reference and read or write values to. The CBA code is shown in appendix 3. In this instance a simple array is created which contains a single column and three rows, this is shown graphically in fig 6.6. At the inception of the Monte Carlo trial the array is initialised and all values set to 0. At the first call to the subroutine the current time of transition, T_1, is stored in the array at location A1. On the second call the old value, T_1, is copied from position A1 to position A2 and the new value of system transition, T_2, is stored in the position A1. On the third call the oldest value, T_1, is copied from position A2 to A3. The old value of system transition, T_2, is copied from position A1 to A2 and the new value of system transition, T_3, is stored at position A1. On the next call the oldest value of system transition, T_1, is overwritten as value T_2 moves from position A2 to A3. The value of system transition, T_3, moves from A1 to A2 and a new value of system transition, T_4, is stored in position A1. The algorithm continues in this way storing and copying values, whilst always discarding the oldest value, allowing the program to read the times of the last three system transitions.

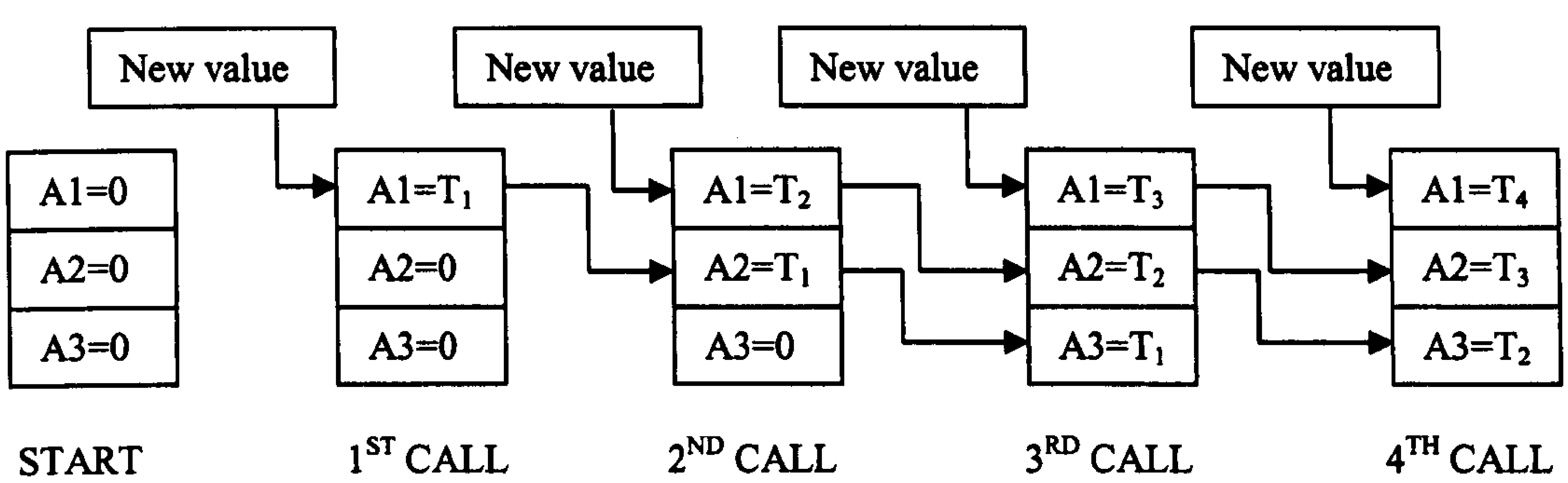

Fig 6.6 Graphical representation of the circular buffer algorithm

6.4. Case Study Results

The failure rates for the individual components in the program were taken from the OREDA handbook (OREDA 2002). The failure rate columns show an estimate of the failure rate for each failure mode. For this study the accumulated number of failures for each component was used and is presented as 'all modes'.

Table 6.2 shows the failure data for centrifugal pumps in the marine industry, the data encompasses a population of 350 units over 59 offshore installations.

Table 6.2 OREDA failure data for centrifugal machinery pumps

Failure mode	No. of failures	Failure rate (per 10^6 hours)					Active repair hours	Repair (man-hours)		
		Lower	Mean	Upper	SD	n/τ		Min	Mean	Max
All modes	1949	172.90	1277.00	3233.73	1001.75	339.22	21.3	0.5	30.5	1025.0

Table 6.3 shows the failure data for ball valves, the data includes a population of 316 units over 18 offshore installations.

Table 6.3 OREDA failure data for ball valves

Failure mode	No. of failures	Failure rate (per 10^6 hours)					Active repair hours	Repair (man-hours)		
		Lower	Mean	Upper	SD	n/τ		Min	Mean	Max
All modes	328	8.80	43.70	100.54	29.47	35.18	6.4	1.0	10.7	113.0

Table 6.4 shows the failure data for plate heat exchangers, water→sea water, the data encompasses a population of 8 units over 3 offshore installations.

Table 6.4 OREDA failure data for plate heat exchangers

Failure mode	No. of failures	Failure rate (per 10^6 hours)					Active repair hours	Repair (man-hours)		
		Lower	Mean	Upper	SD	n/τ		Min	Mean	Max
All modes	9	0.27	39.75	137.51	50.71	35.25	13.7	6.0	29.0	109.0

OREDA also documented failure data which was specific to centrifugal pumps in cooling systems. This data only covered 18 units from 2 offshore installations and therefore the data for centrifugal pumps in general was chosen for the analysis. The mission time was based on the passage time from Liverpool→Shanghai→New York based on an average speed of 25 knots.

The passage time with a speed of 25 knots works out at 22 days and 4 hours. This gives a mission time of 532 hours. After a full mission time the components are treated as 'same as new'. For the initial analysis table 6.5 shows the input values used.

Table 6.5 Input values for the Monte Carlo analysis

T_m (hrs)	532
$\lambda_{v1}(hrs^{-1})$	4.37×10^{-5}
$\lambda_{v2}(hrs^{-1})$	4.37×10^{-5}
$\lambda_{v3}(hrs^{-1})$	4.37×10^{-5}
$\lambda_{p1}(hrs^{-1})$	1.277×10^{-3}
$\lambda_{p2}(hrs^{-1})$	1.277×10^{-3}
$\lambda_{p3}(hrs^{-1})$	1.277×10^{-3}
$\lambda_{pc1}(hrs^{-1})$	3.975×10^{-5}
$\lambda_{pc2}(hrs^{-1})$	3.975×10^{-5}
t (hrs)	10

The number of staff was increased in increments of one person and the results are shown in table 6.6.

Table 6.6 Results of the analysis

Number of Staff	System Downtime
1	434.93
2	15.37
3	10.00
4	10.00
5	10.00
6	10.00
7	10.00
8	10.00
9	10.00
10	10.00

It can be seen from the analysis that once the staff level reaches three members of staff an increase in the level reflects no further reduction in system downtime.

The cost model is based on the information provided by Mr Ramin Riahi. A detailed description of Mr Riahi's industrial experience and academic qualifications is listed in appendix 4. Mr Riahi stated that the average costs per day of three different engine room staff members are those presented in table 6.7.

Table 6.7 Cost per day of engine room staff

Staff Member	Income/day ($)
Chief Engineer	300-400
2ND Engineer	250-350
3RD Engineer	200-300

Based on an average eight hour shift in a twenty four hour period and taking the mid range of each income, the cost per hour of each member of staff was calculated, shown in Table 6.8.

Table 6.8 Cost per hour of engine room staff

Staff Member	Income/hour($)
Chief Engineer	43.75
2ND Engineer	37.5
3RD Engineer	31.25

The cost results were plotted against system downtime and are shown graphically in fig 6.7. The graph shows the cost associated with two different cases. In the first case the first member of staff employed is a chief engineer, followed by a second engineer and then multiple third engineers depending on the staff level. In the second case a single chief engineer is employed first followed by multiple third engineers depending on the staff level.

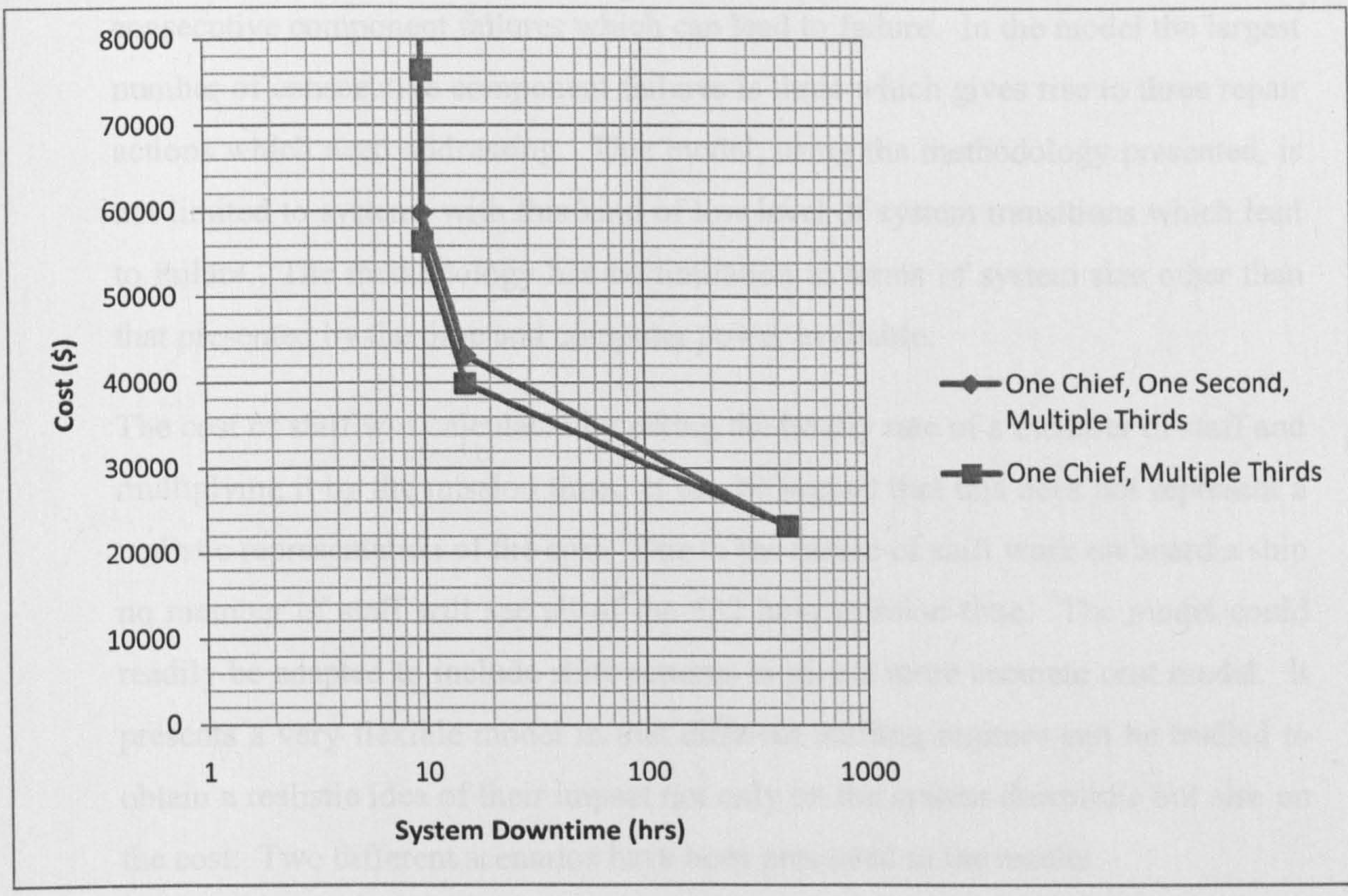

Fig 6.7 Graph showing system downtime against cost

6.5. Discussion

The validation for the model used has already been presented in Chapter 4 of the thesis.

The model demonstrates how a Monte Carlo Simulation can be extended to give information pertaining to both system downtime and the effect of different members of staff. The cost model which has been developed will change with each different analysis performed. Different companies and vessels will have different staff payment regimes and different recommended minimum safe manning levels. The development of the cost model based on the expert data demonstrates how the information from the simulation can be used. There is a definite staffing level which produces a minimised system downtime, in this case 3 members of staff. Increasing the number of staff after this point has no effect other than increasing the staff cost. The model presented achieved the minimum system downtime relatively quickly. This level of staff is linked to the number of consecutive component failures which can lead to failure. In the model the largest number of consecutive component failures is three which gives rise to three repair actions which need addressing. This model, using the methodology presented, is not limited to systems with this kind of low level of system transitions which lead to failure. The methodology has no limitation in terms of system size other than that presented by the time and computer power available.

The cost of staff was calculated by taking the hourly rate of a member of staff and multiplying it by the mission time. It can be argued that this does not represent a realistic representation of the cost. Due to the nature of shift work on board a ship no member of staff will see all of the 532 hour mission time. The model could readily be adapted to include shift patterns to give a more accurate cost model. It presents a very flexible model in that different staffing regimes can be trialled to obtain a realistic idea of their impact not only on the system downtime but also on the cost. Two different scenarios have been presented in the results.

The concept of introducing periods in which the system is left 'waiting' repair could also be extended to include unforeseen problems. The model could be modified to randomly include time waiting for parts without great difficulty. It could also include time taken for the system to 'start-up' before it is in a full mode of operation. In the model presented it is assumed that the system is put back online instantaneously, which is often an unrealistic assumption.

The strength of this model is its simplicity it shows something that most experienced systems engineers can deduce intuitively. This does not detract from the fact that the model has allowed for the formulation of a methodical process which can be applied to a much larger system. The more complex a system gets the more subtle the interactions between various states can become, it is often in this case that misplaced confidence can be put into intuition.

6.6. Conclusion

This chapter has demonstrated how Monte Carlo Methods can be extended to give information which can be used to assess appropriate staff level. The model presented represents an extension of work from a previous chapter. Staff level will ultimately be dictated by the ships safe manning document. However information which can be gleaned from any system analysis can be helpful when making decisions. The methodology developed is readily applicable to larger systems where appropriate staff level for maintenance and supervision is not easily discerned. Large systems present greater challenges in terms of the shear number of system states that can exist and the computer power required however the methodology could be applied to a full engine room. This would be a mammoth task and at the present time would not be feasible but it is entirely possible.

Chapter 7 – Conclusions and Recommendations for Future Work

Summary

This chapter will draw together the main conclusions from this research work, highlighting the contribution that has been made to the research field. It will also discuss the limitations of the work done and highlight the opportunities for further work.

7.1 Main Conclusions

The application of MCM in the marine environment has been the main area of this research work. Specifically the analysis of a main engine cooling system using Monte Carlo Methods has been central to all of the technical research. The technique randomly samples data from known statistical distributions using component failure probabilities to simulate system behaviour. The results of the analyses provide engineers with the information to assess the impacts of decisions concerning maintenance and operations.

The Monte Carlo analysis of the complex system (chapter 4) showed how the behaviour of a complex system is often different to that which human logic might dictate. The study highlighted the fact that a simulation program could provide a useful tool for marine engineers on board ship. If a simulation program could be produced, which would allow the marine engineer to define a system and perform an analysis, then decisions concerning maintenance could be tested before any changes were made to the real life systems.

The DTA using MCM (chapter 5) provided a model which could be used by a wide range of engineers without the prerequisite of extensive mathematical knowledge. Based on the evidence of the results presented the methodology outlined for performing the analysis will provide optimal inspection periods for a given set of data. This work also demonstrates the power and flexibility contained within the MCM to consider a number of different models and methods.

Monte Carlo Simulation to facilitate decision making (chapter 6) showed how simulation models of complex systems can be extended to provide quantifiable information as to the impact of a decision. The methodology outlined will provide cost and system downtime information for a given staff level. The methodology developed is readily applicable to larger systems where appropriate staff level for maintenance and supervision is not easily discerned.

7.2 Research Contribution

The main contribution of this work is the application of MCM in the marine environment. The methodology for a simulation based maintenance model has been developed. It has been shown how the incorporation of FTA and BRM techniques can be used to aid the formulation of the Monte Carlo Model. The practical application of MCM in the marine environment has been achieved with several case studies presented. The application of MCM in marine systems analysis has shown how useful a tool this could be to the marine sector. It has application both in ship maintenance planning and ship design.

In addition to a classical Monte Carlo approach the method has been extended to encompass the DTA to establish optimal inspectional intervals through minimising downtime relating to maintenance and inspection. It has been shown that the MCM applied to DTA allow for non-homogeneous Poisson process to be considered without a significant increase in mathematical rigour. In this way the model was extended so that a non-constant failure rate was considered. The failure rate deteriorated based on an imperfect inspection regime. The model allowed for results to be produced when considering both imperfect failure and imperfect repair.

A model was also developed to establish optimal staff level through minimising system downtime and staff cost. The proof of such situations can sometimes seem intuitive however the methodology produced has significant scientific rigour for application to larger systems where intuition may be unable to play a part. Overall it has been demonstrated how MCM could be used to produce simulation based models of marine systems to provide quantitative data to aid in decision making processes regarding operation and maintenance.

The models used incorporated OREDA failure data and full methodologies were outlined for the application of method and how the data should be used. The feasibility of MCM within the marine industry is ultimately reliant on the availability of failure data. OREDA provides a great source of information from the offshore industries. In order for this and similar research to progress, develop and become practicable, further efforts must be made by the marine industry as a whole, to collect and collate failure data. Finally, it should be remembered that the accuracy of the simulation is always dependant on the accuracy of the failure data.

7.3 Limitations

The Monte Carlo Models presented in chapters 4 and 6 have failure rates based on the exponential distribution. This represents components with constant failure rate where past events have no bearing on the system going forward. The models also assume that after repair components are returned to a 'good as new' state. These assumptions are unrealistic but were necessary to simplify the analysis. This may limit the application of the methodologies to certain components or systems.

In the case study presented in chapter 4 only case 3 allows the system to be put back online as soon as enough components are available for normal operation. However this case only deals with equal deterministic repair times, when consideration is given to the application of this operating scenario to repair times that are different for each component the analysis becomes much more complex.

The DTA presented in chapter 5 illustrates the worst case scenario condition. The condition states that if a failure has not occurred before the inspection then it will occur during the inspection repair.

In all the Monte Carlo models presented no consideration was given to the variance of the final results. As such all the models presented are 'analogue' Monte Carlo models.

In order to fully appreciate the limitations of the methodology presented in this thesis a number of different case studies on different marine equipment must be conducted.

7.4 Future Work

The potential for the MCM and its application to systems in the marine environment is vast. Having identified the limitations of the research work presented in this thesis future work could certainly be done to address some of the problems highlighted.

Monte Carlo models can be developed to sample from numerous distributions. Some which could be considered are the Weibull, gamma, normal, multivariate and lognormal distributions. The selection of a distribution which is most appropriate for the available failure data could be achieved. This would involve the use of 'goodness of fit' testing to decide upon the appropriate distributions. A number of distributions can be used in the same analysis using the composition method. In this way the distribution could change to reflect the changing failure characteristics of a component throughout its lifetime. This would provide a much more dynamic analysis of the system, at the moment the majority of commercial system analysis tools work on the assumption of constant failure rates.

A model could be developed in which components are not always returned to a 'good as new' state but reflects the possible degradation of a component as a consequence of a number of repairs. Furthermore the assumption that all repairs are perfect could also be addressed.

In some applications of MCM dealing with so called 'rare' events it is necessary to implement variance reduction techniques. One such method is the use of 'forced' Monte Carlo. In the cooling system presented, due to the harsh nature of the marine environment, the failure rates were such that biasing was not required. However there may be cases in future applications where it is. The development of a methodology which could incorporate biased MCM would be advantageous. It would also allow investigation as to the point at which an event becomes 'rare' and could specify a point at which variance reduction is required.

Due to the run-time inherent to all applications of MCM it is often the case that real-time use of models is not considered feasible. Work could be done to use MCM in conjunction with feed forward artificial neural networks. If the MC model can be used to produce a sufficient data set then a neural network could be trained to produce the desired output values in a number of different circumstances. This is advantageous in that the artificial neural network could produce the same results as those in the MC model, however a neural network will often execute and produce results in a fraction of a second.

The ultimate long-term goal for further development of the research work would be the development of a 'user friendly' software package. This software would have a user interface which was accessible to people who are not well versed in the application of MCM. All the user has to do is model the system within the package, enter the relevant failure data and the software package would run the analysis independently. It is also envisaged that the MCM could be advanced with the development of an object oriented MC analysis tool. This would involve a graphical interface, where various pumps, valves and coolers are represented by their appropriate symbols. Attached to each of these graphics would be a module of code which contained all of the appropriate information for that particular piece of equipment. The user is required only to be able to recreate a line diagram of the system under consideration by 'dragging and dropping' the system component into place. Simple lines connecting the various components determine whether the components have are in parallel or series and check boxes used to define cold standby units. If constructed as envisaged the software package would construct a fault tree of the system, produce the minimum cut-sets and run the analysis. Producing as much information about the system as is required by the user. While post processing the results the user could remove standby equipment, alter failure rates and re-run the analysis. This software would then be able to give quantitative estimates of the effect of these changes. Such a software package is feasible however to be useable, i.e. used to analyse real systems with little simplification, a great deal of computer power would need to be made available. Run times would be long and often unacceptable to be used for real time application. This would be less of a problem in a ship design office where computers could be dedicated to this type of analysis and left to run when longer run times are required. A genetic algorithm tool could be integrated into the software package to provide full validation of the results.

References

Aalbers A.B., Cooper C.K., Nowak S., Lloyd J.R., Leenaars C.E.J., Vollen F. (2001) 'SafeTrans: A New Software System for Safer Rig Moves'. *City University Jack-Up Conference.*

Accumolli A. (1996) 'Manufacturing process problem analysis using Monte Carlo simulation'. *Quality and Reliability Engineering International.* 12, 39-41.

Anderson D.R., Sweeney D.J., Williams T.A. (2003) 'An Introduction to Management Science: Quantitative Approaches to Decision Making'. Melissa Accuna, 10th Ed.

Andrews J.D., Moss T.R. (2002) 'Reliability and Risk Assessment'. Second Edition, John Wiley & Sons Ltd.

Apostolakis G.E., Salem S.L., Wu J.S. (1978) 'CAT: A Computer Code for Automated Construction of Fault Trees'. *EPRI Report*, USA.

Arslan T. Khisty C.J. (2005) 'A Rational Reasoning Method from Fuzzy Perceptions in Route Choice'. *Fuzzy Sets and Systems.* 150(3), 419-435.

Backlund F., Akersten P.A. (2003) 'RCM Introduction: Process and Requirements Management Aspects'. *Journal of Quality in Maintenance Engineering.* 9(3), 250-264.

Baker R., Wang W. (1993) 'Developing and Testing the Delay-Time Model'. *Journal of Operational Research Society.* 44(4), 361-374.

Barata J., Guedes Soares C., Marseguerra M., Zio E. (2002) 'Simulation Modelling of Repairable Multi-Component Deteriorating Systems for 'On Condition' Maintenance Optimisation'. *Reliability Engineering and System Safety.* 76, 255-264.

Barbera F., Schneider H., Kelle P. (1996) 'A Condition Based Maintenance Model with Exponential Failures and Fixed Inspection Intervals'. *Journal of the Operational Research Society.* 47, 1037-1045.

Beebe R. (2003) 'Condition Monitoring of Steam Turbines Performance by Analysis'. *Journal of Quality in Maintenance Engineering.* 9(2) 2003.

Ben-Daya M. (2000) 'You may need RCM to enhance TPM implementation'. *Journal of Quality in Maintenance Engineering.* 6(2), 82-85.

Berrittella M., Certa A., Enea M., Zito P. (2007) 'An Analytical Hierarchy Process for the Evaluation of Transport Policies to Reduce Climate Change Impacts'. *Note di Lavoro Della Fonadazione Eni Enrico Mattei.* 12.

Bevilacqua M., Braglia M., Gabrielli R. (2000) 'Monte Carlo Simulation Approach for a Modified FMECA in a Power Plant'. *Quality and Reliability Engineering International.* 16, 313-324.

Brown A., Haugene B. (1998) 'Assessing the Impact of Management and Organisational Factors on the Risk of Tanker Grounding'. *Proceedings of the International Offshore and Polar Engineering Conference.* 4, 469-476.

Brown M., Proschan F. (1983) 'Imperfect Repair'. *Journal of Applied Probability.* 20(4), 851-859.

Buffon G. (1777) 'Essai d'Arithmetique Morale'. *Supplement a l'Histoire Naturelle.* 4.

Calder, N. (2007) 'Marine Diesel Engines: Maintenance, Troubleshooting and Repair'. 3rd ed., Adlard Coles Nautical.

Cantoni M., Marseguerra M., Zio E. (2000) 'Genetic Algorithms and Monte Carlo Simulation for Optimal Plant Design'. *Reliability Engineering and System Safety.* 68, 29-38.

Cashwell E.D., Everett C.J. (1959) 'A Practical Manual on the Monte Carloe Methods for Random Walk Problems'. Pergamon Press, New York.

Christer A.H. (1999) 'Developments in Delay-Time Analysis for Modelling Plant Maintenance'. *Journal of the Operational Research Society.* 50, 1120-1137.

Christer A.H., Walker W.M. (1984) 'Delay-Time models of Industrial Inspection Maintenance Problems'. *Journal of Operational Research Society.* 35, 401-406.

Christer A.H., Wang W. (1995) 'A Delay-Time Based Maintenance Model of Multi-Component System'. *IMA Journal of Mathematics Applied in Business and Industry.* 6, 205-222.

Curtiss J.H. (1954) 'Monte Carlo Methods for the Iteration of Linear Operators', *Journal of Maths & Physics.* 32, 209-232.

Dey P.K., Gupta S.S. (2001) 'Feasibility Analysis of Cross-Country Petroleum Pipeline Projects: A Quantitative Approach'. *Project Management Journal.* 32(4), 50-58.

Dhillon, B.S. (1983) 'Reliability Engineering in Systems Design and Operation'. Van Nostrand, New York.

Dixon P. (1964) 'Decision Tables and Their Applications'. *Computer and Automation.* 13(4), 376-386.

Dowd M. (2006) 'A Sequential Monte Carlo Approach for Marine Ecological Prediction'. *Environmetrics.*17, 435-455.

Drake P.R. (1998) 'Using the Analytic Hierarchy Process in Engineering Education'. *International Journal of Engineering Education.* 14(3), 191-196.

Dubi A. (1986) *'Monte Carlo Calculations for Nuclear Reactors'.* CRC Handbook of Nuclear Reactor Calculations, Y. Ronen, Ed., CRC PRESS, Vol. II, 1.

Dubi A. (2000) *'Monte Carlo Applications in Systems Engineering'.* John Wiley & Sons Ltd.

El-Haram M.A., Horner M.W. (2002) 'Practical Application of RCM to Local Authority Housing: A Pilot Study'. *Journal of Quality in Maintenance Engineering.* 8(2), 135-143.

Emblemsvåg J., Tonning L. (2003) 'Decision Support in Selecting Maintenance Organization'. *Journal of Quality in Maintenance Engineering.* 9(1), 11-24.

Forsythe G.E., Leibler R.A. (1950) 'Matrix Inversion by the Monte Carlo Method'. *Journal of Maths Tables and Other Aids to Computing.* 4, 213-257.

Fussel J.B. (1973) 'Synthetic Tree Model – Formal Methodology for Fault Tree Construction'. *ANCR-1098*. Spring Field, USA.

Gerbard M., Spanier J. (1969) ' Monte Carlo Principles and Neutron Transport Problems'. Addison-Wesley.

Goldfield A., Dubi A. (1987) 'Monte Carlo Methods in Reliability Engineering'. *Journal of Quality and Reliability Engineering International.* 3(2), 83-91

Grandzol J.R. (2005) 'Improving the Faculty Selection Process in Higher Education: A Case for the Analytical Hierarchy Process'. *Association for Institutional Research.* 6.

Guedes Soares C., Garbatov Y. (1996) 'Fatigue Reliability of the Ship Hull Girder Accounting for Inspection and Repair'. *Reliability Engineering and System safety* 51, 341-351.

Guedes Soares C., Santos T.A. (2004) 'Monte Carlo Simulation of Damaged Ship Survivability'. Proc. ImechE, Journal of Engineering for the Maritime Environment. 219(1), 25-36.

Halton J.H. (1962) 'Sequential Monte Carlo'. *Proc. Cambridge Philos. Soc.* 58, 57-78.

Halton J.H. (1994) 'Sequential Monte Carlo techniques for the Solution of Linear Systems'. *Journal of science and computing.* 9, 213-257.

Henley E.J., Kumamoto H. (1992) 'Probabilistic Risk Assessment'. *IEEE Press.* New York.

Kececioglu, D. (1991) 'Reliability Engineering Handbook'. 2, Prentice Hall Inc., New Jersey.

Labeau P.E. (2000) 'Partially Unbiased Estimators for Reliability and Availability Calculations'. *Progress in Nuclear Energy.* 36(2), 131-187.

Labeau P.E., Zio E. (2002) 'Procedures of Monte Carlo Transport Simulation for Applications in System Engineering'. *Reliability Engineering and System Safety.* 77, 217-228.

Lambert J.H., Peterson K.D., Joshi N.N. (2006) 'Synthesis of Quantitative and Qualitative Evidence for Accident Analysis in Risk-Based Highway Planning'. *Accident Analysis and Prevention.* 38(5), 925-935.

Lehmer D.H. (1951) 'Mathematical Methods in Large Scale Computing Units'. *Ann. Comp. Lab. Harvard University,* 26, 141-146.

Leung F., Kit-leung M. (1996) 'Using Delay-Time analysis to Study the Maintenance Problem of Gearboxes'. *International Journal of Operations & Production Management.* 16(12), 98-105.

Lewis,E.E. (1987)' Introduction to Reliability Engineering'. John Wiley & Sons Ltd, New York, 1987.

Lim T.C., Thanopoulou H.A., Beynon M.J., Bereford A.K.C (2004) 'An Application of AHP on Transhipment Port Selection: A Global Perspective'. *Maritime Economics and Logistics.* 6, 70-91.

Lux I., Koblinger L. (1991) 'Monte Carlo Transport Methods: Neutron and Photon Calculations'. CRC Press.

MAIB (1991-2005) 'Marine Accident Reports', Marine Accident Investigation Branch of the Department of the Environment Transport and the Regions, London.

Manno I. (1999) 'Introduction to the Monte Carlo Method'. Akadémiai Kiadó.

Marseguerra M., Zio E. (1993) 'Nonlinear Monte Carlo Reliability Analysis with Biasing towards Top Event'. *Reliability Engineering and System Safety.* 40, 31-42.

Marseguerra M., Zio E. (2000) 'Monte Carlo Biasing in Reliability Calculations with Deterministic Repair Times'. *Annals of Nuclear Energy.* 27, 639-648.

Marseguerra M., Zio E. (2002) *'Basics of the Monte Carlo Method with Application to System Reliability',* LiLoLe – Verlag GmbH Publ. Co. Ltd.

Marseguerra M., Zio E., Cadini F. (2002) 'Biased Monte Carlo Unavailability Analysis for Systems with Time Dependant Failure Rates'. *Reliabilty Engineering and System Safety.* 76, 11-17.

Marseguerra M., Zio E., Devooght J., Labeau P.E. (1998) 'A Concept Paper on Dynamic Reliability via Monte Carlo Simulation'. *Mathematics and Computers in Simulation.* 47, 371-382.

Marseguerra M., Zio E., Podofillini L. (2005) 'First-order Differential Sensitivity Analysis of a Nuclear Safety system by Monte Carlo Simulation'. *Reliabilty Engineering and System Safety.* 90, 162-168.

McGrath E.J., Irving D.C. (1975) 'Techniques for Efficient Monte Carlo Simulation'. *ORNL-RSIC-38,* 2.

Mokashi A.J., Wang J., Vermar A.K. (2002) 'A Study of Reliability-Centred Maintenance in Marine Operations'. *Marine Policy.* 26, 325-335.

Moubray J. (1997) 'Reliability Centred Maintenance'. 2nd Edition, Industrial Press.

Nakajima S. (1988) 'Introduction to TPM'. Productivity Press, Portland, OR.

Nowlan F.S., Heap H.F. (1978) 'Reliability Centred Maintenance'. United Airlines Publications, San Francisco, CA.

OREDA (2002) 'OREDA: Offshore Reliability Data Handbook'. SINTEF industrial.

Pillay A., Wang J. (2003) 'Technology and Safety of Marine Systems'. Elsevier Science Publishers Ltd.

Pillay A., Wang J., Wall A.D., Ruxton T. (2001) 'Maintenance Study of Fishing Vessel Equipment Using Delay-Time Analysis'. *Journal of Quality in Maintenance Engineering.* 7(2), 118-127.

Pillay A., Wang J., Wall A., Ruxton T. (2001) 'Optimal Inspection Period for Fishing Vessel Equipment: A Cost and Downtime Model Using Delay-Time Analysis'. *Marine Technology, SNAME.* 38(2), 122-129.

Pinelton L., Nagarur N., Van Puyvelde F. (1999) 'Case Study: RCM – Yes, No or Maybe?'. *Journal of Quality in Maintenance Engineering.* 5(3), 182-191.

Powers G.J., Tompkins F.C. (1974) 'Fault Tree Analysis for Chemical Processes'. *AIChE Journal.* 20(2), 376-386.

Qi X., Chen T., Tu F. (1999) 'Scheduling Maintenance on a Single Machine'. *Journal of the Operational Research Society.* 50, 1071-1078.

Rich N. (1999) 'TPM: The Lean Approach'. Tudor Business Publishing/Prentice-Hall, London.

Rubinstein R.Y. (1981) 'Simulation and the Monte Carlo Method'. John Wiley & Sons, inc.

Saaty T.L. (1980) 'The Analytic Hierarchy Process'. McGraw Hill International, New York.

Saaty T.L., Forman E.H. (1992) 'The Hierarchon'. RWS Publications.

Saaty T.L., Ozdemir M. (2005) ' The Encyclicon'. RWS Publications.

Saaty T.L. (2007) 'Decisions, Structure and Judgement and Natural Law'. *ISAHP 2007.*

Saaty T.L. (2008) ' Decision Making with the Analytic Hierarchy Process'. *International Journal of Services Sciences.* 1(1), 83- 98.

Saaty T.L. (2008) 'Relative Measurement and its Generalization in Decision Making, why Pairwise Comparisons are Central in Mathematics for the Measurement of Intangible Factors, The Analytic Hierarchy/Network Process'. *Statistics and Operations Research.* 102(2), 251-318.

Salem S.L. (1977) 'A New Methodology for the Computer Aided Construction of Fault Tree'. *Ann Nucl Energy.* 4, 417-433.

SEC(2005)791 'Impact Assessment Guidlines', *European Commission.*

Shang J.S., Tjader Y., Ding Y. (2004) 'A Unified Framework for Multi-Criteria Evaluation of Transport Projects'. *IEEE Transactions on Engineering Management.* 51(3), 300-313.

Sii H.S., Ruxton T., Wang J. (2001) 'A Fuzzy Logic Based Approach to Qualitative Safety Modelling for Marine Systems'. *Reliability Engineering & System Safety.* 73(1), 19-34.

Sii H.S., Wang J. (2003) 'A Design-Decision Support Framework for the Evaluation of Design Options/Proposals using a Composite Structure Methodology based on The Approximate Reasoning Approach and Evidential Reasoning Method'. *Proceedings of the Institution of Mechanical Engineers.* E 17(1), 59-76.

Smith A.M. (1993) 'Reliability Centred Maintenance'. New York: McGraw-Hill, Inc.

Soncini G. (1996) 'PMS on ships: Dream or Reality?'. *Institute of Marine Engineering Safe and Efficient Ships: New Approaches for Design Operation and Maintenance – ICMES96.* 105-109.

Tsang A. (1995) 'Condition Based Maintenance: Tools ad Decision Making'. *Journal of Quality in Maintenance Engineering.* 1(3), 3-17.

Ugboma C., Ugboma O., Ogwude I.C. (2006) 'An Analytic Hierarchy Process (AHP) Approach to Port Selection Decisions – Empirical Evidence from Nigerian Ports'. *Maritime Economics and Logistics.* 8(3), 251-266.

Wang J. (2006) 'Maritime Risk Assesment and its Curent Status'. *Quality and Reliability Engineering International.* 22, 3-19.

Wang C.H., Hwang S.L. (2004) 'A Stochastic Maintenance Management Model with Recovery Factor'. *Journal of Quality in Maintenance Engineering.* 10(2), 154-164.

Wang W., Majid H.B.A. (2000) 'Reliability Data Analysis and Modelling of Offshore Oil Platform Plant'. *Journal of Quality in Maintenance Engineering.* 6(4), 287-295.

Wang W., Scarf P.A., Smith M.A.J. (2000) 'On the Application of a Model of Condition-Based Maintenance'. *Journal of the Operational Research Society.* 51, 1218-1227.

Whitaker L., Samaniego F. (1989) 'Estimating the Reliability of Systems Subject to Imperfect Repair'. *Journal of the American Statistical Association.* 84(405) 301-309.

Woods D.D. (1984) 'Some Results on Operator Performance in Emergency Events'. *Institute of Chemical Engineers Symposium Series.* 90, 13-21.

Zio E. (1995) 'Biasing the Transition Probabilities in Direct Monte Carlo'. *Reliability Engineering and System Safety.* 47, 59-63.

Zio, E. (2007) 'An Introduction to the Basics of Reliability and Risk Analysis, Series in Quality, Reliability and Engineering Statistics'. 13, World Scientific.

Zio E., Baraldi P., Patelli E. (2004) 'Application of Monte Carlo Simulation for the Evaluation of the Production Availability of an Offshore Installation'. *Doc. code W75.5POLIMI-MOFFSHORE.52(1), Task 7.5*, www.mar.istutl.pt/saferelnet.

Zio E., Marella M., Podofillini L. (2007) 'A Monte Carlo Simulation Approach to the Availability Assessment of Multi-State Systems with Operational Dependancies'. *Relaibility Engineering and System Safety.* 92, 871-882.

Zio E., Sansavini G. (2007) 'Reliability Analysis of Multi-State Networks: Making Monte Carlo Simulation Feasible Through Biasing'. *Computational Models of Risks to Infrastructure.* 13, 121-132.

Appendices

Appendix 1 – FORTRAN Code Complex Model

```
PROGRAM COOLING_3
!
!PURPOSE:
!THE PURPOSE OF THE PROGRAM IS TO PERFORM A RAMS ANALYSIS ON A COMPLEX
!COOLING SYSTEM. THE SYSTEM IS SUCH THAT A MINIMUM OF TWO PUMPS MUST BE
!FUNCTIONING IN ORDER FOR THE SYSTEM TO FUFILL THE COOLING DEMANDS, IT
!IS ESSENTIALLY AN UPDATED VERSION OF AN EARLIER PROGRAM THE MAIN
!DIFFERENCE IS THAT IN THIS VERSION THE DIFFERENT COMPONENTS HAVE
!DIFFERENT DETERMINISTIC REPAIR TIMES. THE DATE CONCERNING THE ORIGINAL
!CODE RELATES TO THE PREVIOUS PROGRAM.
!
!DATE          PROGRAMMER          REVISIONS
!======        ===========         =============
!25/07/08      A.CUNNINGHAM        ORIGINAL CODE
!26/08/08      A.CUNNINGHAM        INDIVID. REPAIR TIMES
!
!DECLARE AND INITIALISE THE VARIABLES USED IN THE PROGRAM

INTEGER::F
INTEGER::FV1
INTEGER::FV2
INTEGER::FV3
INTEGER::FP1
INTEGER::FP2
INTEGER::FP3
INTEGER::FPC1
INTEGER::FPC2
INTEGER::N
INTEGER::B
DOUBLE PRECISION::RT
DOUBLE PRECISION::RC
DOUBLE PRECISION::COMP1
DOUBLE PRECISION::COMP2
DOUBLE PRECISION::COMP3
DOUBLE PRECISION::COMP4
DOUBLE PRECISION::COMP5
DOUBLE PRECISION::COMP6
DOUBLE PRECISION::COMP7
DOUBLE PRECISION::COMP8
REAL::T0
REAL::T
REAL::TM
REAL::DTV1
REAL::DTV2
REAL::DTV3
REAL::DTP1
REAL::DTP2
REAL::DTP3
REAL::DTPC1
REAL::DTPC2
REAL::TAUV=10.
REAL::TAUP=10.
REAL::TAUPC=10.
```

```
DOUBLE PRECISION::lambdav1=0.0000437
DOUBLE PRECISION::lambdav2=0.0000437
DOUBLE PRECISION::lambdav3=0.0000437
DOUBLE PRECISION::lambdap1=0.001277
DOUBLE PRECISION::lambdap2=0.001277
DOUBLE PRECISION::lambdap3=0.001277
DOUBLE PRECISION::lambdapc1=0.00003975
DOUBLE PRECISION::lambdapc2=0.00003975
REAL::LAMBDAS

PARAMETER(N=10.**9)
PARAMETER(TM=532.)




HISTORIES_LOOP:DO i=1,N
!WRITE(*,*)lambdav1,lambdap1,lambdapc1

T0=0
B=1
DTV1=0
DTV2=0
DTV3=0
DTP1=0
DTP2=0
DTP3=0
DTPC1=0
DTPC2=0
INNER:DO

SYSST:IF((B==1).OR.(B==2).OR.(B==10).OR.(B==11).OR.(B==19).OR.(B==20))TH
EN
        LAMBDAS=lambdav1+lambdav3+lambdap1+lambdap3+lambdapc1
        COMP1=(lambdav1/LAMBDAS)
        COMP2=(2*COMP1)
        COMP3=((lambdav1+lambdav3+lambdap1)/LAMBDAS)
        COMP4=((lambdav1+lambdav3+lambdap1+lambdap3)/LAMBDAS)
        COMP5=((lambdav1+lambdav3+lambdap1+lambdap3+lambdapc1)/LAMBDAS)
       ELSE IF((B==3).OR.(B==12).OR.(B==21))THEN
        LAMBDAS=lambdav1+lambdav3+lambdap1+lambdap3+lambdapc2
        COMP1=(lambdav1/LAMBDAS)
        COMP2=(2*COMP1)
        COMP3=((lambdav1+lambdav3+lambdap1)/LAMBDAS)
        COMP4=((lambdav1+lambdav3+lambdap1+lambdap3)/LAMBDAS)
        COMP5=((lambdav1+lambdav3+lambdap1+lambdap3+lambdapc2)/LAMBDAS)
       ELSE IF((B==4).OR.(B==5).OR.(B==13).OR.(B==14))THEN
        LAMBDAS=lambdav1+lambdav2+lambdap1+lambdap2+lambdapc1
        LAMBDAS=lambdav1+lambdav2+lambdap1+lambdap2+lambdapc1
        COMP1=(lambdav1/LAMBDAS)
        COMP2=(2*COMP1)
        COMP3=((lambdav1+lambdav2+lambdap1)/LAMBDAS)
        COMP4=((lambdav1+lambdav2+lambdap1+lambdap2)/LAMBDAS)
        COMP5=((lambdav1+lambdav2+lambdap1+lambdap2+lambdapc1)/LAMBDAS)
       ELSE IF((B==6).OR.(B==15))THEN
        LAMBDAS=lambdav1+lambdav2+lambdap1+lambdap2+lambdapc2
        COMP1=(lambdav1/LAMBDAS)
        COMP2=(2*COMP1)
```

```
         COMP3=((lambdav1+lambdav2+lambdap1)/LAMBDAS)
         COMP4=((lambdav1+lambdav2+lambdap1+lambdap2)/LAMBDAS)
         COMP5=((lambdav1+lambdav2+lambdap1+lambdap2+lambdapc2)/LAMBDAS)
        ELSE IF((B==7).OR.(B==8).OR.(B==16).OR.(B==17))THEN
         LAMBDAS=lambdav2+lambdav3+lambdap2+lambdap3+lambdapc1
         LAMBDAS=lambdav2+lambdav3+lambdap2+lambdap3+lambdapc1
         COMP1=(lambdav2/LAMBDAS)
         COMP2=(2*COMP1)
         COMP3=((lambdav2+lambdav3+lambdap2)/LAMBDAS)
         COMP4=((lambdav2+lambdav3+lambdap2+lambdap3)/LAMBDAS)
         COMP5=((lambdav2+lambdav3+lambdap2+lambdap3+lambdapc1)/LAMBDAS)
        ELSE IF((B==9).OR.(B==18))THEN
         LAMBDAS=lambdav2+lambdav3+lambdap2+lambdap3+lambdapc2
         COMP1=(lambdav2/LAMBDAS)
         COMP2=(2*COMP1)
         COMP3=((lambdav2+lambdav3+lambdap2)/LAMBDAS)
         COMP4=((lambdav2+lambdav3+lambdap2+lambdap3)/LAMBDAS)
         COMP5=((lambdav2+lambdav3+lambdap2+lambdap3+lambdapc2)/LAMBDAS)
        ELSE
         !INVALID SYSTEM STATE
         WRITE(*,*)'INVALID SYSTEM STATE'
        END IF SYSST

!WRITE(*,*)B

CALL RANDOM_NUMBER(RT)
T=T0-(1./LAMBDAS)*LOG(1-(RT))
MISS:IF(T<TM)THEN
        CALL RANDOM_NUMBER(RC)
NO_1:IF(B==1)THEN
NO_2:IF (RC.LE.COMP1) THEN
        !VALVE 1 HAS FAILED
        FV1=FV1+1
        T0=T0+T
        DTV1=T
        B=7
       ELSE IF (RC.LE.COMP2)THEN
        !VALVE 3 HAS FAILED
        FV3=FV3+1
        T0=T0+T
        DTV3=T
        B=13
       ELSE IF(RC.LE.COMP3)THEN
        !PUMP 1 HAS FAILED
        FP1=FP1+1
        T0=T0+T
        DTP1=T
        B=16
       ELSE IF(RC.LE.COMP4)THEN
        !pump 3 has failed
        FP3=FP3+1
        T0=T0+T
        DTP3=T
        B=4
       ELSE
        !PLATE COOLER 1 HAS FAILED
        T0=T0+T
        DTPC1=T
        B=3
       END IF NO_2
       ELSE IF(B==2)THEN
```

```
NO_3:IF(RC.LE.COMP1)THEN
      !VALVE 1 HAS FAILED
      FV1=FV1+1
      T0=T0+T
      DTV1=T
NO_23:IF(T<DTPC2+TAUPC)THEN
      !VALVE1 AND PLATE COOLER 2 HAVE FAILED
      FPC2=FPC2+1
      DTPC2=T
      B=8
      ELSE !T>DTPC2+TAUPC
      !VALVE 1 FAILED PLATE COOLER 2 IS REPAIRED
      B=7
      END IF NO_23
      ELSE IF(RC.LE.COMP2)THEN
      !VALVE 3 HAS FAILED
      FV3=FV3+1
      T0=T0+T
      DTV3=T
NO_24:IF(T<DTPC2+TAUPC)THEN
      !VALVE 3 AND PLATE COOLER 2 HAVE FAILED
      FPC2=FPC2+1
      DTPC2=T
      B=14
      ELSE !T>DTPC2+TAU
      !VALVE 3 FAILED PLATE COOLER 2 IS REPAIRED
      B=13
      END IF NO_24
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 1 HAS FAILED
      FP1=FP1+1
      T0=T0+T
      DTP1=T
NO_25:IF(T<DTPC2+TAUPC)THEN
      !PUMP 1 AND PLATE COOLER 2 HAVE FAILED
      FPC2=FPC2+1
      DTPC2=T
      B=17
      ELSE !T>DTPC2+TAUPC
      !PUMP 1 FAILED PLATE COOLER 2 IS REPAIRED
      B=16
      END IF NO_25
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 3 HAS FAILED
      FP3=FP3+1
      T0=T0+T
      DTP3=T
NO_26:IF(T<DTPC2+TAUPC)THEN
      !PUMP 3 AND PLATE COOLER 2 HAVE FAILED
      FPC2=FPC2+1
      DTPC2=T
      B=5
      ELSE !T>DTPC2+TAUPC
      !PUMP 3 FAILED PLATE COOLER 2 IS REPAIRED
      B=4
      END IF NO_26
      ELSE
      !PLATE COOLER 1 HAS FAILED
      FPC1=FPC1+1
      T0=T0+T
      DTPC1=T
```

```
NO_27:IF(T<DTPC2+TAUPC)THEN
      !PLATE COOLER 1 AND PLATE COOLER 2 HAVE FAILED
      !B=28
      FPC2=F
      F=F+1
      T0=T0+TAUPC
      B=1 !SYSTEM RETURNED TO NOMINAL CONDITION
      ELSE !T>DTPC2+TAUPC
      !PLATE COOLER 1 FAILED PLATE COOLER 2 IS REPAIRED
      B=3
      END IF NO_27
      END IF NO_3

ELSE IF(B==3)THEN
NO_4:IF(RC.LE.COMP1)THEN
      !VALVE 1 HAS FAILED
      FV1=FV1+1
      T0=T0+T
      DTV1=T
NO_28:IF(T<DTPC1+TAUPC)THEN
      !VALVE1 AND PLATE COOLER 1 HAVE FAILED
      FPC1=FPC1+1
      DTPC1=T
      B=9
      ELSE !T>DTPC1+TAUPC
      !VALVE 1 FAILED PLATE COOLER 1 IS REPAIRED
      B=7
      END IF NO_28
      ELSE IF(RC.LE.COMP2)THEN
      !VALVE 3 HAS FAILED
      FV3=FV3+1
      T0=T0+T
      DTV3=T
NO_29:IF(T<DTPC1+TAUPC)THEN
      !VALVE 3 AND PLATE COOLER 1 HAVE FAILED
      FPC1=FPC1+1
      DTPC1=T
      B=15
      ELSE !T>DTPC1+TAUPC
      !VALVE 3 FAILED PLATE COOLER 1 IS REPAIRED
      B=13
      END IF NO_29
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 1 HAS FAILED
      FP1=FP1+1
      T0=T0+T
      DTP1=T
NO_30:IF(T<DTPC1+TAUPC)THEN
      !PUMP 1 AND PLATE COOLER 1 HAVE FAILED
      FPC1=FPC1+1
      DTPC1=T
      B=18
      ELSE !T>DTPC1+TAUPC
      !PUMP 1 FAILED PLATE COOLER 1 IS REPAIRED
      B=16
      END IF NO_30
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 3 HAS FAILED
      FP3=FP3+1
      T0=T0+T
      DTP3=T
```

```
NO_31:IF(T<DTPC1+TAUPC)THEN
      !PUMP 3 AND PLATE COOLER 1 HAVE FAILED
      FPC1=FPC1+1
      DTPC1=T
      B=6
      ELSE !T>DTPC2+TAUPC
      !PUMP 3 FAILED PLATE COOLER 1 IS REPAIRED
      B=4
      END IF NO_31
      ELSE
      !PLATE COOLER 2 HAS FAILED
      FPC2=FPC2+1
      T0=T0+T
      DTPC2=T
NO_32:IF(T<DTPC1+TAUPC)THEN
      !PLATE COOLER 1 AND PLATE COOLER 2 HAVE FAILED
      !B=28
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE !T>DTPC1+TAUPC
      !PLATE COOLER 2 FAILED PLATE COOLER 1 IS REPAIRED
      B=2
      END IF NO_32
      END IF NO_4

ELSE IF(B==4)THEN
NO_5:IF(RC.LE.COMP1)THEN
      !VALVE 1 HAS FAILED
      FV1=FV1+1
      T0=T0+T
      DTV1=T
NO_33:IF(T<DTP3+TAUP)THEN
      !VALVE1 AND PUMP 3 HAVE FAILED
      !B=30
      FP3=FP3+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !T>DTP3+TAUP
      !VALVE 1 FAILED PUMP 3 IS REPAIRED
      B=7
      END IF NO_33
      ELSE IF(RC.LE.COMP2)THEN
      !VALVE 2 HAS FAILED
      FV2=FV2+1
      T0=T0+T
      DTV2=T
NO_34:IF(T<DTP3+TAUP)THEN
      !VALVE 2 AND PUMP 3 HAVE FAILED
      !B=32
      FP3=FP3+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !T>DTP3+TAUP
      !VALVE 2 FAILED PUMP 3 IS REPAIRED
      B=10
      END IF NO_34
      ELSE IF(RC.LE.COMP3)THEN
```

```
      !PUMP 1 HAS FAILED
      FP1=FP1+1
      T0=T0+T
      DTP1=T
NO_35:IF(T<DTP3+TAUP)THEN
      !PUMP 1 AND PUMP 3 HAVE FAILED
      !B=26
      FP3=FP3+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !T>DTP3+TAU
      !PUMP 1 FAILED PUMP 3 IS REPAIRED
      B=16
      END IF NO_35
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 2 HAS FAILED
      FP2=FP2+1
      T0=T0+T
      DTP2=T
NO_36:IF(T<DTP3+TAUP)THEN
      !PUMP 3 AND PUMP 2 HAVE FAILED
      !B=27
      FP3=FP3+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !T>DTP3+TAUP
      !PUMP 2 FAILED PUMP 3 IS REPAIRED
      B=19
      END IF NO_36
      ELSE
      !PLATE COOLER 1 HAS FAILED
      FPC1=FPC1+1
      T0=T0+T
      DTPC1=T
NO_37:IF(T<DTP3+TAUP)THEN
      !PLATE COOLER 1 AND PUMP 3 HAVE FAILED
      FP3=FP3+1
      DTP3=T
      B=6
      ELSE !T>DTP3+TAUP
      !PLATE COOLER 1 FAILED PUMP 3 IS REPAIRED
      B=3
      END IF NO_37
      END IF NO_5

ELSE IF(B==5)THEN
NO_6:IF(RC.LE.COMP1)THEN
      !VALVE 1 HAS FAILED
      FV1=FV1+1
      T0=T0+T
      DTV1=T
NO_38:IF((T<DTP3+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 1 AND PUMP 3 AND PLATE COOLER 2 HAVE FAILED
      !B=30
      FP3=FP3+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
```

```
      ELSE IF((T>DTP3+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 1 AND PLATE COOLER 2 HAVE FAILED PUMP 3 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=8
      ELSE IF((T<DTP3+TAUP).AND.(T>DTPC2+TAUPC))THEN
      !VALVE 1 HAS FAILED PUMP 3 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=30
      FP3=FP3+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTP3+TAUP).AND.(T>DTPC2+TAUPC))
      !VALVE 1 HAS FAILED PUMP 3 AND PLATE COOLER 2 ARE REPAIRED
      B=7
      END IF NO_38
      ELSE IF(RC.LE.COMP2)THEN
      !VALVE 2 HAS FAILED
      FV2=FV2+1
      T0=T0+T
      DTV2=T
NO_39:IF((T<DTP3+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 2 AND PUMP 3 AND PLATE COOLER 2 HAVE FAILED
      !B=32
      FP3=FP3+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTP3+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 2 AND PLATE COOLER 2 HAVE FAILED PUMP 3 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=11
      ELSE IF((T<DTP3+TAUP).AND.(T>DTPC2+TAUPC))THEN
      !VALVE 2 HAS FAILED PUMP 3 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=32
      FP3=FP3+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTP3+TAU).AND.(T>DTPC2+TAU))
      !VALVE 2 HAS FAILED PUMP 3 AND PLATE COOLER 2 ARE REPAIRED
      B=10
      END IF NO_39
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 1 HAS FAILED
      FP1=FP1+1
      T0=T0+T
      DTP1=T
NO_40:IF((T<DTP3+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !PUMP 1 AND PUMP 3 AND PLATE COOLER 2 HAVE FAILED
      !B=26
      FP3=FP3+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTP3+TAUP).AND.(T<DTPC2+TAUPC))THEN
```

```
      !PUMP 1 AND PLATE COOLER 2 HAVE FAILED PUMP 3 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=17
      ELSE IF((T<DTP3+TAUP).AND.(T>DTPC2+TAUPC))THEN
      !PUMP 1 HAS FAILED PUMP 3 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=26
      FP3=FP3+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTP3+TAUP).AND.(T>DTPC2+TAUPC))
      !PUMP 1 HAS FAILED PUMP 3 AND PLATE COOLER 2 ARE REPAIRED
      B=16
      END IF NO_40
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 2 HAS FAILED
      FP2=FP2+1
      T0=T0+T
      DTP2=T
NO_41:IF((T<DTP3+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !PUMP 2 AND PUMP 3 AND PLATE COOLER 2 HAVE FAILED
      !B=25
      FP3=FP3+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTP3+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !PUMP 2 AND PLATE COOLER 2 HAVE FAILED PUMP 3 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=20
      ELSE IF((T<DTP3+TAUP).AND.(T>DTPC2+TAUPC))THEN
      !PUMP 2 HAS FAILED PUMP 3 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=25
      FP3=FP3+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTP3+TAUP).AND.(T>DTPC2+TAUPC))
      !PUMP 2 HAS FAILED PUMP 3 AND PLATE COOLER 2 ARE REPAIRED
      B=19
      END IF NO_41
      ELSE
      !PLATE COOLER 1 HAS FAILED
      FPC1=FPC1+1
      T0=T0+T
      DTPC1=T
NO_42:IF((T<DTP3+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !PLATE COOLER 1 AND PUMP 3 AND PLATE COOLER 2 HAVE FAILED
      !B=28
      FP3=FP3+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTP3+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !PLATE COOLER 1 AND PLATE COOLER 2 HAVE FAILED PUMP 3 IS REPAIRED
```

```
      !B=28
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T<DTP3+TAUP).AND.(T>DTPC2+TAUPC))THEN
      !PLATE COOLER 1 HAS FAILED PUMP 3 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      FP3=FP3+1
      DTP3=T
      B=6
      ELSE !((T>DTP3+TAUP).AND.(T>DTPC2+TAUPC))
      !PLATE COOLER 1 HAS FAILED PUMP 3 AND PLATE COOLER 2 ARE REPAIRED
      B=3
      END IF NO_42
      END IF NO_6

ELSE IF(B==6)THEN
NO_7:IF(RC.LE.COMP1)THEN
      !VALVE 1 HAS FAILED
      FV1=FV1+1
      T0=T0+T
      DTV1=T
NO_43:IF((T<DTP3+TAUP).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 1 AND PUMP 3 AND PLATE COOLER 1 HAVE FAILED
      !B=30
      FP3=FP3+1
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTP3+TAUP).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 1 AND PLATE COOLER 1 HAVE FAILED PUMP 3 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=9
      ELSE IF((T<DTP3+TAUP).AND.(T>DTPC1+TAUPC))THEN
      !VALVE 1 HAS FAILED PUMP 3 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=30
      FP3=FP3+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTP3+TAUP).AND.(T>DTPC1+TAUPC))
      !VALVE 1 HAS FAILED PUMP 3 AND PLATE COOLER 1 ARE REPAIRED
      B=7
      END IF NO_43
      ELSE IF(RC.LE.COMP2)THEN
      !VALVE 2 HAS FAILED
      FV2=FV2+1
      T0=T0+T
      DTV2=T
NO_44:IF((T<DTP3+TAUP).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 2 AND PUMP 3 AND PLATE COOLER 1 HAVE FAILED
      !B=32
      FP3=FP3+1
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
```

```
      ELSE IF((T>DTP3+TAUP).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 2 AND PLATE COOLER 1 HAVE FAILED PUMP 3 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=12
      ELSE IF((T<DTP3+TAUP).AND.(T>DTPC1+TAUPC))THEN
      !VALVE 2 HAS FAILED PUMP 3 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=32
      FP3=FP3+1
      F=F+1
      TO=TO+TAUP
      B=1
      ELSE !((T>DTP3+TAUP).AND.(T>DTPC1+TAUPC))
      !VALVE 1 HAS FAILED PUMP 3 AND PLATE COOLER 1 ARE REPAIRED
      B=10
      END IF NO_44
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 1 HAS FAILED
      FP1=FP1+1
      TO=TO+T
      DTP1=T
NO_45:IF((T<DTP3+TAUP).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 1 AND PUMP 3 AND PLATE COOLER 1 HAVE FAILED
      !B=26
      FP3=FP3+1
      FPC1=FPC1+1
      F=F+1
      TO=TO+TAUPC
      B=1
      ELSE IF((T>DTP3+TAUP).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 1 AND PLATE COOLER 1 HAVE FAILED PUMP 3 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=18
      ELSE IF((T<DTP3+TAUP).AND.(T>DTPC1+TAUPC))THEN
      !PUMP 1 HAS FAILED PUMP 3 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=26
      FP3=FP3+1
      F=F+1
      TO=TO+TAUP
      B=1
      ELSE !((T>DTP3+TAU).AND.(T>DTPC1+TAU))
      !PUMP 1 HAS FAILED PUMP 3 AND PLATE COOLER 1 ARE REPAIRED
      B=16
      END IF NO_45
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 2 HAS FAILED
      FP2=FP2+1
      TO=TO+T
      DTP2=T
NO_46:IF((T<DTP3+TAUP).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 2 AND PUMP 3 AND PLATE COOLER 1 HAVE FAILED
      !B=27
      FP3=FP3+1
      FPC1=FPC1+1
      F=F+1
      TO=TO+TAUPC
      B=1
      ELSE IF((T>DTP3+TAUP).AND.(T<DTPC1+TAUPC))THEN
```

```
       !PUMP 2 AND PLATE COOLER 1 HAVE FAILED PUMP 3 IS REPAIRED
       FPC1=FPC1+1
       DTPC1=T
       B=21
       ELSE IF((T<DTP3+TAUP).AND.(T>DTPC1+TAUPC))THEN
       !PUMP 2 HAS FAILED PUMP 3 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
       !B=27
       FP3=FP3+1
       F=F+1
       T0=T0+TAUP
       B=1
       ELSE !((T>DTP3+TAUP).AND.(T>DTPC1+TAUPC))
       !PUMP 2 HAS FAILED PUMP 3 AND PLATE COOLER 1 ARE REPAIRED
       B=20
       END IF NO_46
       ELSE
       !PLATE COOLER 2 HAS FAILED
       FPC2=FPC2+1
       T0=T0+T
       DTPC2=T
NO_47:IF((T<DTP3+TAUP).AND.(T<DTPC1+TAUPC))THEN
       !PLATE COOLER 1 AND PUMP 3 AND PLATE COOLER 2 HAVE FAILED
       !B=28
       FPC1=FPC1+1
       FP3=FP3+1
       F=F+1
       T0=T0+TAUPC
       B=1
       ELSE IF((T>DTP3+TAUP).AND.(T<DTPC1+TAUPC))THEN
       !PLATE COOLER 1 AND PLATE COOLER 2 HAVE FAILED PUMP 3 IS REPAIRED
       !B=28
       FPC1=FPC1+1
       F=F+1
       T0=T0+TAUPC
       B=1
       ELSE IF((T<DTP3+TAUP).AND.(T>DTPC1+TAUPC))THEN
       !PLATE COOLER 2 HAS FAILED PUMP 3 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
       FP3=FP3+1
       DTP3=T
       B=5
       ELSE !((T>DTP3+TAUP).AND.(T>DTPC1+TAUPC))
       !PLATE COOLER 2 HAS FAILED PUMP 3 AND PLATE COOLER 1 ARE REPAIRED
       B=2
       END IF NO_47
       END IF NO_7

ELSE IF(B==7)THEN
NO_8:IF(RC.LE.COMP1)THEN
       !VALVE 2 HAS FAILED
       FV2=FV2+1
       T0=T0+T
       DTV2=T
NO_48:IF(T<DTV1+TAUV)THEN
       !VALVE 2 AND VALVE 1 HAVE FAILED
       !B=22
       FV1=FV1+1
       F=F+1
       T0=T0+TAUV
       B=1
```

```
      ELSE !T>DTV1+TAUV
      !VALVE 2 FAILED VALVE 1 IS REPAIRED
      B=10
      END IF NO_48
      ELSE IF(RC.LE.COMP2)THEN
      !VALVE 3 HAS FAILED
      FV3=FV3+1
      TO=TO+T
      DTV3=T
NO_49:IF(T<DTV1+TAUV)THEN
      !VALVE 3 AND VALVE 1 HAVE FAILED
      !B=23
      FV1=FV1+1
      F=F+1
      TO=TO+TAUV
      B=1
      ELSE !T>DTV1+TAUV
      !VALVE 3 FAILED VALVE 1 IS REPAIRED
      B=13
      END IF NO_49
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 2 HAS FAILED
      FP2=FP2+1
      TO=TO+T
      DTP2=T
NO_50:IF(T<DTV1+TAUV)THEN
      !PUMP 2 AND VALVE 1 HAVE FAILED
      !B=29
      FV1=FV1+1
      F=F+1
      TO=TO+TAUP
      B=1
      ELSE !T>DTV1+TAUV
      !PUMP 2 FAILED VALVE 1 IS REPAIRED
      B=19
      END IF NO_50
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 3 HAS FAILED
      FP3=FP3+1
      TO=TO+T
      DTP3=T
NO_51:IF(T<DTV1+TAUV)THEN
      !PUMP 3 AND VALVE 1 HAVE FAILED
      !B=30
      FV1=FV1+1
      F=F+1
      TO=TO+TAUP
      B=1
      ELSE !T>DTV1+TAUV
      !PUMP 3 FAILED VALVE 1 IS REPAIRED
      B=4
      END IF NO_51
      ELSE
      !PLATE COOLER 1 HAS FAILED
      FPC1=FPC1+1
      TO=TO+T
      DTPC1=T
NO_52:IF(T<DTV1+TAUV)THEN
      !PLATE COOLER 1 AND VALVE 1 HAVE FAILED
      FV1=FV1+1
      DTV1=T
```

```
      B=9
      ELSE !T>DTV1+TAUV
      !PLATE COOLER 1 FAILED VALVE 1 IS REPAIRED
      B=3
      END IF NO_52
      END IF NO_8

ELSE IF(B==8)THEN
NO_9:IF(RC.LE.COMP1)THEN
      !VALVE 2 HAS FAILED
      FV2=FV2+1
      T0=T0+T
      DTV2=T
NO_53:IF((T<DTV1+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 2 AND VALVE 1 AND PLATE COOLER 2 HAVE FAILED
      !B=22
      FV1=FV1+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV1+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 2 AND PLATE COOLER 2 HAVE FAILED VALVE 1 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=11
      ELSE IF((T<DTV1+TAUV).AND.(T>DTPC2+TAUPC))THEN
      !VALVE 2 HAS FAILED VALVE 1 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=22
      FV1=FV1+1
      F=F+1
      T0=T0+TAUV
      B=1
      ELSE !((T>DTV1+TAUV).AND.(T>DTPC2+TAUPC))
      !VALVE 2 HAS FAILED VALVE 1 AND PLATE COOLER 2 ARE REPAIRED
      B=10
      END IF NO_53
      ELSE IF(RC.LE.COMP2)THEN
      !VALVE 3 HAS FAILED
      FV3=FV3+1
      T0=T0+T
      DTV3=T
NO_54:IF((T<DTV1+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 3 AND VALVE 1 AND PLATE COOLER 2 HAVE FAILED
      !B=23
      FV1=FV1+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV1+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 3 AND PLATE COOLER 2 HAVE FAILED VALVE 1 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=14
      ELSE IF((T<DTV1+TAUV).AND.(T>DTPC2+TAUPC))THEN
      !VALVE 3 HAS FAILED VALVE 1 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=23
      FV1=FV1+1
```

```
      F=F+1
      T0=T0+TAUV
      B=1
      ELSE !((T>DTV1+TAUV).AND.(T>DTPC2+TAUPC))
      !VALVE 3 HAS FAILED VALVE 1 AND PLATE COOLER 2 ARE REPAIRED
      B=13
      END IF NO_54
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 2 HAS FAILED
      FP2=FP2+1
      T0=T0+T
      DTP2=T
NO_55:IF((T<DTV1+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !PUMP 2 AND VALVE 1 AND PLATE COOLER 2 HAVE FAILED
      !B=29
      FV1=FV1+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV1+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !PUMP 2 AND PLATE COOLER 2 HAVE FAILED VALVE 1 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=20
      ELSE IF((T<DTV1+TAUV).AND.(T>DTPC2+TAUPC))THEN
      !PUMP 2 HAS FAILED VALVE 1 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=29
      FV1=FV1+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTV1+TAUV).AND.(T>DTPC2+TAUPC))
      !PUMP 2 HAS FAILED VALVE 1 AND PLATE COOLER 2 ARE REPAIRED
      B=19
      END IF NO_55
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 3 HAS FAILED
      FP3=FP3+1
      T0=T0+T
      DTP3=T
NO_56:IF((T<DTV1+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !PUMP 3 AND VALVE 1 AND PLATE COOLER 2 HAVE FAILED
      !B=30
      FV1=FV1+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV1+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !PUMP 3 AND PLATE COOLER 2 HAVE FAILED VALVE 1 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=5
      ELSE IF((T<DTV1+TAUV).AND.(T>DTPC2+TAUPC))THEN
      !PUMP 3 HAS FAILED VALVE 1 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=30
      FV1=FV1+1
      F=F+1
```

```
      T0=T0+TAUP
      B=1
      ELSE !((T>DTV1+TAUV).AND.(T>DTPC2+TAUPC))
      !PUMP 3 HAS FAILED VALVE 1 AND PLATE COOLER 2 ARE REPAIRED
      B=4
      END IF NO_56
      ELSE
      !PLATE COOLER 1 HAS FAILED
      FPC1=FPC1+1
      T0=T0+T
      DTPC1=T
NO_57:IF((T<DTV1+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !PLATE COOLER 1 AND VALVE 1 AND PLATE COOLER 2 HAVE FAILED
      !B=28
      FV1=FV1+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV1+TAU).AND.(T<DTPC2+TAU))THEN
      !PLATE COOLER 1 AND PLATE COOLER 2 HAVE FAILED VALVE 1 IS REPAIRED
      !B=28
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T<DTV1+TAUV).AND.(T>DTPC2+TAUPC))THEN
      !PLATE COOLER 1 HAS FAILED VALVE 1 HAS FAILED AND PLATE COOLER 2
 IS REPAIRED
      FV1=FV1+1
      DTV1=T
      B=9
      ELSE !((T>DTV1+TAUV).AND.(T>DTPC2+TAUPC))
      !PLATE COOLER 1 HAS FAILED VALVE 1 AND PLATE COOLER 2 ARE REPAIRED
      B=3
      END IF NO_57
      END IF NO_9

 ELSE IF(B==9)THEN
      NO_10:IF(RC.LE.COMP1)THEN
      !VALVE 2 HAS FAILED
      FV2=FV2+1
      T0=T0+T
      DTV2=T
 NO_58:IF((T<DTV1+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 2 AND VALVE 1 AND PLATE COOLER 1 HAVE FAILED
      !B=22
      FV1=FV1+1
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV1+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 2 AND PLATE COOLER 1 HAVE FAILED VALVE 1 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=12
      ELSE IF((T<DTV1+TAUV).AND.(T>DTPC1+TAUPC))THEN
      !VALVE 2 HAS FAILED VALVE 1 HAS FAILED AND PLATE COOLER 1 IS
 REPAIRED
      !B=22
```

```
      FV1=FV1+1
      F=F+1
      TO=TO+TAUV
      B=1
      ELSE !((T>DTV1+TAUV).AND.(T>DTPC1+TAUPC))
      !VALVE 2 HAS FAILED VALVE 1 AND PLATE COOLER 1 ARE REPAIRED
      B=10
      END IF NO_58
      ELSE IF(RC.LE.COMP2)THEN
      !VALVE 3 HAS FAILED
      FV3=FV3+1
      TO=TO+T
      DTV3=T
NO_59:IF((T<DTV1+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 3 AND VALVE 1 AND PLATE COOLER 1 HAVE FAILED
      !B=23
      FV1=FV1+1
      FPC1=FPC1+1
      F=F+1
      TO=TO+TAUPC
      B=1
      ELSE IF((T>DTV1+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 3 AND PLATE COOLER 1 HAVE FAILED VALVE 1 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=15
      ELSE IF((T<DTV1+TAUV).AND.(T>DTPC1+TAUPC))THEN
      !VALVE 3 HAS FAILED VALVE 1 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=23
      PCV1=PCV1+1
      F=F+1
      TO=TO+TAUV
      B=1
      ELSE !((T>DTV1+TAUV).AND.(T>DTPC1+TAUPC))
      !VALVE 3 HAS FAILED VALVE 1 AND PLATE COOLER 1 ARE REPAIRED
      B=13
      END IF NO_59
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 2 HAS FAILED
      FP2=FP2+1
      TO=TO+T
      DTP2=T
NO_60:IF((T<DTV1+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 2 AND VALVE 1 AND PLATE COOLER 1 HAVE FAILED
      !B=29
      FV1=FV1+1
      FPC1=FPC1+1
      F=F+1
      TO=TO+TAUPC
      B=1
      ELSE IF((T>DTV1+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 2 AND PLATE COOLER 1 HAVE FAILED VALVE 1 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=21
      ELSE IF((T<DTV1+TAUV).AND.(T>DTPC1+TAUPC))THEN
      !PUMP 2 HAS FAILED VALVE 1 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=29
      FV1=FV1+1
```

```
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTV1+TAUV).AND.(T>DTPC1+TAUPC))
      !PUMP 2 HAS FAILED VALVE 1 AND PLATE COOLER 1 ARE REPAIRED
      B=19
      END IF NO_60
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 3 HAS FAILED
      FP3=FP3+1
      T0=T0+T
      DTP3=T
NO_61:IF((T<DTV1+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 3 AND VALVE 1 AND PLATE COOLER 1 HAVE FAILED
      !B=30
      FV1=FV1+1
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV1+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 3 AND PLATE COOLER 1 HAVE FAILED VALVE 1 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=6
      ELSE IF((T<DTV1+TAUV).AND.(T>DTPC1+TAUPC))THEN
      !PUMP 3 HAS FAILED VALVE 1 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=30
      FV1=FV1+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTV1+TAUV).AND.(T>DTPC1+TAUPC))
      !PUMP 3 HAS FAILED VALVE 1 AND PLATE COOLER 1 ARE REPAIRED
      B=4
      END IF NO_61
      ELSE
      !PLATE COOLER 2 HAS FAILED
      FPC2=FPC2+1
      T0=T0+T
      DTPC2=T
NO_62:IF((T<DTV1+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PLATE COOLER 2 AND VALVE 1 AND PLATE COOLER 1 HAVE FAILED
      !B=28
      FV1=FV1+1
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV1+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PLATE COOLER 2 AND PLATE COOLER 1 HAVE FAILED VALVE 1 IS REPAIRED
      !B=28
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T<DTV1+TAUV).AND.(T>DTPC1+TAUPC))THEN
      !PLATE COOLER 2 HAS FAILED VALVE 1 HAS FAILED AND PLATE COOLER 1
IS REPAIRED
      FV1=FV1+1
```

```
      DTV1=T
      B=8
      ELSE !((T>DTV1+TAUV).AND.(T>DTPC1+TAUPC))
      !PLATE COOLER 2 HAS FAILED VALVE 1 AND PLATE COOLER 1 ARE REPAIRED
      B=2
      END IF NO_62
      END IF NO_10

ELSE IF(B==10)THEN
NO_11:IF(RC.LE.COMP1)THEN
     !VALVE 1 HAS FAILED
     FV1=FV1+1
     T0=T0+T
     DTV1=T
NO_63:IF(T<DTV2+TAUV)THEN
     !VALVE 1 AND VALVE 2 HAVE FAILED
     !B=22
     FV2=FV2+1
     F=F+1
     T0=T0+TAUV
     B=1
     ELSE !T>DTV2+TAUV
     !VALVE 1 FAILED VALVE 2 IS REPAIRED
     B=17
     END IF NO_63
     ELSE IF(RC.LE.COMP2)THEN
     !VALVE 3 HAS FAILED
     FV3=FV3+1
     T0=T0+T
     DTV3=T
NO_64:IF(T<DTV2+TAUV)THEN
     !VALVE 3 AND VALVE 2 HAVE FAILED
     !B=24
     FV2=FV2+1
     F=F+1
     T0=T0+TAUV
     B=1
     ELSE !T>DTV2+TAU
     !VALVE 3 FAILED VALVE 2 IS REPAIRED
     B=13
     END IF NO_64
     ELSE IF(RC.LE.COMP3)THEN
     !PUMP 1 HAS FAILED
     FP1=FP1+1
     T0=T0+T
     DTP1=T
NO_65:IF(T<DTV2+TAUV)THEN
     !PUMP 1 AND VALVE 2 HAVE FAILED
     !B=31
     FV2=FV2+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !T>DTV2+TAUV
     !PUMP 1 FAILED VALVE 2 IS REPAIRED
     B=16
     END IF NO_65
     ELSE IF(RC.LE.COMP4)THEN
     !PUMP 3 HAS FAILED
     FP3=FP3+1
     T0=T0+T
```

```
     DTP3=T
NO_66:IF(T<DTV2+TAUV)THEN
     !PUMP 3 AND VALVE 2 HAVE FAILED
     !B=32
     FV2=FV2+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !T>DTV2+TAUV
     !PUMP 3 FAILED VALVE 2 IS REPAIRED
     B=4
     END IF NO_66
     ELSE
     !PLATE COOLER 1 HAS FAILED
     FPC1=FPC1+1
     T0=T0+T
     DTPC1=T
NO_67:IF(T<DTV2+TAUV)THEN
     !PLATE COOLER 1 AND VALVE 2 HAVE FAILED
     FV2=FV2+1
     DTV2=T
     B=12
     ELSE !T>DTV2+TAUV
     !PLATE COOLER 1 FAILED VALVE 2 IS REPAIRED
     B=3
     END IF NO_67
     END IF NO_11

ELSE IF(B==11)THEN
NO_12:IF(RC.LE.COMP1)THEN
     !VALVE 1 HAS FAILED
     FV1=FV1+1
     T0=T0+T
     DTV1=T
NO_68:IF((T<DTV2+TAUV).AND.(T<DTPC2+TAUPC))THEN
     !VALVE 1 AND VALVE 2 AND PLATE COOLER 2 HAVE FAILED
     !B=22
     FV2=FV2+1
     FPC2=FPC2+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTV2+TAUV).AND.(T<DTPC2+TAUPC))THEN
     !VALVE 1 AND PLATE COOLER 2 HAVE FAILED VALVE 2 IS REPAIRED
     FPC2=FPC2+1
     DTPC2=T
     B=8
     ELSE IF((T<DTV2+TAUV).AND.(T>DTPC2+TAUPC))THEN
     !VALVE 1 HAS FAILED VALVE 2 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
     !B=22
     FV2=FV2+1
     F=F+1
     T0=T0+TAUV
     B=1
     ELSE !((T>DTV2+TAUV).AND.(T>DTPC2+TAUPC))
     !VALVE 1 HAS FAILED VALVE 2 AND PLATE COOLER 2 ARE REPAIRED
     B=7
     END IF NO_68
     ELSE IF(RC.LE.COMP2)THEN
     !VALVE 3 HAS FAILED
```

```
      FV3=FV3+1
      T0=T0+T
      DTV3=T
NO_69:IF((T<DTV2+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 3 AND VALVE 2 AND PLATE COOLER 2 HAVE FAILED
      !B=24
      FV2=FV2+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV2+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 3 AND PLATE COOLER 2 HAVE FAILED VALVE 2 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=14
      ELSE IF((T<DTV2+TAUV).AND.(T>DTPC2+TAUPC))THEN
      !VALVE 3 HAS FAILED VALVE 2 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=24
      FV2=FV2+1
      F=F+1
      T0=T0+TAUV
      B=1
      ELSE !((T>DTV2+TAUV).AND.(T>DTPC2+TAUPC))
      !VALVE 3 HAS FAILED VALVE 2 AND PLATE COOLER 2 ARE REPAIRED
      B=13
      END IF NO_69
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 1 HAS FAILED
      FP1=FP1+1
      T0=T0+T
      DTP1=T
NO_70:IF((T<DTV2+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !PUMP 1 AND VALVE 2 AND PLATE COOLER 2 HAVE FAILED
      !B=31
      FV2=FV2+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV2+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !PUMP 1 AND PLATE COOLER 2 HAVE FAILED VALVE 2 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=17
      ELSE IF((T<DTV2+TAUV).AND.(T>DTPC2+TAUPC))THEN
      !PUMP 1 HAS FAILED VALVE 2 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=31
      FV2=FV2+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTV2+TAUV).AND.(T>DTPC2+TAUP))
      !PUMP 1 HAS FAILED VALVE 2 AND PLATE COOLER 2 ARE REPAIRED
      B=16
      END IF NO_70
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 3 HAS FAILED
      FP3=FP3+1
```

```
    T0=T0+T
    DTP3=T
NO_71:IF((T<DTV2+TAUV).AND.(T<DTPC2+TAUPC))THEN
    !PUMP 3 AND VALVE 2 AND PLATE COOLER 2 HAVE FAILED
    !B=32
    FV2=FV2+1
    FPC2=FPC2+1
    F=F+1
    T0=T0+TAUPC
    B=1
    ELSE IF((T>DTV2+TAUV).AND.(T<DTPC2+TAUPC))THEN
    !PUMP 3 AND PLATE COOLER 2 HAVE FAILED VALVE 2 IS REPAIRED
    FPC2=FPC2+1
    DTPC2=T
    B=5
    ELSE IF((T<DTV2+TAUV).AND.(T>DTPC2+TAUPC))THEN
    !PUMP 3 HAS FAILED VALVE 2 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
    !B=32
    FV2=FV2+1
    F=F+1
    T0=T0+TAUP
    B=1
    ELSE !((T>DTV2+TAUV).AND.(T>DTPC2+TAUPC))
    !PUMP 3 HAS FAILED VALVE 2 AND PLATE COOLER 2 ARE REPAIRED
    B=4
    END IF NO_71
    ELSE
    !PLATE COOLER 1 HAS FAILED
    FPC1=FPC1+1
    T0=T0+T
    DTPC1=T
NO_72:IF((T<DTV2+TAUV).AND.(T<DTPC2+TAUPC))THEN
    !PLATE COOLER 1 AND VALVE 2 AND PLATE COOLER 2 HAVE FAILED
    !B=28
    FV2=FV2+1
    FPC2=FPC2+1
    F=F+1
    T0=T0+TAUPC
    B=1
    ELSE IF((T>DTV2+TAUV).AND.(T<DTPC2+TAUPC))THEN
    !PLATE COOLER 1 AND PLATE COOLER 2 HAVE FAILED VALVE 2 IS REPAIRED
    !B=28
    FPC2=FPC2+1
    F=F+1
    T0=T0+TAUPC
    B=1
    ELSE IF((T<DTV2+TAUV).AND.(T>DTPC2+TAUPC))THEN
    !PLATE COOLER 1 HAS FAILED VALVE 2 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
    FV2=FV2+1
    DTV2=T
    B=12
    ELSE !((T>DTV2+TAUV).AND.(T>DTPC2+TAUPC))
    !PLATE COOLER 1 HAS FAILED VALVE 2 AND PLATE COOLER 2 ARE REPAIRED
    B=3
    END IF NO_72
    END IF NO_12

ELSE IF(B==12)THEN
NO_13:IF(RC.LE.COMP1)THEN
```

```
      !VALVE 1 HAS FAILED
      FV1=FV1+1
      T0=T0+T
      DTV1=T
NO_73:IF((T<DTV2+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 1 AND VALVE 2 AND PLATE COOLER 1 HAVE FAILED
      !B=22
      FV2=FV2+1
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV2+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 1 AND PLATE COOLER 1 HAVE FAILED VALVE 2 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=9
      ELSE IF((T<DTV2+TAUV).AND.(T>DTPC1+TAUPC))THEN
      !VALVE 1 HAS FAILED VALVE 2 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=22
      FV2=FV2+1
      F=F+1
      T0=T0+TAUV
      B=1
      ELSE !((T>DTV2+TAUV).AND.(T>DTPC1+TAUPC))
      !VALVE 1 HAS FAILED VALVE 2 AND PLATE COOLER 1 ARE REPAIRED
      B=7
      END IF NO_73
      ELSE IF(RC.LE.COMP2)THEN
      !VALVE 3 HAS FAILED
      FV3=FV3+1
      T0=T0+T
      DTV3=T
NO_74:IF((T<DTV2+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 3 AND VALVE 2 AND PLATE COOLER 1 HAVE FAILED
      !B=24
      FV2=FV2+1
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV2+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 3 AND PLATE COOLER 1 HAVE FAILED VALVE 2 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=15
      ELSE IF((T<DTV2+TAUV).AND.(T>DTPC1+TAUPC))THEN
      !VALVE 3 HAS FAILED VALVE 2 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=24
      FV2=FV2+1
      F=F+1
      T0=T0+TAUV
      B=1
      ELSE !((T>DTV2+TAUV).AND.(T>DTPC1+TAUPC))
      !VALVE 3 HAS FAILED VALVE 2 AND PLATE COOLER 1 ARE REPAIRED
      B=13
      END IF NO_74
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 1 HAS FAILED
```

```
      FP1=FP1+1
      T0=T0+T
      DTP1=T
NO_75:IF((T<DTV2+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 1 AND VALVE 2 AND PLATE COOLER 1 HAVE FAILED
      !B=31
      FV2=FV2+1
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV2+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 1 AND PLATE COOLER 1 HAVE FAILED VALVE 2 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=18
      ELSE IF((T<DTV2+TAUV).AND.(T>DTPC1+TAUPC))THEN
      !PUMP 1 HAS FAILED VALVE 2 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=31
      FV2=FV2+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTV2+TAUV).AND.(T>DTPC1+TAUPC))
      !PUMP 1 HAS FAILED VALVE 2 AND PLATE COOLER 1 ARE REPAIRED
      B=16
      END IF NO_75
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 3 HAS FAILED
      FP3=FP3+1
      T0=T0+T
      DTP3=T
NO_76:IF((T<DTV2+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 3 AND VALVE 2 AND PLATE COOLER 1 HAVE FAILED
      !B=32
      FV2=FV2+1
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV2+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 3 AND PLATE COOLER 1 HAVE FAILED VALVE 2 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=6
      ELSE IF((T<DTV2+TAUV).AND.(T>DTPC1+TAUPC))THEN
      !PUMP 3 HAS FAILED VALVE 2 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=32
      FV2=FV2+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTV2+TAUV).AND.(T>DTPC1+TAUPC))
      !PUMP 3 HAS FAILED VALVE 2 AND PLATE COOLER 1 ARE REPAIRED
      B=4
      END IF NO_76
      ELSE
      !PLATE COOLER 2 HAS FAILED
      FPC2=FPC2+1
```

```
     TO=TO+T
     DTPC2=T
NO_77:IF((T<DTV2+TAUV).AND.(T<DTPC1+TAUPC))THEN
     !PLATE COOLER 1 AND VALVE 2 AND PLATE COOLER 2 HAVE FAILED
     !B=28
     FV2=FV2+1
     FPC1=FPC1+1
     F=F+1
     TO=TO+TAUPC
     B=1
     ELSE IF((T>DTV2+TAUV).AND.(T<DTPC1+TAUPC))THEN
     !PLATE COOLER 1 AND PLATE COOLER 2 HAVE FAILED VALVE 2 IS REPAIRED
     !B=28
     FPC1=FPC1+1
     F=F+1
     TO=TO+TAUPC
     B=1
     ELSE IF((T<DTV2+TAUV).AND.(T>DTPC1+TAUPC))THEN
     !PLATE COOLER 2 HAS FAILED VALVE 2 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
     FV2=FV2+1
     DTV2=T
     B=11
     ELSE !((T>DTV2+TAUV).AND.(T>DTPC1+TAUPC))
     !PLATE COOLER 2 HAS FAILED VALVE 2 AND PLATE COOLER 1 ARE REPAIRED
     B=2
     END IF NO_77
     END IF NO_13

ELSE IF(B==13)THEN
NO_14:IF(RC.LE.COMP1)THEN
     !VALVE 1 HAS FAILED
     FV1=FV1+1
     TO=TO+T
     DTV1=T
NO_78:IF(T<DTV3+TAUV)THEN
     !VALVE 1 AND VALVE 3 HAVE FAILED
     !B=23
     FV3=FV3+1
     F=F+1
     TO=TO+TAUV
     B=1
     ELSE !T>DTV3+TAUV
     !VALVE 1 FAILED VALVE 3 IS REPAIRED
     B=7
     END IF NO_78
     ELSE IF(RC.LE.COMP2)THEN
     !VALVE 2 HAS FAILED
     FV2=FV2+1
     TO=TO+T
     DTV2=T
NO_79:IF(T<DTV3+TAUV)THEN
     !VALVE 2 AND VALVE 3 HAVE FAILED
     !B=24
     FV3=FV3+1
     F=F+1
     TO=TO+TAUV
     B=1
     ELSE !T>DTV3+TAUV
     !VALVE 2 FAILED VALVE 3 IS REPAIRED
     B=10
```

```
      END IF NO_79
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 1 HAS FAILED
      FP1=FP1+1
      T0=T0+T
      DTP1=T
NO_80:IF(T<DTV3+TAUV)THEN
      !PUMP 1 AND VALVE 3 HAVE FAILED
      !B=33
      FV3=FV3+1
      F=F+1
      T0=T0+TAUV
      B=1
      ELSE !T>DTV3+TAUV
      !PUMP 1 FAILED VALVE 3 IS REPAIRED
      B=16
      END IF NO_80
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 2 HAS FAILED
      FP2=FP2+1
      T0=T0+T
      DTP2=T
NO_81:IF(T<DTV3+TAUV)THEN
      !PUMP 2 AND VALVE 3 HAVE FAILED
      !B=34
      FV3=FV3+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !T>DTV3+TAUV
      !PUMP 2 FAILED VALVE 3 IS REPAIRED
      B=19
      END IF NO_81
      ELSE
      !PLATE COOLER 1 HAS FAILED
      FPC1=FPC1+1
      T0=T0+T
      DTPC1=T
NO_82:IF(T<DTV3+TAUV)THEN
      !PLATE COOLER 1 AND VALVE 3 HAVE FAILED
      FV3=FV3+1
      DTV3=T
      B=15
      ELSE !T>DTV3+TAUV
      !PLATE COOLER 1 FAILED VALVE 3 IS REPAIRED
      B=3
      END IF NO_82
      END IF NO_14

ELSE IF(B==14)THEN
NO_15:IF(RC.LE.COMP1)THEN
      !VALVE 1 HAS FAILED
      FV1=FV1+1
      T0=T0+T
      DTV1=T
NO_83:IF((T<DTV3+TAUV).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 1 AND VALVE 3 AND PLATE COOLER 2 HAVE FAILED
      !B=23
      FV3=FV3+1
      FPC2=FPC2+1
      F=F+1
```

```
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTV3+TAUV).AND.(T<DTPC2+TAUPC))THEN
     !VALVE 1 AND PLATE COOLER 2 HAVE FAILED VALVE 3 IS REPAIRED
     FPC2=FPC2+1
     DTPC2=T
     B=8
     ELSE IF((T<DTV3+TAUV).AND.(T>DTPC2+TAUPC))THEN
     !VALVE 1 HAS FAILED VALVE 3 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
     !B=23
     FV3=FV3+1
     F=F+1
     T0=T0+TAUV
     B=1
     ELSE !((T>DTV3+TAUV).AND.(T>DTPC2+TAUPC))
     !VALVE 1 HAS FAILED VALVE 3 AND PLATE COOLER 2 ARE REPAIRED
     B=7
     END IF NO_83
     ELSE IF(RC.LE.COMP2)THEN
     !VALVE 2 HAS FAILED
     FV2=FV2+1
     T0=T0+T
     DTV2=T
NO_84:IF((T<DTV3+TAUV).AND.(T<DTPC2+TAUPC))THEN
     !VALVE 2 AND VALVE 3 AND PLATE COOLER 2 HAVE FAILED
     !B=24
     FV3=FV3+1
     FPC2=FPC2+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTV3+TAUV).AND.(T<DTPC2+TAUPC))THEN
     !VALVE 2 AND PLATE COOLER 2 HAVE FAILED VALVE 3 IS REPAIRED
     FPC2=FPC2+1
     DTPC2=T
     B=11
     ELSE IF((T<DTV3+TAUV).AND.(T>DTPC2+TAUPC))THEN
     !VALVE 2 HAS FAILED VALVE 3 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
     !B=24
     FV3=FV3+1
     F=F+1
     T0=T0+TAUV
     B=1
     ELSE !((T>DTV3+TAUV).AND.(T>DTPC2+TAUPC))
     !VALVE 2 HAS FAILED VALVE 3 AND PLATE COOLER 2 ARE REPAIRED
     B=10
     END IF NO_84
     ELSE IF(RC.LE.COMP3)THEN
     !PUMP 1 HAS FAILED
     FP1=FP1+1
     T0=T0+T
     DTP1=T
NO_85:IF((T<DTV3+TAUV).AND.(T<DTPC2+TAUPC))THEN
     !PUMP 1 AND VALVE 3 AND PLATE COOLER 2 HAVE FAILED
     !B=33
     FV3=FV3+1
     FPC2=FPC2+1
     F=F+1
     T0=T0+TAUPC
```

```
     B=1
     ELSE IF((T>DTV3+TAUV).AND.(T<DTPC2+TAUPC))THEN
     !PUMP 1 AND PLATE COOLER 2 HAVE FAILED VALVE 3 IS REPAIRED
     FPC2=FPC2+1
     DTPC2=T
     B=17
     ELSE IF((T<DTV3+TAUV).AND.(T>DTPC2+TAUPC))THEN
     !PUMP 1 HAS FAILED VALVE 3 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
     !B=33
     FV3=FV3+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !((T>DTV3+TAUV).AND.(T>DTPC2+TAUPC))
     !PUMP 1 HAS FAILED VALVE 3 AND PLATE COOLER 2 ARE REPAIRED
     B=16
     END IF NO_85
     ELSE IF(RC.LE.COMP4)THEN
     !PUMP 2 HAS FAILED
     FP2=FP2+1
     T0=T0+T
     DTP2=T
NO_86:IF((T<DTV3+TAUV).AND.(T<DTPC2+TAUPC))THEN
     !PUMP 2 AND VALVE 3 AND PLATE COOLER 2 HAVE FAILED
     !B=34
     FV3=FV3+1
     FPC2=FPC2+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTV3+TAUV).AND.(T<DTPC2+TAUPC))THEN
     !PUMP 2 AND PLATE COOLER 2 HAVE FAILED VALVE 3 IS REPAIRED
     FPC2=FPC2+1
     DTPC2=T
     B=20
     ELSE IF((T<DTV3+TAUV).AND.(T>DTPC2+TAUPC))THEN
     !PUMP 2 HAS FAILED VALVE 3 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
     !B=34
     FV3=FV3+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !((T>DTV3+TAUV).AND.(T>DTPC2+TAUPC))
     !PUMP 2 HAS FAILED VALVE 3 AND PLATE COOLER 2 ARE REPAIRED
     B=19
     END IF NO_86
     ELSE
     !PLATE COOLER 1 HAS FAILED
     FPC1=FPC1+1
     T0=T0+T
     DTPC1=T
NO_87:IF((T<DTV3+TAUV).AND.(T<DTPC2+TAUPC))THEN
     !PLATE COOLER 1 AND VALVE 3 AND PLATE COOLER 2 HAVE FAILED
     !B=28
     FV3=FV3+1
     FPC2=FPC2+1
     F=F+1
     T0=T0+TAUPC
     B=1
```

```
     ELSE IF((T>DTV3+TAUV).AND.(T<DTPC2+TAUPC))THEN
     !PLATE COOLER 1 AND PLATE COOLER 2 HAVE FAILED VALVE 3 IS REPAIRED
     !B=28
     FPC2=FPC2+1
     F=F+1
     TO=TO+TAUPC
     B=1
     ELSE IF((T<DTV3+TAUV).AND.(T>DTPC2+TAUPC))THEN
     !PLATE COOLER 1 HAS FAILED VALVE 3 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
     FV3=FV3+1
     DTV3=T
     B=15
     ELSE !((T>DTV3+TAUV).AND.(T>DTPC2+TAUPC))
     !PLATE COOLER 1 HAS FAILED VALVE 3 AND PLATE COOLER 2 ARE REPAIRED
     B=3
     END IF NO_87
     END IF NO_15

ELSE IF(B==15)THEN
NO_16:IF(RC.LE.COMP1)THEN
     !VALVE 1 HAS FAILED
     FV1=FV1+1
     TO=TO+T
     DTV1=T
NO_88:IF((T<DTV3+TAUV).AND.(T<DTPC1+TAUPC))THEN
     !VALVE 1 AND VALVE 3 AND PLATE COOLER 1 HAVE FAILED
     !B=23
     FV3=FV3+1
     FPC1=FPC1+1
     F=F+1
     TO=TO+TAUPC
     B=1
     ELSE IF((T>DTV3+TAUV).AND.(T<DTPC1+TAUPC))THEN
     !VALVE 1 AND PLATE COOLER 1 HAVE FAILED VALVE 3 IS REPAIRED
     FPC1=FPC1+1
     DTPC1=T
     B=9
     ELSE IF((T<DTV3+TAUV).AND.(T>DTPC1+TAUPC))THEN
     !VALVE 1 HAS FAILED VALVE 3 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
     !B=23
     FV3=FV3+1
     F=F+1
     TO=TO+TAUV
     B=1
     ELSE !((T>DTV3+TAUV).AND.(T>DTPC1+TAUPC))
     !VALVE 1 HAS FAILED VALVE 3 AND PLATE COOLER 1 ARE REPAIRED
     B=7
     END IF NO_88
     ELSE IF(RC.LE.COMP2)THEN
     !VALVE 2 HAS FAILED
     FV2=FV2+1
     TO=TO+T
     DTV2=T
NO_89:IF((T<DTV3+TAUV).AND.(T<DTPC1+TAUPC))THEN
     !VALVE 2 AND VALVE 3 AND PLATE COOLER 1 HAVE FAILED
     !B=24
     FV3=FV3+1
     FPC1=FPC1+1
     F=F+1
```

```
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV3+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !VALVE 2 AND PLATE COOLER 1 HAVE FAILED VALVE 3 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=12
      ELSE IF((T<DTV3+TAUV).AND.(T>DTPC1+TAUPC))THEN
      !VALVE 2 HAS FAILED VALVE 3 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=24
      FV3=FV3+1
      F=F+1
      T0=T0+TAUV
      B=1
      ELSE !((T>DTV3+TAUV).AND.(T>DTPC1+TAUPC))
      !VALVE 2 HAS FAILED VALVE 3 AND PLATE COOLER 1 ARE REPAIRED
      B=10
      END IF NO_89
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 1 HAS FAILED
      FP1=FP1+1
      T0=T0+T
      DTP1=T
NO_90:IF((T<DTV3+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 1 AND VALVE 3 AND PLATE COOLER 1 HAVE FAILED
      !B=33
      FV3=FV3+1
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTV3+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 1 AND PLATE COOLER 1 HAVE FAILED VALVE 3 IS REPAIRED
      FPC1=FPC1+1
      DTPC1=T
      B=18
      ELSE IF((T<DTV3+TAUV).AND.(T>DTPC1+TAUPC))THEN
      !PUMP 1 HAS FAILED VALVE 3 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
      !B=33
      FV3=FV3+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTV3+TAUV).AND.(T>DTPC1+TAUPC))
      !PUMP 1 HAS FAILED VALVE 3 AND PLATE COOLER 1 ARE REPAIRED
      B=16
      END IF NO_90
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 2 HAS FAILED
      FP2=FP2+1
      T0=T0+T
      DTP2=T
NO_91:IF((T<DTV3+TAUV).AND.(T<DTPC1+TAUPC))THEN
      !PUMP 2 AND VALVE 3 AND PLATE COOLER 1 HAVE FAILED
      !B=34
      FV3=FV3+1
      FPC1=FPC1+1
      F=F+1
      T0=T0+TAUPC
```

```
     B=1
     ELSE IF((T>DTV3+TAUV).AND.(T<DTPC1+TAUPC))THEN
     !PUMP 2 AND PLATE COOLER 1 HAVE FAILED VALVE 3 IS REPAIRED
     FPC1=FPC1+1
     DTPC1=T
     B=21
     ELSE IF((T<DTV3+TAUV).AND.(T>DTPC1+TAUPC))THEN
     !PUMP 2 HAS FAILED VALVE 3 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
     !B=34
     FV3=FV3+1
     F=F+1
     TO=TO+TAUP
     B=1
     ELSE !((T>DTV3+TAUV).AND.(T>DTPC1+TAUPC))
     !PUMP 2 HAS FAILED VALVE 3 AND PLATE COOLER 1 ARE REPAIRED
     B=19
     END IF NO_91
     ELSE
     !PLATE COOLER 2 HAS FAILED
     FPC2=FPC2+1
     TO=TO+T
     DTPC2=T
NO_92:IF((T<DTV3+TAUV).AND.(T<DTPC1+TAUPC))THEN
     !PLATE COOLER 2 AND VALVE 3 AND PLATE COOLER 1 HAVE FAILED
     !B=28
     FV3=FV3+1
     FPC1=FPC1+1
     F=F+1
     TO=TO+TAUPC
     B=1
     ELSE IF((T>DTV3+TAUV).AND.(T<DTPC1+TAUPC))THEN
     !PLATE COOLER 2 AND PLATE COOLER 1 HAVE FAILED VALVE 3 IS REPAIRED
     !B=28
     FPC1=FPC1+1
     F=F+1
     TO=TO+TAUPC
     B=1
     ELSE IF((T<DTV3+TAUV).AND.(T>DTPC1+TAUPC))THEN
     !PLATE COOLER 2 HAS FAILED VALVE 3 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
     FV3=FV3+1
     DTV3=T
     B=14
     ELSE !((T>DTV3+TAUV).AND.(T>DTPC1+TAUPC))
     !PLATE COOLER 2 HAS FAILED VALVE 3 AND PLATE COOLER 1 ARE REPAIRED
     B=2
     END IF NO_92
     END IF NO_16

ELSE IF(B==16)THEN
NO_17:IF(RC.LE.COMP1)THEN
     !VALVE 2 HAS FAILED
     FV2=FV2+1
     TO=TO+T
     DTV2=T
NO_93:IF(T<DTP1+TAUP)THEN
     !VALVE 2 AND PUMP 1 HAVE FAILED
     !B=31
     FP1=FP1+1
     F=F+1
```

```
     T0=T0+TAUP
     B=1
     ELSE !T>DTP1+TAUP
     !VALVE 2 FAILED PUMP 1 IS REPAIRED
     B=10
     END IF NO_93
     ELSE IF(RC.LE.COMP2)THEN
     !VALVE 3 HAS FAILED
     FV3=FV3+1
     T0=T0+T
     DTV3=T
NO_94:IF(T<DTP1+TAUP)THEN
     !VALVE 3 AND PUMP 1 HAVE FAILED
     !B=33
     FP1=FP1+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !T>DTP1+TAUP
     !VALVE 3 FAILED PUMP 1 IS REPAIRED
     B=13
     END IF NO_94
     ELSE IF(RC.LE.COMP3)THEN
     !PUMP 2 HAS FAILED
     FP2=FP2+1
     T0=T0+T
     DTP2=T
NO_95:IF(T<DTP1+TAUP)THEN
     !PUMP 2 AND PUMP 1 HAVE FAILED
     !B=25
     FP1=FP1+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !T>DTP1+TAUP
     !PUMP 2 FAILED PUMP 1 IS REPAIRED
     B=4
     END IF NO_95
     ELSE IF(RC.LE.COMP4)THEN
     !PUMP 3 HAS FAILED
     FP3=FP3+1
     T0=T0+T
     DTP3=T
NO_96:IF(T<DTP1+TAUP)THEN
     !PUMP 3 AND PUMP 1 HAVE FAILED
     !B=26
     FP1=FP1+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !T>DTP1+TAU
     !PUMP 3 FAILED PUMP 1 IS REPAIRED
     B=4
     END IF NO_96
     ELSE
     !PLATE COOLER 1 HAS FAILED
     FPC1=FPC1+1
     T0=T0+T
     DTPC1=T
NO_97:IF(T<DTP1+TAUP)THEN
     !PLATE COOLER 1 AND PUMP 1 HAVE FAILED
```

```
     FP1=FP1+1
     DTP1=T
     B=18
     ELSE !T>DTP1+TAUP
     !PLATE COOLER 1 FAILED PUMP 1 IS REPAIRED
     B=3
     END IF NO_97
     END IF NO_17

ELSE IF(B==17)THEN
NO_18:IF(RC.LE.COMP1)THEN
     !VALVE 2 HAS FAILED
     FV2=FV2+1
     TO=TO+T
     DTV2=T
NO_98:IF((T<DTP1+TAUP).AND.(T<DTPC2+TAUPC))THEN
     !VALVE 2 AND PUMP 1 AND PLATE COOLER 2 HAVE FAILED
     !B=31
     FP1=FP1+1
     FPC2=FPC2+1
     F=F+1
     TO=TO+TAUPC
     B=1
     ELSE IF((T>DTP1+TAUP).AND.(T<DTPC2+TAUPC))THEN
     !VALVE 2 AND PLATE COOLER 2 HAVE FAILED PUMP 1 IS REPAIRED
     FPC2=FPC2+1
     DTPC2=T
     B=11
     ELSE IF((T<DTP1+TAUP).AND.(T>DTPC2+TAUPC))THEN
     !VALVE 2 HAS FAILED PUMP 1 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
     !B=31
     FP1=FP1+1
     F=F+1
     TO=TO+TAUP
     B=1
     ELSE !((T>DTP1+TAUP).AND.(T>DTPC2+TAUPC))
     !VALVE 2 HAS FAILED PUMP 1 AND PLATE COOLER 2 ARE REPAIRED
     B=10
     END IF NO_98
     ELSE IF(RC.LE.COMP2)THEN
     !VALVE 3 HAS FAILED
     FV3=FV3+1
     TO=TO+T
     DTV3=T
NO_99:IF((T<DTP1+TAUP).AND.(T<DTPC2+TAUPC))THEN
     !VALVE 3 AND PUMP 1 AND PLATE COOLER 2 HAVE FAILED
     !B=33
     FP1=FP1+1
     FPC2=FPC2+1
     F=F+1
     TO=TO+TAU
     B=1
     ELSE IF((T>DTP1+TAUP).AND.(T<DTPC2+TAUPC))THEN
     !VALVE 3 AND PLATE COOLER 2 HAVE FAILED PUMP 1 IS REPAIRED
     FPC2=FPC2+1
     DTPC2=T
     B=14
     ELSE IF((T<DTP1+TAUP).AND.(T>DTPC2+TAUPC))THEN
     !VALVE 3 HAS FAILED PUMP 1 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
```

```
     !B=33
     FP1=FP1+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !((T>DTP1+TAUP).AND.(T>DTPC2+TAUPC))
     !VALVE 3 HAS FAILED PUMP 1 AND PLATE COOLER 2 ARE REPAIRED
     B=13
     END IF NO_99
     ELSE IF(RC.LE.COMP3)THEN
     !PUMP 2 HAS FAILED
     FP2=FP2+1
     T0=T0+T
     DTP2=T
NO_100:IF((T<DTP1+TAUP).AND.(T<DTPC2+TAUPC))THEN
     !PUMP 2 AND PUMP 1 AND PLATE COOLER 2 HAVE FAILED
     !B=25
     FP1=FP1+1
     FPC2=FPC2+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTP1+TAUP).AND.(T<DTPC2+TAUPC))THEN
     !PUMP 2 AND PLATE COOLER 2 HAVE FAILED PUMP 1 IS REPAIRED
     FPC2=FPC2+1
     DTPC2=T
     B=20
     ELSE IF((T<DTP1+TAUP).AND.(T>DTPC2+TAUPC))THEN
     !PUMP 2 HAS FAILED PUMP 1 HAS FAILED AND PLATE COOLER 2 IS REPAIRED
     !B=25
     FP1=FP1+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !((T>DTP1+TAUP).AND.(T>DTPC2+TAUPC))
     !PUMP 2 HAS FAILED PUMP 1 AND PLATE COOLER 2 ARE REPAIRED
     B=19
     END IF NO_100
     ELSE IF(RC.LE.COMP4)THEN
     !PUMP 3 HAS FAILED
     FP3=FP3+1
     T0=T0+T
     DTP3=T
NO_101:IF((T<DTP1+TAUP).AND.(T<DTPC2+TAUPC))THEN
     !PUMP 3 AND PUMP 1 AND PLATE COOLER 2 HAVE FAILED
     !B=26
     FP1=FP1+1
     FPC2=FPC2+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTP1+TAUP).AND.(T<DTPC2+TAUPC))THEN
     !PUMP 3 AND PLATE COOLER 2 HAVE FAILED PUMP 1 IS REPAIRED
     FPC2=FPC2+1
     DTPC2=T
     B=5
     ELSE IF((T<DTP1+TAUP).AND.(T>DTPC2+TAUPC))THEN
     !PUMP 3 HAS FAILED PUMP 1 HAS FAILED AND PLATE COOLER 2 IS REPAIRED
     !B=26
     FP1=FP1+1
     F=F+1
```

```
     T0=T0+TAUP
     B=1
     ELSE !((T>DTP1+TAUP).AND.(T>DTPC2+TAUPC))
     !PUMP 3 HAS FAILED PUMP 1 AND PLATE COOLER 2 ARE REPAIRED
     B=4
     END IF NO_101
     ELSE
     !PLATE COOLER 1 HAS FAILED
     FPC1=FPC1+1
     T0=T0+T
     DTPC1=T
NO_102:IF((T<DTP1+TAUP).AND.(T<DTPC2+TAUPC))THEN
     !PLATE COOLER 1 AND PUMP 1 AND PLATE COOLER 2 HAVE FAILED
     !B=28
     FP1=FP1+1
     FPC2=FPC2+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTP1+TAUP).AND.(T<DTPC2+TAUPC))THEN
     !PLATE COOLER 1 AND PLATE COOLER 2 HAVE FAILED PUMP 1 IS REPAIRED
     !B=28
     FPC2=FPC2+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T<DTP1+TAUP).AND.(T>DTPC2+TAUPC))THEN
     !PLATE COOLER 1 HAS FAILED PUMP 1 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
     FP1=FP1+1
     DTP1=T
     B=18
     ELSE !((T>DTP1+TAUP).AND.(T>DTPC2+TAUPC))
     !PLATE COOLER 1 HAS FAILED PUMP 1 AND PLATE COOLER 2 ARE REPAIRED
     B=3
     END IF NO_102
     END IF NO_18

ELSE IF(B==18)THEN
NO_19:IF(RC.LE.COMP1)THEN
     !VALVE 2 HAS FAILED
     FV2=FV2+1
     T0=T0+T
     DTV2=T
NO_103:IF((T<DTP1+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !VALVE 2 AND PUMP 1 AND PLATE COOLER 1 HAVE FAILED
     !B=31
     FP1=FP1+1
     FPC1=FPC1+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTP1+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !VALVE 2 AND PLATE COOLER 1 HAVE FAILED PUMP 1 IS REPAIRED
     FPC1=FPC1+1
     DTPC1=T
     B=13
     ELSE IF((T<DTP1+TAUP).AND.(T>DTPC1+TAUPC))THEN
     !VALVE 2 HAS FAILED PUMP 1 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
     !B=31
```

```
     FP1=FP1+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !((T>DTP1+TAUP).AND.(T>DTPC1+TAUPC))
     !VALVE 2 HAS FAILED PUMP 1 AND PLATE COOLER 1 ARE REPAIRED
     B=10
     END IF NO_103
     ELSE IF(RC.LE.COMP2)THEN
     !VALVE 3 HAS FAILED
     FV3=FV3+1
     T0=T0+T
     DTV3=T
NO_104:IF((T<DTP1+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !VALVE 3 AND PUMP 1 AND PLATE COOLER 1 HAVE FAILED
     !B=33
     FP1=FP1+1
     FPC1=FPC1+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTP1+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !VALVE 3 AND PLATE COOLER 1 HAVE FAILED PUMP 1 IS REPAIRED
     FPC1=FPC1+1
     DTPC1=T
     B=15
     ELSE IF((T<DTP1+TAUP).AND.(T>DTPC1+TAUPC))THEN
     !VALVE 3 HAS FAILED PUMP 1 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
     !B=33
     FP1=FP1+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !((T>DTP1+TAUP).AND.(T>DTPC1+TAUPC))
     !VALVE 3 HAS FAILED PUMP 1 AND PLATE COOLER 1 ARE REPAIRED
     B=13
     END IF NO_104
     ELSE IF(RC.LE.COMP3)THEN
     !PUMP 2 HAS FAILED
     FP2=FP2+1
     T0=T0+T
     DTP2=T
NO_105:IF((T<DTP1+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !PUMP 2 AND PUMP 1 AND PLATE COOLER 1 HAVE FAILED
     !B=25
     FP1=FP1+1
     FPC1=FPC1+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTP1+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !PUMP 2 AND PLATE COOLER 1 HAVE FAILED PUMP 1 IS REPAIRED
     FPC1=FPC1+1
     DTPC1=T
     B=21
     ELSE IF((T<DTP1+TAUP).AND.(T>DTPC1+TAUPC))THEN
     !PUMP 2 HAS FAILED PUMP 1 HAS FAILED AND PLATE COOLER 1 IS REPAIRED
     !B=25
     FP1=FP1+1
     F=F+1
```

```
     T0=T0+TAUP
     B=1
     ELSE !((T>DTP1+TAUP).AND.(T>DTPC1+TAUPC))
     !PUMP 2 HAS FAILED PUMP 1 AND PLATE COOLER 1 ARE REPAIRED
     B=19
     END IF NO_105
     ELSE IF(RC.LE.COMP4)THEN
     !PUMP 3 HAS FAILED
     FP3=FP3+1
     T0=T0+T
     DTP3=T
NO_106:IF((T<DTP1+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !PUMP 3 AND PUMP 1 AND PLATE COOLER 1 HAVE FAILED
     !B=26
     FP1=FP1+1
     FPC1=FPC1+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTP1+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !PUMP 3 AND PLATE COOLER 1 HAVE FAILED PUMP 1 IS REPAIRED
     FPC1=FPC1+1
     DTPC1=T
     B=6
     ELSE IF((T<DTP1+TAUP).AND.(T>DTPC1+TAUPC))THEN
     !PUMP 3 HAS FAILED PUMP 1 HAS FAILED AND PLATE COOLER 1 IS REPAIRED
     !B=26
     FP1=FP1+1
     F=F+1
     T0=T0+TAUP
     B=1
     ELSE !((T>DTP1+TAUP).AND.(T>DTPC1+TAUPC))
     !PUMP 3 HAS FAILED PUMP 1 AND PLATE COOLER 1 ARE REPAIRED
     B=4
     END IF NO_106
     ELSE
     !PLATE COOLER 2 HAS FAILED
     FPC2=FPC2+1
     T0=T0+T
     DTPC2=T
NO_107:IF((T<DTP1+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !PLATE COOLER 2 AND PUMP 1 AND PLATE COOLER 1 HAVE FAILED
     !B=28
     FP1=FP1+1
     FPC1=FPC1+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTP1+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !PLATE COOLER 2 AND PLATE COOLER 1 HAVE FAILED PUMP 1 IS REPAIRED
     !B=28
     FPC1=FPC1+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T<DTP1+TAUP).AND.(T>DTPC1+TAUPC))THEN
     !PLATE COOLER 2 HAS FAILED PUMP 1 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
     FP1=FP1+1
     DTP1=T
     B=17
```

```
    ELSE !((T>DTP1+TAUP).AND.(T>DTPC1+TAUPC))
    !PLATE COOLER 2 HAS FAILED PUMP 1 AND PLATE COOLER 1 ARE REPAIRED
    B=2
    END IF NO_107
    END IF NO_19

ELSE IF(B==19)THEN
NO_20:IF(RC.LE.COMP1)THEN
    !VALVE 1 HAS FAILED
    FV1=FV1+1
    TO=TO+T
    DTV1=T
NO_108:IF(T<DTP2+TAUP)THEN
    !VALVE 1 AND PUMP 2 HAVE FAILED
    !B=29
    FP2=FP2+1
    F=F+1
    TO=TO+TAUP
    B=1
    ELSE !T>DTP2+TAUP
    !VALVE 1 FAILED PUMP 2 IS REPAIRED
    B=7
    END IF NO_108
    ELSE IF(RC.LE.COMP2)THEN
    !VALVE 3 HAS FAILED
    FV3=FV3+1
    TO=TO+T
    DTV3=T
NO_109:IF(T<DTP2+TAUP)THEN
    !VALVE 3 AND PUMP 2 HAVE FAILED
    !B=34
    FP2=FP2+1
    F=F+1
    TO=TO+TAUP
    B=1
    ELSE !T>DTP2+TAU
    !VALVE 3 FAILED PUMP 2 IS REPAIRED
    B=13
    END IF NO_109
    ELSE IF(RC.LE.COMP3)THEN
    !PUMP 1 HAS FAILED
    FP1=FP1+1
    TO=TO+T
    DTP1=T
NO_110:IF(T<DTP2+TAUP)THEN
    !PUMP 1 AND PUMP 2 HAVE FAILED
    !B=25
    FP2=FP2+1
    F=F+1
    TO=TO+TAUP
    B=1
    ELSE !T>DTP2+TAUP
    !PUMP 1 FAILED PUMP 2 IS REPAIRED
    B=16
    END IF NO_110
    ELSE IF(RC.LE.COMP4)THEN
    !PUMP 3 HAS FAILED
    FP3=FP3+1
    TO=TO+T
    DTP3=T
NO_111:IF(T<DTP2+TAUP)THEN
```

```
      !PUMP 3 AND PUMP 2 HAVE FAILED
      !B=27
      FP2=FP2+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !T>DTP2+TAUP
      !PUMP 3 FAILED PUMP 2 IS REPAIRED
      B=4
      END IF NO_111
      ELSE
      !PLATE COOLER 1 HAS FAILED
      FPC1=FPC1+1
      T0=T0+T
      DTPC1=T
NO_112:IF(T<DTP2+TAUP)THEN
      !PLATE COOLER 1 AND PUMP 2 HAVE FAILED
      FP2=FP2+1
      DTP2=T
      B=21
      ELSE !T>DTP2+TAUP
      !PLATE COOLER 1 FAILED PUMP 2 IS REPAIRED
      B=3
      END IF NO_112
      END IF NO_20

ELSE IF(B==20)THEN
NO_21:IF(RC.LE.COMP1)THEN
      !VALVE 1 HAS FAILED
      FV1=FV1+1
      T0=T0+T
      DTV1=T
NO_113:IF((T<DTP2+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 1 AND PUMP 2 AND PLATE COOLER 2 HAVE FAILED
      !B=29
      FP2=FP2+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTP2+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 1 AND PLATE COOLER 2 HAVE FAILED PUMP 2 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=8
      ELSE IF((T<DTP2+TAUP).AND.(T>DTPC2+TAUPC))THEN
      !VALVE 1 HAS FAILED PUMP 2 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=29
      FP2=FP2+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTP2+TAUP).AND.(T>DTPC2+TAUPC))
      !VALVE 1 HAS FAILED PUMP 2 AND PLATE COOLER 2 ARE REPAIRED
      B=7
      END IF NO_113
      ELSE IF(RC.LE.COMP2)THEN
      !VALVE 3 HAS FAILED
      FV3=FV3+1
      T0=T0+T
```

```
      DTV3=T
NO_114:IF((T<DTP2+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 3 AND PUMP 2 AND PLATE COOLER 2 HAVE FAILED
      !B=34
      FP2=FP2+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTP2+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !VALVE 3 AND PLATE COOLER 2 HAVE FAILED PUMP 2 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=14
      ELSE IF((T<DTP2+TAUP).AND.(T>DTPC2+TAUPC))THEN
      !VALVE 3 HAS FAILED PUMP 2 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
      !B=34
      FP2=FP2+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTP2+TAUP).AND.(T>DTPC2+TAUPC))
      !VALVE 3 HAS FAILED PUMP 2 AND PLATE COOLER 2 ARE REPAIRED
      B=13
      END IF NO_114
      ELSE IF(RC.LE.COMP3)THEN
      !PUMP 1 HAS FAILED
      FP1=FP1+1
      T0=T0+T
      DTP1=T
NO_115:IF((T<DTP2+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !PUMP 1 AND PUMP 2 AND PLATE COOLER 2 HAVE FAILED
      !B=25
      FP2=FP2+1
      FPC2=FPC2+1
      F=F+1
      T0=T0+TAUPC
      B=1
      ELSE IF((T>DTP2+TAUP).AND.(T<DTPC2+TAUPC))THEN
      !PUMP 1 AND PLATE COOLER 2 HAVE FAILED PUMP 2 IS REPAIRED
      FPC2=FPC2+1
      DTPC2=T
      B=17
      ELSE IF((T<DTP2+TAUP).AND.(T>DTPC2+TAUPC))THEN
      !PUMP 1 HAS FAILED PUMP 2 HAS FAILED AND PLATE COOLER 2 IS REPAIRED
      !B=25
      FP2=FP2+1
      F=F+1
      T0=T0+TAUP
      B=1
      ELSE !((T>DTP2+TAUP).AND.(T>DTPC2+TAUPC))
      !PUMP 1 HAS FAILED PUMP 2 AND PLATE COOLER 2 ARE REPAIRED
      B=16
      END IF NO_115
      ELSE IF(RC.LE.COMP4)THEN
      !PUMP 3 HAS FAILED
      FP3=FP3+1
      T0=T0+T
      DTP3=T
NO_116:IF((T<DTP2+TAUP).AND.(T<DTPC2+TAUPC))THEN
```

```
!PUMP 3 AND PUMP 2 AND PLATE COOLER 2 HAVE FAILED
!B=27
FP2=FP2+1
FPC2=FPC2+1
F=F+1
T0=T0+TAUPC
B=1
ELSE IF((T>DTP2+TAUP).AND.(T<DTPC2+TAUPC))THEN
!PUMP 3 AND PLATE COOLER 2 HAVE FAILED PUMP 2 IS REPAIRED
FPC2=FPC2+1
DTPC2=T
B=5
ELSE IF((T<DTP2+TAUP).AND.(T>DTPC2+TAUPC))THEN
!PUMP 3 HAS FAILED PUMP 2 HAS FAILED AND PLATE COOLER 2 IS REPAIRED
!B=27
FP2=FP2+1
F=F+1
T0=T0+TAUP
B=1
ELSE !((T>DTP2+TAUP).AND.(T>DTPC2+TAUPC))
!PUMP 3 HAS FAILED PUMP 2 AND PLATE COOLER 2 ARE REPAIRED
B=4
END IF NO_116
ELSE
!PLATE COOLER 1 HAS FAILED
FPC1=FPC1+1
T0=T0+T
DTPC1=T
NO_117:IF((T<DTP2+TAUP).AND.(T<DTPC2+TAUPC))THEN
!PLATE COOLER 1 AND PUMP 2 AND PLATE COOLER 2 HAVE FAILED
!B=28
FP2=FP2+1
FPC2=FPC2+1
F=F+1
T0=T0+TAUPC
B=1
ELSE IF((T>DTP2+TAUP).AND.(T<DTPC2+TAUPC))THEN
!PLATE COOLER 1 AND PLATE COOLER 2 HAVE FAILED PUMP 2 IS REPAIRED
!B=28
FPC2=FPC2+1
F=F+1
T0=T0+TAUPC
B=1
ELSE IF((T<DTP2+TAUP).AND.(T>DTPC2+TAUPC))THEN
!PLATE COOLER 1 HAS FAILED PUMP 2 HAS FAILED AND PLATE COOLER 2 IS
REPAIRED
FP2=FP2+1
DTP2=T
B=21
ELSE !((T>DTP2+TAUP).AND.(T>DTPC2+TAUPC))
!PLATE COOLER 1 HAS FAILED PUMP 2 AND PLATE COOLER 2 ARE REPAIRED
B=3
END IF NO_117
END IF NO_21

ELSE !(B==21)
NO_22:IF(RC.LE.COMP1)THEN
!VALVE 1 HAS FAILED
FV1=FV1+1
T0=T0+T
DTV1=T
```

```
NO_118:IF((T<DTP2+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !VALVE 1 AND PUMP 2 AND PLATE COOLER 1 HAVE FAILED
     !B=29
     FP2=FP2+1
     FPC1=FPC1+1
     F=F+1
     TO=TO+TAUPC
     B=1
     ELSE IF((T>DTP2+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !VALVE 1 AND PLATE COOLER 1 HAVE FAILED PUMP 2 IS REPAIRED
     FPC1=FPC1+1
     DTPC1=T
     B=9
     ELSE IF((T<DTP2+TAUP).AND.(T>DTPC1+TAUPC))THEN
     !VALVE 1 HAS FAILED PUMP 2 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
     !B=29
     FP2=FP2+1
     F=F+1
     TO=TO+TAUP
     B=1
     ELSE !((T>DTP2+TAUP).AND.(T>DTPC1+TAUPC))
     !VALVE 1 HAS FAILED PUMP 2 AND PLATE COOLER 1 ARE REPAIRED
     B=7
     END IF NO_118
     ELSE IF(RC.LE.COMP2)THEN
     !VALVE 3 HAS FAILED
     FV3=FV3+1
     TO=TO+T
     DTV3=T
NO_119:IF((T<DTP2+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !VALVE 3 AND PUMP 2 AND PLATE COOLER 1 HAVE FAILED
     !B=34
     FP2=FP2+1
     FPC1=FPC1+1
     F=F+1
     TO=TO+TAUPC
     B=1
     ELSE IF((T>DTP2+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !VALVE 3 AND PLATE COOLER 1 HAVE FAILED PUMP 2 IS REPAIRED
     FPC1=FPC1+1
     DTPC1=T
     B=15
     ELSE IF((T<DTP2+TAUP).AND.(T>DTPC1+TAUPC))THEN
     !VALVE 3 HAS FAILED PUMP 2 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
     !B=34
     FP2=FP2+1
     F=F+1
     TO=TO+TAUP
     B=1
     ELSE !((T>DTP2+TAUP).AND.(T>DTPC1+TAUPC))
     !VALVE 3 HAS FAILED PUMP 2 AND PLATE COOLER 1 ARE REPAIRED
     B=13
     END IF NO_119
     ELSE IF(RC.LE.COMP3)THEN
     !PUMP 1 HAS FAILED
     FP1=FP1+1
     TO=TO+T
     DTP1=T
NO_120:IF((T<DTP2+TAUP).AND.(T<DTPC1+TAUPC))THEN
```

```
       !PUMP 1 AND PUMP 2 AND PLATE COOLER 1 HAVE FAILED
       !B=25
       FP2=FP2+1
       FPC1=FPC1+1
       F=F+1
       T0=T0+TAUPC
       B=1
       ELSE IF((T>DTP2+TAUP).AND.(T<DTPC1+TAUPC))THEN
       !PUMP 1 AND PLATE COOLER 1 HAVE FAILED PUMP 2 IS REPAIRED
       FPC1=FPC1+1
       DTPC1=T
       B=18
       ELSE IF((T<DTP2+TAUP).AND.(T>DTPC1+TAUPC))THEN
       !PUMP 1 HAS FAILED PUMP 2 HAS FAILED AND PLATE COOLER 1 IS REPAIRED
       !B=25
       FP2=FP2+1
       F=F+1
       T0=T0+TAUP
       B=1
       ELSE !((T>DTP2+TAUP).AND.(T>DTPC1+TAUPC))
       !PUMP 1 HAS FAILED PUMP 2 AND PLATE COOLER 1 ARE REPAIRED
       B=16
       END IF NO_120
       ELSE IF(RC.LE.COMP4)THEN
       !PUMP 3 HAS FAILED
       FP3=FP3+1
       T0=T0+T
       DTP3=T
NO_121:IF((T<DTP2+TAUP).AND.(T<DTPC1+TAUPC))THEN
       !PUMP 3 AND PUMP 2 AND PLATE COOLER 1 HAVE FAILED
       !B=27
       FP2=FP2+1
       FPC1=FPC1+1
       F=F+1
       T0=T0+TAUPC
       B=1
       ELSE IF((T>DTP2+TAUP).AND.(T<DTPC1+TAUPC))THEN
       !PUMP 3 AND PLATE COOLER 1 HAVE FAILED PUMP 2 IS REPAIRED
       FPC1=FPC1+1
       DTPC1=T
       B=6
       ELSE IF((T<DTP2+TAUP).AND.(T>DTPC1+TAUPC))THEN
       !PUMP 3 HAS FAILED PUMP 2 HAS FAILED AND PLATE COOLER 1 IS REPAIRED
       !B=27
       FP2=FP2+1
       F=F+1
       T0=T0+TAUP
       B=1
       ELSE !((T>DTP2+TAUP).AND.(T>DTPC1+TAUPC))
       !PUMP 3 HAS FAILED PUMP 2 AND PLATE COOLER 1 ARE REPAIRED
       B=4
       END IF NO_121
       ELSE
       !PLATE COOLER 2 HAS FAILED
       FPC2=FPC2+1
       T0=T0+T
       DTPC2=T
NO_122:IF((T<DTP2+TAUP).AND.(T<DTPC1+TAUPC))THEN
       !PLATE COOLER 2 AND PUMP 2 AND PLATE COOLER 1 HAVE FAILED
       !B=28
       FP2=FP2+1
```

```
     FPC1=FPC1+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T>DTP2+TAUP).AND.(T<DTPC1+TAUPC))THEN
     !PLATE COOLER 2 AND PLATE COOLER 1 HAVE FAILED PUMP 2 IS REPAIRED
     !B=28
     FPC1=FPC1+1
     F=F+1
     T0=T0+TAUPC
     B=1
     ELSE IF((T<DTP2+TAUP).AND.(T>DTPC1+TAUPC))THEN
     !PLATE COOLER 2 HAS FAILED PUMP 2 HAS FAILED AND PLATE COOLER 1 IS
REPAIRED
     FP2=FP2+1
     DTP2=T
     B=20
     ELSE !((T>DTP2+TAUP).AND.(T>DTPC1+TAUPC))
     !PLATE COOLER 2 HAS FAILED PUMP 2 AND PLATE COOLER 1 ARE REPAIRED
     B=2
     END IF NO_122
     END IF NO_22
     END IF NO_1
     ELSE !OUTSIDE MISSION TIME SO NO FAILURES
EXIT INNER
END IF MISS
END DO INNER
END DO HISTORIES_LOOP

WRITE(*,*) FV1,FV2,FV3,FP1,FP2,FP3,FPC1,FPC2,F,N

END PROGRAM
```

Appendix 2 – Delay-Time FORTRAN Code

```
PROGRAM DELAYTIMEFINAL

IMPLICIT NONE

!DECLARE LOCAL VARIABLES

INTEGER,PARAMETER ::TMAX=48
INTEGER, PARAMETER ::TINCR=1
INTEGER ::T,IERROR,bINT,REL_INT,i,J
INTEGER, PARAMETER :: N = 10
REAL::DTE,CU1,b,U1,H1,T01,RI
REAL::KF1=0.0015324
REAL::TM=87600.
REAL, DIMENSION(TMAX) :: DTE_AVERAGES

DTE_AVERAGES = 0.0


!BODY OF CODE
CALL RANDOM_SEED

AVG_LOOP: DO J=1,N

HISTORIES:DO T=1,TMAX,TINCR
T01=0.0
DTE=0.0
CU1=0.0

INNER:DO
!CALL RANDOM U AND H VALUE
CALL EXP_RND_NO(U1,KF1,T01,CU1)
CALL WBLRAND(H1)

!IF CUMULATIVE U VALUE IS BIGGER THAN TM MISSION FINISHES
IF(CU1.GE.TM)EXIT INNER

!CALCULATION OF INSPECTION INTERVAL
b=CU1/T
bINT=INT(b)
REL_INT=(bINT*T)+T

IF(CU1.GE.(REL_INT-H1))THEN
!INSPECTION
T01=T01+U1
!CALL RANDOM_NUMBER(RI)
!     IF(RI.LE.0.9)THEN
!     !INSPECTION
!     KF1=0.0015324
!     T01=T01+U1
!     ELSE
!     !BREAKDOWN
!     KF1=0.001277
!     T01=T01+U1
!     DTE=DTE+1.0
!     END IF
ELSE !(U.LT.(REL_INT-H))
!BREAKDOWN
```

```
!KF1=0.001277
T01=T01+U1
DTE=DTE+1.0
END IF

END DO INNER

!WRITE RESULTS
!WRITE(*,*) T,DTE

!WRITE RESULTS IN OUTPUT TXT FILE
!OPEN(30,FILE='RESULTS.txt',STATUS='REPLACE',ACTION='WRITE',IOSTAT=IERROR)
!WRITE(30,*) T,DTE
DTE_AVERAGES(T) = DTE_AVERAGES(T) + DTE

END DO HISTORIES
END DO AVG_LOOP
DTE_AVERAGES = DTE_AVERAGES / REAL(N)

WRITE (*,*) DTE_AVERAGES
!WRITE RESULTS IN OUTPUT TXT FILE
OPEN(30,FILE='RESULTS.txt',STATUS='REPLACE',ACTION='WRITE',IOSTAT=IERROR)
WRITE(30,*) DTE_AVERAGES

END PROGRAM DELAYTIMEFINAL

!=====================================================================

SUBROUTINE EXP_RND_NO(U,KF,T0,CU)

IMPLICIT NONE

REAL, INTENT(OUT) ::U,CU
REAL, INTENT(IN) ::KF
REAL, INTENT(INOUT) ::T0
REAL ::RT

CALL RANDOM_NUMBER(RT)
U=-(1./KF)*LOG(1-(RT))
CU=CU+T0

END SUBROUTINE
!=====================================================================

SUBROUTINE WBLRAND(H)

IMPLICIT NONE
REAL, INTENT(OUT) ::H
REAL ::alpha=10.
REAL ::beta=5.
REAL ::RT

    CALL RANDOM_NUMBER(RT)
    H=alpha*(-log(1-RT))**(1/beta)

END SUBROUTINE


!=====================================================================
```

Appendix 3 – Circular Buffer Algorithm FORTAN Code

```
PROGRAM ARRAY_TEST

!PURPOSE:
!TO TEST THE SUBROUTINE THAT STORES AND THEN REPRODUCES THE LAST THREE
!TIMES OF SYSTEM TRANSITIONS
!
!DATE        PROGRAMMER       REVISION
!========    ============     ============
!14/12/09    A.CUNNINGHAM     ORIGINAL CODE

IMPLICIT NONE

INTEGER, PARAMETER ::N=10000
INTEGER ::i,S=3
REAL ::T=1.0
REAL, DIMENSION(3) ::TIMES

TIMES=0.0 !INITIALISE THE ARRAY

!BODY OF CODE

TEST:DO i=1,N
        T=T+1.0
        CALL LAST_THREE(TIMES,S,T)
     END DO TEST

     IF(IN_STAFF==1)THEN
       T0=TM
       SYS_DT=TM-DTP3
     ELSE IF(IN_STAFF==2)THEN
       T0=T0+((D+TAU)-DTP3)+TAU
       SYS_DT=((DTP2+TAU)-DTP3)+TAU
     ELSE !STAFF>=3
       T0=T0+TAU
       SYS_DT=TAU
     END IF


WRITE(*,*) TIMES(3)

END PROGRAM ARRAY_TEST

!=======================================================================

SUBROUTINE LAST_THREE(TIMES,S,T)

IMPLICIT NONE

INTEGER, INTENT(IN) ::S
REAL, INTENT(IN) ::T
REAL, INTENT(INOUT), DIMENSION(S) ::TIMES

TIMES(3)=TIMES(2)
TIMES(2)=TIMES(1)
TIMES(1)=T

END SUBROUTINE
```