Whole life cost methods for computer systems

This item was submitted to Loughborough University's Institutional Repository by the/an author.

Additional Information:

- A Doctoral Thesis. Submitted in partial fulfillment of the requirements for the award of Doctor of Philosophy of Loughborough University.

Metadata Record: https://dspace.lboro.ac.uk/2134/7152

Publisher: © Malcolm Bradley

Please cite the published version.
This item is held in Loughborough University’s Institutional Repository (https://dspace.lboro.ac.uk/) and was harvested from the British Library’s EThOS service (http://www.ethos.bl.uk/). It is made available under the following Creative Commons Licence conditions.

For the full text of this licence, please go to:
http://creativecommons.org/licenses/by-nc-nd/2.5/
Whole Life Cost Methods for Computer Systems

by

Malcolm Bradley

A Doctoral Thesis

Submitted in partial fulfilment of the requirements for the award of Doctor of Philosophy of Loughborough University

25th October 1998

©Malcolm Bradley 1998
For my wife Josephine
Abstract

This thesis provides an analysis of cost of ownership issues and techniques, and provides the supporting data to enable future system designers to make rational decisions on design options. It represents the experience gained whilst collecting cost and cost relationship data in the Rolls-Royce group over a period of more than four years. This, in a time of continuous change, in both the company and the wider IT industry.

The thesis is arranged in chapters, each representing a milestone conference or journal paper. The exception to this is chapter 11 - the conclusion and summary of the work in the thesis.

The Chapter topics cover firstly the background of whole life cost and the aims and objectives of the research. A relationship between whole life cost and quality is considered and why whole life cost is a useful measure of quality. This is examined in practical terms of tools and methods. Case studies are used to illustrate the measurement and use of whole life cost. The impact of obsolescence risk is next considered, identifying the causes and implications of obsolescence.

Case studies are used to show how the IT help desk can be used to identify and reduce whole life costs both in a deterministic and a probabilistic approach. This is followed
by an examination of the costs of database systems at Rolls-Royce and Associates. Case studies of database systems are also used to show the need to collect in service data, and genetic algorithms are shown to be a useful tool for analysing the data.

Whole life costing techniques applied to engineering systems at Rolls-Royce is examined. It is shown that a reliability centred maintenance database is a cost effective tool in collecting data. Network monitoring software is shown to be an effective tool for reducing the cost of ownership of IT systems.

The overall conclusion is that whole life cost techniques have been shown to work for computer based systems, further work in this area is still needed to enable costs to be fully understood and optimised.

**Keywords**

Whole life cost, Metric, Reliability, Optimisation, Availability, Genetic Algorithm, Quality, Software, Computer
Acknowledgements

I wish to thank my supervisor Ray Dawson of Loughborough University, for his ever present advice and support over the five years of association with the University.

To my colleagues at Rolls-Royce, my grateful thanks for practical support and consideration over the same period. To the company I extend my sincere thanks for the financial and material support without which I could not have attempted the work.

Most of all I wish to thank my wife Josephine, whose moral support and encouragement gave purpose to it all.

Malcolm Bradley

October 1998
Contents

Chapter 1

Introduction, Existing Standards, Background Information & Aims and Objectives

1.1 Introduction 1
1.2 Standards 5
1.3 Background Information to Whole Life Cost 6
1.4 Aims and Objectives 15
1.5 Research Methodology 15
1.6 Summary of Research in the United Kingdom 17
1.7 Thesis Layout 22

Chapter 2

What is a Quality System Anyway? 34

2.1 What is Life Cycle Costing? 35
2.2 Why Whole Life Cost 37
2.3 Implementing Design to Whole Life Cost 40
2.4 Commonly Available Tools for Design to Whole Life Cost 47
2.5 Practical Implementation of the Whole Life Cost Approach 49
Chapter 3

Gaining Acceptance of the Whole Life Cost Approach

3.1 What is Whole Life Cycle Costing? 55
3.2 Who is using Whole Life Cost Methods? 57
3.3 A Case Study: The SEEI Systems Engineering Project 59
3.4 A step Beyond the SE4 Project 62
3.5 General Considerations of Whole Life Cost 63
3.6 Summary 66

Chapter 4

An Analysis of Obsolescence Risk in IT Systems 68

4.1 Introduction 68
4.2 The Root Causes of Obsolescence 71
4.3 The Software Upgrade Problem 72
4.4 When Demand Outstrips Performance 75
4.5 A Break in the Supply Chain 78
4.6 The Problem of the Business Moving On 79
Chapter 5

Using the IT Helpdesk to Deterministically Reduce the Whole Life Cost

5.1 Introduction 84
5.2 Case Studies 85
5.3 Case Study 1 Determining the Availability and Reliability of PCs 86
5.4 Case Study 2 Applying Availability principles to A complete Software System 96
5.5 A Generalised Method for establishing Cost of Ownership 104
5.6 Conclusions 106

Chapter 6

Using the IT Helpdesk to Probabilistically Reduce the Whole Life Cost

6.1 Introduction 109
6.2 Deriving Failure Profiles 111
6.3 Calculation of Ownership Costs 115
6.4 Calculation of Replacement Costs 116
Chapter 7

Determining the Significance Costs of Software Systems

7.1 Introduction

7.2 Experience of Life Cycle Costing at Rolls-Royce and Associates

7.3 The CAMS System Life Cycle Costs

7.4 The PIMS System Life Cycle Costs

7.5 using Cost Categories for Estimating Full Life Cycle Costs

7.6 Attempted application of existing development estimating algorithms to RDBMS applications

7.7 Proposed metrics for 4GL applications

7.8 Conclusions

Chapter 8

Data Collection Versus World Data

8.1 Why Collect Data

8.2 Why Should we not use world data?

8.3 Some Database Application Case Studies

8.4 Learning from the case studies
<table>
<thead>
<tr>
<th>Chapter 9</th>
<th>Cost Effectiveness</th>
<th>152</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.1</td>
<td>Introduction</td>
<td>153</td>
</tr>
<tr>
<td>9.2</td>
<td>History of the Reliability Centred Maintenance</td>
<td>154</td>
</tr>
<tr>
<td>9.3</td>
<td>Integration with existing systems</td>
<td>155</td>
</tr>
<tr>
<td>9.4</td>
<td>The Cost of Data Collection</td>
<td>158</td>
</tr>
<tr>
<td>9.5</td>
<td>The Detailed Operation of the Database</td>
<td>159</td>
</tr>
<tr>
<td>9.6</td>
<td>Future Developments</td>
<td>162</td>
</tr>
<tr>
<td>9.7</td>
<td>Conclusions</td>
<td>163</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 10</th>
<th>Network Monitoring Software</th>
<th>165</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.1</td>
<td>Introduction</td>
<td>165</td>
</tr>
<tr>
<td>10.2</td>
<td>Monitoring Computer Use</td>
<td>166</td>
</tr>
<tr>
<td>10.3</td>
<td>The Decisions Made</td>
<td>168</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Chapter 11</th>
<th>Summary, Conclusions and Recommendations for Future Work</th>
<th>174</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.1</td>
<td>Summary</td>
<td>174</td>
</tr>
</tbody>
</table>
11.2 Achievements 178
11.3 Conclusions 179
11.3 Future Work 180

References 183

Acknowledgements 193

Appendices
1 Letter to BSI 21 April 1995
2 Failure Probability Results
3 Life Cycle Costing - a paper from Engineering Through Life Support for Profit
4 Retail Price Index 1982 - 1998
5 Chronological Listing of Conference Papers and Journal Articles
CHAPTER I
Introduction, Existing Standards, Background Information & Aims and Objectives

Chapter Preface

This chapter puts the work of this thesis into an overall context. It begins by introducing the current state of the whole life cost of ownership / cost estimating task in an IT context, drawing from both personal experience in industry and from the literature.

1.1 Introduction

During 1993 I spent around six months as one of a five member systems engineering team that had been formed to move the company to an improved document production method. Documents were / are the life blood of the company. Rolls-Royce and Associates Ltd is the Design Authority for the Royal Navy's nuclear steam raising plant. Design documentation and dissemination of a range of technical information is the major company deliverable. This was an important task that the company had set itself.
The systems engineering team collected a great deal of data, analysed, deliberated and then issued a final report that was accepted for implementation. The conclusions were in two parts:

- Procedural change would be undertaken to ensure that documents were as short as possible, delivering a compact message that had been agreed at synopsis before the author task was started.

- A new company-wide networked word processor would be installed replacing the typing pool's local Wang system.

The new system would use Word Perfect 5.1, chosen because:

- The software was already widely used within Rolls Royce Industrial Power Group.

- Both local colleges were teaching version 5.1, making training easy with no special need for in house courses.

- Word perfect 5.1 could be run on all existing PCs, and only a limited number of new PC's would need to be purchased - mainly for the typing pools.
The cost benefit analysis produced with the final report showed very large savings could be made in implementing these simple changes, and on this basis the project was given the go ahead.

For a variety of reasons none of the Systems Engineering team stayed with the project during the implementation phase. Instead technologists were used with experience of network installation. Word perfect version 6 was purchased. This caused a considerable increase in the implementation costs but also caused some surprising side effects, e.g.

- Typists were introduced to the mouse, and output slowed

- Windows™ software was introduced progressively on all machines on the site, which then led to a rapid increase in demand for windows™ application software. Previously the company had used DOS based software as a standard.

My attention was thus focused on an installation which had been previously been cost justified and which was now being installed in a completely different configuration and with a potentially much increased cost of ownership.

The more street wise of computer users have also noted similar effects. Hewson (1) in an article in the Sunday Times on the use of the PC mouse provides an amusing,
but pointed, example. The article was a provocative, semi serious poke in the eye for the windows™ software industry. Hewson takes the line that the use of the mouse actually slows down the use of a software package. He argues that using the mouse indiscriminately can cost (him) a staggering 20 minutes a day. That is two weeks in every year. If this sounds like an extreme viewpoint, it can be quickly verified by carrying out simple experiments to give comparative figures for driving software through the mouse or through 'quick keys' (those key combinations that directly carry out a specific function).

My reaction to the use of windows™ based software was to attempt to compare the different approaches taken by the two teams (the original systems engineering team and the follow on implementation team). It was immediately apparent that in fact the problem was much larger than the simple comparison that I was attempting and that these were only two design solutions from a very large set. Here was an opportunity to derive a method of establishing an optimal design based on the cost of ownership.

I was arguing that any design decision should take all future costs into account when making procurement decisions. Personal (anecdotal) experience in database application development also gave the indication that software development was not generally taking the full costs of ownership into account at design time, notwithstanding any recommendations made by individual design methods. E.g. SSADM includes a requirement to carry out a cost benefit analysis of alternative design solutions.
Coincidentally at this time I was offered the job of Life Cycle Cost facilitator for the company's engineering business. This provided a window into a parallel universe. The Society of Logistic Engineers (SOLE) have been concerned with through life cost for over twenty years and the texts are available for their growing appreciation of reliability, maintainability, availability and whole life costs of systems. But not for software systems!

1.2 Standards

At the time of writing there are no standards specific to the whole life cost of software systems. There is, however, an international standard published by BSI (2) for general whole life costing. This standard, or guide as it is described in the document title, was commented on in full in March 1995, at the end of the first public draft stage and those comments are included in this thesis as Appendix 1. The guide is deficient in several respects, not the least of which is the fact that it is a guide rather than a standard. A major difficulty in producing product whole life cost statements is that there is no accepted, definitive way of presenting the data. This means that a supplier could distort cost data about his product by presenting data in a particular way, or by excluding disadvantageous cost elements.

In addition, the guide implicitly associates high reliability and dependability with low cost of ownership. There is not necessarily a causal link between these attributes of a particular system or element. It is possible to arrange for a system to
be 'unreliable' but quickly repairable, to give an overall lower cost of ownership than an equivalent system with a high inherent reliability and long repair times.

On the subject of modelling of data the draft guide also does not acknowledge that a whole life cost model produced by a purchaser, by its very existence, contains explicit requirement statements about the systems being built. For the future it will need to be a requirement that such models will need to be available to all parties involved in contracts that contain cost of ownership clause(s).

Given that the idea of whole life cost modelling is not a new idea, the first papers in the literature dating from 1957 (3), it appears strange that the techniques should not have become sufficiently well established to have dealt with the points raised in Appendix 1. This might have suggested that whole life cost as a technique is one which although used by a number of industries has not found a permanent place in the design organisations of any one industry. This point is now taken up in the next section.

1.3 Background Information to Whole Life Cost

Industries appear to have a slow awakening to the issues and opportunities of whole life cost, (or total cost of ownership, or whole life cost depending on the industry). This is exemplified by the curves shown in figures 1.1 and 1.2 below. These are both
accumulative counts of papers from the Bath Information Database Service (BIDS) for two industry categories, electronics and computing.

![Cost of Ownership Articles in BIDS - 1988..1997](image)

Figure 1.1 Electronics Articles

Both categories can be seen to have a slow build up of interest when measured by the accumulating count of published papers and journal articles. One key reason for the rise in interest in industry categories is commercial pressure to survive in an industry whose margins are being squeezed. Retail price indices are published monthly in Labour Market Trends, and the January 1998 issue (4) reveals an
interesting insight into some of the industry groups that are concerning themselves with whole life cost techniques.

The retail price index contains approximately 120 separate indices, each covering a group of prices. The overall index, covering all items, was published as 159.6 compared to an index of 100 for January 1987.

![Cost of Ownership Articles in BIDS - 1993..1997](image)

**Figure 1.2 Computing Articles**

Using the indices, a price or cost can be escalated from January 1987 to January 1998 by use of the following formula:

\[
\frac{\text{New index number}}{\text{Old index number}} \times \text{Old price} = \text{Escalated Price}
\]
Thus for a PC cost of £500 in 1987 and using the 'All item' index the escalated cost in January 1998 would be:

\[(159.6 / 100) \times £500 = £798\]

The grouping of indices, however, allows a more accurate estimate of the cost by choosing the correct index for the industry group under study. If the group indices are compared to the 'All Items' index, certain groups can be seen not to be maintaining their prices over time. Two groups whose prices have fallen since 1987 are:

Oil (and other fuels) 118.3
Electrical appliances, including audio visual goods 100.1

These groups are identifiable as those under direct financial pressure from intense competition. They are also industries with players that are increasingly seeking whole life cost advantages over the competition.

**Oil**

The oil industry promoted a Joint Industry Project (JIP) for oil production platform cost of ownership reduction in 1994. This was supported financially by four major oil and gas companies with platforms in the North Sea. The objective of the JIP was to lower the whole life cost of assets following a period of low oil prices. The wording in the JIP proposal document is significant:
However, low oil prices, which have been observed since the mid 1980's, and which are expected to persist ... have increased pressure within the oil and gas industry to reduce operating expenditure... and consistently apply alternative investment criteria.’

Electrical Appliances

The data from Labor Market Trends (4) give an indication that this particular industry is under pressure to find a competitive advantage in a shrinking market place. This is confirmed by the data shown at figure 1.1 where the industry appears to be looking for whole life cost of ownership solutions.

Similarly, the aero engine business is currently very competitive with just three major UK and USA engine manufacturers competing for shrinking engine orders. Here the whole life cost approach is reported by Pearce (5) to be in use, to both design and market aeroplane engines. Other parts of the Rolls-Royce Group are concerned with lowering the cost of ownership of marine engines for the UK Ministry of Defence. Rolls-Royce, unlike the oil industry, is attempting to offer their customer a product, which has inherently lower cost of ownership (than the competition).
These are the two fundamental approaches being used, reflecting the different positions of the customer/owner and the supplier. One position is that of the asset owner seeking to reduce the whole life cost of an asset, and the other is the position of the supplier attempting to hold or gain a greater foothold in difficult market conditions.

Arguably, the Computer hardware business is merely a part of the electronics industry. If this is the case then the computer hardware business can be said to have had an easier time than the majority of the electronics industry until the last four/five years. The situation now appears to have changed. Large corporations can no longer support the constant demands from within to upgrade hardware. This has led hardware suppliers to adopt initiatives designed to convince purchasers that a particular hardware or network design choice will have a whole life cost advantage over all other solutions in the market place. Supplier companies identified, that at least have marketing initiatives, include:

Gartner Group (6), Kyocera (7), Compaq (8), Sun™ (9), Microsoft (10) and Wyse (11).

Each of these organisations can be said to be delivering the marketing message that their product is the one, which provides the minimum cost of ownership solution. Gartner wish to market their cost of ownership software package. Context in their report have provided a comprehensive breakdown of ownership costs for a range of
printers to show that Kyocera printers have the lowest whole life cost, despite
having a slightly higher purchase cost. Compaq, Sun\textsuperscript{TM} Microsoft and Wyse\textsuperscript{1} on the
other hand are all marketing computer hardware, servers and workstations, that offer
low cost of ownership solutions through particular combinations of hardware and
software packages. All six references are recent, the oldest being the Context report
of 1995.

![Entry Level Computer prices 1995 - 1998](image)

Figure 1.3 Entry Level Computer prices

Both Wyse (12) and Kyocera (7) have commissioned 'Cost of Ownership' studies to
support their particular products and these are discussed later in the thesis.

If entry level PC prices are compared for the period 1995 to 1998 (the same period
that has seen a growth in the interest in computing whole life cost of ownership),
with prices escalated using the Retail Price Index (RPI), the result is as shown in figure 1.3. Here the declining value of PC sales can be seen to mirror the rise in interest in Cost of Ownership. The RPI Indices are included for reference in Appendix 4 for years 1982 to 1998. Values are taken from the appropriate issues of the 'Employment Gazette' and 'Labour Market Trends'.

Given the observed increase in interest in whole life cost of ownership in the electronics and computing industries, it is useful to try to understand the mechanisms that might generally influence a particular market sector or industry in adopting or abandoning the whole life cost process.

Figure 1.4 shows the total accumulated whole life cost of ownership articles located from all sources in the reference list that are related to the construction industry to 1994. The pattern here appears to be similar to that exhibited in the early years by the electronics and computing industry, i.e. a slow build up, followed by a heavy concentration of reported research. In the construction industry there has been a decline in the quantity of whole life cost research papers in the late eighties. From contact with the University of Derby (13) it is clear that undergraduate and diploma courses in the construction management and architectural technology areas have included whole life cost as a continuous thread in the appropriate modules of the courses mentioned. These modules include Maintenance and Regeneration, Facilities Management, Economics, Cost Estimation and Planning. Importantly the subject has been included since 1989 - 1990. This time coincides with:
a) Recession in the construction industry, and hence a demand for graduates that were aware of the available whole life cycle tools and techniques.

b) A body of research material being available to support a teaching programme.

At the start of the research programme that supports this thesis Life Cycle Cost / Whole Life Cost / Total Cost of Ownership research in software systems was limited to the cost estimation of the software development process. The intervening years (1994 - 1998) have seen supplier efforts to provide cost of ownership driven marketing, but no reported asset- owner effort other than those supporting this thesis.

Figure 1.4 Construction Industry Whole Life Cost Articles
1.4 Aims and Objectives

Given the background information in the above sections the underlying aims of the thesis are:

a) To test the hypothesis:
   Whole life cost, design methods are appropriate and effective for IT and software based systems.

b) To develop generalised models for the process of designing software systems to minimum Whole Life Cost (WLC).

c) To demonstrate by example, contrary to existing Whole Life Costing beliefs (54) for physical systems that WLC techniques can provide lower whole life costs for software based systems at any time in the perceived system life.

1.5 Research Methodology

The overall method of approach to the research described in this thesis has been as follows.
• Initial literature search using Loughborough University –of the Pilkington library resources, World Wide Web, Books in print, University of Bath BIDS database

• Periodic updates of the literature search during the four years of research.

• Analysis of the Computing hardware and software in use within Rolls-Royce.

• Adjustment of the company practice, based on the analyses.

• Dissemination of the analyses in journals and conferences.

At each stage the research direction of research was assessed to ensure that the aims and objectives were being addressed, and the research direction modified as required. As Rolls-Royce was used heavily for the source of data it was clearly important to ensure that what was being analysed and reported was not atypical of the practices being adopted by the UK industrial base. This was addressed by regular comparisons of RR practice with the recorded industry practice in the weekly computing journals.

The thesis draws heavily on statistical methods. It was necessary to update my skills in this area, and I attended a course on the statistics of failure run by University of Plymouth. As a result of the course Rolls-Royce installed the Minitab statistical software that was used throughout the research.
Whole life cost assessments are, by their nature, a practical business and it was an important feature of the research that company processes were changed on the basis of the research results.

1.6 Summary of Research in the UK

A review of Current Research in Britain (58) revealed no specific research into the cost of ownership of computer/software based systems with searches that used the following keywords

- Total cost of ownership
- Cost
- Cost estimation
- Cost of ownership
- Life cycle cost
- Whole life cost

The conference audiences and reviewers of papers and articles supporting this thesis provided some useful feedback to the proposals and conclusions presented. However, at no time did any reviewer or audience provide a pointer to a reference that either supported or contradicted the ideas being presented.
Similarly, searches using both BIDS and the available internet search engines located very few articles. Of these only a very small number had even a slight connection with computers and computer software. The very phrase 'life cycle' caused problems, as the software fraternity regard the software development life cycle as the life cycle, (for this reason the thesis refers to the subject matter as Whole rather than Life or Total cost, in an attempt to differentiate between the different perceptions of the whole product life cycle). It remains possible to find documents in the literature that reference 'life cycle' but which are only concerned with the cost of the software development. Norman Fenton's book (Reference 49) is typical in this sense. The second edition of his text, written jointly with Shari Pfleeger, contains much of the background work that is necessary for whole life costing but the book is predominantly concerned with software, rather than both the software and the hardware it runs on. Fenton and Pfleeger do concern themselves with issues of maintainability, usability and reliability. These are issues that are of concern in whole life costing, but these are dealt with as engineering topics more fully by other authors, e.g. Knezevic (41). Other major areas of concern like obsolescence are not so easily found in the literature, hence the paper covering this subject and re presented in Chapter 4. Obsolescence of semiconductors is a manufacturing issue for the electronics industry, so it is not surprising that companies like Taktec in the USA and DERA in the UK are providing data services for the many thousands of semiconductor devices. Services include advice on pending component obsolescence, part finding for obsolete devices, and advice on
alternatives for parts which are either obsolete or facing pending obsolescence. No similar advice is available for the software components in widespread use.

The investment to achieve any progress is significant, and the investors tend, therefore, to be industrial companies with a potential profit motive. It is instructive to analyse the categories of authors in the reference list to this thesis:

Figure 1.5 provides a breakdown of the references to the thesis from the standpoint of the author. The chart shows that for this thesis the observer category is the largest, this category including independent researchers and consultancies, with manufacturers and tied consultancies the next largest. Manufacturers are perceived to be the only major group with the finance and access to the raw data to enable studies with a practical outcome.

![Figure 1.5 Taxonomy of References by Author](image-url)
This may be the crucial feature of this research theme, whole life cost studies are by their very nature a practical business. Independent observers are not in the position to make a difference because the organisations that have access to the data are unwilling to release it to external organisations or individuals. Any cost data is guarded carefully in commercial organisations. Whenever such Rolls-Royce data has been released in papers it has aroused interest. Equally the release itself has received scrutiny from the Company to ensure that commercial advantage is not being given away.

Turning to the companies in the references, with the exception of the Kyocera printer studies carried out by Context (7), no hard facts are made available. Even when direct questions about cost savings are put to company representatives (this happened at a Compaq event in Manchester in 1997) the result is a promise to direct the question to another company representative. Claims are rarely backed up with fact.

Figure 1.6 shows the split on those references in the thesis that are specifically connected to computer hardware and software. The categories are evenly split between hardware and software. The important point to note, however, is that eight software and three hardware references included are those of the author. The most exhaustive searches reveal only very small numbers of hits.
Finally, it is worth noting that the same phenomenon occurs in other areas. During 1997 and 1998 I undertook a large scale and significant study in Rolls-Royce on the cost of ownership of the Royal Navy's submarine fleet. This was a study carried out from the perspective of a manufacturer seeking to consolidate a market position by improving the understanding of costs for the product in its manufacture, use and retirement. Only two references of substance were found in a comprehensive literature search. The first was a publication from the Rand Corporation concerned with US experience, and funded by the US Department of Defence. The second was the published paper detailing the 1996/7 UK Defence Estimates. These two documents are almost unique in that they provide financial detail that can not be found anywhere else. This probably points the way for the future, in that government is best placed to
• direct standard makers,
• provide the funding
• provide access to the necessary data

and should therefore take the lead in ensuring that UK industry has universal access to published whole life cost data in a standard format. This is an issue that is taken up in the summary and conclusions.

1.7 Thesis layout

'A house may only be constructed from the bricks that are available' - Chinese proverb

It is true that progress in whole life cost research is only possible after taking measurements, as the whole subject is based on understanding the cost implications of various choices of actions. Measurement is an expensive, and sometimes disruptive, business. The experience during the research has been that whilst specific elements of research have been pursued by my employer others have only been completed by persuading the employer that the work would result in a productive outcome. A further group of research tasks have been conducted covertly, as they were not supported at all. This reflects on the perceived value that different
elements of the work has to the company, and also the changing fortunes of the IT
departments in the organisation.

The net result is that the research is supported by the case studies that were available
possible rather than the case studies that would have been desirable. Thus, the rest
of the thesis is laid out in chapters that follow the case studies. All of the chapters
except 11 are based on published journal or conference papers. Chapters are
organised in four main groups; chapters 2 and 3 are concerned with method and the
close link between quality and cost of ownership. Chapters 4, 5 and 6 cover the
subjects of obsolescence, reliability and availability. The next three chapters, 7 to 9,
are concerned with relational database applications. Chapter 10 describes the more
useful outputs of available network monitoring software, whilst chapter 11
summarises the work.

Each chapter is summarised below, with the details of the original paper on which
each is based:
Chapter 1 Introduction, Existing Standards, Background Information & Aims and Objectives


Chapter 2 What is a Quality System Anyway?


Chapter 2 provides:

- Definition of whole life cost
- Current measures of software quality
- The proposition that whole life cost is a measure of quality and the use of whole life cost data as a marketing tool.
- A tested method for implementing design to minimum whole life cost
Commonly available tools for use with the method

A case study provides an example of quantitative measures being used in a whole life cost design decision task.

Chapter 3 Gaining Acceptance of Whole Life Cost

Taken from 'Gaining Acceptance of Software Whole Life Cost by IT Staff' (60), Proceedings of the First Psychology of Programming Interest Group, Postgraduate Student Workshop, September 1996, Matlock, Derbyshire.

Chapter 3 provides:

- A view of the difficulties associated with commonly used design methods and where design strategies are dominated by purchase price
- A case study for design and implementation of a desktop system, demonstrating the value of the whole life cost approach.
- Comparison between systems methods, involving a single pass through the design process, and the iterative, top-down approach of the whole life cost method.
- Hardware and software examples in which influences other than cost often determine a design or procurement decision.
- Summary of the benefits to the customer of the whole life cost approach, and the need for changes to the way that IT staff design and procure, education of
IT staff, and implementation of metrics programmes for the high cost phases of the life cycle.

Chapter 4 Obsolescence Risk in IT Systems


Chapter 4 provides:

- The definition and root causes of obsolescence, and the relationship to the whole life cost.
- Description of the four identified issues surrounding obsolescence, software obsolescence, computer obsolescence, breaks in the supply chain, and changes in the direction of the business.
- A proposition that there is a continuing need to manage risk of obsolescence in order to control the cost of ownership during the operational phase of a system.
Chapter 5 Using the IT Helpdesk to deterministically reduce the whole life cost

Taken from 'Reducing the Cost of IT Ownership Using Feedback from the IT Helpdesk' (36), Fifth International Conference on Software Quality Management, March 1997 and reprinted in the Software Quality Journal, June 1997 (66). 

Chapter 5 provides:

- A description of the observed link between reliability, availability and the cost of ownership of IT systems, including two linked case studies and the deterministic supporting data.
- Listing of simple hardware and software attributes that enable cost of ownership studies / design to minimum whole life cost.

Chapter 6 Using the IT Helpdesk to probabilistically reduce the whole life cost


Chapter 6 provides:

- A probabilistic approach to computer reliability based on the deterministic case study described in Chapter 5.
- The impact of the probability papers, determined from field data, is explained by showing that the cost of part replacement and lost availability declines over the life of the machine.

Chapter 7 Determining the Significant Costs of Software Systems


Chapter 7 provides:

A view of the major cost elements of two significant database applications, providing through two case studies, an indication that:

- Application use is at least a high contributor to whole life cost
- Pareto methods can be used to predict overall system costs
- Comparison of function point analysis and Putnam estimation model when applied to 4GL application development and a description of the poor resulting estimates.
• A derived method for development cost estimation based on significant aspects of the design (obtained by interview of experienced personnel in the Rolls-Royce development team.)
• The relationship between development effort and full life costs for the two application case studies.

Chapter 8 Data collection versus world data

Taken from 'What Can Software Engineering teach Industry?' (63), Data, Information and Decision Making Seminar, Atomic Energy Authority, March 1996.

Chapter 8 provides:

• Comparison of software failures with construction industry project failures, with the proposition that failure may be the result of the step changes often found between one project and the next.
• The proposition that collection of local cost and reliability data for the operational phase of the life cycle, rather than:
  • a) Use of world data, which may not be applicable to the organisation
  • b) Data for less significant phases of the life cycle
• Four case studies provide the evidence that use and design are amongst the top whole life cost elements.
• An introduction to the idea that genetic algorithms can be used successfully for determination of the minimum whole life cost design.

• A method for model building suitable for use with genetic algorithm, based on influence diagrams / causal networks.

Chapter 9 Cost effectiveness

Taken from 'Lifetime Management Maximises Submarine Power Plant Availability' (64), Presented at the International Structural Integrity Conference, Cambridge University September 1998.

Chapter 9 provides

History and whole life cost profile of a database application designed to capture nuclear component failure data.

Whilst this Chapter provides a view of the cost effectiveness of an application to store and manipulate non IT failures, it is clear that a similar application would provide cost effectiveness data about IT systems / components. The future development section of the chapter is therefore considered as much about the future recommended whole life research for IT as for nuclear systems. It is anticipated that the model for data manipulation presented by the Reliability Centred Database
(RCM) database application will be reconfigured to establish the most cost effective options for a range of database applications.

**Chapter 10 Network Monitoring Software**

Taken from 'Computer Evaluation' (65), a chapter for an Encyclopedia of Electronics Engineering accepted for publication by the publishers John Wiley.

This Chapter provides a snapshot of the software available for network monitoring- software that was used in case studies elsewhere in the thesis.

**Chapter 11 Summary, Conclusions and Recommendations for Further Work**

This chapter summarises the whole thesis, relating the work to the aims and objectives set out in chapter 1.

**Appendix 1**

Appendix 1 is the copy of a letter to the BSI with comments on the DRAFT IEC 300-3-3
Appendix 2

Appendix 2 contains the collection of probability density functions for the PC sub components that are covered by Chapter 7. The density functions were derived from collected data, rather than by calculation, and are not published elsewhere.

Appendix 3

Appendix 3 is a copy of the paper Life Cycle Costing (15), In 'Engineering Through Life Support for Profit' delivered at the Midland Hotel in Derby in October 1994. This was a closed seminar, with the papers being published for the attendees only. As the Life Cycle Cost paper was for a mixed audience - it contains a mixture of software, and engineering system examples. This makes the full paper of limited value to the thesis, but it does contain material, which is referenced by other papers, justifying the inclusion as an appendix.

Appendix 4

Appendix 4 contains RPI indices for the years 1982 to 1998. RPI can be usefully used to escalate costs / prices from earlier years to current year. Data included is from the 'Employment Gazette' and 'Labour Market Trends'. These two monthly journals have, over several decades recorded the escalating prices of over a hundred commodities. The indices recorded in Appendix 4 are for the 'all items excluding
seasonal food’. Where escalation has been used in papers in this thesis values have been calculated using the indices in Appendix 4. Because the two journals above have used different index reference dates the data in Appendix 4 uniquely combines the two sources of data into a single data set.
CHAPTER 2

What is a Quality System Anyway?

Chapter Preface

The conventional wisdom is that a design methodology will provide a quality software solution and that qualitative judgements would be concerned with the quality of the delivered code or algorithms, in matching the customer requirements.

This chapter offers a supplementary design method based on the whole life cost of ownership and establishes that the whole life cost is increasingly regarded as the principal criterion for comparison of software systems, especially in the defence industry. Existing costing algorithms for software systems tend to concentrate on the development costs only (14,20). However, whole life costs for a software system must include the full cost of ownership with costs such as hardware, operation, and data administration. It is shown that these costs can have a significant if not dominant bearing on the design process for a software system. A nine step approach is suggested for the design process based on whole life costing; some commonly available software tools can support this approach.
Industrial experience has shown that obsession with software development costs tend to obscure the full cost of ownership. In conclusion, therefore, this chapter provides a case for an alternative definition of a quality software system as one, which demonstrably provides the required functionality at the lowest cost to the owner, where ALL system costs are taken into consideration.

2.1 What is Whole Life Costing?

Whole life costing is sometimes referred to as life cycle costing or through life costing. Whatever the name it amounts to much the same thing, but this text will refer to the collection of methods as whole life costing (WLC). As we shall see this will act as a reminder that the costs being assembled are holistic, in the sense that the whole of the project is considered, rather than either

- the acquisition cost of the complete product or

- the whole life cost of one element of the delivered product.

Whole Life Cost estimation and analysis is not simply concerned with the software development life cycle. Whole life costs are concerned with all the costs for a project, and if that project is a software system then normally all the costs of ownership of that system are included, not merely the costs associated with the development cycle.
Existing cost estimating methods, e.g. COCOMO from Boehm, (14) or Putnam methods from Putnam, (20) are concerned with software development costs only. Other significant costs are not dealt with. These might typically include the cost of:

- Hardware
- Hardware maintenance
- Software operation
- Network installation and support
- Software licences
- Database administration
- Data validation
- Training

A draft international standard issued for comment in January 1995 by the British Standards Institution, (2), provided a definition of whole life cost as the cumulative cost of a product over its life cycle.

Alternative descriptions are available; e.g. the sum of all costs incurred during the lifetime of an item from Dhillon, (18).

In application, the costs included in the life cycle cost can be categorised as:

- Research and Development
• Production and construction
• Operation and Support
• Retirement

The sum of these costs for a product can then be used for two ends:

• Establishing affordability.
• Comparing two or more products on the basis of total cost of ownership.

Significantly for the software industry, although software is not explicitly covered by the draft standard, the panel for the standard does include representation from National Computing Centre Limited. We might still assume from this that software systems should and can be dealt with in the same way as physical systems from a whole life cost point of view.

2.2 Why Whole Life Cost?

Major users of whole life cost techniques to date have been the civil engineering fraternity, and the defence industry. The industries that are applying the techniques are those that are facing major competition to sales or budgetary constraints to purchases. Ministry of Defence (MoD) contracts are now including whole life cost requirements at the bid stage, and other industries employing whole life cost
arguments to gain business include printer (Kyocera 17) and car manufacturers (e.g. Daewoo in their various advertisements). From a purchasers perspective MoD objectives are simple: guaranteed affordability, coupled with a design that demonstrably has the minimum whole life cost, often for a design life measured in decades. Demanding whole life cost at the bid phase forces potential contractors at least to think in terms of the most cost effective set of design solutions. Whole life cost has become an integral part of the decision process. It is probably significant therefore that a whole life cost seminar organised by the Institution of Electrical Engineers, and held at the Institution of Mechanical Engineers in London, March 1995, was well represented by MoD contractors.

Professional software engineers need also to be concerned with the benefits of the whole life cost approach in application to the 'software' design process. Best software practice currently concerns itself with the application of an accepted structured design method. According to Sommerville, (21), there appears not to be a definitive way of establishing what is 'good design'. Conventional wisdom might suggest that quality judgements would be concerned with the quality of the delivered code or algorithms, in matching the customer requirements. These qualitative judgements might include:

- Cost of the delivered software
- Timeliness of the delivery
- Uncorrected bugs
- Functionality
- Ease of use
- Development cost
- Lines of code / day
- Response time
- Reliability
- Availability

The qualitative judgements in the alternative whole life cost environment are directly quantified in cash terms in a whole life cost profile. This profile by definition incorporates all project costs and, as we shall describe later, can be further reduced to a single cash figure for comparison purposes.
The justification for a whole life cost approach for software based projects is founded in research that suggests that only 25% (17) to 40% (23) of through life costs are incurred designing and implementing systems. The balance of ownership costs (between 60 and 75%) are incurred in the operation and retirement phases.

2.3 Implementing Design to Whole Life Cost

The process shown in Figure 2.1, taken from Bradley, (15), provides a model of the process of Design to Whole Life Cost for an organisation using formal design reviews. Although not specifically designed for software systems the model has been found to be equally applicable to both hardware and software systems. Here feasible alternatives are identified, typically using brainstorming techniques, and these are then costed individually to provide cost profiles for each of the competing design options. A subsequent design review meeting decides which design option is to be taken, taking a range of qualitative data into account as well as the whole life cost comparisons. The process is iterative, reflecting both:

- the need for the design review right to veto

- the need to design to WLC in a top down sense, in greater detail at each successive level in the design hierarchy.
Figure 2.1 Design to Minimum Cost

The complete process has nine steps centred on the Design Review Meeting. The process in summary is shown in Figure 2.1, with amplification of the individual steps in the following paragraphs:
1) Initial Design Review to Establish the Feasible Alternatives for a Design Solution.

This initial step is the starting point for the design process and provides the analyst and/or the designer with guidance on the costing task, alternatives to be costed, the overall life span, financial discounting, the need for an accurate estimate or comparative study and the boundaries of the costs to be included. This is also the opportunity for the organisation to set out explicit requirements on the sources of data that are to be used in the design process, and the extent to which the designer might be expected to gather new data to support the case for the winning design. Chapters 4 to 10 of this thesis are concerned with the many aspects of data collection to support whole life cost analysis.

2) Development of a Suitable Cost Breakdown Structure

All elements of the project, system or component need to be cost accounted for. The cost breakdown structure used must also be one that will be supported by the subsequent project management tasks. This enables the estimate to be compared with the actual costs incurred during the project life. The cost breakdown structure is, in effect, multi-dimensional, with Product, Life cycle phases and elapsed time dimensions. The nature of the cost model is that it will contain a significant number of cost elements. The size of the data problem dictates the extent to which the model will be filled since data acquisition itself has a cost. Clearly, where all the competing design options have an identical sub component, then that part of the design may not be modelled at all, and the designer may
choose to omit the component from the cost structure. Section 2.6 of this chapter provides an example of the categories that might appear in a cost structure for a simple typing example.

3) Selection of a Suitable Cost Model for the Analysis.

For each element of the cost breakdown structure created in step (2) costs for the Acquisition, Operation and Disposal need to be separately included for each year (or other time period) of life for the cost elements included. An audit trail is an essential feature of any model used, to allow identification of the source of any cost data used.

The degree of sophistication required in construction of the model is open to some debate. This thesis, in Chapters 5 and 6, presents case studies of deterministic and probabilistic approaches to the same problem. Probabilistic methods undoubtedly provide design decision-makers with a clearer picture of a design problem, but also attract greater design costs. An example of such costs exists in the determination of failure probabilities for the PC sub-components. This task alone, in support of the work of this thesis, cost approximately 150 man-hours.

4) Development of Cost Estimates

Cost estimates are attached to the cost elements in the model. These can be based on actual data from historical records, quotations or estimates obtained from the
many estimating techniques available, e.g. parametric, comparative, detailed. This thesis does not provide guidance on cost estimating algorithms and methods, however references (16) and (17) provide algorithms for estimating the development timescales of INGRES™ applications. More direction on estimating is provided by Fabrycky and Blanchard (54) who cover the more generally used methods of mentioned above. As can be seen from later sections of this text:

- the production phase for hardware elements can be costed relatively easily in IT systems,
- the design and production phases of software elements are difficult to cost, hence the chapter on Ingres™ application development.
- the operation phase of IT systems (hardware + software + people) is effectively impossible to estimate without taking measurements. This is covered in Chapter 10.

5) Development of Cost Profiles

Cost profiles are created by the summation of all costs for each year. These are generally presented as a histogram. An example of a typical histogram is shown in Figure 6.4. The cost profile may be adjusted to take account of the time value of money using the following formula:

\[ P = F \left( \frac{1}{(1+i)^n} \right) \]
where

\[ F = \text{Future value} \]
\[ P = \text{present value} \]
\[ i = \text{discount rate} \]
\[ n = \text{years} \]

If Present Values are used to generate a cost profile then the whole life cost practitioner must take care to use a discount value and a range of years appropriate for the software system being studied. For relational database applications a typical value for \( n \) would be between 3 and 10 years. Discount rates (\( i \)) vary with industry but are generally greater than 5%. Rates are often fixed centrally in an organisation and may even be fixed by government agencies. Different combinations of \( i \) and \( n \) can lead to radically different design decisions. It can therefore be helpful to have profiles of both present value and undiscounted costs and benefits.

Profiles of discounted values are normally expressed as a single value, or Net Present Value (NPV). NPV is the sum of the present values for the life of the item.
6) Carry out a Break Even Analysis (Optional)

A combination of cost profiles and net present value will normally provide a sound basis for comparison. In some cases, however, where competing designs have very similar cost profiles it can be useful to compare the accumulating costs (rather than the year on year costs in the profile). As the name implies this identifies the year(s) in which the competing designs have identical whole life costs, and where one subsequently becomes more competitive than all other design options.

An example of such a breakeven graphic is shown in Appendix 3, Figure 5. It has been found that the relatively short lifespans of IT systems provides few opportunities for the use of a breakeven analysis. It is more usual to make a simple presentation of the option costs in a table of the kind shown in table 3.1.

7) Identify High Cost Contributors

A review of all cost elements in the structure will identify the high cost drivers. Once identified these can be nominations for redesign with a view to whole life cost reduction, causing an iteration of the process.

A useful tool for identification is the Pareto presentation. The use of Pareto curves in the context of Ingres™ applications is given in Chapter 7. Here the significant cost of relational database applications is shown to be data maintenance.
8) Sensitivity and Uncertainty Analysis

Sensitivity analysis, or the determination of the impact of changing estimates, can be useful in identifying risks to the project. Commercial software is available to carry out sensitivity analysis automatically in a spreadsheet. Similarly uncertainty can be handled to provide decision makers with guidance on the range of likely whole life cost for each design option.

9) Select the Design Option

Ultimately decision-makers have to balance all the information being provided to the design review. Whole life costs are but one input to the decision. Others include risks associated with novel design, political acceptance, obsolescence, etc. These risks and other issues will be those that were excluded from the whole life cost analysis in step 2. As previously stated, cost collection / estimation can be expensive, and some skill is required in the selection of those cost elements to be excluded from the design study. The very act of carrying out a study can reveal alternative options that carry a lower whole life cost than those dictated by step 1. This was certainly true in the case study described in section 5.3.4.

2.4 Commonly Available Tools for Design to Whole Life Cost

Probably the most useful tool in the practitioner’s armoury is the humble spreadsheet. This facility allows what-if? calculations, sensitivity analysis, graphical presentation
of cost profiles and break-even comparisons all as standard functions. In addition, so-called add-ins are available to enhance the basic spreadsheet functions. Two useful additions are Probability and Genetic Algorithm.

Probability add-ins, such as @Risk™, are available that enable a range of values to be used in conjunction with a distribution type (e.g., Weibull) in place of a single value. Spreadsheet cells incorporating a distribution are carried forward in calculations, typically providing a range of values in cost profiles. This provides system designers with the range of likely whole life cost values.

By comparison the genetic algorithm (GA) add-ins, such as Evolver™, provide an optimising function. For whole life cost the GA is used to provide the combination of inputs for a minimised output. Thus the GA can be used to determine the optimum combination of elements in a design that taken as a whole will result in the minimum whole life cost given the range of feasible options presented. A simple example would be to present a minimum whole life cost solution to the problem of which combination of word processor, word processor version, personal computer, specific typist skills, etc should be purchased; or indeed whether a word processor should be purchased at all.
2.5 Practical Implementation of the Whole Life Cost Approach

Many of the factors that influence the cost of a system will be hard to determine. Is it possible, for example, to determine the cost of lost future business through the poor reliability of a system? In practice, however, it is often not necessary to consider all the costs if the approach is being used to compare different systems. Only those costs that will differ between alternatives need be considered. A demonstration of the principle that quality measures can be reduced to cost from industrial experience is given in section 3.5.

When used as a comparative measure of system or software quality all the costs and benefits are compiled into a single cost profile which is then reduced to a single cost using the Net Present Value. A full description of the methods used to generate a Whole Life Cost comparison is contained in Bradley and Dawson (23), and are presented in section 2.3.

2.6 Example of Quality Measurement using Whole Life Cost from Industry

An example of quality measurement taken from industrial experience at Rolls-Royce and Associates is the redesign and implementation of a networked word processing system.
In this example, the alternative means of producing the company's large number of letters were compared to see which was the best option. The "best option" and therefore the highest quality system, was determined by examining which system would give the overall minimum cost of ownership. This is a simple example with only a few variables to consider but it nevertheless shows how the method can work and give a significant overall cost saving.

Figure 2.2 Causal Network – Typing Pool
The alternatives considered were the different versions of word processor, the hardware platform and who should type the letters, i.e. whether they should be hand written and passed to a typing pool or whether the originators should type the letters themselves. The direct and indirect cost influences were determined and plotted in a causal network shown in figure 2.2. These were then quantified using the current market values of the hardware and software and the current staff rates. A genetic algorithm, Evolver™, was used to derive the least cost option.

Genetic Algorithms, previously used with some success in the early design optimisation of a wide variety of systems, are described by Davis, (24). Genetic Algorithms are now being focused on data systems. Early indications are that the following method is appropriate to derive an optimum design to minimum whole life cost, replacing the steps 3 to 5 of the main method described in Section 2.3.

- Deriving through brainstorming techniques a causal network of the relationships affecting the whole life cost of an information system incorporating a relational database application.
- Creation of a list of relations from the causal network in step 1.
- Deriving by experiment a set of heuristics and constraints for each relation
- Using the heuristics and constraints, create a mathematical model incorporating a Genetic Algorithm (GA) to establish the minimum whole life cost design.
The resulting figures given below show that, for managers, the traditional typing pool using a DOS based version of word processor on low specification computers would save the company several hundred pounds / letter writer / year compared with the other option combinations. This clearly is a finding that needs to be kept under review as software, hardware and staff rates all change regularly. This in turn is a pointer to a major difference between computer based and lower technology systems. Continuous change is a feature of computer solutions. The problem of obsolescence in the context of whole life cost is dealt with in Chapter 4. The results show that for the simple case of letter production, the better quality solution is the lower technology solution.

<table>
<thead>
<tr>
<th>Word Processor Version</th>
<th>1 (DOS)</th>
<th>2 (DOS)</th>
<th>3 (WIN 3 x)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PC Spec 1 (80386)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Handwritten for typing</td>
<td>5882</td>
<td>6174</td>
<td>6355</td>
</tr>
<tr>
<td>Author types own</td>
<td>7397</td>
<td>8272</td>
<td>8816</td>
</tr>
<tr>
<td><strong>PC Spec 2 (80486)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Handwritten for typing</td>
<td>6403</td>
<td>6551</td>
<td>6631</td>
</tr>
<tr>
<td>Author types own</td>
<td>8334</td>
<td>8779</td>
<td>9018</td>
</tr>
</tbody>
</table>

Table 2.1 Typical three year costs of producing letters with a number of competing options
2.7 Other Results of the Whole Life Cost Approach

The whole life cost approach has provided the company with measurable improvements in cost of ownership, as a result of applying the described methodology to software systems. Example savings are shown below for projects that involved the author.

**Saving to the Company**

- PC Maintenance Options £30,000 / year
- Document Production System £450,000 / year
- Network availability £250,000 / year estimated - not yet implemented

2.8 Conclusion

For the future the professional designer will need to be concerned with all of the costs of ownership of systems, and not merely the acquisition cost. The business of designing to whole life cost criteria is complex. To enable the decision process, models, tools and cost data all need to be available. Traditionally software engineers have collected data to improve their understanding of the software development process. The experience at Rolls-Royce and Associates Limited has shown that in the future metrics will be needed on software, hardware, data and the interactions between these elements for the whole of the information system life cycle. This
includes the all important system operation phase, including the support to the hardware, software and data.

The overall system cost can be regarded as an effective measure of the quality of the system and is particularly useful when comparing alternative system proposals. This conclusion does not seem so surprising when the nature of quality is considered, which raises the question of what is a quality software system anyway? Traditional quality considerations such as timeliness of delivery, necessary functionality, ease of use, and reliability are all reflected in the whole life system cost. The timeliness of delivery will affect the development and possibly the operation costs, the functionality and ease of use affect the operation costs, and the reliability is reflected in the operation and maintenance costs. If a quality system is one that is regarded as one that inflicts the least grief on its owner then clearly the whole life cost is a measure of this. The only exception would be for safety critical systems, largely because it is not possible or even ethical, to put a cost on human life. However, for most systems whole life costing becomes a particularly effective measure of quality as it allows simple comparisons of different systems and different aspects of quality within a system. Indeed it could be argued for most systems that whole life costing is the only measure of quality that is needed. It follows that for the future the focus in whole life costing for design to minimum through life cost will need to be in the area of accurate prediction of costs of ownership.
CHAPTER 3

Gaining Acceptance of the Whole Life Cost Approach

Chapter Preface

This chapter discusses the need to educate Senior Management and IT staff in the need for the concept of Whole Life Cost (WLC), and supports the need through presentation of a case study that led to the adoption of a Whole Life Costing method. The chapter concludes with a description of some general considerations of the need for whole life cost and a summary of the benefits to companies of adopting the WLC method on software projects.

3.1 What is Whole Life Costing?

'This is a thing you can easily explain twice before anybody knows what you are talking about.' A. A. Milne

Whole life costing was defined, described and demonstrated by case study in Chapters 1 and 2. This chapter is concerned primarily with the problems that will face a practitioner or champion of the techniques in introducing the concept of
accounting for the costs of the whole life cycle when making procurement or design decisions.

### 3.1.2 Why Whole Life Cost?

Historically, purchase decisions have been made on the basis of the initial procurement cost, rather than the total cost of ownership through the product life. Today however information professionals are becoming aware that the cost of ownership is an important issue. In a recent issue of 'Computing' magazine a quotation (25) from a Railtrack spokesman that '...the challenge is business management and cost of ownership.' underpins the claim in Chapter 1 that the IT industry is facing competition that is forcing a whole life cost approach.

### 3.1.3 The Need to Educate IT Staff

Professionalism, paradoxically, can stand in the way of achieving the 'best' design. Designers of any discipline will generally hold the view that their design method takes account of all the necessary parameters and that the final design approaches the best possible. This leads to resistance to design by whole life costing. Adopting a whole life cost approach to design provides a new rigour to the design process. At every stage of the top down design process the question must be asked by the IT professional, 'which design will give the lowest through life cost?'. However whole life cost trade offs in the design process are not inexpensive, and the design process
will inevitably cost more. The maxim must be spend more now and save much more later.

Because the design and purchase costs are likely to be greater this is a maxim that is not easily sold to IT staff. Breaking down the barriers is really only achieved by proving that solid savings can be made by using the technique. Success breeds success.

3.2 Who is Using Whole Life Cost Methods?

3.2.1 A Historical Perspective

Whole life cost methods have been used to date largely by the civil engineering fraternity, and the defence industry. Civil engineering references may be found in the literature as early as 1970 (26) and the subject is to be found on the syllabus of building and construction degree courses in the UK. The industries that are applying the techniques tend to be those that are:

- mature and / or
- facing major competition to sales and / or
- presented with budgetary constraints to purchases
In a literature search conducted at the start of the research supporting this thesis, for the years prior to 1994, twenty per cent of the whole life cost literature was concerned directly with civil engineering and defence systems. Sadly of more than eight hundred references less than 1% were concerned with software systems.

Demanding whole life cost at the bid phase forces potential contractors to think in terms of the most cost effective set of design solutions. For those industries adopting it, whole life cost thinking has become an integral part of the decision making process. For software systems the situation is less certain. Periodically, computer hardware manufacturers use whole life cost in an attempt to increase their market share of a particular product. Examples of this approach are the Context report on page printers commissioned by Kyocera (27) and the Benchmark Research report on PC ownership commissioned by Compaq (28). These studies are useful demonstrations of a comparison of competing options (27) and of the current attitude to whole life cost issues by IT managers (28). In fact, the Benchmark report (28) epitomises the attitude of management in wanting reduced cost of ownership but of continuing to adopt purchase and design strategies that are dominated by purchase price and low design costs.
3.3 A Case Study - the SE4 Systems Engineering Project

3.3.1 What was SE4?

SE4 was the name given to a systems engineering project carried out by Rolls-Royce and Associates Limited in 1993. The project was to look at the typing process used to produce company documents and the project was to use a systems method that had been used successfully by the Rolls-Royce group on a number of previous occasions. This was the Project 2000 systems method described below.

3.3.2 The Project 2000 Methodology

The project 2000 method was utilised by the SE4 team because it had previously been used with success by both the parent Rolls-Royce PLC and by Rolls-Royce and Associates Limited to carry out process improvement. The method is one of hard systems analysis and re-design to achieve the 'world's best performance'.

The whole method is driven by

a) a small core team - that provides the ideas, and researches the current process,
b) a support team drawn from the management - that challenge and appraise the ideas of the core team, and ultimately buy in to the re designed process.

The project 2000 method comprises eleven steps that are carried out in a predominantly sequential manner. For this chapter not all of the steps are discussed, only those that have direct relevance to the subject. Steps 0 and 1 are those concerned with agreement of the project boundary whilst steps 7 to 9 are the comparison (of old and new design), simulation and cost benefit steps.

3.3.3 Demonstrating the need for whole life costing

The original task definition only required re-design of the typing process, but following initial analysis it was clear to the core team that a much reduced cost of ownership would be achieved if the whole of the document production process was included in the task. The new task included all process steps from authoring to the reproduction of the finished document, as well as the typing of the document included in the original project definition. The increased scope of the task was met with some resistance from the management team supporting the project, and an initial estimate of the relative cost savings was required to convince the team of the merits of the increased task size. The revised project definition was formally accepted in May 1993. What is important to note here is that:

a) a system design with a more expensive initial cost was finally chosen
b) early demonstration of the benefits of the more expensive option was vital to convince the team that the extra spend was worth while.

3.3.4 The Results

A number of different options were costed over the new system design life of three years. The chosen option was the one with the minimum whole life cost and offered a potential saving in the second and third years of around 0.5 Million Pounds a year. The cost profiles were generated on a spreadsheet and based on cost estimation relationships derived from simple measurements of the original document production systems. The spreadsheets employed a discount rate of 0%, i.e. for the three years of the system life all costs and benefits were calculated at the rates for year 1.

The spreadsheets produced included all the costs and benefits in the design life and included:

- Document planning
- Training
- Hardware cost
- Hardware installation
- Software licensing
- Document conversion (old word processor to new)
• Hardware maintenance
• Implementation costs
• Avoidance of a Mainframe upgrade
• Expanded ethernet
• File servers
• Document production time

Monitoring of the system introduced has confirmed that the savings predicted were achieved in the second full year of operation.

3.4 A Step Beyond the SE4 Project

The SE4 project was motivated by the vision of one manager, using a method that had been used successfully by Rolls-Royce on a wide range of process improvement projects. The Project 2000 experience provided invaluable experience in a new area for the company. At the time of the SE4 project the core team members had no concept of the 'whole life cost approach to design'. It was driven by the accountants. The SE4 project was the catalyst in bringing the subject of Whole Life Cost, and the benefits to be gained to the company. The SE4 experience has been combined with the published body of knowledge to produce a whole life cost method that has wide applicability. Project 2000 and the Whole Life Cost method have some apparently common aims, the Project 2000 goal is the worlds best process, whilst the Whole Life Cost aim is the process with the minimum whole life cost. The two approaches
have some almost common steps, step 1 of both processes being almost identical. The main differences between the two methods is that in:

Project 2000
Performance is the main emphasis rather than cost, and the method is used as a once through, or serial process, compared to the iterative Whole Life Cost method.

Whole Life Cost
The method is iterative, rather than serial, and the opportunity is taken to design the best process or product, rather than the being better than the rest of the world. This is a software quality issue that was addressed by Bradley & Dawson in (23) in the sense that a quality software system can be regarded as one which has the lowest cost of ownership for a particular function set.

The complete Whole Life Cost process has nine steps centred on the Design Review Meeting. The full process is described in Chapter 2.

3.5 General Considerations of Whole Life Cost

In general it is a human failing that easy options are taken in preference to difficult options. This is certainly true of software procurement and design decisions. In making procurement and design decisions we sometimes make expensive mistakes. The formalism of a method can help to avoid these mistakes.
3.5.1 A Hardware Purchase Example

Probably, in common with most large organisations, Rolls-Royce and Associates purchases its PC's through a separate buying function. The IT specifiers provided a technical and support specification and this has been the basis for the buying function to place a competitive tender on potential suppliers. More recently the purchase of a PC has included the cost of installation to desk and three year on site warranty.

Whilst this strategy appears attractive it also has some problems. The first is that PC's have very few failures in the first three years of life and virtually no failures in the first year. Of the seven 486 PC's in a department of Rolls-Royce and Associates, in the first year of operation no equipment failures have been reported out of the sixty two requests for assistance from the company help desk associated with the seven machines.

The second problem is that by the end of the third year of ownership a PC is almost certainly obsolete, in the sense that the machine will probably need to be upgraded to handle the latest software (see next chapter). Money spent on maintenance might well be better spent on an upgrade to a higher specification if a significant failure is experienced during the third year.
A third disadvantage is that Help Desk staff always carries out initial analysis of a reported problem, and help desk staff are involved when the external maintenance staff carry out a repair. The overall personnel costs are largely unchanged by the existence of the maintenance contract.

A whole life cost comparison between paying for maintenance at the time of procurement and paying for maintenance at the time of probable failure quickly confirms that an unlikely and significant number of major failures would need to occur before the upfront maintenance option becomes attractive.

3.5.2 A Software Example

Pure software examples abound. A common problem is that of choosing a software tool. Often the purchase cost of competing tools is near identical, the price that the market will bear rather than the value of the tool. A whole life cost approach includes the assessment of risks for each competing tool:

a) becoming obsolescent

b) the required skills for the tool becoming obsolescent in the life of the software product

What all too often happens is that demand for specific software skills in the market place has the greatest influence on the choice of the tool. On a design and implementation project two years ago, technical decisions made on the choice of a
tool led to a PC based database tool being chosen. A recent combination of a demand for maintenance on the project, and the loss by natural wastage of the small number of personnel with the necessary coding skills has led to a sharp increase in cost of ownership of the project. We may now be faced with either a re-write or the use of contract staff. If a whole life cost approach had been used it is likely that a different tool, different operating system and hardware would have been used because the risks associated with the tool would have been assessed on the common base of cost.

3.6 Summary

3.6.1 The Benefits of Design to Whole Life Cost

In summary the benefit to a company of adopting the WLC method on software projects is:

Under design to whole life cost all design decisions are taken with a rationale of reducing the cost of ownership at every design step taken. This rigour in design forces decisions to be made in favour of the client that pays for the product or service, and provides a design which:

a) Pleases the customer in a way, which is more likely both to generate future business, and create a reputation for a quality product.
b) Records design decisions making them auditable against the data that was available at the time of the decision.

Design to whole life cost criteria demands data or best estimates of a range of parameters for the product being designed. This focuses attention on those metrics which really are important instead of the development process metrics traditionally taken by IT management.

3.6.2 The Way Forward

A psychological change is required in the way that IT staff design and procure software products. Cost has to be considered at every step in the design process to ensure delivered products are those with the minimum whole life cost, based on the information available at design time.

The immediate need is the education of IT staff, through WLC case studies that show demonstrable cost savings over traditional design methods.

In parallel with the education process the necessary metrics need to be collected to ensure that decisions are taken with the benefit of all available and relevant data.
CHAPTER 4

An Analysis of Obsolescence Risk in IT Systems

Chapter Preface

This chapter is concerned with technological and functional obsolescence in the context of common computing applications and from a user perspective. Both forms of obsolescence are defined and the effect on software and hardware are discussed, together with the consequence for cost of ownership of IT systems. Supporting data is provided from a number of industrial surveys of commercial off the shelf software.

4.1. Introduction

There are two principle forms of obsolescence, technological and functional, and these affect the provider and user of products in different ways.
Technological obsolescence is a problem that afflicts every delivered product and service to a degree. The greatest degree of affect is felt in rapidly changing technologies, the least degree in those that change only slowly. Examples of specific products drawn from a slowly changing technology that are affected to only a small degree by obsolescence are bridges and roads. Recent examples of products that are affected to a greater degree by obsolescence are the media used by audio industry. Here, technological advance has inflicted effective obsolescence on various standards of records and tapes over several decades. The effect of technological change and obsolescence is to force new purchase on to the user, whilst providing opportunity and disaster to the owners of the incoming and outgoing technologies.

Functional obsolescence is a secondary problem, generally caused by technological advance elsewhere, and is exemplified by the disuse of sections of Roman roads. Whilst roads in general are not yet obsolete, some roads will become functionally obsolescent as the destination is no longer in use. Functional obsolescence leaves society with a piece of technology that has gone out of use, leaves the owner of the outgoing technology with reduced opportunity, and may provide another technology's owner with a new opportunity.

The crucial difference between the two forms of obsolescence is that the Technological form is caused by a technology owner whilst Functional obsolescence
is caused either by the user, or by a different technology entirely - a new technology owner with a step change.

In addition to the typing of obsolescence, there is also an issue on the standpoint from which obsolescence is viewed. There are a great number of articles in the literature written from the position of a manufacturer, (29,30,31,32,33), providing advice on the strategies to retain or gain market leadership. There is also advice available from the literature for organisations in the middle of the supply chain (33), where it is clearly important to monitor component obsolescence to avoid both lost sales and obsolete stock.

This chapter is concerned with both forms of obsolescence described above in the context of common computing applications, the hardware required and the consequent implications for cost of ownership of IT systems. The context is that of the end user. To provide a common understanding of the subject the following dictionary definitions are provided, taken from Chambers (34):

Obsolescent: 'adj. - going out of use, in course of disappearance; tending to become obsolete'
Obsolescence: 'n. - < the process of > ' going out of use'

4.2. The Root Causes of Obsolescence.

Obsolescence impacts on a PC in a business environment in a number of ways, these are typically caused by:

- software being subject to an upgrade by the software supplier creating technological obsolescence of the earlier software version, and a possible consequential obsolescence of the computing hardware.

- the demands made on a PC are increased within the capability of the software, but beyond the capability of the hardware, creating functional obsolescence of the existing PC.

- the PC is no longer supported because some component is now no longer in production or is no longer in demand by the market place, an example of technological obsolescence.

- the business changes and the software changes in line with the new business, causing functional obsolescence of the software (version), again with a possible consequential obsolescence of the hardware.

An understanding of the likely occurrence of obsolescence is important in understanding the cost of ownership of a PC in the organisation. The first three of
these categories are dealt with below, the fourth category is not, as this category is business specific and is best dealt with by carrying out a risk analysis of the business future.

4.3. The Software Upgrade Problem

This form of obsolescence occurs where the software being used is subject to an upgrade by the software supplier and where the upgrade requires an increased specification PC, or where the software is no longer supported at all - withdrawal.

An indication of the volatility of software can be gained from the time between full releases of packaged business software. Clearly, not all releases cause an improved specification to be purchased, but usually the pressure is increased on the business to upgrade. A review of releases for some popular windows™ based software carried out in May 1997 revealed:

Software vendors generally provide support for software for the current and previous releases only. Some may provide informal support for earlier releases, but may not formally enter into support agreements based on old software versions. This implies that the time between alternate releases is significant. It has also been quite normal for three contiguous PC software releases to move the software from DOS, through Windows™ 3. X to Windows 95™, with significant changes to hardware specification
at each change. Based on a total of eight popular software packages with a total of twenty five releases the time between alternate software releases is estimated to follow a Lognormal probability distribution, with a Hazard plot as shown in Figure 4.1. The Hazard function provides an indicator of the change in instantaneous renewal (or obsolescence) rate with respect to time t over the life of a component. The hazard function is normally used to present the 'Bathtub' curve used by many texts to show the normally expected failure characteristics of engineering components and explained in a software context by Calverley (56).

Figure 4.1 provides the indication that the instantaneous risk of a second re-release of a single PC software package increases sharply with time, and that the expected time to re release of any one piece of software is 1160 calendar days. This effectively means that if an organisation was fortunate enough to buy a software licence for a business application on the first day that the version was available the chances are that it would be considered obsolete around 1160 days later; or approximately three years. Unfortunately the situation is not normally as clear cut. Most organisations operate a small number of key PC software packages, but also a larger number of lesser used software packages. At the time of writing Rolls-Royce and Associates Limited (RRA, now known simply as Rolls-Royce PLC) are making widespread use of the Windows™ operating system software and two such packages, WordPerfect™ and Groupwise. These are the word processing and internal communications packages, accounting for around two thirds of PC use as determined by Green (35). In total
however around twenty commercial, off the shelf (COTS) packages are in use, with an approximately random spread of release dates. If two COTS application packages are considered to be running on one PC both of the packages are likely to become obsolete, and if we assume that one of the licences is purchased mid version, the time to obsolescence drops to 870 days. As the number of key software packages increase the expected time to a second re release decreases to around half the expected time for two releases, (580 days or 21 months). What happens in practice is that the dominant packages in an organisation are the ones that decide if the organisation upgrades its hardware. Where specific machines are used with specialist packages, e.g. CAD or statistics, then this smaller number of machines are upgraded in isolation. For the majority of machines the software technological obsolescence causes expenditure on both software and hardware between 580 and 870 days from the joint purchase of hardware and software, despite the fact that the hardware in isolation has a low probability of failure as described by Bradley & Dawson (36). It is clear that most businesses are committed to COTS application software. Given this situation the minimum cost of ownership solution for a small organisation would therefore appear to be:

- the adoption of a policy of software purchase at the release date of the major application(s) used by the organisation,
- a hardware upgrade to the most technologically advanced hardware within the organisations budget.
This particular form of software obsolescence also has an impact on the software development fraternity; as compilers and their versions change so too must the developer. Developer skill obsolescence is therefore exactly paralleled by the obsolescence characteristics of the software that the developers use. Specifically, this means that there is a likelihood that at two to three year intervals software or software version will change with a corresponding risk to hardware and training.

![Software Releases - Based on elapsed time for two releases](image)

**Figure 4.1 Software Releases – Based on Elapsed time for two releases**

### 4.4. When Demand Outstrips Performance

Obsolescence can occur where the demands made on a PC are increased within the capability of the software but where the PC is not capable of meeting the demand. This can occur where the original PC was not fully capable of handling all aspects of the software; this has happened at RRA even when the PC was the latest specification.
machine and purchased to 'match' the software, reflecting the software drive for ever better / faster machines. An example of the phenomenon is an initial use of a word processing package to produce simple letters and documents, followed by ever more sophisticated and larger documents as users grow in confidence and awareness of the software.

Data from RRA can demonstrate this. The company implemented a networked WordPerfect™ word processor system in early 1994. Training was restricted to the typing pool and a proportion of the rest of the Company - around thirty per cent of the total organisation receiving training in the new word processor. The first machines on the

![Figure 4.2 RRA Document Statistics](image-url)
network were 80486, 33Mhz machines with 8 Mbytes of RAM. Training was largely
to a basic level as it was expected that the corporate helpdesk would handle more
complex problems. Figure 4.2 provides a view of the changing workload experienced
by the typing pool in the three and a half years from 1994. That workload was of
course dependent on the level of business, although that was relatively constant, with
year on year turnover varying by less than ten per cent over the period.

Figure 4.2 shows that the mean document size being produced by the pool increased
steadily over the three and a half years to 250k. Additionally it can be seen that
production of small documents (of 20k or less) had initially peaked and then almost
disappeared entirely after the third year.

This data appears to show the growing confidence of the RRA population of (non-
typing pool) WordPerfect™ users, who, over the period, have gradually undertaken
the production of all but the very largest of documents. To move from novice to a
very capable and confident user of this comprehensive word processor has taken, we
believe, around two years, during which time the PC being used has been stretched in
terms of processor speed and memory. This is demonstrated by Green (35) who
recommended in 1997 that ten Pentium 90 machines less than 2 years old should be
replaced with large memory Pentium 200s on the grounds of inadequate performance.
As Green pointed out, RRA is an organisation whose PC use is dominated by the use
of WordPerfect™, with 50% of PC use restricted to the word processor. The Green
recommendation does, however, lead to an organisation with a continuing problem of replacement and where the technology, rather than the organisation appears to be in control.

The impact of this problem is focused on the hardware, with the organisation effectively pressured into the purchase of latest technology that will run the older software more effectively. If the experience with WordPerfect™ at RRA is typical then the need to upgrade hardware might be expected to occur at around two years after a major word processor version change.

4.5. A Break in the Spares Supply Chain

Effective obsolescence occurs where the PC is no longer supported because some component is no longer in production, or has been overtaken by new technology. In either situation spares to support in-service machines may become unavailable.

The clearest example of this problem, and the one which to date has probably determined PC obsolescence more than any other, is the processor installed in the PC. Data from Intel (37) shows that obsolescence of the chip has accelerated with each successive design improvement. Figure 4.3 shows the days between significant design improvements of the 80x86 chip from the original 8086 to the PentiumII. Strictly,
Figure 4.3 shows virtual obsolescence, as the chips can still generally be purchased from the manufacturer. However the supply chain tends to support the current production, and/or the hardware that is covered by warranty. In the event of a motherboard failure that requires a replacement chip this effectively puts an average limit on the spares availability, and hence machine obsolescence of around three years. Chip improvements, though, have been occurring more frequently recently and this average has fallen to less than two years for the most recent PC sales.

4.6. The Problem of the Business Moving on

The final cause of obsolescence is found where the business moves on, and the software in use has to change. This will often be to meet the demands of a (new) prime customer, or because part of the core business has changed. It is an issue that needs careful attention from both the IS / IT management and the main board of an organisation. As it will be different for different organisations it is only possible to generalise on policy for handling the risks as methods for handling these are likely to be the same for all organisations.
4.7. The Need to Identify and Manage Risk

Organisations investing in IT have a need to understand the various risks that can affect the cost of the investment. RRA have had a well developed risk management method since 1994, and largely similar to the principles described by Pressman (38). This involves identification and assessment of the risks to a project, mapping of the probability and impact of each risk, and putting risk reduction measures in place at the project start. Risk management can be expensive though, and at RRA is therefore usually used only on projects with a value greater than £10M.

The importance of risk management and continuous risk review is illustrated by Sillitoe (39). In this case the company was requested 'informally' to move from WordPerfect™ to Word by the main customer. The change, if carried out, would have involved a move not only to a new word processor, but also to a new operating
system, Windows 95™, which at the time would not run on 70% of the company's PCs. The cost of a change of the full PC population (around 700 were involved) would be in excess of £1M for hardware alone. The risk that might have been anticipated in this case was the obsolescence of the operating software for the extant PC population. Continuous review of the risk would have identified the need for an earlier and gradual migration to a new operating system. At the time of writing (1998) at RRA only one example of formal software risk management exists. This was for the implementation of the networked PC / WordPerfect™. For this project no account was taken of the impact of obsolescence, as the project was for the implementation of the system, rather than the long term operation of the new network. Risk management needs to be performed more widely for IT systems. Specifically risk management for IT needs to include obsolescence of software and hardware.

4.8. Conclusions

Obsolescence is a significant root cause of high whole life costs, and yet is not often considered at the planning and implementation stages of software based projects.

Different forms of obsolescence need to be considered and each of these forms an impact on hardware and software.
Both hardware and software can suffer from obsolescence, and this can have an impact on skill obsolescence.

It is possible to gauge the probability and impact of obsolescence of both software and hardware elements of an IT system in the context of both risk assessment and cost of ownership.
CHAPTER 5

Using The IT Helpdesk To Deterministically Reduce The Whole Life Cost

Chapter Preface

The chapter provides a discourse on the utilisation of data derived from corporate data systems in reducing the cost of ownership of IT. It is based on a series of case studies carried out on systems in use at Rolls-Royce and Associates Limited. The case studies draw on data contained in the corporate IT help desk system, and system monitors. It describes the methodology used to draw together metrics for failure and use in determining the whole life cost of IT systems. This is demonstrated through examples of cost improvement initiatives applied to Personal Computers.

The chapter concludes with a summary of simple system attributes that can both improve our understanding of the more cost significant system elements, and which can then be used to carry through a TQM cost reduction initiative.
5.1 Introduction

This chapter provides a record of the approach taken to establish the failure characteristics of computing systems and their component parts in use at Rolls-Royce and Associates Limited (RRA). Whole life cost of ownership techniques are used to make comparisons because of the principle stated in chapter 2 that whole life cost is a true and full measure of the quality of an IT component or system (23).

The motivation for the study was:

- Increased understanding of the cost of ownership of software based systems in a business context with a view to minimising costs.

- Assessment of the transferability of failure based methods (of cost and availability estimation) from hardware to software systems.

- An assessment of failure based methods of cost and availability estimation to provide a quantitative measure of different service alternatives.

- Evaluation of a helpdesk in providing the necessary data to support the determination of whole life cost by using failure based methods. This in part provides a solution to the service measurement problem identified.
by Marcella and Middleton (40) in their research into the critical success factors for helpdesk systems.

5.2. Case Studies

Two linked case studies provide the vehicle for demonstrating the methods proposed for the use of helpdesk data. The first is an assessment of PC reliability and maintenance cost, and the second describes the use of reliability data to establish availability of an IT system incorporating PCs. The case studies presented are based on data from the computing systems in use at RRA during 1995 / 1996.

Many of the techniques and definitions used in this chapter are taken from Knezevic (41). The approach used by Knezevic is probabilistic rather than deterministic, however the available data in the case studies is too small to enable a fully probabilistic approach. Simpler deterministic methods have been used where required by small data sets.

All the data used in the case studies have been taken from the systems in corporate use, and hence reflect the environment in which the computing systems are used. Data collection and the data used are not ideally matched to the requirements of failure based methods that have been used. Despite this, when used with care, the
data provide a method of improving the quality and availability of systems in the business environment.

5.3 Case Study 1, Determining Reliability & Cost of Ownership of PCs

5.3.1 The systems in place at RRA to support the case study

All of the data used have been derived from three sources:

(i) The helpdesk in use at RRA,

(ii) The output of the Deskwatch™ network monitoring software

(iii) The register of electrical equipment owned by RRA

Data concerned with network events were extracted from archived helpdesk files. Initial analysis was concerned with categorisation of events into hardware / non hardware 'failures', with later analysis attempting a lower level understanding of failure by ascribing the cause of failure at the sub component level. The RRA implementation of a corporate IT helpdesk system is currently only partial. For example, no record is currently maintained of the actual corrective action taken
following a reported event, the spares consumed in a corrective action, or the time taken to effect a repair. The taxonomy of events was only achieved by inspection of all of the text statements for the telephone reporting. With some sixteen thousand records created in the first year of operation this made the analysis particularly difficult.

Data concerned with the use of hardware and software were taken from the Deskwatch™ system. This network software enabled direct comparison on the use of the various software packages used in the company. Deskwatch™ also provided statistics on the degree of use of individual PCs.

Purchase dates of computing equipment were obtained from the Company's register of electrical equipment.

5.3.2 Approach

This case study focuses on a subset of the PCs in current use. PCs are those for which a purchase date was available in one of the two databases. The PC population in the study is therefore 341 out of more than 1200 PCs owned by the company. The reason for limiting the study in this way lies in a future research intention to monitor a significant body of computers through life. The second case study, covering a further subset of the 341 PCs, focuses on the computers in use by a company business unit to
demonstrate the importance of availability at the system, rather than the component level.

5.3.3 PC Reliability

The 341 PCs in this part of the study were supplied by a number of different manufacturers (Figure 5.1). To test the consistency of data across manufacturers a comparison of days to failure was performed, (Figure 5.2). This figure does show some differences between different manufacturers' PCs, but because the populations are small for all but two manufacturers the differences are disregarded.

![Figure 5.1 Computer manufacturers for the study machines](image)

Figure 5.1 Computer manufacturers for the study machines
As a further comparative test of the failure characteristics of differently manufactured PCs a more detailed study was made of PC sub-component failure rates for the principal suppliers. The data again showed that failure rates for the two main suppliers were substantially similar. Until a more substantial data set is available, the data will be used as a single set, i.e. for this chapter all PCs are assumed to behave approximately similarly from a reliability standpoint. Reliability characteristics will be represented by the single parameter, Mean Time To Failure (MTTF), i.e. the failure distribution will be assumed to be exponential, until a larger set of failure data is available.
To assist in subsequent cost estimation a corrected set of component failure rates has been created for the 341 PCs in the study. Table 5.1 presents failure characteristics as MTTF in hours. At RRA the working day is variable due to flexible working hours. The calculation of the actual working day for a particular PC is therefore not an easy matter. For the calculations supporting Table 5.1 the working day is taken to be 7.5 hours. Similarly, to avoid some difficult programming on the database used, the days of use are taken to be 5/7 of the difference in days between the purchase date and 5th March 1996 (the last failure record in the data set). Strictly this is incorrect because it ignores:

- the national holidays which introduces optimism,
- the day of the week that the purchase was made which introduces both positive and negative errors to the calculation,
- the difference between the purchase date and the installation date, again introducing optimism to the MTTF figures produced.
Table 5.1 Corrected Failure Rates for PC Sub Components

<table>
<thead>
<tr>
<th>Description</th>
<th>Failures</th>
<th>MTTF</th>
<th>Replacement</th>
</tr>
</thead>
<tbody>
<tr>
<td>A DRIVE</td>
<td>19</td>
<td>20456</td>
<td>20</td>
</tr>
<tr>
<td>BATTERY</td>
<td>6</td>
<td>64778</td>
<td>5</td>
</tr>
<tr>
<td>FAN</td>
<td>2</td>
<td>194333</td>
<td>6</td>
</tr>
<tr>
<td>HD</td>
<td>28</td>
<td>13881</td>
<td>179</td>
</tr>
<tr>
<td>KEYBOARD</td>
<td>15</td>
<td>25914</td>
<td>20</td>
</tr>
<tr>
<td>MONITOR</td>
<td>18</td>
<td>21593</td>
<td>169</td>
</tr>
<tr>
<td>MOTHERBOARD</td>
<td>4</td>
<td>97166</td>
<td>150</td>
</tr>
<tr>
<td>MOUSE</td>
<td>24</td>
<td>16194</td>
<td>15</td>
</tr>
<tr>
<td>NETCARD</td>
<td>2</td>
<td>194333</td>
<td>23</td>
</tr>
<tr>
<td>OTHER</td>
<td>14</td>
<td>27762</td>
<td>-</td>
</tr>
<tr>
<td>Complete PC</td>
<td>132</td>
<td>2944</td>
<td>-</td>
</tr>
</tbody>
</table>

Finally, the calculation takes account of the time that PCs are in use as RRA have network software which monitors network use. This includes PCs for all periods of use, i.e. when the PC is network connected or not. This shows that PCs are typically powered for 90% of the core time of 7.5 hours.

This gives the MTTF calculation as:

\[
MTTF = \frac{1}{\sum_{i=1}^N ((\text{Endate} - \text{Idate}) \times 7.5 \times 0.9 \times 5.7) / N}
\]

Where i = Number of the PC in the data set
Endate = 5th March 1996

Pdate = Purchase date for each PC

N = Number of recorded failures for each PC

And where 7.5 is the standard number of hours in a working day

0.9 represents the 90% of the core time

5/7 accounts for the 5 working days in a seven day week.

The cost data included in Table 5.1 are typical values taken from the September 1996 trade magazines for the components listed.

Using the helpdesk data, it has been shown that maintenance costs will not be significantly affected by changing the manufacturers of the hardware. However, the determination of the Mean Time To Failure of the various components enables the calculation of the likely cost of maintenance of the PCs for any given repair or replacement policy. In the next section it is shown how the maintenance option with the lowest through life cost has been determined at Rolls-Royce and Associates.

5.3.4 The Cost of Maintaining PCs

Recent PC purchases have included a maintenance contract extension to provide on site hardware maintenance for the first three years of life. This practice is a poor fit with the corporate view that PCs should be retained for 'at least four years', because
the fourth (and any subsequent) year has to be supported separately to the maintenance contract. Nevertheless the maintenance contracts were seen as a good buy. Therefore an objective of this part of the case study was to establish whether the maintenance purchased was a 'good buy', from a pure cost point of view.

Costs for the more straightforward options for maintenance are shown in Figure 5.3. Given that an initial assessment of a hardware failure is invariably made by Company staff and that the equipment is largely modular it appears that the function of a maintenance contract is largely in the provision of spares. As an assessment of the effectiveness of the current maintenance contract a number of alternative maintenance options were considered as benchmarks. Four of the more obvious, alternative options are:

- Purchase spares support only for the whole of life, i.e. years 1 to 4. Diagnosis and support being provided by Company staff. (Referred to as 'Spares only' in Table 5.2 and Figure 5.3)
- Purchase support for the first three months only (to cover infantile failures), providing support from Company held spares for the rest of life. ('In house')
- Bulk purchase PCs complete with spares for life. ('Bulk purchase')
- Support the PCs using Company staff with spares procurement as follows:
a) a bulk buy of the high obsolescence items (i.e. motherboard, hard drive and fan) which are likely to be obsolete within the life of the PC.

b) spares only support for the low obsolescence items after the first three months

This provides protection from the possibility that a replacement would be unavailable due to obsolescence, but minimises the initial spend on through life spares. ('High Obsolescence')

If the original option of purchasing support for three years ('As is') is included this provides a total of five options in the comparison.

A number of assumptions were made in deriving the cost curves for the five options:

- The labour costs used are the 'without profit' rate for the grade of Company staff used for maintenance.

- Hardware costs are typical values for single purchases from the September 1996 trade press, but reduced by 15% and 10% respectively for bulk purchases in excess of £40000 and £25000.
- Maintenance times are assumed to be average figures of 20 minutes for a visit to either establish the cause or to fix a fault. These figures are not available from the Company database and estimated times are used instead. Errors from these estimates have only a small impact, however, because in practice the costs are small and are present in all maintenance options.

The Net Present Values (NPV) for these options are tabulated in Table 5.2. The discount rate for NPV calculations is 6%. The definition of NPV can be found in section 2.3 of chapter 2.

Figure 5.3 Comparison of PC Maintenance Options
Table 5.2 Net Present Values (NPV) for Maintenance Options

<table>
<thead>
<tr>
<th>Maintenance Option</th>
<th>NPV (£)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spares Only</td>
<td>71592</td>
</tr>
<tr>
<td>In House</td>
<td>88312</td>
</tr>
<tr>
<td>Bulk Purchase</td>
<td>108955</td>
</tr>
<tr>
<td>High Obsolescence</td>
<td>88437</td>
</tr>
<tr>
<td>As is</td>
<td>61630</td>
</tr>
</tbody>
</table>

From the results shown in Figure 5.3 and Table 5.2, it can be seen that the current option, subcontracting the maintenance of PC hardware, appears to be the more cost effective solution. However this is not necessarily the complete picture as we shall see in the next case study.

5.4. Case Study 2, Applying Availability Principles to A Complete Software System

Availability is defined by Nicholas (42) as:

$$A_0 = \frac{MTTF}{MTTF + MTTR + MTTS}$$

where $A_0$ = Operational availability
MTTF = mean time to failure
MTTR = mean time to repair
MTTS = mean time to support

It can be seen that for a given repair scenario Ao will have a maximum or inherent value when MTTS = 0, (that is the time to provide the support actions is zero). MTTF is an expression of reliability, normally quoted in hours, and is defined in the first case study of this chapter. For software systems the concept of reliability provides some difficulties. The British Standards Institution provide a definition of software reliability (43) as 'the probability that the system of which it is a component will not experience a design failure due to a fault in the software under given conditions for a given time interval.' Design failure in this context is the loss of a function which was required by the software system.

This case study is concerned with an area of the Company network; the part chosen for analysis is small to make the analysis more manageable, but the results have relevance for the whole network. In this case study we look at the computing system used by a group of eight engineers. These engineers have access to seven PCs, of varying manufacture and specification, and of varying ages. From the Deskwatch™ software we know that:

- for around 90% of the normal working day the seven PCs are powered up
• the principal software package, WordPerfect™, is in use for 50% of the active time.

In addition it is clear from Deskwatch™ that in this office the active PC use is less than 50%. Taking a judgement on these figures we have assessed that the minimum availability requirement for the PCs is probably 4 out of 7 for the office use of WordPerfect™ to be 'available'. That is, three of the seven PCs could have reduced function before the local computer system was considered unavailable to the engineers.

Knowing the inherent or actual availability of a system is only of value if we also know what the cost of achieving that availability is, and conversely the cost of any unavailability. Recent research by the Gartner Group (44) gave an indication of the cost of ownership of a networked PC as £8460 / year. This figure is similar to the annual cost of a Windows™ 3.1 PC at RRA. Since no company that wants to stay in business will be likely to have assets that are providing a return that is less than the cost, we have taken the quoted cost to be the cost and/or benefit of the PC's function in assessing the availability of the system for a full year of operation. It is conceded that the figure could be higher, but it is extremely unlikely to be lower.
5.4.1 Inherent Availability of the WordPerfect™ Function

All of the intelligence in the previous paragraphs is incorporated in Figure 5.4, a Reliability Block Diagram for the subsystem under consideration. Reliability Block Diagrams are described in (43) as a tool in the assessment of software reliability. As the principal software package is WordPerfect™, only this software function is included in the diagram. Reliability figures (MTTF) are those derived from the earlier case studies, and from similar exercises covering the server, network and WordPerfect™ elements. For the period of the study there were no PC failures recorded with the help desk for any of the PCs. MTTF figures used in the Reliability Block Diagram are therefore those for the full set of 341 PCs in the first case study.

The single printer in use by the group of engineers is also excluded from this study. This in effect defines the function for which we wish to know the availability as 'a word processing function complete with spell checker and drawing package, available on the RRA network and with file storage and retrieval to the server, operating in a Windows™ environment'.

WordPerfect™ 'failures' include any software based query that has been placed with the helpdesk. This of course includes a significant set of 'design failures' of WordPerfect™ training-courses. Ideally these would be shown separately in the Reliability Block Diagram but for simplicity in this analysis all the failures related to
WordPerfect™ are grouped under one heading. It must be stated at this point that the 'failures' described here are almost exclusively due to the lack of understanding of particular features of the software on the part of an individual user and not associated with some inherent defect with the WordPerfect™ software. Finally, the installation of WordPerfect™ is as network software, that is the executable on a particular PC is dependent on the continuing existence of a session protocol between the PC and its network server. This relationship is described by the Reliability Block Diagram.

From the reliability figures (MTTF) shown on the diagram the total system reliability can be determined by use of the following formulae from Knezevic (41)

5.4.1.1 Series Configurations

\[ R_s(t) = \prod_{i=1}^{NCI} \left( 1 - e^{-\frac{t}{\Lambda_i}} \right) \]

where \( R_s \) = system reliability

\( t \) = the required time for the system to be functional

\( \Lambda \) = the scale parameter of the exponential distribution

\( NCI \) = the number of consisting items
5.4.1.2 r out of n Parallel Configurations

\[
R_d(t) = \cdot (\binom{N}{x} \cdot R_i(t)^x \cdot (1-R_i(t))^{N-x})
\]

where \( R_i(t) \) is the item reliability

\( \binom{N}{x} \) is the total number of combinations of (NCI) consisting items taken \( x \) at a time.

Using the formulae in paragraphs 5.4.1.1 and 5.4.1.2 the maximum availability that can be achieved (Inherent Availability) is 99.99%, and the availability achieved with the current support options is 92.58%. 
Figure 5.4 Reliability Block Diagram

These results are based on MTTR and MTTS derived from:

a) the helpdesk data for elapsed time between start of an event and its subsequent clearing by the helpdesk staff (MTTR + MTTS)

b) interviews with staff and contractors (on the proportion of the timing in (a) that is related to the active repair time - MTTR)
This results in a combined system value for MTTR, and MTTS of approximately 11 hours.

In monetary terms this leads to losses to the organisation of the difference between $A_{inherent}$ and $A_{actual}$, approximately equal to 100 hours / year for the seven PC sub system shown in the Reliability Block Diagram. After adjustments for the 50% usage of WordPerfect™ determined from the Deskwatch™ software this equates to approximately £59000 / year for the population of 341 PCs, or about £172 / PC.

5.4.1.3 More Sensible Options?

If the Reliability Block Diagram is inspected it can be seen that major causes of loss of availability stem from the use of a server and network. In this particular group of engineers the losses due to the PC unreliability and associated maintainability and supportability problems are offset by the use of seven PCs in parallel. A number of system configuration alternatives could be considered to increase availability and / or decrease whole life cost but these are beyond the scope of this chapter. For the next section the availability for the system configuration shown in the RBD is used to aid decision making.
5.4.1.4 Returning to PC Maintenance

The figures for availability that are provided above now provide access to an additional cost consideration for PC maintenance. The cost of availability was missing from the cost profiles generated for maintenance and shown in Figure 5.3. Figure 5.5 shows adjustments for the added cost of unavailability.

Figure 5.5 shows a very different picture to Figure 5.3. The lowest cost option can be seen to be the 'Spares only' option. This has a Net Present Value of £71592, compared to the new figure of £89964 for the 'As is' maintenance arrangements. The Company is reverting to a single on site maintenance organisation following
a) the work leading to this report, and
b) problems with loss of availability.

5.5. A Generalised Method for Establishing the Cost of Ownership

5.5.1 The Process

The process shown in Chapter 2, Figure 2.1 provides a model of the process of 'Design to Minimum Whole Life Cost'. This section provides attributes that are needed to support the whole life cost approach defined in chapter 2.
5.5.2 Simple System Attributes That Support The Method

Fig 5.5 Comparison of PC options

Any attribute of the system that will act as a measure of cost should be measured, either directly or indirectly. The helpdesk can support measurement of some of the attributes. They include:

- **Consumables**
  
  eg Pages printed for laser printers (RRA have used toner replacements as an indirect measure of printer use). These two measures are examples of direct and indirect measures of use.

- **Failures of all components including training, and documentation of hardware and software**
• The corrective actions taken following a failure.

• Time to correct failures or replacement of consumables, and the split between the actual repair time and any waiting time

• Other software eg Deskwatch™, can monitor the use of hardware and software, which provides the data for statistical calculations.

5.6. Conclusions

1) Whole life costing, including both hardware and software elements can provide real cost benefits in shaping IT system maintenance philosophy. Purchase and obsolescence could also clearly benefit from the same technique, although this is beyond the scope of this chapter.

2) Failure based methods of cost and availability estimation have been shown to be appropriate for IT components and systems and provide a means of determining the maintenance component of the whole life cost of software, a measure of quality of the delivered IT.
3) The data required for the whole life cost / availability estimation can be derived from a combination of a corporate IT helpdesk and system monitoring software.

4) Helpdesk software could be improved by incorporating more detail, for example the number of pages printed by a printer, time to effect corrective action, and the detail of the actual corrective action. Whilst some of these may be available from helpdesk software they are not always used.
CHAPTER 6

Using the helpdesk to probabilistically reduce the whole life cost

Chapter Preface

The running cost of over 300 PCs has been studied at Rolls-Royce and Associates. The company helpdesk system was employed to provide detail of the frequency and nature of the PC component failure and the time taken for the repairs. The cost of unreliability was then calculated as the cost of replacing each faulty component plus the cost of unavailability while waiting for the repair.

It was found that for most PC components there is a higher probability of failure, leading to a higher cost of unreliability, in the early part of life. The conclusion is drawn that bulk purchase of PCs will lead to higher running costs in the initial period and for many organisations a rolling replacement policy would spread costs more evenly. The helpdesk was found to be an excellent source of information for improving the quality of maintenance procedures.
6.1 Introduction

During the period February 1995 to November 1996 a case study to examine the cost of owning a PC was carried out on PCs in use at Rolls-Royce and Associates Limited (RRA). This study focused on a subset of the PCs in current use. PCs studied were those for which a purchase date was available and which had been the subject of reporting through a central IT helpdesk system for software and hardware failures. The PC population in the study was the 341 PCs reported in the last chapter. In the previous chapter and the associated paper, Bradley & Dawson (36) reported interim reliability results and quoted the failure rates for the PC sub-components as Mean Time To Failure (MTTF). It was recognised at that time that the adoption of a probabilistic approach to financial and reliability decisions might be desirable, as it would provide more accurate and dependable information about costs over time. This chapter provides an update on the PCs being studied, a view on the distributions that best describe the PC component failures and the financial implications. In examining the PCs, costs were calculated based on both the costs of replacement of component parts and the costs of un-availability resulting from the PC failures.
6.1.1 The specification for the sample set

The machines being monitored were generally to the base specification for the company at the time they were purchased. Most were fitted with a 486 processor, although the full breakdown on processor is given in Figure 6.1: all had a 3.5" A drive, 14" or 15" monitor, keyboard, mouse and a hard disk drive. Hard disk drives varied widely in size, but the typical size was less than 500 Megabyte. Random access memory was generally 8 Megabyte but was increased in a small percentage of machines to 16 Megabyte. Few machines had a CD ROM fitted as standard and these are not reported. The specific processors covered by this study are mainly the DX 50 and DX 66 but the detailed results show no differences between their failure characteristics and any other individual processor type in the study.

6.1.2 The Changes at the Company

Since the start of this research, Rolls-Royce and Associates Limited has outsourced its IT function. It might have been expected that the need for the company to have a view of helpdesk data would have diminished, but as will be shown later the need for financial information is even more focused.
6.2 Deriving Failure Profiles

A PC can be regarded as a modular assembly, and one that can be constructed from the assembled component parts in a matter of fifteen minutes or less.

The failure probability of a component or system can be estimated experimentally from field data, provided the population studied is reasonably large and the rate of occurrence of failure is sufficiently high as to provide a high confidence level in the derived failure probabilities.
For this chapter a large body of computers was studied for a period of a little short of two years. During that time the majority of sub component design categories experienced more than thirty failures each. A probability plot for one sub component is shown at Figure 6.2. The plot is for the population of hard disks in the 341 PCs studied. The distribution is a Weibull with a shape parameter of 0.518. The Weibull distribution is fully described in the context of engineering reliability by Knezevic (41) but a Weibull distribution with a shape parameter approaching unity is consistent with an Exponential distribution as can be shown by the following taken from the Microsoft Excel™ Function Reference,

$$f(x; \alpha, \beta) = (\alpha \beta^{\alpha})x^{\alpha-1}e^{-\beta x}$$

when $$\alpha = 1$$, $$f(x; \alpha, \beta) = \beta e^{-x\beta}$$

where $$\alpha$$ is the shape parameter and $$\beta$$ is the scale parameter or the mean. An exponential failure distribution exhibits a constant failure rate, and it is therefore legitimate in such cases to use the Mean Time To Failure (MTTF) as the measure of reliability. A shape parameter, $$\alpha$$, which is less than unity implies that the failure rate is not constant with a tendency for more early failures. Similarly a shape parameter above unity implies a tendency for more later failures. The importance to this chapter
is that it shows the failure characteristics of the majority of PC components tend towards early failure.

Only one sub component within the PC population under study had a shape parameter approaching unity. This was the motherboard with a shape parameter of 0.99. This sample of the total population suggests a MTTF of around 49000 hours for a motherboard. The motherboard, with a near exponential distribution could be expected to yield failures at an approximately constant rate. The remaining sub components would be expected to yield more failures in the early part of the computer's life.

Using the statistical analysis software tool MINITAB™, the results given in table I have been obtained from source data given by the RRA Helpdesk. Data for this study has been collected over a period of approximately two years, giving a total PC usage of over 600 PC - years.

A limitation of this approach (i.e. of fitting empirical data to a standard probability distribution) is that the output of the statistical analysis software provides a projection beyond the time to the last recorded failure. As can be seen in later chapters this is not a problem in a PC environment as with the data available the optimum time to replacement of a PC is significantly less than the time to last failure record, i.e. the data projection from Minitab is not actually required.
Table 6.1 Experimentally determined values of Weibull parameters - PC sub components

<table>
<thead>
<tr>
<th>Equipment</th>
<th>Alpha</th>
<th>Beta</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mouse</td>
<td>0.86</td>
<td>22440</td>
</tr>
<tr>
<td>Motherboard</td>
<td>0.99</td>
<td>49171</td>
</tr>
<tr>
<td>Keyboard</td>
<td>0.76</td>
<td>41919</td>
</tr>
<tr>
<td>Hard Disk</td>
<td>0.51</td>
<td>136752</td>
</tr>
<tr>
<td>A Drive</td>
<td>0.55</td>
<td>196436</td>
</tr>
<tr>
<td>Monitors</td>
<td>0.76</td>
<td>58395</td>
</tr>
</tbody>
</table>

Figure 6.2 Weibull Probability Distribution – Hard Disk

The data from Figure 6.2 is redrawn at figure 6.3 in the more convenient representation of the Cumulative Density Function. The steeper curve during early life shows a probability of failure that reduces over time. The full study has resulted in similar outputs for the major PC components with the exception of the motherboard.
The Weibull parameters for each component are provided in Table 6.1, whilst the full set of results are displayed graphically in Appendix 2.

![Cumulative Density Function - Hard disk](image)

Figure 6.3 Cumulative density function – Hard disk

### 6.3 Calculation of ownership costs

The calculation of the post installation cost of computer hardware ownership can be summarised as follows:

\[ C_o = \text{Cost of component replacement} + \text{Cost of lost availability} \]

\[ = \sum_i P_i \cdot N \left[ C_{c_i} + C_{r_i} + C_{a_i} \right] \]
where

\( C_c = \text{cost of the replaced component} \)

\( C_r = \text{(labour) cost of replacing the component} \)

\( C_a = \text{cost of lost availability to the organisation} \)

\( P_t = \text{Cumulative probability of component failure at time } t \text{ after installation} \)

\( N = \text{Number of computers} \)

\( n = \text{number of components in the computer} \)

### 6.4 Calculation of Replacement Costs

The collected reliability profiles from all the major PC components have been assembled and used to establish the direct cost of unreliability of the complete PC, and the indirect cost of unavailability. This is provided at Figure 6.4 for the PC and its major components for the full population of 1000 computers in the RRA network.
Figure 6.4 shows the costs accrued over a period of 7000 hours (approximately four years) in equal time slices of 500 hours. For each timeslice the number of failures for each component design that could be expected was calculated in an EXCELTM spreadsheet. This was achieved by substituting the Weibull shape and scale parameters from Table 1, together with the time to date in hours, into the EXCELTM Weibull function. The replacement cost calculated in each case then being the sum of labour and material costs for the individual component. Labour costs used are those applicable in the company prior to outsourcing. The labour times used are those derived for the individual components from interviews with staff and contractors on the proportion of the time that is related to the active repair.

In our model each PC component is treated independently such that if it is replaced it will count as new, with the failure probability of a new component, while all other components in the same PC continue their ageing. The additional, replacement components therefore create new sample populations for each new time slice, giving a new but smaller population for each time slice. This process continues for each subsequent time slice until there are too few components remaining in any population to perform meaningful calculations.
6.5 Calculation of Cost of Un-availability.

The cost of un-availability used in the financial arguments was derived in the previous chapter and is also given in Bradley & Dawson (36). The method is summarised here, amended to reflect the use of probabilistic rather than deterministic methods.

As stated in the previous chapter availability is defined by Nicholas (42) as:

\[ A_0 = \frac{MTTF}{MTTF + MTTR + MTTS} \]

where \( A_0 \) = Operational availability

MTTF = Mean time to failure

MTTR = Mean time to repair

MTTS = Mean time to support

Unavailability can be represented as the time that the PC is not available, or

\[ (1 - A_0) \]

Time lost to unavailability can therefore be represented as:
(1 - A_o) * Hours in the period

Knowing the lost availability of a PC is, however, only of value if we also know what the cost of the unavailability is to the organisation. As in the previous chapter the cost of ownership of a networked PC of £8460 is taken as the cost of unavailability.
6.6 Conclusions

1. Because PC component failures appear to be skewed to the early part of life the cost of unavailability and unreliability is similarly skewed into the early life of the machines. This is important for purchasers and maintainers. Where large numbers of machines are replaced at the same time in an organisation the immediate effect is likely to be a significant number of early failures - damaging to the organisation's operational availability, and a busy period for the maintainer - possibly damaging to the reputation. This suggests that a programme of rolling replacement for PCs would be less damaging for many organisations.

2. Helpdesk data can be used to provide the costs essential to the planning of a quality maintenance programme, as demonstrated by Bradley & Dawson, (45). The data collected here has been used in a very specific way, to provide through-life costs for the PC population. This is a use for Helpdesk information not normally considered, e.g. Marcella & Middleton, (40). It is important to recognise the relevance of service metrics to the core business regardless of the presence of an outsourcing agent. Helpdesk data can be used to establish the actual cost of the maintenance operation, and then used to confirm that a fair price is agreed in an outsourcing agreement.
CHAPTER 7

Determining the Significant Costs of Software Systems

Chapter Preface

The increased use of computers and computer software means that software procurement decisions are frequently made by inexperienced people. The importance of full whole life costing is becoming recognised.

Historically, software costing techniques have been based on the initial purchase and development costs. Experience at Rolls-Royce and Associates has shown that when full whole life costs are examined for database systems the data maintenance costs are dominant.

Methods taken from the construction industry show that a simple method for costing systems can be based on the most significant 80%. This has the implication that whole life costing of database software systems can be based on the costs of data maintenance without the need to estimate costs of code implementation.

7.1 Introduction

Increased use of computers and computer software in most Industrial and Business applications create the need for whole life costing. Increased use exposes increasingly greater numbers of personnel to software procurement decisions.
Decision takers are often ill equipped to take the decisions. The construction and military hardware industries are already rationalising the procurement decision by incorporating the requirement for a whole life costing estimate, including maintenance, into invitations to tender. Historically software life cycle costing has been concerned only with the software element of the complete system. Typical among life cycle models is the waterfall model described by Boehm (14). Even so it has long been known Balzer et al (46) that the maintenance cost of even the software elements can represent 80% of the total cost.

Previous authors working in the military, e.g. Beddoes (47), have based whole life cost estimates on a cycle time of 25 years; this cycle time is more usually associated with military hardware. It is suggested that for commercial applications a software life cycle of more than ten years is unlikely. Commercial & military organisations have different strategies for coping with the impact of hardware and software obsolescence. The relational database used for the first generation of the Rolls-Royce and Associates' database systems saw only eight years of service. There is no reason to expect the second generation will be significantly different. The commercial computer systems described here are costed on the basis of a ten-year service life.

A definition of software system maintenance is also important. The normally accepted definition is suggested to be inadequate. This might typically be 'the addition, adaptation or correction of the software code first delivered'. A more
appropriate definition, and one that describes the complete software system would include reference to:

- Development and maintenance of code
- Development and maintenance of hardware
- Data maintenance

'Development and maintenance of code' is a recognised part of software whole life costing, and there is an obvious need to carry out hardware maintenance of the complete computer system. Data maintenance is an ongoing activity in database systems. Typical of the activities covered by this category of maintenance is:

- Addition and update of reference data in the database
- Re-indexing or redistribution of files within the system disc drives.
- Backing up the database for file and commercial security.

Many of these activities will remain unseen to the majority of users, indeed organisations may post the costs of the activities to an overhead account and they may be unseen by even the application owners. Cost estimates that exclude any of the maintenance categories will not be a total cost of ownership.
7.2 Experience of Whole Life Costing at Rolls Royce and Associates

Experience of these costs has been gained on two software based projects at Rolls Royce and Associates Ltd. These projects are PIMS and CAMS. PIMS is a computerised configuration management system for engineering parts. CAMS, the asset management system, is the maintenance adjunct of PIMS. An earlier system of the same name was written in COBOL on a MIMER RDBMS. Both CAMS and PIMS applications are interactive with a total of around 400 input and retrieval screens. The PIMS/CAMS suite is a 4GL application designed to run on INGRES™ on a SUN™ UNIX network. The CAMS application was subject to a detailed design phase with a long user consultation period to establish an extended requirement. No formal methodology was in use in the company at the time and the analysis was recorded in natural English supplemented by Frame Flow diagrams, an INGRES™ design method.

The PIMS system was conceived in 1990, with the design and coding taking place during the following three years. In this time the development environment was subject to considerable change. Project management, development methods, development tools and software configuration control all changed at least once.

Both software systems were designed and brought into use as 'Projects'. This project emphasis caused difficulties to the cost collection activity. Cost categories for each of
the systems were not located in single accounts. Instead the company had arranged for several budgets to be used for different parts of the delivered system. This unfortunately makes collating the information difficult and it was not until the research for this paper was undertaken that the full picture of the whole life costing became evident.

Cost categories collected were those associated with the waterfall model, except that:

(a) the requirements specification and product design have been incorporated into 'Design'
(b) detailed design and coding have been combined with system integration
(c) additional categories have been added to cover licensing and support for the RDBMS
(d) hardware related categories have been added.

Changes (a) and (b) were forced by the cost recording arrangements. (c) and (d) are categories contributing to the cost of ownership but not dealt with by the waterfall model. The two applications are analysed separately in the next sections.

7.3 The CAMS System Whole Life Costs

The proportions of these categories are shown in Fig. 7.1 for CAMS. Costs are shown here as accumulating percentages of the total cost of ownership. The largest category
is shown to be 'Data maintenance'. Over 80% of ownership costs are contained in half of the ten categories. Code related costs are all contained in the smaller categories.

Data entry, other than for monitoring and maintenance purposes is excluded from the costs of ownership. Data entry activities are costed separately and as part of the underlying tasks and are excluded as they would exist if the database project had not been started.

The cost categories in Figure 7.1 are all directly related to the application with the exception of the licensing and support for the RDBMS. The costs of these two categories are related to the size and number of applications using the RDBMS. The greater the number of applications the lower the cost per application. It is therefore sensible to remove the two categories. This is shown in Fig 7.2.

With the removal of the RDBMS costs the largest three categories; Data Maintenance, Design and Hardware Maintenance represent approximately 80% of the total cost of ownership. Again the code related categories are contained wholly in the remaining 20% of Whole Life Cost.
7.4 PIMS Whole Life Costs

The whole life costs for the PIMS system are represented in the same format as those for CAMS. The overall cost breakdown is shown at Fig 7.3. Although the proportions of cost are changed they are to be expected as the PIMS system was created with little initial design, with programmers largely copying an earlier system. This has meant that the lack of time spent in the design phase of this project has caused more time to be spent in the coding phase.

Figure 7.1 Cams Life Cycle Costs
Figure 7.2 CAMS Costs exc. Licence

It is believed that had the system been fully analysed and designed in the first instance the coding effort would have reduced in significance, giving a similar cost graph to the CAMS project.
CAMS was designed with a significant initial cost reflecting a new and novel application. The importance of Fig 7.3 is that the dominant cost is that of data maintenance. Removing the RDBMS licence and support cost shows a similar picture to the CAMS system with the top four categories identical.

7.5 Using Cost Categories for Estimating Full Whole Life Costs.

The construction industry has recently established through Asif (48) an estimating tool for use in the early stages of a project to avoid the need to calculate bills of
This relies on the relationships between cost categories for a class of deliverable. Research has shown that typically 80% of cost may often be found in only 20% of the cost categories. Historical records of these cost significant categories can then be used to establish an early estimate of total cost for a particular instance of a deliverable. Typically

\[ \text{Cost Estimate} = \text{Estimate for Significant 20\%} \times 0.8 \]

for a deliverable where 80% of cost is vested in 20% of the categories. This is illustrated by considering the costing of buildings. Typically the costs will primarily consider the area of flooring rather than the cost of assembling each brick. In database systems the data can be considered to be the "flooring" and the code can be considered to be the "bricks" of the system.

Significantly, for database software systems it does appear that where a full design method has been employed, the software implementation costs could be ignored when estimating the whole life cost. This will clearly simplify the cost estimation process.
7.6 Attempted application of existing development estimating algorithms to RDBMS applications

Although, from the preceding paragraphs, it appears possible to estimate the whole life cost of a database application based on accurate cost estimates of the design and maintenance phases there will always be a requirement for project managers to estimate subtasks separately. The major subtask of the implementation phase is probably the software development itself. The estimation of this task is therefore addressed in the following paragraphs. Initially at the Function Point technique and also the Putnam estimation model were examined as possible methods of estimating database applications in Rolls-Royce. Both these techniques were disappointing when applied to the CAMS system described earlier.

Function Point Analysis was performed on the specification, some adjustment being made for the differences in terminology, e.g. Number of Files was interpreted as number of Tables, and Number of User Enquiries equated to Retrievals. With these adjustments the spread of function point productivity ranged from 0.3 to a little over 2 FP/ Hour. It was felt that the analysis was therefore inappropriate to the 4GL under review.

The Putnam model, when applied to the whole of CAMS, gave a Technology constant of 51000, but values of approximately half that value for the five CAMS modules. Pressman (38) quotes typical values of between 2000 and 11000. Fenton (49) advises
the use of the Putnam model on projects of > 70000 lines of code. CAMS has a line count of 54000 including the comment and SCCS header lines. Consequently Putnam, and other Lines of Code estimating methods were abandoned.

7.7 Proposed metrics for 4GL applications

A different approach was required, and the personnel who had designed and coded the CAMS system were interviewed to obtain views on the most cost significant aspects of the design, from the point of view of initial coding. The most striking feature of the interviews was the differences in opinion. The three interviewees had wide programming experience, with approximately one years experience of Ingres™ at the time of the interviews. The developer comments were distilled to a small number of common parameters:

- Number of screens in the interactive application
- Number of fields on the screens
- Number of SQL database operators in the application (e.g. Select, Delete, Insert)
- Number of user functions in the application (e.g. Print, Exit, Previous)

These were then subject to regression analysis, using the data available from the CAMS project. Initially the data was analysed in its raw form with a significant lack
of success in correlating the various parameters. After this initial disappointment the whole data set was more closely analysed and in some cases cost / times were reallocated into more appropriate categories. The regression analysis was then repeated and the following relationships observed:

- No correlation existed between the number of user functions and the development effort.

- An apparent linear relationship existed between the number of screens + number of fields and the development effort consumed.

- A linear relationship with a correlation coefficient of 0.96 was found between the screens + fields + database operators and the development effort as shown in equation 1, where a and c are both < 10 and b < 1.

  \[ \text{Hours} = (a \times \text{fields} + b \times \text{SQL Operators} - c \times \text{screens} + K) \quad (1) \]

- Equation 2 shows regression of the same three parameters (screen fields + SQL operators + screens) and reported defects in the code. This also provides a linear relationship, this time with correlation coefficient of 0.97. Values a, b and c have values < 1.

  \[ \text{Defects} = (x \times \text{fields} + y \times \text{SQL operators} + z \times \text{screens} + K') \quad (2) \]
These last two relationships provide some further interest because they confirm anecdotal observations made by the author whilst using Ingres™ 4GL:

- The same functionality provided on a greater number of screens appears to cost less to design and code. (Within the limits of available data)
- The same functionality provided on a greater number of screens appears to cost more in reported coding errors.

Combining these two relationships it is clear that an increase in the number of screens for a particular functionality will lead to a decrease in development costs but at the same time an increase in errors and correction costs.

Author's note: The exact values of the parameters are in effect a statement of the effectiveness of the Rolls-Royce development team. As such it is not possible to publish exact values.

7.8 Conclusions

- There does appear to be evidence to support the construction of a cost estimation technique based on the design and maintenance phase costs for database software systems. It may be possible to establish whole life cost estimates quickly, based only on data maintenance, hardware maintenance and initial design estimates. Avoiding the estimation of the implementation costs of coding, testing and installation will greatly
simplify the estimation process enabling software tenders to be made or evaluated quickly and with greater confidence.

- Existing cost estimation methods based on Function points and Lines of code appear not to fit the 4GL applications used in RDBMS systems. It would appear that estimating for high level languages need higher level metrics.

- Initial investigation shows that the number of screens, fields and SQL operators may give simple but effective metrics for development cost estimation.

Authors note to the Chapter:

The algorithms described in this chapter have been used subsequently to estimate the development time for the application described in chapter 11. The estimated time for development was:

(a) Within 5% of the subsequent actual

(b) almost double the development team estimate

The development team were blind to the authors estimate. Whilst this single event is hardly proof that the estimating method is robust, it is an encouraging start. New
Ingres™ applications are not common in the organisation, consequently it may be some time before further estimation is required.

Future research efforts need to concentrate on verifying the simple estimation models proposed. Clearer definition and possibly expansion of the cost categories for the Pareto based model is also required. In addition the predominant cost of the data maintenance is likely to be affected by the development tools and methodology. Research is required on the effects on the whole life costs of varying:

- The database management system
- The 4th generation application development tool.
- The design methodology

with the intention of producing a Whole Life Cost Estimation tool with wide application
CHAPTER 8

Data Collection Versus World Data

Chapter Preface

This chapter presents learning experiences in Software Engineering resulting from work carried out by Rolls-Royce and Associates Ltd and Loughborough University and relates the relevance of these experiences to other industries. The reasons for collecting data are explored in the software context and specifically the Whole Life Cost of database systems. The dangers and difficulties of using software - industry data are explained, against the background of four software case studies. The case studies are centred on computerised inspection, maintenance and failure databases. Whole life cost profiles are provided which show that the significant costs are not those generally accepted by the software engineering fraternity and emphasises the need to collect local data.

In focusing on the cost relationships that cause these high cost categories, optimisation of second generation designs becomes a possibility. The chapter concludes that for database systems the most likely significant costs will be those
associated with data in the operational phase of the life cycle. The use of genetic algorithms is proposed for the optimisation of system designs based on collected data. Using methods proposed, at the design stage, will give the maximum cost benefit, which in turn may influence decision makers.

8.1 Why Collect Data?

The collection of data is driven by the need for greater understanding and control of processes and products. As an example, engineering designers seek information on the performance of existing designs as a basis for providing new and improved designs. At Rolls-Royce and Associates this has resulted in the implementation of computer based systems for the collection of in service data for the company's engineered product. Traditional, iterative design allows this data collection activity. Very small changes in the design of a product are seen to be desirable and are enabled by the act of observing the performance of the current design. The discrete design changes that result generally improve the design with a minimum of risk. At Rolls-Royce and Associates many of the company's hardware designs are required to have a very high inherent reliability, typically better than $10^{-6}$ failures per hour of operation. Often these designs are in small installed populations which prohibits the collection of in service failure data, as the failures occur so infrequently. Where the populations are larger and the designs are stable, then performance data is recorded in detail. The resulting failure statistics are used to confirm the bottom up failure rate estimates.
produced at the design stage. The desired end results are cost reductions and improved availability, from changed maintenance regimes. In this environment gradual improvement, the Japanese 'Kaizen', can be seen to be desirable. Small changes can be argued not to affect data collected over a long period and deterioration of a particular product can be detected by continuous data collection and monitoring.

This comfortable position is disturbed by large step changes. New requirements, technology or materials can provide a major design problem when there is no prior design as a starting point.

There are a great many examples of projects that have included challenging new designs failing in one way or another. The channel tunnel and the Humber bridge are two examples from civil engineering where failure has been financial. There are doubtless some similar examples in other engineering fields.

Being in a challenging new design situation is the norm for the software industry. Most projects are started with some significant step change, and change is usually a continuing feature through the life of a project. Change might be a new language, new application, new and untrained staff, new hardware, or a new communication protocol.
One of the case studies discussed involved changes to the following aspects of the design compared to previous projects completed for the company:

- Requirement
- Hardware
- Relational database and language
- Operating system
- Design method
- Configuration management tools
- Project management method

It is therefore small wonder that software based systems run into problems. The software industry has recognised that it has at least one problem, this is the so called 'Software Crisis'. This is software engineering's apparent inability to deliver quality software on time and to budget. Examples abound of software projects that are either late or are cancelled because of lateness. It is quite normal for software projects to be twice the originally quoted price. This same software crisis though has a beneficial side effect. Software engineers world wide have been measuring anything that can be measured in an effort to gain control and conquer the crisis. Gibbs (50) writing in the September 1994 issue of Scientific American notes: 'Intuition is slowly yielding to analysis as programmers begin using quantitative measurements of the quality of the software they produce to improve the way they produce it.'
The net result has been the growth of an industry within an industry. Software engineers call it "Metrics", the business of taking measurements. This relatively new software industry has grown methods and analysis techniques to direct the process of measuring, in its own industry. Industry and academia have in many cases collaborated to gain a better understanding of software. The collaboration between Rolls-Royce and Associates and Loughborough University is one of many. Rolls-Royce and Associates also has its own independent software metrics programme covering its various software contracts.

In the UK the AMI Consortium (Application of Metrics in Industry) have produced a guide (51) to their quantitative approach to software management, which includes a basic set of software metrics, whilst in Canada the Software Productivity Centre have created their Metricate program for software development companies interested in starting a metrics program.

All of this activity has resulted in software tools for data collection and, importantly for the discussion on data, measurement of the tools used for data collection. Data collection in our research area has been concerned with the cost of ownership of database systems. The importance of this work will become clear in the rest of this paper. Firstly we should examine some reasons for collecting data specific to our own product.
8.2 Why should we not use world data?

A reasonable approach to data requirements might be to minimise the organisation's own costs by sharing experience with others in related fields. This strategy of course does have its place, e.g. in situations where the failure rates are very low or the product is genuinely identical in form and use. But it also has a major drawback. In software, as with other disciplines the product may be specific to manufacturer and application. An example from real time software demonstrates this. Keene & Keene (52) conclude that the largest contribution to software whole life cost is the maintenance costs associated with the fielded code. This conclusion was reached in the context of real time software. As we shall see from the following collection of case studies this conclusion has been found to be invalid for different product forms, suggesting that it might be wise to collect at least some local data rather than rely on the wholly generic.

8.3 Some Database Application Case Studies

Four case studies identified as A, B, C and D are described briefly below in the context of a study of their whole life cost. Whole life cost is defined as the 'sum of all the project costs that would not have been accrued had the project not been started'.

The categories of cost associated with whole life cost are:
Development

Production

Operation

These can be further broken down, for a software project, as follows:

Project management - Development

Design - Development

Code implementation - Production

Hardware procurement - Production

User Test - Production

Hardware Maintenance - Operation

Code Maintenance - Operation

Data Maintenance - Operation

Case Study A

This system is an interactive asset management application with a total of around 60 input and retrieval screens. The suite of screens is a 4GL application designed to run on Ingres™ on a SUN™ UNIX network. It has a user population less than twenty in number. The customer requirement was a reduction in cost of maintenance.
Case Study B

The second system is again an interactive Ingres™ application. The application is smaller, with only six main screens and with less than five users. It is concerned with recording and manipulating equipment reliability data. It was designed to provide feedback to reduce maintenance.

Case Study C

System C is another Ingres™ application running on a SUN™ UNIX network. The system is unusual in that it has no users. The application is a central data repository for component inspection data from a number of satellite databases. All data is loaded through overnight batch runs from the satellite systems. This does not remove the cost of data maintenance since database administrator costs are still a significant contributor to whole life cost.

Case Study D

The final system is substantially different to the first three. The main source of difference is the use of Visual Basic™ to create the application. The system records environmental data and has a small user base of less than twenty.
All the case studies discussed have significant whole life costs, ranging between £130,000 and £1M for a nominal ten year life. A ten year life cycle is considered to be the maximum life of a system without a major upgrade to function, software or hardware. Cost categories are those for which costs are generally grouped in any software organisation and which cover the complete life cycle of the systems concerned.

Figure 8.1 displays the data for the four systems as accumulating curves of the percentage of whole life cost for each cost category. It can be seen that the significant category for all four of these database systems is the cost of maintaining the data in
the operational phase of the life cycle. The average contribution of the operational
phase (the sum of the maintenance activities) for the four case studies is over 50% of
the total whole life cost for this class of software system. This is not consistent with
the normally held view (quoted in Reference 52) that maintenance of the fielded code
(code maintenance in Figure 8.1) is the most significant cost in the life cycle. Case
studies A, B and C are from the Rolls-Royce environment whilst Case study D is from
Loughborough University. The different profiles of the curves from the two different
organisations suggests that collecting data specific to the application and environment
can be a wise move.

Combining and smoothing these four projects into an envelope of a pair of curves
provides the results shown at Figure 8.2. A further inspection of the two curves also
shows that for these four databases, chosen for their differences in use, the
relationships between categories are the same. The design costs are the second most
significant category, hardware the fourth and so on. The range of cost distribution,
however, does seem to vary significantly for different database type.
8.4 Learning from the Case Studies

Having discovered that the most significant part of the whole life cost of a database system can be expected to be in its operational phase, the next objective is the exploration of the design parameters that influence these significant whole life cost categories. This understanding provides cost relationships that can be used in a model to provide optimised second generation designs, optimisation being carried out on the whole life cost, including hardware, software, data and user costs.

Database systems are invariably used to support decisions in the business. The cost of ownership is a direct overhead to the business and the whole life cost needs to be minimised for a required level of decision support. The commitment curve from
Fabrycky (54), shown in Figure 8.3, shows that opportunity to efficiently minimise the whole life cost of these categories is only in the design phase of a project. After this the potential to amend the design to reduce the operational cost rapidly reduces.

8.5 Optimisation using Genetic Algorithms

Genetic Algorithms, previously used with some success in the early design optimisation of a wide variety of systems, are described in detail by Davis (24). Genetic Algorithms are now being focused on data systems. Early indications are that the following method is appropriate to derive an optimum design to minimum whole life cost.

i) Deriving through brainstorming techniques a causal network of the relationships affecting the whole life cost of an information system incorporating a relational database application. An example of a causal network is shown in Figure 8.4.

ii) Creation of a list of relations from the causal network in step 1.

iii) Deriving by experiment a set of heuristics and constraints for each relation

iv) Using the heuristics and constraints, create a mathematical model incorporating a Genetic Algorithm (GA) to establish the minimum whole life cost design.
Use of experiments to derive cost relationships is not essential, work on physical systems at Rolls-Royce, described by Pearce (5) has shown that expert opinion can be used within a whole life cost genetic algorithm instead of experimentally obtained data, however the requirement for audit may prevent widescale use of such opinion.

An early version of a causal network for relational applications, shown at Figure 8.4, demonstrates the extreme complexity of the relationships and the number of interactions that can affect the whole cost of a data system. This very complexity is the main justification for the use of historical data and genetic algorithms. The search time for optimised solutions on large problems is reduced by the use of the GA, whilst the use of historical data increases the probability of an accurate solution.

![Figure 8.3 The Commitment Curve](image-url)
8.6 Conclusions

This chapter emphasises that the collection and storage of data costs money. A breakdown of costs can reveal that the more significant cost categories may not be as expected and local information is needed to gauge the major contributors to the whole life cost of the system.
Experience has shown that it is the operational costs of relational database applications that are the more significant and may even outweigh the desired benefits.

Once the significant cost categories are identified system whole life cost can be minimised using optimising methods. This chapter proposes a method that uses existing or experimental data to derive the heuristics and constraints that can be employed in a genetic algorithm to design a system with the minimum whole life cost. Clearly, such methods need to be employed in the design stage to maximise the benefits.
CHAPTER 9

Cost Effectiveness

Chapter Preface

The Lifetime Support Department at Rolls-Royce and Associates Limited has a role that is focused on providing the Customer of the nuclear steam raising plant with '...lifetime management that maximises submarine availability with minimum upkeep costs, consistent with plant safety and operational flexibility.' as defined by the Lifetime Management Strategy (57). A central need for management of Lifetime Support is information on the reliability of all the items fitted in operational plants. This chapter presents experience of creating and operating a database that provides component reliability and maintainability data, providing lifetime management staff with information to allow a focus on plant items that cause either low availability or high through life costs. Statistical/ graphical database output of a typical plant item is included as well as the breakdown of ownership costs for the database application.
9.1 Introduction

Rolls-Royce, who design, supply and support the naval nuclear steam raising plant are constantly examining ways of improving reliability and reducing ownership costs, and have recently undertaken a major re-appraisal of their management processes and organisation which support these aims.

A support organisation has been established with the prime objective of maximising availability without compromising safety. This support team is the focus for all support activities. A reactor plant for the Royal Navy's Submarine Programme comprises more than two thousand separate component designs. To enable an effective lifetime support organisation to make good management decisions on maintenance and spares provisioning and the need for future design change, each of these designs need to be monitored. This monitoring process ensures that the perceived availability, reliability and maintainability is not deteriorating. The monitoring process is provided through a database. The database has been given the same name as the process that uses its data, Reliability Centred Maintenance (RCM).

What follows is a summary of part of the monitoring process that Rolls-Royce are undertaking.
9.2 History of the Reliability Centred Maintenance Database

Historically, the product monitoring process has relied on two databases that have been used to monitor and track site generated problem reports and maintenance activities. Both databases have existed since the early days of the Naval programme but neither database holds data about the system design or the fitted history of individual components. This effectively means that if either of the two systems is used for reliability assessment they are obliged to assume that hardware fails at a constant rate, despite the fact that the assumption defies an engineering understanding of failure mechanisms. The probabilistic approach to failure taken by Knezevic (41) provides a mechanism to predict time to failure by mapping failures to one of a number of standard probability distributions. The approach taken by the Reliability Centred Maintenance (RCM) database, and described in this chapter is, therefore, a step change from the practice of the last two decades. The approach taken has been motivated by the need for increased availability and lower cost, based on a better understanding of the failure process.

The RCM process is documented in Mil. Std 2173. This process has been piloted several times at Rolls-Royce on a variety of systems and components. In all of these RCM studies the problem of lack of easy access to dependable reliability and maintainability data has been acknowledged. Failure rates can be obtained from the existing systems, (one such data set is given later in the chapter), but the cost and time to retrieve such data is high, thus reducing the benefit of the RCM process. This was recognised after the earliest RCM study and led to promotion of a Reliability and Maintainability database. Work on
the database commenced in 1995, with implementation occurring in the spring of 1997. The anticipated ten year cost of ownership for the database is shown in figure 9.1. The costs for the ten year cycle are categorised as shown in figure 9.2. The cost split shown in figure 9.2 is similar to previous projects managed by Rolls-Royce and Associates Limited and recorded in Bradley and Dawson (17).

9.3 Integration with Existing Information Systems

When the RCM database was conceived the company was two years into the implementation of a Parts Information Management System, (PIMS), designed to track both manufactured / installed components and their design states. The responsibility for PIMS has been placed with the Rolls-Royce team responsible for delivering increased availability. The system was a separate requirement to provide a technology update to a
long standing requirement to track the design and build configurations of the Nuclear Steam Raising Plant through the whole life-cycle.

It was natural to build the RCM database on the earlier PIMS system because:

- PIMS provided the necessary data on design state and install date required for reliability calculations
- the system pre-existed the RCM database
- it was under the control of the same team
- the costs for PIMS were already accounted for
- the concepts of Time To Failure and Time Between Failure were supported by the component fit-date data held by PIMS for all serial numbered components.

In addition the system holds manufacturer details for all serial numbered parts, holding out a future prospect of reliability comparisons between manufacturers of the same parts.

The data held by the RCM database is from two sources:

- the problem reporting system referred to earlier.
- paper based deficiency reports from the customer.

Each of these data sets are held against the appropriate serial numbered part records in the parent PIMS database. Separately, the two sources of failure data only provide a partial picture of failure, the first for the maintenance and the second for the operational phase of the life cycle. However the two systems between them contain both the customer's and
the company's design authority perceptions of the hardware failures for the complete cycle. They are therefore the most complete record that is available.

What was therefore implemented was a combination system that allowed both sources to be referenced. Because the two systems overlapped, it was also arranged for data for the same failure to be cross-validated to the other system.

![Pie Chart: User Test 10.0%, Design 24.0%, Code 10.0%, Data Lead 47.0%, Code Rework 6.0%, Project Management 3.0%]

Figure 9.2 RCM Database - Ten Year Costs

Dunbar (53) suggests that the type of paper based sources that we are using are not ideal, on the basis of untimeliness and high costs. The timeliness issue is not, however, a problem as the majority of components have very low mean failure rates for all failure modes; such components have an equivalent time between failure measured in years, and the time between failure and reporting can therefore be more relaxed than in a system.
built from commercial components. Similarly the cost of operating the system can be seen to be low. One consequence of components with long times to failure is that the number of failure details to be loaded into the database is small.

9.4 The Cost of Data Collection

We can compare the cost of reliability data from the RCM database with the cost of obtaining similar data from other sources. One such comparison can be made with the costs for an exercise carried out on a commonly used component. This exercise was to establish the effect of environment on failure distribution. A typical failure probability distribution plot from this work is shown at Figure 9.3. Importantly the anticipated ten year cost of the complete RCM database is only one seventh of the cost of the environment study. The environment study, though, provided a reliability profile for only one component design out of the two thousand in the complete plant. The RCM database therefore compares very favourably on cost with the assessment used for the environmental study.

There is a timescale problem with the very reliable components to be found in reactor systems. Those components designs that have long time to failure characteristics and are also in small populations may not experience a failure in the life of a plant. Given that only 30% of the plant component designs are installed in quantities of five per vessel or greater, it can be seen that the short term benefit of the database will be limited to around
600 component designs. Even with this pessimism it is clear that the cost of the database is justified when compared with the alternative manual methods used in historical studies.

When comparing the environmental study results with the RCM database output for a typical component it is clear that some differences exist. Figures 9.3 and 9.4 show comparative results for the same component design. Figure 9.3 is taken from the environmental study, whilst figure 9.4 is output from the RCM database. Both show Weibull distributions with a similar shape, but the characteristic values for the two figures are very different. This was expected, as the two figures are for different samples from the same population and the level of reporting for certain failure modes is known to have been different for the two samples. We can conclude that the RCM database is probably a more accurate reflection of the current failure profile than earlier studies, even though the earlier studies were based on larger failure samples spread over a long time period.

### 9.5 The Detailed Operation of the Database

All defect / failure reports received are reviewed for inclusion in the RCM database but as the database is concerned with hardware failures those that are clearly non hardware related are excluded. All reports are treated in exactly the same way when entered onto the database, and indeed the two sources of data are both entered when reports are received for the same failure from both company and customer representatives. The data on each report is significant but the basic data comprises:
- Vessel
- Class
- Part No
- Serial No
- Failure Mode
- Recovery Action
- Date
- Time
- Time to repair (Active)
- Time to repair (Total inc Logistic)
- Assessments

1 Failure modes are those provided by the designer for the design, with mode assessments being made by the database owner.

2 The assessments are made by the database owner on the basis of the available data to enable root cause analysis. As data used for evaluation can change, e.g. when a customer report is made some time after the initial company report, the assessments are date and time stamped. The latest assessment is then the one used for root cause analysis.

When required, data can be retrieved from the database by class, vessel or part number. The data retrieved is made available in a format compatible with Minitab™ statistical
software to enable a statistical analysis to be performed. Final output is in the standard forms available from Minitab™ as shown in figures 9.3 and 9.4 below, or alternatively as a survival or hazard plot. This option was chosen to avoid having to write expensive, custom built software when good quality commercial software was available. Retrieved data includes recovery and logistic times as well as time to failure.

Figure 9.3 Environmental Study Output
9.6 Future Developments

For the future the distributions of failure probability derived from the RCM database will
be used in part to support the modelling of availability and cost of ownership of the
complete plant. In this way the best available data will have been used in a mathematical
model to choose the maintenance option and maintenance timing to maximise plant
availability at minimum cost. At the time of writing this thesis the full model,
incorporating a genetic algorithm is in the formative stage, however a prototype model has been created to justify continued spending. The early indications from the prototype are that it will be able to optimise whole plant availability and cost of ownership with the data generated from the RCM database, modifying maintenance activities at the component level. This modelling approach, of course, requires detailed data that is not available from the RCM database. Additional data items include component costs, costs of lost availability and repair infrastructure cost contributions.

9.7 Conclusions

From the experience gained at Rolls-Royce it can be shown that:

- reliability and maintainability data can be collected at a reasonable cost for long lived systems.
- for systems containing high reliability components it is possible to use paper systems as the source data without a timeliness problem
- distributions of failure probability are increasingly being used to describe the failure profile of components.
- It seems likely that the approach taken with the RCM database should / could also be extended to the software components in software systems. The approach taken by the RCM database is also that attempted by the author in describing the reliability of the hardware component in Chapters
5 and 6, emphasising the similarity of computer systems with more general classes of system.
CHAPTER 10

Network Monitoring Software

Chapter Preface

Network monitoring software used by Rolls-Royce is described, and the implications for the future of IT cost of ownership measurement is reviewed.

10.1 Introduction

Software now exists that will continuously monitor the use of a computer, or networked computer and record the extent of use of the machine, the software that is used and the effectiveness of individual users in executing a particular software function. In a case study Green (35) illustrates in his MSc dissertation how such software was used at Rolls-Royce to enable decision making for PC cost of ownership reductions.
10.2 Monitoring Computer Use

In the study the Deskwatch™ software was used to monitor the company’s computer network. The software was originally purchased to provide data for hard disk procurement decisions, but proved invaluable in support of whole life cost reduction decisions. It provides detailed PC system, performance and configuration data using a combination of Windows™ executables, Windows™ function libraries and DOS executables on both networked and non-networked PCs.

The software consists of a 'capture agent' component installed onto each PC, performing a monitoring function, and an 'agent manager' providing administrative control of the agent community. This allows decisions to be made on the scope of collected data. The capture agent continuously monitors the PCs' activity and configuration details, the 'capture' process, and records them to Event Record Files on the hard drive of each configured PC.

The Deskwatch™ analysis showed, not surprisingly, that a users productivity is directly impacted by the speed of their PC.

The analysis identified that 1.5% of all PCs had a 'good' performance level, 25% an 'acceptable' performance level, 42% a 'poor' performance level, whilst 31% had a 'critical' performance level. Effectively this shows that at the time of the analysis approximately 73% of desktop PCs were failing to meet acceptable service levels.
This illustrated a significant target area within which large potential productivity increases could be obtained.

Following the analysis a programme was initiated to:

a) swap little used high specification machines for high use low spec machines
b) buy in new machines on a rolling programme

In addition the software identified through the following figures 10.1 to 10.4 that:

- WordPerfect™ and Groupwise were the top applications with 50% of active machine time being used with these two applications.
- Active machine use was largely only for 1 to 2 hours / day. Only a very small number of machines saw 6 or more hours of use / day.
- Applications demands on processor and disk are variable, and need to be accounted for in system design.
- Cost of ownership of specific PC models is affected by the applications used and their extent of use, giving rise to a range of annual costs of £1000 to £22000 at 1997 price levels.

The importance of this précis account of Green’s application of Deskwatch™ to the RR network is that the data derived can be combined with reliability and availability costs (not covered by Deskwatch™) to give a view of the change of whole life costs.
over time. This can be used to make more objective / rational decisions about time to replacement. The decision making potential is described graphically in Chapter 11.

10.3 The Decisions Made

The Green report (35) was not fully implemented. The reasons for this are quite complex, and follow from the outsourcing of the corporate IT function in the months prior to Green working in the company on his MSc project. EDS, the company that won the outsource contract, committed to a rolling upgrade of all low specification 486 PC’s. Part of the Green recommendations was the exchange of these low specification machines into the areas with high PC usage. The objective here was to anticipate machine changes to higher specification, and to avoid the problem of areas with low usage having high specification machines. In the event, most areas were reluctant to change if it involved (even temporarily) having a lower specification machine. The exceptions have been the more aware department managers, who, in some cases have relinquished PC ownership entirely to provide lower grades of staff better access. EDS themselves were equally reluctant to move machines twice.

The Green report also recommended that the rolling programme was accelerated but instead the company and EDS called a halt to the rolling programme, and the associated change to the Windows™ 95 operating system. At the time of writing it is understood that the 32 bit Windows™ NT operating system is likely to be
implemented on the desktop in favour of Windows™ 95. The immediate effect of this change is the continued reliance on a machine population that is growing older by the day, and whose performance is relatively poor when compared with either of the incoming operating systems. The wisdom of this situation is questioned graphically in chapter 11, but it is regrettable that having invested a significant amount of time and money in the Deskwatch™ software that at least some of the recommendations were implemented.

The savings quoted by the Deskwatch™ software, and based on the company financial data provided to Green, were significant. The performance based annual savings quoted for the top fifty machine upgrade recommendations was £200000. This of course must be tempered with the increased costs associated with installation and maintenance. This is taken up in the final chapter.

It appears, though, that it is not enough to be right, and be able to demonstrate that significant cost savings can be made by adopting a particular design choice, but also it is important to carry the organisation with the proposals. This almost certainly implies the need for political awareness of the organisation involved as well as the technical and financial skills required to carry out whole life cost demonstrations.
Figure 10.1 Application user Counts
Figure 10.2 Average PC use per day
Figure 10.3 Performance level by application
Figure 10.4 PC Cost by Specification
CHAPTER 11

Summary, Conclusions and Recommendations for Further Work

Chapter Preface

This chapter summarises the whole thesis, relating the work to the aims and objectives set out in Chapter 1. Conclusions and recommendations for further work are included to provide direction for future research work by Rolls-Royce and / or other interested persons.

11.1 Summary

Chapter 1 set out the aims and objectives for the work described by this thesis as:

a) To test the hypothesis:

Whole life cost, design methods are appropriate and effective for IT and software based systems.
b) To develop generalised methods for the process of designing software systems to minimum Whole Life Cost (WLC).

c) To demonstrate by example, contrary to existing Whole Life Costing beliefs (54) for physical systems that WLC techniques can provide lower whole life costs for software based systems at any time in the perceived system life.

These objectives are discussed in the context of the rest of the thesis in the following paragraphs.

11.1.1 Testing the Hypothesis

Proof, in the scientific sense, cannot be provided for the hypothesis. This thesis has, though provided in Chapters 1 to 10 evidence that the:

- Whole life cost approach is necessary for IT systems to ensure that systems provided are cost effective and of good quality (Chapters 1 and 2).
- Whole life cost methods that work for non IT systems also work for the software based systems to be found in industry (Chapters 2 and 8).
- Data collection to support the methods are, if anything, easier to arrange than for hard systems because the data collection can in some cases be automated (Chapters 5,6, and 10).
• Cost drivers are not necessarily the same as for hard systems, e.g. obsolescence is a significant driver for both software and desktop machines (Chapter 4).

• Cost drivers tend to be similarly ranked across systems, enabling design solutions to be transferable from project to project (Chapter 7).

• Resistance to whole life cost methods is acknowledged to be present, indicating that the need for change / education (Chapters 1 and 3).

• Genetically grown design solutions involving a probabilistic approach appear to be the future for IT systems, to take account of the differences between IT and hard systems, e.g. short times to obsolescence (Chapter 9)

11.1.2 Generalised methods

A method for designing to minimum whole life cost is described in Chapters 2, where the method is laid out in some detail. An optimising adjunct to the method is also described in Chapter 8 (Section 8.5) where causal networks / influence diagrams are initially used to conceptualise cost relationships, prior to optimisation of a system design using Genetic Algorithms. The method is demonstrated using a simple system example in Chapter 2 where typical cost savings are shown.
11.1.3 Lowering Costs Through Life

The classical understanding of what is possible using whole life cost techniques is shown in Fig 1 of Appendix 3. Here Fabrycky and Blanchard (54) claim that the value of possible savings fall as the whole life cycle progress. For computer systems where a fleet of similar equipment is in use, with progressive replacement of ageing equipment, it is possible to make continuous assessment and upgrade - replacement of system components. In this way the whole life cost can be trimmed as part of a rolling programme. Figure 11.1 shows the effect of combining the availability, maintenance and lost performance cost data derived in Chapters 5 and 6 with the performance data from Chapter 10. At the current state of computer and software development in a company like Rolls-Royce it can be seen that the minimum cost / time to replacement is in the order of 2000 hours (approximately 1 year), for a PC that was seeing heavy use.

The two curves also show a disturbingly different picture to the one described by Gartner (44). Here the cost of PC ownership is given as less than £9000 / year. The whole cost of PC ownership at the desktop clearly is dynamic with respect to usage patterns, age, and system configuration. The data shown in Figure 11.1 is only one such combination.
11.2 Achievements

The thesis, apart from exploring the evidence that is available in industry to support the hypothesis has provided a number of original results that are of value to the IT community in their own right and quite separate to the needs of the original hypothesis:

- Methods for design to minimum cost, Chapters 2 and 8
- IT system obsolescence defined (Chapter 4)
• The need for change to the British Standard, based on universal access to common cost profiles, (Chapter 1)
• PC Component failure probabilities established experimentally (Chapter 6)
• Cost estimation relationship for the 4GL development cost established experimentally, and the method for repeating this relationship defined, (Chapter 7).

11.3 Conclusions

The whole life cost technique or method has been shown to work for computer based systems. There are some major differences between physical systems or products and software systems. Obsolescence is an example of a major and growing problem with computers and software. The fact that a computer can be used for a wide variety of processes by different users on the same day provides problems for analysis of whole life cost. Paradoxically the ability of software systems to collect data whilst on-line provides a major advantage for analysts. By example, the latest versions of helpdesk software are now recording failures of network objects automatically. This points the way to the future where automatic logging will need to be extended to failure logging for each of the network objects affected by the failure of another (hardware or software) component. In this way the service provider will, in future, be monitored on the basis of actual availability at the point of use, and the costs of the whole service will begin to be understood for each desktop machine.
There are also some psychological problems with whole life cost methods. Although the method is intuitive, in that it is clearly sensible to take account of all costs when making a major purchase, it is also clear that in a business sense denial often precedes acceptance of the wisdom of the concept. This was the theme of Chapter 3. It is recognised that existing design methods often call for an 'investment appraisal' or a 'cost benefit analysis'. These are less than a design carried out to a whole life cost method, as they seek to establish the worth of a single design rather than to establish the best investment solution from all those possible.

11.4 Future Work

This is not the end. It is not even the beginning of the end. But it is, perhaps, the end of the beginning.

Winston S Churchill

A great deal of work is left unfinished. It is clear that whole life cost issues as they affect computer systems should be a major concern for UK PLC. Poor network design, component replacement strategies not based on a full understanding of costs, in house software development programmes which do not face up to the realities of the whole life cost profile in providing cost effective operation are all examples of targets for whole life cost reduction.
As a minimum, the following activities are needed to carry the work started in this thesis forward:

Underpinning the teaching of the technique there is a need to accumulate a fundamental-data set for the components and subsystems to be found in computer based systems. Specifically there is a need for the industry to have access to reliability and / or cost data for these common objects. This was a point made by the Chairman of the BCS Quality Special Interest Group at the 5th QSIG conference in Bath, March 1997 on listening to the presentation of the paper presented in Chapter 5. Where that responsibility should lie is open for debate. It could rest with the BSI, in a form of reflective standard, or with the BCS as a part of the professional responsibility owned by that society. The data referred to here, in effect, is the world data referred to in Chapter 8.

Extension of the work to cover the automatic logging and availability calculation described above.

Optimised design rules for a variety of products to minimise whole life costs for the customer, i.e. the product should always be customer focused. Likely products include:
Database structures / applications

Network configurations for different use patterns

The author's own interests currently follow the issue of continuous systems integration - the process of continuous assessment of installed systems for cost based improvement. An EPSRC call for systems integration proposals offers such an opportunity. The focus for this call is on 'managing the product and process complexity' to make 'the business transformations needed for long term success in a global market'. The scope of the call is the 'holistic approach to the design, development and production of products and services' and 'creation and operation of customer focused processes'. The fact that this call is for traditional engineering systems rather than IT systems should not prevent the application of the same outlook to computers and software. Ultimately they are not so different.
References

1. Hewson D., 'The mouse is turning into a pest on PCs', Sunday Times. 14th April 1996.


6. TCO Analyst, Gartner Group, 1998


25. Robertson C., Railtrack signals £32M spend on systems strategy overhaul, Computing, pp1, 18th July 1996.


43. BS5760 Part 2, Section 19 *Assessment of Software Reliability* British Standards Institution, 1994.


ACKNOWLEDGEMENTS

@RISK™ is a product of the Palisade Corporation

Evolver™ is a product of Axcelis Inc

Deskwatch™ is a trademark of Prometrics Limited

Windows™ is a trademark of the Microsoft Corporation

WordPerfect™ is a trademark of the Corel Corporation

Sun™ is a tradename of Sun™ Microsystems Inc

Ingres™ is a tradename of Computer Associates

Visual Basic™ is a trademark of the Microsoft Corporation
APPENDIX 1
Dear Sirs

QMS 23 - DRAFT IEC 300-3-3

This letter is in response to a request for comments made by Mr M Watts at the I Mech E in March of this year.

We have reviewed the full content of the draft as written and our comments without responsibility are contained in the attached appendix.

It may however be helpful to you to consider the following for inclusion:

Ultimately the information derived by LCC estimation will need to be disseminated by the customer(s) for the product. To enable meaningful comparison between products from competing suppliers, the form of the output from the LCC estimation process, and all of the supporting assumptions will need to be common. It is suggested therefore that the following minimum information should be available to the customer:

1) Cost Profile in pounds at a declared base date.

2) List of assumed values representing a use profile for the LCC provided. This would typically include:
   - Assumed Life in years
- Use rate
- Environment conditions
- Excluded costs

3) NPV at a declared discount rate for the assumed life.

Should you require any further information on this subject please do not hesitate to contact us.

Yours faithfully
for ROLLS-ROYCE AND ASSOCIATES LIMITED

M Bradley
Life Cycle Cost Facilitator
PAGE 4 - INTRODUCTION

Para 1, Line 6 & 7:

The challenge for suppliers is in fact '... to design products to meet a particular service at minimum through life cost. The through life cost being expressed as the sum of the acquisition and subsequent ownership cost.'

Reliability is not necessarily an issue in this respect. The product (to meet a particular function) that has the lowest Life Cycle Cost may not be the product with longest MTBF. For example it would be feasible to allow a lower MTBF together with a correspondingly lower MTTR and still achieve the lowest LCC for the product function being designed.

Thus, the optimisation process is present but on design parameters rather than the Life Cycle Cost. It should always be at the minimum possible value for the product function. Note of course that if a customer expresses a desire for a specific reliability then that is part of the functional specification and optimization involving reliability is denied.

Para 2, Line 6:

The Design rather than LCC is optimized, the wording should read '... (to assist product users) to minimize LCC.'

Clause 1, Scope:

Page 5, Para 3:

This contains a repeat of the last sentence of para 1. The complete sentence should be deleted:

'General guidance for ........ is provided'.
It would have been helpful to include in this draft a title or resume' of the 'Further supporting sections'. For example Software Systems LCC is clearly an important issue. The additional considerations for software would be a valuable contribution. Is this a section under consideration? Are comments and contributions required?

Page 5, Notes:

These appear to be misplaced, Note 1, whilst valid to the standard, would be better either placed in the Introduction section or deleted completely as it is already covered in Clause 4. Note 2 appears to be completely out of context. It brings nothing to the standard, as there appears not to be any contained gender references. If the note has any place it should be in the foreword.

Clause 2, Normative References:

Page 6, Para 1, Line 5:

'investigated' should read 'investigate'.

Clause 3, Definitions:

Page 6:

Additional definitions would include:

Whole Life Cost, Through Life Cost - These are used interchangeably with Life Cycle Cost in industry.

Sunk Costs

Uncertainty

Sensitivity

Since the list of definitions will grow, it may be helpful to include these as an alphabetically ordered list.
Clause 4, Life Cycle Cost Concept:

Clause 4.2 Product Life Cycle phases:

The life cycle phases lists as a) through f) are specifically those for construction or hardware projects. This should be made clear, and the distinction made that LCC applies equally to Software and Software systems, through with a different set of Life Cycle Phases.

Clause 4.2, Page 8, Para 1, Line 12:

Missing 'the' before 'decision making process'.

Clause 4.3, Page 8, Para 2, Line 7:

Spelling of 'analyses' should be 'analysis.

This clause correctly states that the early LCC estimation is crucial to product optimization. What is not made clear is that when used in the design process, LCC may need to be applied iteratively. the Design process can be considered to be a continuum with refinement of both the design and the LCC taking place progressively. A model for this process is attached.

Clause 4.4, Page 10:

A Life Cycle Cost may be either a forward Estimate or an Actual Cost of ownership. Actual LCC statements can take the form of the recent study carried out on behalf of the Kyocera printer manufacturer, i.e. a profile of the actual costs together with any qualification statements of the prevailing conditions, escalation rates, equipment use profile etc. The suggested effect on the clause is the addition of the following wording after 'reflecting' in the first line:

'either an estimate or an actual Life Cycle Cost typical of the product. This changes the word 'estimate' in lines 2 and 4 to 'cost' and the wording of the fourth line to be changed to 'As such a quantitative ...'.

One effective procedure for the resolution of the quantitative and qualitative debate is the adoption of the model attached. Here the feasible alternatives are identified, and then each of these is subject to LCC estimation. A design review looks at the profile of the option costs with respect to time. If a clear winner is presented on cost, then that option is taken. If a clear winner is not present then the qualitative issues and risk assessment
chooses the option. The point here is that only feasible options are taken through the LCC analysis. Life Cycle Costing is a desperately expensive business.

You may want to review this clause further in the light of this comment.

Clause 5 Dependability and LCC Relationship:

Clause 5.2 Unavailability Costs:

Strictly, unavailability costs are limited to the costs associated with the loss of function. The cost of preventive maintenance is generally the result of a desire for high availability. The cost of corrective maintenance may be the result of either random failure or a policy of allowing an item to fail (often because that policy is cheaper than attempting preemptive maintenance).

We suggest that this clause is re-worded to removed the corrective and preventive maintenance statements.

Clause 5.2, page 11, Item (c):

Replace 'store' with 'restore'.

Clause 5.3, Para 5.3, 2nd Sentence:

The LCC of a product must surely be from one particular view point, i.e. a LCC cannot include every possible attributable cost e.g:

A vendor will typically add between 5-15% to his unit price to include warranty. It is this cost that should be included in the LCC. If the equipment fails more often than expected during the warranty period then this cost has to be absorbed by the vendor and is not seen by the customer i.e. it is a cost that is not seen in the LCC activity because a buffer is provided by the vendor.

Clause 6 LCC Modelling:

Clause 6.1 General:

It would be helpful here to include an additional item (e) covering the use of models owned by the purchaser. It is customary for some purchasers to require potential suppliers to provide raw data on product which is then used to determine likely Life Cycle Costs by the application of the data to a parametric model. This practice should be
discouraged by the standard unless as a minimum the algorithms are known to the
supplier. Ideally the model also should be available to the supplier. This is on the basis
that a model will encapsulate those aspects of the design which are important to the
Purchaser. Prospective suppliers should be given the opportunity to tune their design
solution to the customers implicit statement of requirements, using the customer model.

Clauses 7.4 & 8:

Uncertainty, Risk and sensitivity analysis are mentioned a number of times in these two
clauses. Specialist software is required to carry out these analyses. Whilst the use of
particular software is not to be recommended, it ought to be said here that so called Risk
analysis software is available as an 'Add in' to the more popular spreadsheet packages. In
our opinion these represent the only practical solution to the problems of uncertainty and
sensitivity analysis.
Appendix 2

PC Components Failure Probability -
results from Rolls - Royce and Associates Limited
The probability density papers for the full set of PC components analysed in Chapter 7 are presented on the following pages of this appendix. The data used to calculate the Weibull parameters are derived from the Rolls-Royce and Associates Limited helpdesk as described in Chapter 7, i.e., the parameters are derived from collected data in the RR environment rather than calculated from sub-component failure data. Care must be taken in transferring the failure characteristics to another organisation as the failure characteristics may be specific to the RR Company, its practices, and its computer suppliers.
Weibull Distribution

Censoring Indicator in Censor

Shape: 0.991376
Scale: 49171.1
Char. Life: 48171.1

Figure App2.1 Motherboard Failures
Figure App2.2 Floppy Disk Failures
Weibull Distribution
Censoring indicator in Censor

Shape: 0.518910
Scale: 136752
Char. Life: 136752

Figure App 2.3 Hard Disk Failures
Figure App2.4 Keyboard Failures
Weibull Distribution
Censoring Indicator In Censor

Shape: 0.767125
Scale: 58365.2
Char. Life: 58365.2

Time to Failure

Figure App 2.5 PC Monitor Failures
Weibull Distribution
Censoring Indicator in Censor

Time to Failure

Percent

Shape: 0.881030
Scale: 23440.4
Char. Life: 23440.4

Figure App2.6 Mouse Failures
Weibull Distribution
Censoring indicator in Censor

Shape: 0.735105
Scale: 3993.79
Char. Life: 3993.79

Time to Failure

Percent

Figure App2.7 PC Failures - All components, all modes
Appendix 3

Paper from the 'Engineering Through Life Support for Profit' seminar 11th October 1994, Midland Hotel, Derby, Hosted by Rolls Royce and Associates Limited.

Life Cycle Costing

Malcolm Bradley - RRA Derby
Introduction

My personal interest in the subject of Life Cycle Costing stems from the software environment. Typically companies and individuals will purchase the latest version of a software package or upgrade to a later version without first determining the benefits that the particular software version has for their business need.

At about the same time that I was becoming interested in the through life cost of software, the company was exploring the possibilities of Life Cycle Costing of hardware as part of the Integrated Logistic Support process. I was recruited as the Life Cycle Cost facilitator.

This seminar provides me with around fifteen minutes to introduce the topic of Life Cycle Costing, I will only be able in that time to provide an overview of the technique, What the technique is, What it can be used for, Who should use the technique, What the pitfalls are, and provide some examples of its potential use in a variety of circumstances.

What is Life Cycle Costing?

Life Cycle Costing is defined simply as the sum of all project costs that would not have occurred if the project had not started.

Simplifying this statement, the Life Cycle cost of a project is the sum of the costs of:

- Acquisition
- Operation
- Disposal

Life Cycle Costing require both the identification of all potential costs and benefits and the means to predict those costs over time. Benefits are not normally a consideration in a defence contract, but it is possible that the salvage value of a system or components can
present a return on the investment at disposal.

Why Should We Use Life Cycle Cost Techniques?

The Life Cycle cost idea is useful in two situations:

**BUDGETARY COSTING**

**DESIGN CHOICE COSTING**

Budgetary Costing is the technique used when an indication of a total cost distribution is required for the life of the project to ensure that the project is affordable and to provide a basis for financial planning for future expenditure.

Design choice costing is used where there is a need to determine which of a number of competing designs is the cheapest.

It should be noted that UK Defence contracts are already including contract clauses to the effect that the successful bidder will be the one with the lowest through life cost. This is on contracts for both hardware and software. The benefit to a Defence Contractor is therefore clear, a contractor with the life cycle cost skills is more likely to win defence contracts since he will be able to test competing designs for minimum through life cost.

Who in An Organisation Will Be Best Placed to Perform the Function of Life Cycle Costing?

Here at Rolls-Royce and Associates we have concluded that the equipment or software designer will have the responsibility since he or she will be most likely to get the more accurate costing result, being the person closest to the design. Having responsibility for the through life costs for a design in turn obliges the designer to be aware of support issues. Equally the designer is the person most likely to influence the design to minimize
the through life cost. Of course, during a concept design phase and typically during the bid process, the data gathering, collation and validation is performed by the bid team. The high level data required through is provided by the design team.

There is of course a need for a consistent corporate approach to Life Cycle Cost estimating to ensure uniformity. At RRA we have adopted a facilitation approach to ensure common standards.

RRA bid teams producing LCC estimates are using Integrated Logistic Support (ILS) personnel within the team to carry out the very detailed business of data collation and cost build up. ILS personnel are directly concerned with the record of designs - the Logistic Support Analysis Record (LSAR). The LSAR is the definitive data store.

**When Should Life Cycle Costing Be Employed?**

The principle decisions with the greatest impact on the cost are made right at the beginning of the design process. The technique should therefore be used right from the beginning of the design process. As Fig 1 shows, by the end of the Concept Design Process two thirds of the cost have been committed. To have any financial impact on the project Life Cycle Cost techniques must be in use as a decision making tool from the very early days of the design process.

**What Are the Pitfalls?**

Practitioners of any life cycle cost methodology need to proceed with care. Any estimate of Through Life Cost will be inaccurate but there are a number of pitfalls that can be avoided that would cause such inaccuracies.

- **Lack of Agreement on the boundaries** of what will and will not be included in the study.
- **Failure to identify all the cost items within the boundary.**
Failure to collect the correct data - or incorrect interpretation of historical data.

Figure 1 The Commitment Curve

Not recognising that a design has changed or not updating the LCC estimate to take account of the change.

Use of an inappropriate cost model - different models generate different cost estimates.

There are a number of associated issues.

The need to move the organisation to a position where accurate LCC estimating is possible:

eg

Actual, historical cost databases available
Integrated Logistic Support techniques available.
Agreement on, or a policy for deciding, the extent of use of software tools for the Life Cycle Cost estimate.

Agreement on the financial basis for LCC estimating, (the use of Net Present Value and the use of base dated costs and escalated costs for future years.)

To manage these and other issues and pitfalls it is suggested that a structured methodology is required. RRA has adopted and adapted a method first proposed by Fabrycky and Blanchard in Ref 1, and shown in Figure 2.

The RRA method is a ten step iterating process that continues in parallel with the design process. Iterations are controlled by the Design Review meeting and the process ends with a specific design option being selected. The cost estimate is a by product of the essential underlying design process.

**How Should LCC Be Calculated?**

Most organisations use software to calculate LCC. There is a large number of software solutions available. We know of 32 packages covering hardware and a smaller number covering the LCC of software. Most are expensive, none that we know of are universal in their application and many are based on popular spreadsheets. Consequently we have chosen for most of our own applications to use a spreadsheet incorporating a Risk Analysis tool. This combination allows a user to input multiple values for a single input parameter, covering a range of probabilities, whilst at the same time taking advantage of the spreadsheet's 'what if' capability.

We feel that this is a compromise that provides:

Full user control over all inputs and algorithms.
Dispersed responsibility for LCC by placing low cost template spreadsheet software into all design areas of the company.

Commercial packages offer the opposite option:

Program control of the necessary input and algorithms.

Centralised responsibility for LCC estimating because of the high cost of the software.

Figure 2 - Design to Minimum Cost

The route to be taken by any particular organisation will be influenced by organisational size, the core business and customer expectations.

Having covered What, Why, Who, When and How at some speed example LCC
applications will be presented from Hardware, Software and Process environments. LCC can be applied to any subject at any level, these simple examples are those that are either relative to this company's business interests or provide a widely understood subject for discussion.

**The Barge Heating System**

As a training aid we have devised a series of walkthroughs for ILS techniques that are based on a design task for a Heating System for a Canal Barge.

The Life Cycle Cost estimate presented is for two design choices only; a gas fired hot water radiator system, and a generator powered ducted hot air system. The typical output of a LCC study is shown at Fig 3. This is a bar chart of the cost spread for the Project Life of the Electrical Design Option. There is a similar bar chart for the Gas

![Barge Project - Heating System](image)

**Figure 3**Barge Heating System - Electric Design Costs
design. This type of presentation is useful for cash flow and budgetary purposes but
the real strength of this Life Cycle Costing comes from a break even analysis. This is
shown in Fig 4. Perhaps unsurprisingly, the Gas fired system is shown to be easily the
cheaper system to own over ten years.

Barge Project - Heating System

Electric / Gas Fired - Break Even Analy

Figure 4 Barge Project Break even Analysis

This analysis includes the following input data:

Inflation Rate
Discount Rate
System or Project Life in years
Annual Running House
Mean Time Between Failures for each component
Mean Time to Repair for each component
Labour rates for repairers
Spares Costs
Repair Policy for each component
Scrap Value of the system
Fuel Costs

If we were to assume that the barge was for hire, rather than for personal pleasure or carriage of goods, it might be assumed that the hirer would bear the costs of all fuel. The breakdown analysis would then be as shown in Fig 5. In this case the difference in cost distribution is not significant but in fact it does show that this Life Cycle Cost comparison is not sensitive to fuel costs.

**Barge Project - Heating System**

Break Even Analysis excluding fuel

![Cost vs Year Graph](image)

*Figure 5 Barge Project Break Even Analysis Excluding Fuel Costs*

dModels used for Life Cycle Cost Studies require this ability to detect sensitive input values, ie those that cause large changes in the output.
Computerised Parts System

The parts system is a fully functional RRA system, commenced in 1991. It was written for an INGRES database and has almost 400 input and retrieval screens and is our corporate system for control of designs, drawings and manufactured hardware. The costs used are actuals to August 1994 and estimates from then until the end of its ten year life. For commercial reasons costs are shown as percentages of overall spend. Fig 6 shows the typical cost spread for the ten year life. As with the barge example this presentation is useful for budget and cash flow. Fig 7 however is a more useful presentation of the same information. Here the system cost categories are shown, and ranked in descending order as a Pareto Analysis.

Parts System Cost by Year

![Bar chart showing parts system cost by year from 1991 to 2000.](image)

Figure 6 Parts System Costs

From this it becomes clear that it may be possible to estimate the LCC of a future, similar system in RRA by estimating only the categories of Part/Process/Work Item
that are concerned with the largest categories. Typically it is found that a relationship will exist between cost and categories and that approximately eight percent of the cost will be found in twenty percent of the categories.

In this case it may be possible to conclude that:

Operation
Database
Computer Software Coding
Network Maintenance Costs

generically represent approximately 80% of all costs. A future system total cost would then be derived for the first pass estimate from:
Total Cost = (Estimated Cost of the major items) * 1/0.8

The precise relationship between percentage of cost and percentage of categories of cost can only be derived by inspection of historic data for the class of system or part being considered.

This technique was originally pioneered by Dundee University for use on Construction Projects. We believe it has application in many fields as can be shown in my last example.

**Submarine Maintenance**

Submarine steam raising plant is maintained using a set of maintenance instructions, contained in Planned Maintenance Schedules. Any one schedule relates to a category of equipment (eg Pumps or Valves). Unplanned maintenance is recorded by date, vessel, hours consumed in the activity, and the PMS number for the planned maintenance for that item.

Presenting these in Pareto format demonstrates that we again have a relationship that can be used to identify cost drivers and estimate (maintenance) costs for similar designs of equipment.

Fig 8 shows the Pareto analysis for the unplanned maintenance of a class of submarines. A closely similar relationship exists for a later class of submarines of a slightly different design.
The cost of maintenance of a Nuclear Steam Raising Plant (NSRP) is itself a major contributor to through life cost of the plant. It is suggested therefore that future research may profit from exploiting the Pareto relationship at different levels in the cost structure. ie Maintenance is in the 80% of significant cost for the NSRP, but of the categories of maintenance only 20% will be concerned with 80% of cost. Total cost may be obtained rapidly by applying the 80/20 relationship at several levels in a product breakdown structure.

**Research Direction**

Future Life Cycle Cost Research in the Company will concentrate on:

Establishing cost relationships for company products by building and maintaining a database of Life Cycle Costs for the categories of product.
Establishing and maintaining an understanding of the applicability of the many software packages available.

Developing existing in-house software for Life Cycle Costing.

Through an understanding of Cost relationships develop rapid, accurate estimating technique.

To assist the research and the process of Life Cycle Costing further we are:

Implementing the methodology described earlier.

Facilitating the Life Cycle Costing process by making Software and Training available to key staff.

The same training is to be made available to external organisations.

**Summary**

Life Cycle Costing is being used in RRA primarily as a Design Tool.

In an ILS context Life Cycle Costing is a management tool that enables design effort to be focused on the through life cost drivers.

Life Cycle Cost is a useful tool in driving down the through life cost of ownership by enabling cost comparison of competing designs.

Life Cycle Cost estimating demands skills, understanding and a tailored information infrastructure to be effective.
The life cycle cost technique fits naturally into the Integrated Logistic Support environment, but can be used as a standalone technique.

References


RRA 13853 Issue 1 1994 - Life Cycle Costing - A handbook for use on the NSRP.
Appendix 4

Retail Price Index 1982 - 1998
### Retail Price Index - All Items Excluding Seasonal Food

<table>
<thead>
<tr>
<th>Year</th>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>1982</td>
<td>82.8</td>
</tr>
<tr>
<td>1983</td>
<td>86.7</td>
</tr>
<tr>
<td>1984</td>
<td>90.8</td>
</tr>
<tr>
<td>1985</td>
<td>96.5</td>
</tr>
<tr>
<td>1986</td>
<td>99.7</td>
</tr>
<tr>
<td>1987</td>
<td>101.9</td>
</tr>
<tr>
<td>1988</td>
<td>106.9</td>
</tr>
<tr>
<td>1989</td>
<td>115.2</td>
</tr>
<tr>
<td>1990</td>
<td>126.1</td>
</tr>
<tr>
<td>1991</td>
<td>133.5</td>
</tr>
<tr>
<td>1992</td>
<td>138.5</td>
</tr>
<tr>
<td>1993</td>
<td>140.7</td>
</tr>
<tr>
<td>1994</td>
<td>144.6</td>
</tr>
<tr>
<td>1995</td>
<td>149.1</td>
</tr>
<tr>
<td>1996</td>
<td>151.3</td>
</tr>
<tr>
<td>1997</td>
<td>160.6</td>
</tr>
<tr>
<td>1998</td>
<td>162.6</td>
</tr>
</tbody>
</table>
Appendix 5 – Chronological List Of Articles -


5 Bradley M, and Dawson R.J., What is a Quality Software System Anyway?, In 5th Software Quality Conference, University of Abertay, July 1996.


