The implementation of a field reliability database

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

Additional Information:

- A Master's Thesis. Submitted in partial fulfilment of the requirements for the award of Master of Philosophy of Loughborough University.

Metadata Record: https://dspace.lboro.ac.uk/2134/10995

Publisher: © J.A. Jones

Please cite the published version.
This item was submitted to Loughborough University as an MPhil thesis by the author and is made available in the Institutional Repository (https://dspace.lboro.ac.uk/) 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/
ABSTRACT

In 1984 the International Electronics Reliability Institute (IERI) at Loughborough University of Technology (LUT), then the component technology group (CTG), started to address the problem of collecting field reliability information and storing it in a database in order to provide analysis of the way in which components and systems behave in the field. This database was devised by the members of CTG with the assistance of a number of industrial partners. The database was created and stored on the university’s mainframe computers.

In 1988 these computers were replaced with more modern equipment which was no longer compatible with the initial version of the database and so the database was re-implemented using a new hardware/software combination. At this time the opportunity was taken to review the choice of database management system, the structure of the data, the database administration and operation and the forms of data analysis performed. This thesis describes this review and re-implementation process.
# TABLE OF CONTENTS

1. INTRODUCTION .................................................. 1  
   1.1. Identification of Troublesome Components .......... 1  
   1.2. Investigation of Component Failure Mechanisms .... 1  
   1.3. Investigation of System Behaviour .................. 1  
   1.4. Evaluation of Existing Reliability Models ......... 2  
   1.5. Derivation of New Reliability Models ............ 2  
   1.6. Evaluation of Reliability Indicators ............. 2  

2. AN OVERVIEW OF DATABASES ................................. 4  
   2.1. Database types .......................................... 4  
      2.1.1. Flat files ....................................... 4  
      2.1.2. Navigational databases ......................... 5  
         1) Hierarchical databases .......................... 5  
         2) Network databases ................................ 5  
      2.1.3. Object orientated databases .................... 6  
      2.1.4. Relational databases ............................ 6  
   2.2. The Use of Databases for Reliability Engineering 7  
      2.2.1. Specification databases ......................... 7  
      2.2.2. Prediction databases ............................ 7  
      2.2.3. Field failure databases ....................... 8  
   2.3. Choosing a DBMS for a Field Failure Database .... 8  
      2.3.1. The Flat File System ............................ 9  
      2.3.2. The Navigable Structures (Hierarchical and Network) 10  
      2.3.3. The Object Orientated Databases ............. 11  
      2.3.4. The Relational Approach ....................... 11  

3. THE DATA MODEL FOR A FIELD RELIABILITY DATABASE ... 12
5.4.5. The Common Program Interface ........................................ 45
5.4.6. Debugging support .................................................. 48
   1) The debugging function. ............................................ 49
   2) Journalling .......................................................... 50
   3) Tcl Interface ....................................................... 50
5.4.7. Program Verification ............................................... 51
5.4.8. Documentation ..................................................... 51

6. DATABASE STRUCTURE ................................................... 54
6.1. Company .............................................................. 56
6.2. Failure Record Number ............................................. 56
6.3. Equipment Type ..................................................... 56
6.4. Simultaneous Component Failure Identifier ....................... 56
   1) Board level. ....................................................... 57
   2) Equipment level ................................................... 57
6.5. Equipment Serial Number ........................................... 57
6.6. Company Design/Part Number of Board or Module ............... 58
6.7. Board Serial Number ............................................... 58
6.8. Operating Time of Equipment to Failure ......................... 58
6.9. Operating Time of Board (This Time Around) .................... 59
6.10. Up and Running Date of Board in System ....................... 59
6.11. Up and Running date of Equipment ............................. 60
6.12. End Date ........................................................... 61
6.13. Usage ............................................................... 61
   1) Stuck at faults ..................................................... 62
   2) Wipe clean faults ................................................. 62
   3) Delivery delay .................................................... 62
   4) Systematic underestimation ...................................... 62
5) High variability ........................................ 62
6) Low early usage ........................................ 62
5) Race faults ........................................... 62

6.14. Previous Use ........................................ 63
6.15. Storage ............................................... 65
6.16. Manufacturing Date of Board .......................... 65
6.17. Component Type/Generic Description ................. 65
   1) Equipment level NFFs .............................. 66
   2) Board level NFFs ................................. 66
   3) Component level NFFs. ......................... 66

6.18. Component Part Number ............................. 67
6.19. Component Encapsulation/Mounting Details ............ 68
6.20. Screened Level of Component ........................ 68
6.21. Circuit Reference Position .......................... 69
6.22. Component Manufacturer ............................ 69
6.23. Batch Number ..................................... 69
6.24. Operating Condition of System Immediately Prior to Failure ... 70
6.25. Failure Mode of Equipment .......................... 70
6.26. Reported Failure Mode of Component .................. 70
6.27. Number of Boards of a Particular Design Number ...... 70
6.28. Primary Function of Board/Module ..................... 70
6.29. Number of Components .............................. 71
6.30. Environment ........................................ 71

6.31. Soft Information ..................................... 71
   1) Equipment Type and Design ......................... 71
   2) Maintenance policy and repair strategy ............... 72
   3) System construction ................................ 72
   4) Data collection mechanism ......................... 72
   5) General ........................................... 72
   6) Database specific information ..................... 72

6.32. Failure Mechanism Reports .......................... 72
7. DATABASE PROCEDURES ........................................... 74
   7.1. Raw Data Supply Mechanism ................................. 74
   7.2. Database creation ........................................... 75
       1) Creation of the Main Database Tables .................. 75
       2) Creation of the Auxiliary Tables ....................... 75
   7.3. The Data Format ............................................ 75
   7.4. Data Anomalies and Data Validation ....................... 76
   7.5. Data validation ............................................ 78
       7.5.1. Principles of the Validation of Incoming Data .... 78
       7.5.2. Principles of First Stage Validation ................. 78
           1) Data framing errors ................................ 79
           2) Incorrect format ................................... 79
       7.5.3. Principles of Second Stage Validation ............... 80
           1) Relations within a single record .................... 80
           2) Relations between records .......................... 80
           3) Relations between data files ....................... 81
           4) Relations between data sets ....................... 81
   7.6. The validation used at IERI ................................ 81
       7.6.1. First stage validation ................................ 84
       7.6.2. Second stage validation .............................. 85
       7.6.3. Error reporting and database loading ................ 87
   7.7. The administration of the validation ..................... 89

8. DATABASE DATA ANALYSIS ........................................ 90
   8.1. Database totals ........................................... 90
   8.2. Raw data extraction ...................................... 91
   8.3. Structural Analysis ....................................... 92
       1) Breakdown of equipment environments ................... 92
       2) Equipment Complexity ................................... 93
       3) Component analysis ..................................... 96
   8.4. Time Independent Failure Statistics ....................... 98
1. INTRODUCTION

The reliability industry for many years has been interested in the way in which systems and components behave in the field. However reliable information is difficult to obtain due to the lack of solid facts upon which to base any analysis. Some of the reasons for performing reliability analysis are given below

1.1. Identification of Troublesome Components

By consideration of the failure frequency for particular component types across a large number of different equipments operating in differing environments it becomes possible to build up a picture of the components that are most likely to cause failure. This is done by calculating a failure rate for a component type and then comparing the failure rate obtained between different components. This allows the formation of a failure rate hierarchy at the top of which will be the components that cause most trouble in the field and this identification allows other research work to target component types for further investigations.

1.2. Investigation of Component Failure Mechanisms

It is of interest to know the most likely failure mechanisms that will occur when components are operated normally in the field. If these failure mechanisms can be identified then steps can be taken to remove any design factors that may accelerate the onset of these mechanisms.

1.3. Investigation of System Behaviour

Current thinking in reliability work has shifted the accent of interest away from the component level to system and sub-system level. The behaviour of systems is thus of great interest to reliability engineers.
1.4. Evaluation of Existing Reliability Models

There are many mathematical and physical models used in component reliability, many of these are empirical models based on small quantities of data while others are crude approximations. The collection of large amounts of field data would enable these models to be evaluated and improved where they are lacking.

1.5. Derivation of New Reliability Models

The derivation of new models where none existed previously may be possible by examining in detail the wealth of reliability information that a study of this nature can collect.

1.6. Evaluation of Reliability Indicators

A reliability indicator is a parameter of a component or system that can be measured while the system is new and which will give some indication of the likelihood of failure of that component or system. Information on field behaviour is required to place any data obtained within the laboratory in perspective.

In order to facilitate this work it is necessary to collect field reliability information. In 1984 the International Electronics Reliability Institute (IERI) at Loughborough University of Technology (LUT), then the component technology group (CTG), started to address the problem of collecting this field reliability information and storing it in a database. This database was devised by the members of CTG with the assistance of a number of industrial partners. The database was created and stored on the university's mainframe computers. This original database is described in [1].

In 1988 these computers were replaced with more modern equipment which was no longer compatible with the initial version of the database and so the database was re-implemented using this new hardware/software combination. At this time the
opportunity was taken to review the choice of database management system, the structure of the data, the database administration and operation and the forms of data analysis performed. This thesis describes this review and re-implementation process.
2. AN OVERVIEW OF DATABASES

The International Organisation for Standardization (ISO) has defined an information system to be a predictive system for keeping and manipulating information [2]. Information systems include processing functions, which are implementations of the various derivations and inferences that are required by the information recipient and a information storage repository, which is know as the database component of the information system, which contains the information to be processed. This database component, in order to be useful, must in some way reflect the structure of the information that is stored within it. This reflection of reality is obtained by use of a data model. A data model is a way of thinking about data and how it can be implemented using a database management systems or a DBMS.

2.1. Database types

There are a number of possible types of database that can be used to store information. They are described here in turn.

2.1.1. Flat files

A flat file database consists of only one record type which contains no nested repeating items or groups of items. A flat file is conceptually a matrix where the columns are named data items representing attributes and the rows are entries or records representing entities in the real world. Each element of the matrix contains a single value of the attribute for the entity. Flat files are the simplest type of data storage, a good example being a telephone directory, however there is no reason for a flat file store to keep its entries in alphabetical order.

Many problems are encountered in the use of flat file databases. They tend to be large, since there is no way to reduce the information stored. For example, if a household has more than one telephone number then the address of the house must
be stored twice in the telephone directory. Another problem is speed of searching. If the file contains more than one attribute that can be searched upon, it becomes difficult to decide on the storage order since optimum storage order for the different possible searching criteria may be very different.

2.1.2. Navigational databases

Navigational databases incorporate two slightly different implementations, hierarchical and navigational types.

1) Hierarchical databases

If any data item or group of data items within a flat file can contain multiple values within any given entry then the file becomes a hierarchical data structure. The central characteristic of a hierarchical data structure is the replication of data within the entries of the file. Furthermore, the replicated data has no meaning outside the context of the containing entry. A hierarchical data structure is built up from a set of entity types (nodes) and hierarchical relationships (arcs) between various pairs of nodes such that the resulting nodes and arcs form a tree if drawn. The tree is built up of a root at the top, various nodes or stems in the middle, leading to end-nodes or leaves at the bottom.

Problems exist with the hierarchical structure in that the structure must be known before any data extraction can be performed. This means the user must be conscious of moving or navigating up and down within the hierarchy which leads to this form of database being not particularly user friendly. Because of the rigid structure of the hierarchy, performing database updates, such as insertions and deletions, is impossible without using dummy nodes and arcs.

2) Network databases

Network database are extensions of the hierarchical systems. The data is again represented by nodes and arcs but where the hierarchical was a one-to-many structure the network database is a many-to-many structure. This means that any
particular node can have multiple parents as well as children. The internal structure of a network database can be very complicated indeed. All the problems of the complexity of usage of hierarchical database are present here and in fact are magnified by the complex structure. The network database is suitable for large applications where static relationships amongst the data stored are important. It is not suitable however for ad-hoc querying or small applications.

Navigational database systems tend to be overly complex, thus bringing about poor programmer and user productivity. They are difficult to modify because of the low level at which programs manipulate their structures. They are not based on any well-defined concept, and consequently are difficult to verify by analytical means and they provide poor facilities for interactive access because of their basis on an unfriendly concept of data and data manipulation.

2.1.3. Object orientated databases

Object-orientated database systems have their origins in object-orientated programming languages. The basic idea in both cases is that the user should not have to wrestle with computer-orientated constructs such as records and fields but should rather deal with objects (and operations) that more closely resemble their counterparts in the real world.

2.1.4. Relational databases

In the relational approach all the information is stored in two dimensional tables with no explicit links between the tables. These tables are manipulated using a query language based on relational algebra. Relational databases can have performance problems when used in large applications because database access programs are written at such a high level that they are unable to make use of low-level design decisions. The databases however are analytically testable due to their reliance on set-theoretic mechanisms.
2.2. The Use of Databases for Reliability Engineering

There are three main types of databases used in reliability work. This reflects three slightly different areas where computers have made inroads into the reliability engineering field. Some of these databases are also used in the fields peripheral to the reliability field, such as design.

2.2.1. Specification databases

This form of databases allows a user to select a component part number given a set of component characteristics or vice versa. The various international standards, down to detail specifications, are stored within the database and can be searched in terms of the characteristics of any component.

Most specification database contain more than just the component values, they often contain component outlines and timing diagrams. This sort of database is often used in conjunction with other system design tools such as layout and schematic editors. They allow the system designers access to component types, characteristics and sometimes models during the equipment design phase. An example is the CECC database held by CODUS[5].

2.2.2. Prediction databases

This type of database stores failure rates and prediction algorithms for various component types. The algorithms and failure rates are normally drawn from one of the commercially available reliability handbooks. The database is queried by the prediction software when parts count or parts stress analysis is being performed. Most of the reliability handbook suppliers market a database that performs the same function as the handbook. Many of these databases are based upon the MIL-217 handbook and so have all the drawbacks and advantages that those methods display[3][4]. Often this functionality is built in to a specification database like the CODUS system [5].
2.2.3. Field failure databases

This form of database is used to track the occurrence of failures in the field. There are various ways in which this can be done. Many companies only store information on the symptoms that are observed, the failures that occur and the result of any failure analysis. This information is stored primarily to allow field engineers to identify the fault quickly if it has occurred before. There is no information present on the time to failure or the size of the population at risk. This makes this form of database of limited use to the world at large since it only contains information applicable to the system that the failure are collected from and so is not useful to any other system types.

Other companies, particularly when involved in the provision of military equipment, high quality commercial equipment or equipment where the costs of repair are very high, collect much more information about systems in the field. This allows the company, because it may be a contractual obligation or because of a desire to improve the reliability, to monitor the reliability of their systems in the field on a regular basis. Normally, enough information is collected to provide the ability to calculate failure rates and other reliability statistics. Many of these companies then find it possible to sell this data to other companies in the form of reliability data-books. This was how HRD4 came into existence, when data collected for British Telecom's own purposes became useful to others [6].

2.3. Choosing a DBMS for a Field Failure Database.

Choosing a database implementation for reliability work is difficult because the natural relationship between the equipments, boards and components is complex. In general it can be considered a hierarchical structure where equipments contain an number of boards which in turn contains a number of components. However the definition of component is sometimes elusive. In some implementation it will be the actual physical component while in others it may be a daughter board or subassembly which may contain 'real' components. In general a component is considered to be the
non-repairable part of any system. It is possible, for economic reasons, for a board which is considered repairable in most cases to become disposable when enough of its components have failed. Hence it may become necessary to treat boards, or even equipments, as components in certain cases. This means that a primary concern has to be the flexibility of the database.

A second consideration must be the sort of queries that are to be performed on the system and the amount of information that will be stored. It is likely that any realistic field reliability database would contain information on thousands of equipment and hence millions of components. A query that attempts to count the number of components of a particular type will have great difficulty if the structure of the database is incorrect.

In order to choose a database type each DBMS type was considered in turn and a small example database was considered in order to get an estimate of the size of the database files that would be generated using each particular schema. The example database consists of one thousand equipments each of which contains twenty five boards and each board contains fifty components. The amount of information stored on each component is three hundred bytes, each board is one hundred bytes and each equipment is also one hundred bytes. Estimates of actual overhead in processing and programming are difficult to calculate since they very much depend upon the actual hardware platform and the programmers skill. For that reason they are not considered in the following discussion except in a qualitative sense.

2.3.1. The Flat File System

Using this system would require a data entry for every component tracked in the field. It would also have to contain information regarding the board and equipment that the component is contained within. This means, using the example database, that for each component in the field five hundred bytes is required. This gives an example database size of $500 \times 50 \times 25 \times 1,000 = 6.25 \times 10^9$ bytes or sixty two Megabytes of information.
This amount of data does not include any failure information that may exist. This is rather a large file and would take some time to search or query in any way. This effectively removes the flat-file database implementation from any further consideration.

The Loughborough database is much bigger than this example database and at present contains information on approximately 30,000,000 components, if a flat file implementation had been used then the data file would now be at least 15,000 Megabytes or fifteen Gigabytes in size.

2.3.2. The Navigable Structures (Hierarchical and Network)

The natural structure of a equipment in the field is matched quite well by the normal structure of a hierarchical database which would make the modelling of the real data fairly simple. Many of the problems of equipments, or boards, acting as components can also be solved quite easily. Size of data files would also not be a problems since each level of the database would contain information about the equipment or board only. Using the example database as a size estimate a database size of \((300\times50)+(100\times25)+(100\times1000) = 15,000 + 2,500 + 100,000 = 117,500\) bytes or more than one hundred and seventeen Kbytes of information is obtained.

This is not a particularly large file however when considering database structures it was decided that using this database format would increase the implementation effort since all queries would have to be implemented by a specialist programmer and the database could not be queried on an ad hoc basis, a decision which has been proved to be correct since one of the main uses of the database is to answer ad hoc queries from interested parties. It was this perceived inflexibility in the navigable structures that dismissed them from further consideration.
2.3.3. The Object Orientated Databases

These database were considered to be too close to the forefront of database technology at the time the decision was made and no appreciable benefit could be foreseen from using such a database. Hence this database type was dismissed.

2.3.4. The Relational Approach

The relational approach is flexible enough to allow the hierarchical structure of the actual data to be implemented, which means the size is similar to that quoted for the navigational approaches while remaining flexible enough to allow ad hoc querying. The implementation was felt to be simple enough that specialist programmers would not have to be employed in the implementation. The decision was therefore made that a relational database would be used. The actual commercial DBMS chosen was the Oracle system. This is described briefly in section 5.2.3 and more fully in [7].
3. **THE DATA MODEL FOR A FIELD RELIABILITY DATABASE**

Before a database for field reliability work can be designed it is necessary to have a model of the way in which the equipments being studied are structured and behave when placed in the field. The simplest available model for electronic equipment is to assume a hierarchical structure. Equipment can be considered to contain a number of circuit boards and these circuit boards contain a number of components. This means that in order to understand the way a component behaves information about the circuit board the component is mounted on is required. Then in order to fully understand the circuit board, information about the equipment in which the board is operating is also required.

This leads to the definitions in Table 1 for equipment, board and component.

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equipment</td>
<td>An electronic system that is self contained.</td>
</tr>
<tr>
<td>Board</td>
<td>A sub-system wholly contained within an equipment</td>
</tr>
<tr>
<td>Component</td>
<td>A non-repairable element of a circuit board.</td>
</tr>
</tbody>
</table>

**Table 1:** The definition of equipment, board and component

This gives us a simple model of equipment which is flexible enough to deal with most forms of equipment. This is shown in Figure 1. The boundaries between what is meant by equipment, board and component are vague and the terms can sometimes be interchanged. Specifically it is possible to consider a board to be an equipment or a board to be a component if this is what is necessary to model the actual equipment in terms of these definitions.

It is of course possible to consider a subset of the equipment as an equipment in itself and this allows data collection on systems where the total system structure is unknown since a sub-set of the boards in the system can be followed. This is often
necessary when multiple manufacturers products are used in a system and the data is being provided by only one of the manufacturers who has no access to the board structure information for the boards that are supplied and repaired by the other manufacturers.

![Figure 1: A model of the structure of electronic equipment](image)

The basic model of a component is shown in Figure 2.

![Figure 2: A simple model of a electronic component](image)
In database terms the component is considered to consist of the actual device itself, the encapsulation or packaging and the mounting system. This is all the items contained in the dotted line in the figure. This model has the disadvantage that it makes the tracking of mounting or packaging failures impossible but the advantage that a smaller amount of data has to be collected and this makes the data requirements of the database easier for data suppliers to comply with.

The circuit reference position, or the position on the circuit board where the component is mounted can be considered to be a repairable item. This is sometimes known as the 'socket' but does not necessarily mean that it is in fact a real component socket. This concept of the socket being repairable allows the application of renewal theory to be applied down to component level. (See section 8.5.2)

Another part of the equipment model necessary to design a field failure database is how equipment behaves when operating in the field. Most modern large equipments can be repaired by exchanging circuit boards. This means that the engineer will remove a particular board from an equipment, insert a replacement, and return the faulty board to the workshop for testing and repair. The repaired board will then be placed in storage and used at some later date. It may or may not go back into the equipment it came from. This is illustrated in Figure 3.
Many other scenarios are possible. One that appears with regularity, particularly when large systems such as a telephone exchange are considered is the 'repair pool' system. This occurs when the repair authority holds a stock of replacement boards that are used when an operating circuit board fails, however when the original board is repaired it is immediately replaced in the equipment and the replacement board is removed. This means that there are two sets of boards that have a very different usage figures. One set, which is kept in the repair pool, has a low usage while the boards operating in the equipment have a large usage. Both of these sets are included in the data set stored in the database. In this case the boards almost always return to the system from which they were removed. This is illustrated in Figure 4.

The ability to follow a board between equipments is called board traceability and the movement of these boards is termed board migration. Having this board traceability allows system level analysis to be performed at the circuit board level. Board traceability is automatic if component traceability is present and in order to calculate component lifetimes it is necessary to have component traceability. This is supplied by the ability to identify every particular instance of a component while operating in the field. In order to do this a component descriptor is defined. The most general case is given in Table 2.
Figure 4: Behaviour of systems in the field with a repair pool

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equipment manufacturer</td>
<td>This must uniquely identify the manufacturer within the database system</td>
</tr>
<tr>
<td>Equipment type</td>
<td>This must be unique for this equipment type for this manufacturer.</td>
</tr>
<tr>
<td>Equipment serial number</td>
<td>This must be unique for this instance of an equipment of this type supplied by this manufacturer</td>
</tr>
<tr>
<td>Circuit board manufacturer</td>
<td>This must uniquely identify this manufacturer within the database.</td>
</tr>
<tr>
<td>Circuit board type</td>
<td>This must be unique for this circuit board type for this manufacturer.</td>
</tr>
<tr>
<td>Circuit board serial number</td>
<td>This must be unique for this instance of an circuit board of this type supplied by this manufacturer</td>
</tr>
<tr>
<td>Component reference position</td>
<td>This must be unique for a particular circuit board type.</td>
</tr>
</tbody>
</table>

Table 2: Definition of component descriptor

Throughout the life of a component, the first three elements of this descriptor might change as the circuit board the component is mounted on migrates through a number of systems. The second set of four are constant throughout a components operating life until it fails or the circuit board is scrapped. Since component are considered to be non-repairable items when a component fails it is of no more interest except as to what the failure mode and mechanism were.
For most manufacturers the name of the circuit board manufacturer and the equipment manufacturer will be the same and hence the circuit board manufacturer can be omitted from a practical database.

In order to calculate any time metrics about an equipment a definition of operating time is required. For the IERI database the definition of operating time is taken as the time that the system is powered up and actually operating. This is shown in Figure 5.

![Figure 5: Definition of operating time for a board](image)

The operating time for the equipment is considered to be the sum of the operating times A, B and C. The repair times X and Y are considered to be zero as is the time K, which is the time spent in commissioning etc. Any failure events that occur in any of these periods are ignored. For practical reasons the time a system is operating is measured in calendar time but systems do not operate for the entire time the equipment exists in the field. They may only operate for 8 hours out of every 24. This leads to the concept of system usage. Usage is the percentage time spent actually operating. For example Figure 6 shows a simple usage pattern for an equipment. The usage percentage here would be obtained by using (1). A more complex example is shown in Figure 7. The general equation for usage is given in (2) and this can be applied to the figure in Figure 7 to obtain the actual usage.
Figure 6: Simple example of operating pattern of an equipment

\[
\% \text{Usage} = \frac{B}{A+B+C} \times 100 \quad (1)
\]

\[
\% \text{Usage} = \frac{\sum \text{UpTime}}{\sum \text{UpTime} + \sum \text{DownTime}} \times 100 \quad (2)
\]

Figure 7: A more complex operating pattern for an equipment
Usage can of course be difficult to estimate and is only practically collectable for an equipment type. A further consideration is the fact that some systems have a tri-state operating status. It is possible to consider the equipments to have three operating states, off, powered up and functioning. Not all of the components will be fully operating at all times, they may have power applied but may not be carrying any signals. This is shown in Figure 8.

![Figure 8: Tri-state equipment usage](image)

The model used by the IERI database considers the functional and powered up time to be up time and so the equation for usage, used for cases like that in Figure 8 is given in (3)

\[
\text{\%Usage} = \frac{\sum \text{Functioning} + \sum \text{Powered}}{\sum \text{Off} + \sum \text{Functioning} + \sum \text{Powered}} \times 100
\]

(3)

It must also be noted that the repair process can be an inexact exercise. Whilst seeking a cause of failure the repair engineer may remove more components than necessary. This can occur in two ways, either as multiple replacements, where the extra components are replaced in sequence until the board functions correctly, or as secondary failures, where the component replaced may have been damaged by the original failing component. In both cases it is necessary for the engineer, when filling in the failure report, to make some judgement on the real failed component. This means that in effect the failures logged are actually replacements and not true failures.
4. PROJECT ADMINISTRATION

The Field failure study described in this thesis is based at the International Electronics Reliability Institute (IERI) at Loughborough University of Technology (LUT) and is sponsored by the Ministry of Defense (MOD) via the Defence Research Agency (DRA) Professional Component Services (PCS). The database is a structured component failure database which has been constructed over a period of time using data from a variety of industrial organisations.

4.1. Setting up the database

In order to begin collecting field information and storing it in a database IERI held discussions with many system manufacturers, and although a number of companies expressed a willingness to become involved, IERI finally entered into detailed discussions with the research centres of three major British companies, viz:

GEC-Marconi Material Technology LTD, Caswell (Formerly Plessey Research Ltd).
GEC-Marconi Research Centre, Great Baddow
Northern Telecom Europe Ltd, Paignton, Devon (Formerly STC)

It was recognised that the research centres were in the best position to identify suitable data sources within their groups. Earlier, a pilot study involving two Danish companies, coordinated by the Danish Engineering Academy (DIA) under sub-contract to IERI had been initiated and this provided input from some Danish companies. Figure 9 shows the organisational structure of the project. The number of data sources varies between the companies and is increasing all the time. The figure also shows the technical support that IERI receives from other external agencies, notably on issues of a statistical nature from the Nottingham Trent University and on engineering and data issues from the DIA. Also shown are the support groups that IERI can draw upon for advice or expertise. These include computing and surface analysis services.
The project is unique in that three major companies in competition in the market place are jointly involved in a field study of this type. The success of the project, under such circumstances, is partly due to the confidentiality guaranteed by IERI to each of the individual companies. Individual data sets are desensitised by coding and lose their identity when the data is pooled. More details can be found in [1] and [8].

4.2. The Project Management Structure

Within the framework of the organisation three levels of meetings take place. Project management meetings are held twice yearly. These meetings are a forum which enables senior members of the consortium to meet and discuss progress and future policy. An industrial steering group (ISG) also exists which has within its brief the exploitation of the database information. However, it was realised in the early stages that a technical design committee or 'working group' including those involved in the acquisition, transfer and processing of data was needed to undertake the detailed work of the project. This technical design committee undertook the main database design.
4.3. The Technical Design Committee

The main task initially confronting the working group was to establish a data input format. This format requires information at the system level and subsystem level, in order to obtain the necessary data at the component level. Due to the hierarchical structure of the data, traceability at the system and subsystem level is essential in order to obtain dependable data at the component level. Such traceability, for example, would require a detailed knowledge of the repair loops of PCBs, including knowledge of their use and storage. To achieve meaningful statistical analysis information is required, not only on component failures, but on those components which have not failed and are at risk in the field during the period of observation. The final data format was arrived at after considerations of the minimum information necessary for analysis. After the establishment of the data input format, discussions regarding the selection of types of equipments to be followed in the field were made. The criterion for equipment selection was based only on the dependability of the data. The final equipment selections were based upon the results of a comparison between the companies stored data and the minimum requirements for the database. After the initial equipment selection had been finalised, the technical design committee addressed the practical problems associated with the detailed coding of the data input.

4.4. Data coding and collection issues

The type and manner of data recording, not surprisingly, differs between data sources. Since the data to be transferred to IERI was required in a fixed detailed format, some degree of data translation from the original stored records, be they computer or paper records, was required. Descriptions of the same component type can differ widely between different sources. None of the existing component classification schemes was found to suitable for application across all sources, with the aims of the project in mind. This meant that IERI had to devise a coding based upon the materials and construction of the devices. The essential extracts of the data coding and the data format can be found in appendix one and full details can be found in [9] and [10].
The need for component times to failure requires knowledge of the operating time of equipment, down-times, circuit board replacements etc. In some instances equipments are fitted with elapsed time indicators (ETI) giving an access to accurate time metrics. This is not always the case however and so the required times have to be estimated from delivery dates, commissioning dates and up and running dates etc. This means that some method had to be devised to indicate how the time metrics stored in the database were arrived at. This indicator allows the integrity of the data used in any statistical analysis to be assessed.

The technical design committee addressed many coding problems and the outcome of the decisions made can be seen in the data coding document in appendix one. This document was issued to all data supplying companies to enable them to code their data into the IERI format. It was acknowledged that this document would have to be adapted as time went on and more data source started supplying data. These additions would mainly be in the component type codes where new component codings would need to be added. All additional codes were discussed firstly with the technical committee and any relaxations and modifications to the document were also discussed. The technical committee also devised methods for dealing with incomplete data sets and other coding issues and were the first place where analysis results were discussed. To some extent the meeting guided the direction of the analysis work performed.
5. DATABASE IMPLEMENTATION

The implementation described in this section is completely different to the implementation that existed on the original mainframe due to the change in hardware and software.

5.1. Hardware

Loughborough University of Technology has standardised on Hewlett Packard (HP) hardware for the general purpose computing. The university’s computer centre runs a number of HP machine of varying powers. The machine chosen to host the IERI database was the HPC which is a HP9000 Series 800 Model S/100. This is a so called 'super minicomputer' based upon the Hewlett Packard Precision Architecture (or HP-PA) RISC processor with a capability of addressing 768 Megabytes of main memory and 85 Gigabytes of disk storage and delivering about 90 MIPS.

The HPC is configured with 176 megabytes of random access memory. Disk storage is provided via thirty-two 500 megabytes disks and backup storage via a 1/2" tape system. This machine is part of a four processor system with is connected via a network file system all of which are stored in the university’s computer centre’s secure machine room. Access to this room is strictly controlled via electronic keypad locks and is normally only given to computer centre staff.

Access to the computer from IERI is via a number of network connections that are connected through a gateway on to the universities ethernet backbone. This connection method is the favoured system since it allows the use of X-Window terminal and other network devices. Alternatives are available which use serial lines and packet assemblers and disassemblers (PADs). External access to the computer centre can be through the joint academic network (JANET) system which can in its turn be accessed via PSS and IPSS through a number of gateways. The JANET system is open to the Internet via a system known as JIPS.
5.2. Software

This section describes the software used in the implementation of the IERI database. The database is implemented using an Oracle system which runs on a UNIX based operating system called HP-UX and all of the programming is performed in C.

5.2.1. An Overview of the Unix Operating System

The HPC uses a version of the UNIX operating system known as HP-UX. This is a super-set of the original UNIX system which was described by Richie[11] as a multi-process operating system with facilities functionally similar to other minicomputer and mainframe operating systems. UNIX was designed by a small group of people for their own use and has, with almost no conventional marketing effort, become the most widely used operating system for mid-range computers. This extraordinary popularity can be credited to three fundamental properties

i) The UNIX system is a portable operating systems. New hardware vendors are attracted to it because it can be implemented fairly easily and in so doing a large community of users is inherited. A high-level language implementation of a piece of software can easily be moved from system to system, often with completely different processors and machine architecture, simply by recompiling.

ii) The UNIX system provides a particularly productive environment, particularly for programmers. Included in the UNIX system is a rich set of software development tools which can be applied in combination using simple operating system concepts which allow very many problems to be solved simply without resorting to producing a specific software tool to perform the task. These simple tools are reusable in very many situations.

iii) The UNIX system is an elegant design that is intellectually appealing and satisfying to use, many experienced users would say fun to use. It is difficult to
capture the essence of the systems elegance, but it has to do with power derived from simplicity rather than complexity.

There are of course possible criticisms, the UNIX system presents a rather terse and inconsistent interface to novice users, error reports are sketchy and the system has no interactive help facility. Some commands have strange name such as GREP and AWK. The system also assumes that users know what they are doing and the scope for mistakes is large. These problems are a consequence of the way in which the system was developed and first used. The developers were a Bell Laboratories computer science research group in the early 1970s. The members of this community were computer experts and they were building a system for their own use.

The UNIX system is unquestionably one of the great milestones in the history of computing. Its strengths are substantial and undeniable. It has no fatal flaws. Its shortcomings, which are largely historical artifacts are disappearing. It has proved to be an excellent choice for the development of the IERI database since it is flexible enough to cope with all the development and operating requirements needed for a project of this type. More details about the unix system can be found in [12]

5.2.2. C Language Overview

C is a modern general purpose programming language which features economy of expression, modern flow control and data structures, and a rich set of operators. It is not a very "high level" language, nor a very "big" one and is not specialized to any particular area of application. C was originally designed for and implemented on the UNIX operating system on the DEC PDP-11 by Dennis Ritchie[13], the language however is not tied to any one operating system or machine and compilers exist for a very wide selection of architectures. Many of the important ideas of C stem from the language BCPL indirectly through the language B which was developed for the first UNIX system on the PDP-7. For many years the definition of C was the reference manual in the first edition of The C Programming Language[14].

In 1983, the American National Standards Institute (ANSI) established a committee
to provide a modern comprehensive definition of C. This modern definition, the ANSI standard or ANSI-C was completed late in 1988 and was based on the original reference manual. Most professional level compilers now support this ANSI definition.[15]

5.2.3. ORACLE Overview

Oracle corporation was the first company to offer a true relational DBMS commercially and has continually been in the forefront of RDBMS development. The current Oracle RDBMS is a high performance, fault tolerant database management system especially designed for large database applications. The RDBMS includes the database manager and several tools intended to assist users and DBAs in the maintenance, monitoring and use of data.[7]

5.2.4. SQL Language Overview

SQL - Structured Query Language - formerly SEQUEL and still pronounced thus, was developed in the mid-1970s under the System R project at IBM’s San Jose laboratory. SQL is a concrete interpretation of the relational model. Originally offered commercially by IBM as SQL/DS, the language is now supported by most major relational database management systems. Furthermore an international standard now exists for the language[16] so SQL is therefore is both formally and de facto the standard language for defining and manipulating relational databases.

The language was designed to be used both as a stand-alone query language and as an extension to conventional high-level programming languages. The commercial implementations and standards activities have both confirmed this dual mode of use. Additional language constructs exist to support the embedding of the language within programming languages (referred to as embedded SQL).

An SQL database is a collection of tables. Each table consists of a set of records of fields or columns each of which has a defined type. A query is formulated
as a definition of a required table in terms of a mapping (i.e., derivation) from existing tables. It is possible to define these derived tables as views which can then be treated as base non-derived tables. These mappings are defined by means of a basic mapping block structure of the form shown in Table 3.

| SELECT <target-list> |
| FROM <table-list> |
| WHERE <condition> |

Table 3: Standard SQL Statement

The target-list is a list of the column names in the table being defined. The table-list is a list of tables from which the new table is to be derived. The condition is an expression that defines the extension of the new table. There are of course many other optional statements that can be used with the basic select statement.

All database suppliers have extended their implementations in order to take advantage of the extra facilities they have built into their products. In general, these extensions are purely for data formatting purposes and to control the behaviour of the product being used.

SQL operations can be issued either interactively or through an application program written in a conventional high-level language. Special facilities are provided to support the latter use of the language, resulting in what is normally called embedded SQL. This is a large subject in its own right complicated by the different interfaces required to the different languages. In general, any interactive SQL operation can be embedded within any programming language. More details of SQL implementation can be found in [17]
5.3. Basic Database Administration

This section describes some of the basic administration procedures such as how data is stored and data security and integrity are controlled.

5.3.1. Data and Program Storage

In order to make best use of the facilities available under the HP-UX system a file storage structure was created that allows flexibility while remaining logical. This system was designed so that it has maximum portability, particularly between the PC/MSDOS environment and the HP/UNIX environment. This is particularly important since IERI use PC to communicate with or 'front end onto' the HPC.

1) The Directory Structure.

Any data supplied to the database by the data suppliers is stored within the directory structure set up for the project, as is the source code and the executable files used for validation and analysis. The projects files are grouped in various sub directories according to the file type and the purpose of the actual files. Part of the directory structure is shown in Figure 10 and the directory names and the corresponding descriptions are given in Table 4.
Figure 10: The database directory structure

<table>
<thead>
<tr>
<th>Directory Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROOT</td>
<td>The top level directory below which all other directories reside. This is not necessarily the same as the systems root directory.</td>
</tr>
<tr>
<td>DB</td>
<td>This directory contains all the data sub-directories, the directories of release versions of the executable files and any current database output files. Each program will enforce its own directory structure underneath this directory.</td>
</tr>
<tr>
<td>DBDEV</td>
<td>This directory contains the directories which contain the development versions of the programs, the source code of the programs and the programming library. Each program will enforce its own directory structure underneath this directory.</td>
</tr>
<tr>
<td>TOOLS</td>
<td>This directory contains any programming support tools used by the project development team. Each program will enforce its own directory structure underneath this directory.</td>
</tr>
<tr>
<td>BIN</td>
<td>The directory that contains the compiled analysis programs, validation programs and all executable release version files.</td>
</tr>
<tr>
<td>FORMS</td>
<td>The directory that contains the form definitions of the program SQL*forms which is used to view and edit some data tables.</td>
</tr>
<tr>
<td>REPORTS</td>
<td>The directory that contains the report definitions of the program SQL*reportwriter which is used to produce some database output.</td>
</tr>
<tr>
<td>Directory Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>SQL</td>
<td>The directory that contains the sql scripts that are used to create, manipulate and query the database.</td>
</tr>
<tr>
<td>XX</td>
<td>This directory is representative of a data source directory. All information about a source is stored underneath this directory.</td>
</tr>
<tr>
<td>DATA</td>
<td>This directory contains the raw data files supplied by the source to the project.</td>
</tr>
<tr>
<td>SOFT</td>
<td>This directory contains the soft information files supplied to the project by the company.</td>
</tr>
<tr>
<td>LOG</td>
<td>This directory contains any log files produced by the data validation process.</td>
</tr>
<tr>
<td>REP</td>
<td>This directory contains the report files generated by the validation process.</td>
</tr>
</tbody>
</table>

Table 4: Description of the directory structure

Within the DBDEV directory there are a number of program directories. These directories contain all source code associated with the program under development. There is also a sub-directory used by the SCCS source control system which is discussed in section 5.4.2.

2) File naming conventions

The files used in the project have been allocated standard extensions depending on the file type. The file extension are given in Table 5. The actual name of the file, which replaces the asterisks below, must be no longer than 8 characters, the extension is always less than 4 characters. This allows the greatest portability between operating systems.
<table>
<thead>
<tr>
<th>File Name</th>
<th>File Type description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*.fai</td>
<td>File containing failure information</td>
</tr>
<tr>
<td>*.eqs</td>
<td>File containing equipment structure information</td>
</tr>
<tr>
<td>*.eqi</td>
<td>File containing equipment identification information</td>
</tr>
<tr>
<td>*.bds</td>
<td>File containing board structure information</td>
</tr>
<tr>
<td>*.sft</td>
<td>Soft information</td>
</tr>
<tr>
<td>*.rep</td>
<td>Output file from ERRFMT</td>
</tr>
<tr>
<td>*.log</td>
<td>Log files from validate</td>
</tr>
<tr>
<td>*.doc</td>
<td>Documentation files</td>
</tr>
<tr>
<td>*.sql</td>
<td>SQL query files</td>
</tr>
<tr>
<td>*.txt</td>
<td>Other readable files</td>
</tr>
<tr>
<td>*.rep</td>
<td>Report files</td>
</tr>
<tr>
<td>*.1</td>
<td>Man Pages</td>
</tr>
<tr>
<td>*.c</td>
<td>C source code</td>
</tr>
<tr>
<td>*.pc</td>
<td>Pro*C source code</td>
</tr>
<tr>
<td>*.h</td>
<td>C header files</td>
</tr>
<tr>
<td>*.et</td>
<td>Error table definition files</td>
</tr>
<tr>
<td>*.o</td>
<td>Object code modules</td>
</tr>
<tr>
<td>*.frm</td>
<td>SQL*forms input files</td>
</tr>
<tr>
<td>*.lst</td>
<td>Output files from SQL*plus</td>
</tr>
<tr>
<td>*.a</td>
<td>Object code libraries or archives</td>
</tr>
</tbody>
</table>

**Table 5: Standard File Extensions**

Executable files have no extension under UNIX. These file type are strictly adhered to since the operation of some of the support scripts and make files depend on the file extension to identify the file type and so select the correct tool.

The first part of a source code file name is also controlled as shown in Table 6. This is not an exhaustive list and if a particular extension makes sense for a particular programs function then that should be used. The ?? would be replaced by two letters selected from the name of the program eg Chkdata is shortened to ’cd’

32
<table>
<thead>
<tr>
<th>Stem</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>??info</td>
<td>Informational functions</td>
</tr>
<tr>
<td>??sql</td>
<td>Pro*C functions</td>
</tr>
<tr>
<td>??proc</td>
<td>The main processing section</td>
</tr>
<tr>
<td>??file</td>
<td>All file handling function</td>
</tr>
</tbody>
</table>

Table 6: File stem naming conventions

A file with the same stem as the final program is also defined and this contains the start-up and finishing code for the program as well as any functions that do not fit into the other categories.

5.3.2. Data Security

The companies supplying data to the IERI database were concerned that the failure data they supplied was confidential and not accessible to anybody except the members of IERI. This meant that the security of the data was of prime importance. The data to be secured is available in a number of formats at any one time. These formats include machine readable raw data, structured data files produced by validation and database files.

The raw data is stored in a locked cupboard in a restricted access room where the key holders are limited and known. The data stored on the HPC is subject to the controls of the HP-UX environment. The HPC machine is accessible from any of the university’s network services. At any one time a large number of people can be logged in at once. These will include the university’s staff and students as well as a small number of people from outside the university. The UNIX system however is a relatively secure system and was designed with mechanisms to support this multiple user access. Security is enforced through a system of password log-on and through file access controls. The IERI files are protected by being considered as a special group called ELCR. This means that only members of the institute can access the files. Without implementing access control lists (ACLs), group access is the most granular access that allows more than one person to have access to the files. It is not possible to restrict access to some members of a group without forming another group, nor is
it possible for a file owner to give access to any single individual without creating a special group that contains the two people involved. Group membership can only be changed by a system superuser. The current version of the HP-UX has C2 compliant security level (as defined by the DOD). A more secure system could be set up but this would involve the use of the access control lists already mentioned and this leads to a great deal of processing overhead and since ACLs are not part of the original operating system design they are not transparent during processing and require separate manipulation when data files are changed.

The data stored within the Oracle system itself is subject to the security implemented by the Oracle system which exists over and above that supplied by the UNIX system. All Oracle programs are password protected and all tables within the database are set up so that only the table owner can change the access rights to them. By default only the table owner can read and write to them although this can be relaxed at the owner’s discretion. The IERI database has been implemented so that only one log-on name can write to the database. This prevents any changes being made by anyone reading the data or writing programs that use the system. The person who has write access is a dummy name specially created for the purpose and all changes made by this user can be audited so that any modifications are known.

5.3.3. Data integrity.

Data integrity covers three main areas. The ability to be confident that the data received is good, real data, the ability to recover from system crashes via use of backup systems and the ability to keep track of a data set as it passes through data validation and into the actual database.

In general the data supplied to the IERI database has come from dependable sources via a central point within a data source that is well known to the IERI database operators. This includes personal contact between the suppliers and the operators on a regular basis. It is felt that the building up of a professional relationship
over time and the common aims of the operator and the supplier has reduced the risk of the supply of dubious quality data.

The second issue of data recovery is provided by Loughborough University's computer centre who operate a backup system. This system means that backups are performed weekly, and are stored for a fortnight before being overwritten by a new backup. An annual backup also exists. All are stored remotely from the main computer site. At any one time there are 3 sets of backup stored. All raw data will also be stored in within IERI on floppy disks and cartridge tape. The database itself is dumped occasionally and the data files produced are spooled off onto tape.

Thirdly in order to keep track of a data set while it is at IERI each data set is given an unique number. This number depends on the data source name, the data file name and the data type. This unique number is associated with this data set no matter how many times it is updated. This allows reference to be made via a unique identifier. In order to identify an instance of a data set, every time the validation programs are run a uniquely numbered validation report is generated which can be used by IERI and the data supplier to correct the data and refer to it during this process. All these generated numbers are stored within database tables and reports can be generated that list the history of validation of any particular data-set. More information about this process is given in section 7.7.
5.4. Software methodology

The software methods are defined in order to guarantee a high level of software quality, ensure that end user needs are satisfied, and ensure the conformance of the end product to the actual requirements.

5.4.1. Use of software within the project

Before describing the actual procedures implemented it is necessary to define the purpose of the software developed. There are basically four different areas within this project where software is developed;

1) Data Validation and Data Input

The purpose of the validation software is to take the raw data files that are supplied by the data sources and check the data for inconsistencies, then when all errors have been removed to load the data into the database. This is described in detail in section 7.5. The software supplied for this task is required to be easy for an operator to use, to ensure that the status of any data within the validation scheme can be tracked, and to produce a report for the data supplying company so that data errors can be found and corrected.

2) Raw data extraction

The purpose of the raw data extraction is to extract from the database a set of data that meets some imposed requirement. For example the extraction and reformatting necessary in order to load data into already existing analysis programs such as BMDP or SAS. The requirements of this software are to supply a file, or printout, in the format specified by the final raw data user.

3) Analysis software

The purpose of the analysis software is to provide statistical and investigative results suitable for publication by the database operator. The requirements are that the
software performs according to strictly defined methodologies when producing results and that these results are known to be correct under all circumstances.

4) End user software

The purpose of the end user software is to present to some end user the results produced by the database and associated work. It is required that this software is user friendly and presents information in an obvious way.

In order to provide a framework to meet the requirements discussed above then it becomes necessary to implement a series of methods that apply to all software developed under the project. These methods include the definitions of the tools to be used to build the software, the definition of the coding style and the programming language to be used, a series of guidelines about how programs should be written so that they have a common interface and the creation or definition of the function libraries that can be used.

5.4.2. Development tools

The tools used for development by the programmers on the project must be defined in order that the software can be rebuilt in the quickest possible way be someone not familiar with the development project. This allows the porting of the software to a foreign host providing the defined tools are available on that host. In general the tools chosen are as generic as possible, or are available in source code form and so can be built on the foreign host. In order to allow this the tools chosen have been developed by end users on other systems and so are not tied to any one system supplier. This does however mean that the support for these tools is not always as available as it would be if the tools were standard system tools. However in general the tools are better specified and better performing than any available from the system supplier. The following tools are used by the project.
1) DMAKE

Dmake is a program compilation tool, similar in purpose to the standard Unix make tool, that allows the executable version of the software to be compiled from the source code modules. The unix make program was replaced because of the added functionality required by the project. As already noted there are a number of types of program files that are used by the system. The program files that are required to build the software are described in Table 7.

<table>
<thead>
<tr>
<th>File Name</th>
<th>File Type Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>*.c</td>
<td>Standard C language source files</td>
</tr>
<tr>
<td>*.h</td>
<td>Standard C language header files</td>
</tr>
<tr>
<td>*.pc</td>
<td>Pro*C language source files (These consist of a mixture of standard C and embedded SQL command statements and must be passed through a pre-processor to generate C source files)</td>
</tr>
<tr>
<td>*.et</td>
<td>Error table definition files (These must be passed through a preprocessor to generate standard C files)</td>
</tr>
</tbody>
</table>

Table 7: File types containing source code supported by DMAKE.

The standard unix make only provides make rules for the first two of these and although extra rules could be added to each independent make file to manage the files this would lead to very large and slow make files. A second consideration is the use of the source control system since the standard make does not provide rules to allow the extraction of the source files from the SCCS system by default and performing this task manually each time a compile is required was considered cumbersome. Dmake is a totally configurable program that allows the addition of rules to a single start-up file which affect all the makes done using that program. Hence dmake is used to compile all but the most trivial of programs and is also used in the generation of the database construction scripts that are used when the database is defined. The usage of dmake is described in [18]

2) MEMACS

This tool is a program editor that provides many advantages over the standard unix editors. It has many features that are too numerous to mention here but one of the most important is the configurability of the editor. This allows the provision of start up scripts that configure the editor according to the type of files that are being
edited. This allows the source file layout to be adjusted as it is edited so that it conforms to the project imposed coding style. The editor is also used as a component part in a number of the programs that have been written for use with the project such as the change control system. The usage of memacs is described in [19]

3) SCCS

SCCS is the standard unix source configuration and control system it is generally accessed through a large number of program that each supply one feature that is required to successfully manage the source control. This made the use of the system difficult so a single program was written that allows access to all the standard programs through a single interface. This simplified the SCCS system considerably and was useful when SCCS is used by dmake. This also met a shortcoming of HP-UX which does not supply such a program whereas most other manufacturers do. Details of the SCCS system can be found in [20].

This source code control system is used for the control of all text files within the project. This includes documentation files as well as source files. Every software development directory under the DBDEV directory has a SCCS directory. This directory contains the SCCS control files. The SCCS system allows the retrieval of any previous version of the software at any time.

4) GNUplot Graphical software

The HP-UX machine has a number of very powerful graphical packages but none was suitable for the project since a graph system was required that could be started from inside any of the analysis programs and that could be controlled by a script file. It was also necessary that the graphics package was transportable between the HP and the PC. Hence the program GNUplot was chosen. The use of GNUplot is described in [21]
5.4.3. Coding style control

The C Standard [15] is adhered to for coding purposes. The purpose of this standard is to promote portability, reliability, maintainability and efficient execution of C language programs on a variety of hardware platforms. This ensures that, if at a later date, the hardware platform needs to be changed then this can be achieved with minimum effort. In order to promote a common style between a number of programmers the Indian-Hill guide [22] to style is also adhered to, with suitable changes to accommodate the needs of the IERI database, such as embedded SQL.

5.4.4. Software Library Development and Usage

In order to ensure that the programs have a commonality within the project a number of program libraries are used. Some of the libraries have been taken from external sources because it was felt that duplication of the functions within these libraries was unnecessary. Each library used is discussed in turn below.

1) ParseArgs

Parseargs is a set of functions to parse command-line arguments. Unlike "getopt" and its variants, parseargs does more than just split up the command-line into some canonical form. Parseargs will actually parse the command-line, assigning the appropriate command-line values to the corresponding variables, and will verify the command-line syntax (and print a usage message if necessary). Furthermore many features of its parsing behaviour are configurable at run time. Parseargs also allows for options that take an optional argument, and options that take a (possibly optional) list of one or more arguments. Parseargs consists of a set of function calls to parse command line arguments from the command-line, from strings, from linked-lists and from string-vectors [23].
2) Tcl

Tcl stands for 'tool control language' and is pronounced 'tickle.' It was developed by John Ousterhout of the University of California at Berkeley and is actually two things; a language and a library. First Tcl is a simple textual language, intended primarily for issuing commands to interactive programs such as text editors, debuggers and shells. It has a simple syntax and is also programmable, so Tcl users can write command procedures to provide more powerful commands than those in the built-in set. Second Tcl is a library package that can be embedded in application programs. The Tcl library consists of a parser for the Tcl language, routines to implement the Tcl built-in commands and procedures that allow each application to extend Tcl with additional commands specific to that application. Tcl provides facilities that allow access to an application native variables through the use of Tcl variables and functions. This makes it ideal for a start-up file parser. Many programs use the concept of start up files that are ASCII files that can be edited to change the default settings of the programs. The Tcl interpreter and language is used by all the programs in order to provide this facility. The use of this allows the configuration of the project programs at run time. The Tcl language and library is described by the documentation supplied as part of the distribution package [24]

3) Com_err

This library was developed by the student information processing board of the Massachusetts Institute of Technology and is in the public domain. It is portable to many systems and is available in source code form[25]. In building application software packages, a programmer often has to deal with a large number of libraries, each of which can use a different error reporting mechanism. This is certainly true in the case of the database software. The com_err library is an attempt to present a common error handling mechanism to manipulate the most common form of error code. A list of up to 256 text messages is supplied to a translator program along with a three or four character 'name' of the error table. The translator produces a procedure which can be used to make the new error table known to the com_err library and hence any error code the library generates can be converted to the relevant error message. Error table 0 is defined to match the UNIX system call error table and a
program can have as many error tables as necessary in order to handle the error returned. For the database software each IERI written library has its own error table and the program would add an error table of its own as well. The compiler for the error source file is called et_compile. It takes a single argument, the path name of the file to compile and produces two output files, a c header file, which contains definitions of the numerical values of the error codes described in the input file and a c source file which contains the initialization code for the error table. In order to use the err_com library a simple function is used, this then provides a method for printing error messages to the standard error stream stderr and the library has a number of functions which can alter the behaviour of this function to allows the output to windows, files, printers etc. This library ensures that a standard error reporting mechanism is used in each program and helps the programs produced by the project to have a standard interface.
4) LibCord

This library was developed by IERI programmers. It consists of a number of modules that address different aspects of program execution. The function of each module is described in Table 8.

<table>
<thead>
<tr>
<th>Module</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error handling</td>
<td>To implement a standard method of dealing with program errors. These functions encapsulate the functions provided in the Com_err library</td>
</tr>
<tr>
<td>File handling</td>
<td>To implement a standard, audit-able method of dealing with files. The functions allow access to files by name and by file pointer. All incoming and outgoing data streams can be audited and counts of number of bytes written and read can be made.</td>
</tr>
<tr>
<td>Time Dependent Analysis</td>
<td>To provide time dependent analysis such as failure intensity analysis and M(t) analysis.</td>
</tr>
<tr>
<td>Help system</td>
<td>To implement a help system that can be called interactively from within any program or from the command line. This was developed to allow the implementation of a log in shell for external users. The help system is hypertext based.</td>
</tr>
<tr>
<td>Information retrieval</td>
<td>To implement a standard way of getting information from the system, eg the date and time, and to provide a method of finding user names, program versions etc.</td>
</tr>
<tr>
<td>SQL interface</td>
<td>To implement a standard interface to the Oracle processes. These functions provide logon and logoff functions as well as SQL error functions. No functions are provided for data retrieval</td>
</tr>
<tr>
<td>Startup</td>
<td>To provide a standard way of starting up and exiting a program. Functions here handle the argument passing and startup file processing as well as journalling and output redirection. These functions act as a buffer between the library user and the lower level TCL library and parser/g functions.</td>
</tr>
<tr>
<td>Statistics</td>
<td>To provide statistical support for calculating failure rates, confidence limits etc.</td>
</tr>
<tr>
<td>Low Level</td>
<td>These functions are used within the library to implement journalling and debugging and initialise the libraries internal data structures and global variables. They are not for external use. Other, more high-level, functions exist for the library users.</td>
</tr>
</tbody>
</table>

Table 8: Description of LibCord Modules

The library is defined in a number of header files and the library user includes these files within the program that wishes to use the library. In order to initialise the
library the 'stndCORD.h' header file must always be included. Other header files are optional dependent upon which sections of the library are to be used. More details of these functions can be obtained from the programmers documentation or the system, man pages.

5.4.5. The Common Program Interface

The use of a common program interface is an important feature when using command line based programs since they can be terse and difficult to use. The command line options used by all IERI written programs are shown in Table 9.

<table>
<thead>
<tr>
<th>Options/Arguments:</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-D [&lt;debug&gt;]</code></td>
<td>Debug mode if set</td>
</tr>
<tr>
<td><code>-F &lt;filename&gt;</code></td>
<td>Run TCL file after initialisation</td>
</tr>
<tr>
<td><code>-H</code></td>
<td>Invoke help system, if any</td>
</tr>
<tr>
<td><code>-h</code></td>
<td>Display help page</td>
</tr>
<tr>
<td><code>-i</code></td>
<td>Start TCL interpreter</td>
</tr>
<tr>
<td><code>-j</code></td>
<td>Write journal file</td>
</tr>
<tr>
<td><code>-q</code></td>
<td>Quiet mode if set</td>
</tr>
<tr>
<td><code>-s</code></td>
<td>Suppress use of startup file</td>
</tr>
<tr>
<td><code>-v</code></td>
<td>Give version information</td>
</tr>
</tbody>
</table>

Table 9: The standard command line options

All IERI written programs have at least this common option list, program specific options are added to these common options. These common options are more fully described in Table 10.
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-D [&lt;debug flag&gt;]</td>
<td>Run the program in debug mode. The &lt;debugflag&gt; allows control over the debug messages that are printed. If no &lt;debugflag&gt; is given then full debug information is assumed.</td>
</tr>
<tr>
<td>-F &lt;filename&gt;</td>
<td>This specifies that the tcl initialisation &lt;filename&gt; should be used for initialisation instead of the default which will be named after the program but with a '.ini' extension.</td>
</tr>
<tr>
<td>-H</td>
<td>Start the interactive help system. This option is intended for use inside any user shell that would be provided such as CORDdba.</td>
</tr>
<tr>
<td>-h</td>
<td>Display the programs help or usage page and exit. This page lists command line options and give a terse description of the programs use.</td>
</tr>
<tr>
<td>-I</td>
<td>Start the internal TCL interpreter. In order to process the TCL startup file it is necessary for each program to have a built in tcl interpreter. This can be started using this option and will give a command line feel to any program. Any standard tcl command or any program startup command can be typed.</td>
</tr>
<tr>
<td>-J</td>
<td>Start the journalling process. This causes the program to write to a file '*jou', where * is replaced with the actual program name, the current status at various times in the program. This allows the progress of a program to be considered in detail. Many of the library functions write journal entries, an example is the file handling modules where all files accesses and data counts read/written are placed in the journal file.</td>
</tr>
<tr>
<td>-Q</td>
<td>Redirect the stdout stream to the unix /dev/null device. This has the effect of suppressing any output from the program since any printing to the stdout stream, such as printf, will go to the null device.</td>
</tr>
<tr>
<td>-S</td>
<td>Suppress the use of a start up file. This causes the program to skip the reading of the start up file.</td>
</tr>
<tr>
<td>-V</td>
<td>Display version information and exit.</td>
</tr>
</tbody>
</table>

Table 10: Description of standard program options

All programs obtain information from the environment and display it in a common way and all programs use tcl initialisation files to configure themselves on start up if necessary.
All programs are capable of maintaining a journal file, where they occasionally write their current status as they run. All programs also support the display of debug information.

Another reason for using a common interface is used that this interface can be changed, to X-windows say, by careful re-coding of the common function calls. It also allows the provision of a standard single driving program or shell, which can be used to restrict a user to a subset of a particular operating system commands. This would allow the database to become a remote access database where users could log in across the network but because of a user shell would be unable to compromise the security of the system. Such a shell has been written, it is called CORDdba, and it is used to perform most of the database administration. A further reason for using a common program interface is that help sub-systems can be developed between programs, dynamic linking can be used to reduce the size of executable programs and porting to different operating system becomes easier since once the common library is done then the task is much simplified.

All IERI program should follow the format given in Table 11 for program layout. This skeleton program implements the standard startup and exit functions. The programmer needs to replace the commented lines (/*....CODE.....*/)) with the code necessary to complete the program. Additional startup commands should be added to the parseargs definitions tables at the start of the program where indicated.
5.4.6. Debugging support

As already mentioned the debugging of programs that include embedded SQL can be difficult since the line numbers stored in the executable files does not match the line numbers in the original 'pc' files due to the expansion used to turn the SQL statements into C code. This means that the system debuggers are not usable for the project 'pc' files. In order to get around this problem extensive debugging support has
been built into the library functions and functions are provided to support debugging in the user programs.

1) The debugging function.

This function writes formatted text strings to stderr if debugging has been enabled by the relevant default command line option. The debug output is intended to be a detailed record of what the program did and so should be called at least on entry to every function. Provision is made for the programmer to display only those types of debugging information that are required, so more rather than less calls to debug should be made. The first two characters of the format string can be used to control the type of debug message displayed. These characters are shown in Table 12. The lower case letters are used by the library functions to distinguish the message output from the user code.

<table>
<thead>
<tr>
<th>Character code</th>
<th>Debugging Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>F or f</td>
<td>Details of a function call</td>
</tr>
<tr>
<td>P or p</td>
<td>Progress message</td>
</tr>
<tr>
<td>V or v</td>
<td>Values of variables</td>
</tr>
<tr>
<td>L or l</td>
<td>Loop indicators/counters</td>
</tr>
</tbody>
</table>

Table 12: Debugging flags for the debug call

The debug option on the command line (described in Table 10) is used as shown in Table 13. The bitmask codes are ORed together to get the required level of debug functionality.
### Table 13: Bitmasks for the -D option to programs

<table>
<thead>
<tr>
<th>Bitmask code</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Display function details for user programs</td>
</tr>
<tr>
<td>02</td>
<td>Display progress indicators for user programs</td>
</tr>
<tr>
<td>04</td>
<td>Display variable details for user programs</td>
</tr>
<tr>
<td>08</td>
<td>Display loop details for user programs</td>
</tr>
<tr>
<td>10</td>
<td>Display function details for library functions</td>
</tr>
<tr>
<td>20</td>
<td>Display progress indicators for library functions</td>
</tr>
<tr>
<td>40</td>
<td>Display variable details for library functions</td>
</tr>
<tr>
<td>80</td>
<td>Display loop details for library functions</td>
</tr>
</tbody>
</table>

2) **Journalling**

The journalling feature can be used to assist with the debugging of a program. The philosophy of journaling is that the program writes a message to the journal whenever it completes a particular action. This can be used to print out intermediate results and progress messages.

3) **Tcl Interface**

A Tcl interpreter can be started at any time by calling a single function. This interpreter will contain all the Tcl functions used by the start up processor and can access system variables if functions are included in the source code and are added to the interpreter on startup. The interpreter already contains functions to access and change the global variables, such as debug state, journaling etc. This TCL interpreter would permit the person debugging the code to obtain the value of any variables interactively whilst a program is running. It would also be possible to attach the Tcl interpreter to a system signal and start the interpreter on receipt of that signal. This would provide a built in debugger for programs that are difficult to debug in any other way.
5.4.7. Program Verification

In order to ensure that programs work as they should a modular approach was taken to testing and verification. All modules were tested independently by the programmer and a log book of expected versus actual results was kept. This log book was used in conjunction with the source control system to note exactly which version of a module was tested. The modules are not made available for general use by other programmers until they had been verified and signed out for use by the originator. This testing of modules often meant writing framework programs to test the functionality of the different execution paths through the module.

Actual testing of programs is done using specially generated test data sets. These test data sets are developed in parallel to the program development and as many combinations of possibly troublesome data are added to them. It is recognized that it is impossible to remove all bugs from a program so these test data sets evolve as problems are found with the developed programs in actual use.

All analysis and validation programs append the current version to their output. The version data contains the SCCS generated version number and the date of compilation. Using this data it is possible to identify the version of the program that produced the information and hence correct that version.

5.4.8. Documentation

Any software project can be viewed as consisting of a number of layers. The lowest layer is an actual code module. These modules are grouped into programs, the program into suites (for IERI written programs only) or toolsets (mixture of purchased and written software) and all the programming suites into a project. Each level removes a layer of detail so, for example, the person dealing with the project as a whole need not be concerned by the details of the actual program code that make up the project. This technique of abstraction is known as modular programming. It has a number of advantages which ease the development of the project as a whole. For
instance any number of programmers can work on a module each and as long as the abstraction is managed correctly they need not be concerned as to what use their individual modules will be employed.

<table>
<thead>
<tr>
<th>a) Project level</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>b) Program suite/toolset</td>
<td></td>
</tr>
<tr>
<td>c) Individual program</td>
<td></td>
</tr>
</tbody>
</table>

**Table 14: Different Layers of Project Documentation**

Having these three levels make the task of documenting the system more modular. Some documents can be written that apply to all programming in the project while others will only concern one of the many programs involved. There is also dependency effect within this system. The program level (level-c) documentation assumes knowledge of level-b and level-a documents. It also assumes knowledge about any documents referenced within these levels.

There are also many classes of documentation. These are shown in Table 15

<table>
<thead>
<tr>
<th>a) Management overview.</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>b) User documentation.</td>
<td></td>
</tr>
<tr>
<td>c) Programmer/maintenance documentation.</td>
<td></td>
</tr>
<tr>
<td>d) Technical documentation.</td>
<td></td>
</tr>
</tbody>
</table>

**Table 15: Different Classes of Documentation**

The amount of detail that needs to be conveyed by the documentation is suited to the needs of the person using that documentation. The management overviews only give vague details about the coding techniques used but more detail about why such a task may be performed. The user documents describe the usage of the system and how to interpret system messages as well as techniques for customizing the system to the exact user requirements. The programmer documentation describes exactly the coding techniques and the algorithms used in the programs. These documents are intended to allow the programs to be modified and maintained at some future date, possibly by someone who did not write the programs. They also allow extensions to the programs to be written. The technical documentation is slightly different in context to the main documentation: it contains formal specifications for the programs used, test data sets and test runs for the particular program being described. Its main purpose
is to prove that the programs work as they should and to enable the porting of the software to another language or platform.

This means that there are four levels of documentation for any program in the system eg, a single program will have an overview, a user guide, a programming guide and a technical documentation set. Similarly, both the program suite that the program belongs to and also the project that contains the program suite will have the same four levels of documentation. In most cases the technical documentation consists of the input and output specifications and the verification document. The user guide is the UNIX man page and the programmer documentation consists of the commented source code and library module man pages.
6. DATABASE STRUCTURE

Using the model for electronic equipment developed earlier, in section 3, a database was developed using a relational database structure. This database consists of four main tables and a set of files which contain textural information such as the answers to the soft information questionnaire which are listed in Table 16 along with a description of the content of each table and the alias the table is known by in the project.

<table>
<thead>
<tr>
<th>Table Name</th>
<th>Table Alias</th>
<th>Table Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failure</td>
<td>FAI</td>
<td>This contains information about the actual observed failures.</td>
</tr>
<tr>
<td>Board Structure</td>
<td>BDS</td>
<td>This contains information about the structure of the individual boards</td>
</tr>
<tr>
<td>Equipment Structure</td>
<td>EQS</td>
<td>This contains information about which board is in which equipment</td>
</tr>
<tr>
<td>Equipment Identification</td>
<td>EQI</td>
<td>This contains information about each equipment in the field.</td>
</tr>
<tr>
<td>Soft Information</td>
<td>Not a Database Table</td>
<td>This contains useful information about a data source including the results from the soft information questionnaire</td>
</tr>
</tbody>
</table>

**Table 16: The four tables that make up the IERI database**

The actual contents of these database tables are documented below in Table 17 to Table 20. Entries in emboldened italics are essential to the interrelationship between the tables. Entries in capitals are considered as essential information if any meaningful analysis is to come out of the information.

**Table 17: Database failure table**

<table>
<thead>
<tr>
<th>COMPANY</th>
<th>FAILURE RECORD NUMBER</th>
<th>EQUIPMENT TYPE</th>
<th>SIMULTANEOUS COMPONENT FAILURE IDENTIFIER</th>
</tr>
</thead>
</table>

**Table 17: Database Failure Table Continued**
<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>PART NUMBER</th>
<th>SCREENED LEVEL OF COMPONENT</th>
<th>CIRCUIT REFERENCE POSITION</th>
</tr>
</thead>
</table>

Table 17: Database Failure Table Continued

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>PART NUMBER</th>
<th>ENCAPSULATION AND MOUNTING DETAILS</th>
<th>SCREENED LEVEL OF COMPONENT</th>
<th>CIRCUIT REFERENCE POSITION</th>
</tr>
</thead>
</table>

Table 17: Database Failure Table Continued

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>PART NUMBER</th>
<th>ENCAPSULATION AND MOUNTING DETAILS</th>
<th>SCREENED LEVEL OF COMPONENT</th>
<th>CIRCUIT REFERENCE POSITION</th>
</tr>
</thead>
</table>

Table 17: Database Failure Table Continued

<table>
<thead>
<tr>
<th>OPERATING TIME OF SYSTEM TO FAILURE</th>
<th>OPERATING TIME OF BOARD THIS TIME AROUND</th>
<th>reported failure mode of component</th>
</tr>
</thead>
</table>

Table 17: Database Failure Table Continued

<table>
<thead>
<tr>
<th>COMPANY</th>
<th>EQUIPMENT TYPE</th>
<th>COMPANY DESIGN OR PART NUMBER OF BOARD</th>
<th>NUMBER OF BOARDS OF A PARTICULAR DESIGN NUMBER</th>
<th>primary function of board</th>
</tr>
</thead>
</table>

Table 18: Database Equipment Structure Table

<table>
<thead>
<tr>
<th>COMPANY</th>
<th>COMPANY DESIGN OR PART NUMBER OF BOARD</th>
<th>COMPONENT TYPE GENERIC DESCRIPTION</th>
<th>COMPONENT PART NUMBER</th>
<th>COMPONENT ENCAPSULATION AND MOUNTING DETAILS</th>
<th>NUMBER OF COMPONENTS ON THE BOARDS</th>
</tr>
</thead>
</table>

Table 19: Database Board Structure Table

<table>
<thead>
<tr>
<th>COMPANY</th>
<th>EQUIPMENT TYPE</th>
<th>EQUIPMENT SERIAL NUMBER</th>
<th>UP AND RUNNING DATE OF EQUIPMENT</th>
<th>END DATE</th>
<th>USAGE</th>
<th>ENVIRONMENT</th>
</tr>
</thead>
</table>

Table 20: Database Equipment Identification File
The following provide fuller descriptions of the fields described in the tables

6.1. Company

The company that supplied the information and manufactured the equipment. It is assumed that the manufacturer of an equipment is also the manufacturer of the boards used in that equipment. In some cases the manufacturer means the repair authority for the equipment and not the actual manufacturer. This is part of the component descriptor.

6.2. Failure Record Number

This provides an unique identifier for a failure record and provides a link back to the companies original field failure report if this proved to be necessary after data analysis.

6.3. Equipment Type

This would normally be the equipment part number or the manufacturers stock number for this equipment. It must be unique within a particular manufacturers data set. This is part of the component descriptor.

6.4. Simultaneous Component Failure Identifier

An identifier indicating how many failures occurred simultaneously. Simultaneous failure are those failures that occur at the same time on the same board or in the same equipment.

Simultaneous failure can occur at two distinct levels and can be reported differently by different data suppliers. It is also very difficult to decide whether or not the failure are real failure or just multiple replacements. This last can only be discovered by talking with the data supplier involved.
1) Board level.

This is when two or more failures occur at the same time on the same board. It is possible that they will be reported on the same failure report or they might be reported on separate reports. Board level simultaneous failures are distinct from multiple replacements and the data supplier has to ensure that only actual failures are reported to IERI. Any other removals, due perhaps to repair practices, should not be included.

2) Equipment level

This is when two or more boards in a particular equipment fail at the same time (they may even be multiple failure on both of the boards).

These two cases are distinct from one another and care must be taken when trying to decode from the failure reports what actually happened in the systems. There is also a problem with what is meant by the term 'simultaneous'. Sometimes the granularity of the data collected can cause what appear to be simultaneous failures when in fact the failures occurred some small time interval apart which is shorter than smallest time interval recorded in the data collection process. This of course will be different for each data source so that a two failures that are considered simultaneous by one source may be considered to be two separate failures by a different source. This means that care must be taken when comparing simultaneous failures between sources.

6.5. Equipment Serial Number

The serial number of the equipment. In the failure file this is the equipment that has failed. This must uniquely identify the equipment within the particular equipment type and company data set. This must also identify an instance of an equipment in the field. This is part of the component descriptor.
6.6. Company Design/Part Number of Board or Module

Manufacturers part number of board or module. This must uniquely define the type of board within the manufacturers data set. This is part of the component descriptor.

6.7. Board Serial Number

The serial number of a board. This must uniquely identify the actual board within the particular board type and company data set. This is part of the component descriptor.

6.8. Operating Time of Equipment to Failure

The time to failure since the equipment was up and running in hours. It contains an indicator as to how it has been calculated. Without this parameter it becomes impossible to do any of the more advanced analysis. If the database is assumed to be a constant failure rate database then there is no need to collect this information since number of failures in a time interval is all that is required.
What is actually meant by operating time to failure of an equipment for two separate failures is shown in Figure 11. The first failure occurs after time (A) and the second after time (B). The first time to failure for the equipment is time (A) and the second time to failure is time (B). As can be seen these failure times ignore the down times between the first failure and the equipment restart time.

6.9. Operating Time of Board (This Time Around)

Time since last failure on this board in hours. It should contain an indicator as to how it has been calculated. More details of what is meant by this field can be found in 6.14 under previous use.

6.10. Up and Running Date of Board in System

The date that the board was declared to be up and running in the system. It contains an indicator as to how it has been calculated. More details of what is meant by this field can be found in 6.14 under previous use.
6.11. Up and Running date of Equipment

The date that the equipment was declared to be up and running. It contains an indicator as to how it has been calculated. There are a number of ways of calculating this field, these are shown in Table 21.

<table>
<thead>
<tr>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>1) True, recorded up and running date</td>
</tr>
<tr>
<td>2) Calculated from delivery date</td>
</tr>
<tr>
<td>3) Calculated from first fail</td>
</tr>
<tr>
<td>4) Calculated from planned maintenance date</td>
</tr>
</tbody>
</table>

Table 21: Methods of Calculating Up and Running Date

These different methods are shown in Figure 12.

![Figure 12: Calculation of Up and Running Date for an Equipment](image)

The point marked up+running date is the actual up and running date, Time (a) is the time between delivery and up and running and could be estimated from similar equipment types. Time (b) is the time from the first planned maintenance and time (c) is the time from the first failure. These last two could also be estimated from similar equipments.
The database contains an indicator that informs the user how this field was calculated and so permits the data analysis procedures to use confidence criteria when calculating results.

6.12. End Date

The date of the end of observation. This is the last date that the equipment was known to be operating or it may be the date of equipment is recalled from service. It acts as the cut of date when calculating component hours for the non-failed population.

6.13. Usage

The percentage of possible usage time the equipment was operating. If an equipment is switched on 100% of the time then the usage is easy to calculate. However if two equipments are supplied and one is operated continually, and the other occasionally, say as a back up, It is difficult to estimate what the average usage of the equipment type is. It is often not possible to get usage information about any particular piece of equipment and so it becomes necessary to get an average usage for the equipment type. This also can prove problematical depending on the nature of the equipment being followed. It is likely that a end-user with a telephone exchange will tell you his average usage, but a customer with a military communications equipment is less likely to.

Usage is extremely important since further analysis is being carried out on the data stored and the results of further analysis could be swamped by large inaccuracies in the usage figure. Many pieces of equipment have elapsed time indicators (ETI) which can attempt to monitor the actual usage time. However these too have many problems and can in many cases only give a rough idea of the actual usage time. The problems of estimating usage using ETIs are given below.
1) Stuck at faults
This type of faults occur when the ETI gets stuck or jammed. In this case the ETI reading is very much less than it should be.

2) Wipe clean faults
This type of fault occurs when the ETI loses its reading and reports zero usage.

3) Delivery delay
This type of fault occurs when there is a delay in the delivery which means that the calculated ET is a constant value larger than the recorded ET. If this situation occurs it is easily fixed by subtracting a constant value.

4) Systematic underestimation
This type of fault occurs when the ET is systematically less than it should be. This situation is easily fixed by multiplying raising the estimated usage value.

5) High variability
This type of fault occurs when the ETI is faulty and reports greatly varying values.

6) Low early usage
This occurs when the usage of the equipment is lower than estimated at start-up but eventually settles at the usage estimate. This will cause errors if analysis is made during the early part of the equipment's life.

5) Race faults
This occurs when the relationship between the ETI and the actual operating time is non-linear.

Analysis performed on three of the data-sets stored within the database [26] showed that various combinations of the above faults were present. The effect of this variability of usage on failure rate is shown in Table 22.
Table 22: Error in failure rate prediction given an error in usage

Large though these possible error may be there is little choice but to accept them. A more in depth survey of the problems of usage is given in [26]

6.14. Previous Use

Previous use means the sum of all the previous use experienced by the board prior to the operating hours accrued by the board this time around. A previous use identifier added to the start of the previous use field to allow identification of the order in which the previous use occurred.

As an example and to better clarify the use of many if the time fields used in Figure 13 shows a simple failure map for a piece of equipment which contains one boards and has experienced three failure events. Board one, the initial board, has experienced two failures and the other replacement board one. After the initial board fails the replacement board is inserted until that too fails then the original board is reinserted. This example contains the dates (A),(B)..., failure 1, failure 2 etc. In a real system these would be replaced by actual dates.

The equipment is considered to be placed in the field at the equipment up and running date which is marked by the symbol (A) and a equipment identification file is provided which contains the dates in Table 23

Table 23: The Dates in the Equipment Identification File

The equipment up and running date : (A)
The end date : End of observation period

The equipment operates until the line marked ‘failure 1’. At this time a failure record is supplied which contains the information in Table 24
The up and running date of board in equipment: (B)
Previous use id: A
Previous use: 0000
The equipment operating time: Te1
The board operating time: Tb1a

Table 24: Dates in the failure record at first failure

The equipment is repaired between the times 'failure 1' and (C). However this time is considered to be zero. Then the equipment operates again until time 'failure 2' where a failure record containing the dates in Table 25 is supplied.

The up and running date of board in equipment: (C)
Previous use id: A
Previous use: 0000
The equipment operating time: Te1+Te2
The board operating time: Tb2a

Table 25: Dates in the Failure Record at Second Failure

The equipment is again repaired and the original board reinserted between the times 'failure 2' and (D). Then the equipment operates again until time 'failure 3' where a failure record containing the dates in Table 26 is supplied.

Figure 13: System map for failures showing dates and times
**Table 26: Dates in the Failure Record at Third Failure**

The equipment now operates to the end of observation time with no further failures.

**6.15. Storage**

The sum of the storage period of this board before use in days. It contains an indicator as to how it has been calculated.

**6.16. Manufacturing Date of Board**

The date the board was manufactured. It contains an indicator as to how it has been calculated. This information can be used to look for possible defects introduced into the equipments by the same batch of boards. In effect it gives a mechanism to examine manufacturing processes at the board level.

**6.17. Component Type/Generic Description**

A code describing the component type on a board. There are a large number of component codes available. The coding identifies the component type that exists in the board structure records and identifies the component that has failed in the failure records. The listing of the component codes are contained in appendix one. In general the codes split the components into grouping by production and manufacturing processes and hence possible failure mechanisms. For example the generic codings for ICs are based upon the technology type, Bipolar, FET, etc and the purpose of the device, digital, interface etc. Finally the devices are grouped by complexity. The IC were grouped according to the grouping used by the IC-MASTER [27], A discussion of how this was implemented for the database is described in [28] and a list of the actual IC codings for various IC part numbers is found in [29]
There is one special case of this component coding where the entry is not a component type. It can only occur in the failure table since it is an observed effect when a system fails. It is the coding for no fault founds. A no fault found (NFF) failure occurs when a failure is reported but on subsequent investigation there appeared to be no failure. They occur quite regularly in all sorts of equipment and it is not fully understood what is the main cause. There are a number of possible types;

1). Equipment level NFFs
These occur when the engineer is called out to a faulty equipment and finds no problems when it is tested. There is also the issue of user awareness to be considered. In the early days of use of a new system it is possible that the user is not familiar with the equipment and so will report faults when no such fault exists. This may be because the users expectations of a systems performance do not agree with the actual performance of that equipment.

2) Board level NFFs
This is probably the most common of NFF records. This happens when a board is removed and then replaced in the system and the system works. Many service engineers perform this exercise as a matter of course when called to repair a system. Another type of board level NFF occurs when the board is removed and subsequently tested and is found to work correctly. Reasons for this behaviour are more difficult to suggest. The failure may be due to poor electrical contract between a device and the circuit and the action of testing the board, with all that entails, could correct this kind of fault.

3) Component level NFFs.
These occur when a board is removed and tested and a faulty component is found. It is removed and later tested and it functions perfectly. This may be due to
poor connections between component and board. It could also occur if a system of multiple replacements is made and the non-faulty components are also analyzed.

The NFFs cause problems in this database because of the way in which the data is structured since all the failure information is stored at the component level. This means that if the NFF is at the board level then a large part of the information that is required for validation purposes is unavailable. When a NFF occurs at the equipment level then the amount of information conveyed by the failure record is minimal. This causes serious problems since it becomes impossible to say which order a set of NFFs occurred in since failure time information is conveyed at the component level. Hence equipment level NFF are not stored in the database, but are instead stored in a separate data file.

6.18. Component Part Number

This information is necessary if, after a component has failed, further physical investigation is required to investigate the actual cause of failure. In order to do this it may be necessary to obtain information from the component manufacturers. This fact is also important if wide generic grouping are used, eg. if all digital ICs are grouped together then it becomes impossible to do further breakdowns if no part numbers are known. The main cause of not knowing the part number is second sourcing. This means that the manufacturer gets components from more than one supplier. The parts are often all mixed together in the goods inwards section of a manufacturers plant and it becomes very difficult to tell the components apart if no part number is printed on the device itself. A further problem with NFF is when only an incomplete component part number is supplied. This often happens when the failure report forms filled in by the repair engineer only allows a certain amount of space for the failed part number or the company use shortened generic codes at all times. Some examples are included in Table 27.
The problem lies in not knowing whether the 4028 in the first example is actually a 4028B. There may of course be no difference in actual device function but often the operating parameters of the devices are different and this may be useful for engineering analysis.

6.19. Component Encapsulation/Mounting Details

Information relating to substrate type, mounting method and encapsulation type of component. This allows the investigation of the effect of mounting components using different techniques and the effect of differing packages and substrate types.

6.20. Screened Level of Component

The screened level of a component describes the type of qualification tests that a component has undergone. There are a number of problems that are involved with the collection of screened level. Firstly it is almost impossible to know exactly what is meant by the term 'screened level'. If it is assumed that screened level means a series of tests that a component has undergone and passed then it is inevitable that one company's testing system will be totally different to any others. Similarly the tests etc proposed by the various standard bodies or the military will be completely different to those proposed by any other organisation. This means that although in a 'common sense' way of looking at things a military specification component should have been screened to a higher level than a commercial component and so be more reliable it is quite possible that the commercial component has undergone a more stringent testing procedure than the military component. A second issue is the applicability of the types of screening may not reflect what actually happens to the component in the field. This may be because of a lack of imagination in the test design or because the components are being used in unthought of situations. A third problem concerns the logistics of collecting, it is almost impossible to know the
screened level of every component on a board particularly where there is second souring of components. A distinction should be made between the quality level of a component and the amount of screening a component has undergone. It is possible that a particular company may buy commercial quality components, components that are not BS approved for example, and the impose internal screening that raises the quality level of the components that pass the screening test to something above BS qualification. It is also possible that extra screening my induce an effect known as 'walking wounded'. This is where the stress applied by the screening tests exceeds the components strength and may cause it to be more unreliable than it would have been before screening.

6.21. Circuit Reference Position

The components position on the board. This must be unique for a particular board design. The circuit reference position is normally defined using some standard practise which is often company dependent. This is part of the component descriptor.

6.22. Component Manufacturer

Manufacturer of component. This field allows investigation into the different manufacturing processes used by the different component manufacturers.

6.23. Batch Number

Manufacturer’s batch number. This field allows the investigation of any batch related problems that arise with components.

6.24. Operating Condition of System Immediately Prior to Failure

Codes that identify what task the system was performing prior to its failure. This may be useful in analysing the different stresses exerted on a system by differing operating requirements.
6.25. Failure Mode of Equipment

Code describing the actual failure mode of equipment. Useful for the analysis of causality. This means that if a system only ever displays one particular failure mode when a particular component type fails then it becomes easier to identify the cause of failure. Similarly investigation of sequential failure modes may allow information about repair practises to be assessed.

6.26. Reported Failure Mode of Component

The actual failure mode of the component that has failed.

6.27. Number of Boards of a Particular Design Number

Number of boards of this particular design in equipment type. This is necessary in order to calculate the number of boards at risk at any particular time.

6.28. Primary Function of Board/Module

Primary function of this board. This field allows analysis of the failure behaviour of board designed form different purposes. It may give an insight into the design techniques used by the manufacturing companies.

6.29. Number of Components

Number of components of indicated type mounted on board of this type. Necessary to calculate the number of components at risk in the population at any time.

6.30. Environment

A description of the ambient conditions that the equipment is operating under. This must include an indication of the ambient temperature, the ambient humidity and
the presence of other hazards that might affect the equipment's reliability. Without this parameter we cannot perform any analysis on the effect of ambient conditions on the equipment. This means that no acceleration factors ($\pi$ factors) can be calculated for environment. In the IERI database the definition used with the MIL217 handbook[30] were adopted. Details of these can be found in appendix one.

6.31. Soft Information

This information is not stored in the database but in the soft information directory mentioned in section 5.3.1. It contains information about the data source, the equipment type, the date of tape delivery and any other relevant information. The results of the soft information questionnaire is also stored there. The soft information questionnaire is a series of questions that the company is required to fill in when submitting a new data source to the database. The types of question asked are described below

1) Equipment Type and Design

This information is required in order to understand the nature of equipment being followed in the field. This section should be completed by the equipment designers.

2) Maintenance policy and repair strategy

This information is required in order to understand the nature of the repairs the system under study is undergoing. This can help with the interpretation of failure records during validation and when performing board and system level analysis. This section should be completed by authority performing the maintenance and repair.

3) System construction

This information is required in order to understand the sort of use the system will be put to. This will help in the assessment of the types of failures that should be expected in differing system types. This section should be completed by the equipment designers.
4) Data collection mechanism
This information is required to understand how the failure records are made and stored. This should be completed by the repair authority.

5) General
This section contains general information unclassifiable in any other categories. This should be filled in by any relevant representative.

6) Database specific information
This section contains information about the coding assumptions made while converting the stored failure information into IERI format. This section should be completed by the authority performing the coding.

The actual questions used in the questionnaire are given in appendix two.

6.32. Failure Mechanism Reports

This is not a database field but a series of text files, hypertext documents for windows and printed reports that are associated with the database. These reports contain the results of literature surveys into the failure mechanisms that have been reported in the field for component types in the database. reports exist for the following component types, electrolytic capacitors [31], coil activated relays [32], connectors [33], wire-wound potentiometers [34], Piezoelectric resonator devices[35] and schottky barrier didoes [36]. Many others are under preparation.
7. DATABASE PROCEDURES

This section covers the procedures used to collect data from the company data source, create the database, load data into the database and correct any validation errors.

7.1. Raw Data Supply Mechanism

The companies that provide data to the database supply this data in most cases through the research centre of the company. This allows the data to be manipulated into the format specified by the data coding document included as appendix one. It also gives a central point of contact for the interchange of information which make the validation process, described in section 7.6 much easier to carry out. The transfer of data takes place on magnetic media and either tape or floppy discs may be used.

Five separate files need to be supplied. The formats of these files is discussed in appendix one. The file names must conform to the MSDOS naming pattern of 8 characters in the file name, a dot, and three letters in the extension. The following extensions given in Table 28 must be used.

<table>
<thead>
<tr>
<th>Extension</th>
<th>Contents of file</th>
</tr>
</thead>
<tbody>
<tr>
<td>.eqs</td>
<td>For the equipment structure file</td>
</tr>
<tr>
<td>.bds</td>
<td>For the board structure file</td>
</tr>
<tr>
<td>.eqi</td>
<td>For the equipment identification file</td>
</tr>
<tr>
<td>.fai</td>
<td>For the failure files</td>
</tr>
<tr>
<td>.sft</td>
<td>For the soft information</td>
</tr>
</tbody>
</table>

Table 28: Extension for raw data files

The file names chosen should reflect the nature of the data supplied and should be unique for that particular data source.
7.2. Database creation

1) Creation of the Main Database Tables

All database tables are created using a large number of SQL scripts that are stored in the DB/BIN directory. These scripts create the table space inside the Oracle DBMS and then create the actual base tables and some views on to these tables. These views are identical in format to the database tables but the data retrieved using them can be screened so that not all database users can see all the data. This was implemented to allow interactive logging on, a data supplying company that logged into the database interactively would be allowed to see its own raw data but not that of other companies and would have full access to the statistical and other information tables.

2) Creation of the Auxiliary Tables

After the main tables have been created then a large number (about 100) other tables and views are created. These tables contain summary information that is derived from the main database and auxiliary information that can aid in the decoding of the many coded fields. These summary tables are used to assist in analysis since this can speed up the analysis so that it can be completed in a realistic time. After the auxiliary tables have been created a large number of database administration tables are created. These tables contain information about the processes that are used to administrate the database. They also are used to document the other database tables so that database table catalogues that contain useful information can be created.

7.3. The Data Format

The data coding document describes the format of four data files. These data files are each split into a number of lines or records. Each record consists of a number of fields which are also defined in the data coding document. These fields are semicolon delimited. The structure of a typical record from each field in shown in Table 29, Table 30, Table 31 and Table 32.
The equipment identification file, equipment structure file and board structure file are known as the population information and a data set consists of the combination of the population and failure information for a particular data source.

7.4. Data Anomalies and Data Validation

There are of course a large number of problems that can be encountered with relational databases and these have to be considered before any design decisions are made. Some of the more critical issues are discussed here.

There are four well known forms of anomaly that occur within a relational database that has been poorly designed. These are redundancy, deletion, update and join anomalies. Redundancy means that the database contains information that is stored in too many places. This indicates that the database is not in the most space efficient forms. Deletion anomalies occur when some important information is lost because of the deletion of some other information that is not required. This most often occurs when the database has been over optimised and, for instance, the data is only stored in a temporary table which gets removed. Update anomalies occur when small differences in the data cause multiple retrieval of data. The join anomaly is the most troublesome of the various anomalies. This occurs when doing multiple table searches and a record in one table matches multiple records in another table because of a
poorly specified database key. This last can cause selection of data that is not required and if the user is not aware that it is happening it will badly mislead them.

All these anomalies can be overcome with very careful database design. However they can reappear, particularly join anomalies, if poorly validated data is allowed into the database. The fundamental basis for validation is some idea or expression of what the data ought to look like, or not look like. This process of validation cannot be performed without having some validation criteria. The validation process cannot produce error-free data. Sometimes it is not possible to express or even to know the validation checks that should be performed. More often, what should be checked is known but this cannot be done in a sensible way due either to limitations in the database system or because of uncertainties in the data stored.

Data validation can be performed external to the system by people or within the system based upon formally stated validation criteria. This validation may be performed on input, stored or output data. The input validation should check that the incoming data conforms to the database definition and that interrelated fields are interrelated as expected. Validation of stored data help to keep the database consistent and output validation ensures that the output is reasonable.

This means that as well as implementing a database structure a validation scheme was also devised. This validation scheme depends on the structure of the data being collected and on the coding system used to convert data into a fixed format that is easy to deal with. The validation system is covered in 7.6 and the coding system can be seen in appendix one.
7.5. Data validation

There are a number of basic reasons, shown in Table 33 for performing a data validation procedure before loading data into a database.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1)</td>
<td>To insure the data stored conforms to the data coding standard produced by the technical design committee.</td>
</tr>
<tr>
<td>2)</td>
<td>To ensure the data supplied conforms to the requirements of the database</td>
</tr>
<tr>
<td>3)</td>
<td>To ensure the data is reasonable and valid data</td>
</tr>
</tbody>
</table>

**Table 33: Reasons for performing validation**

To meet these requirements, data validation has to be performed. This validation performs two essential tasks, firstly it checks the data format and secondly the data internal consistency. These are discussed in detail in these sections. This validation is in addition to any validation performed before the data is supplied to IERI. To aid the companies perform this in house validation IERI provided a MS-DOS based program called 'validate' which can perform first stage validation for the IERI database. It can also perform first stage validation for the RACE-DIRAC database.

7.5.1. Principles of the Validation of Incoming Data

In order to get the incoming data into the IERI database it has to pass through two logical stage which can be termed first and second stage validation. The first stage is concerned with ensuring that the data meets the format laid down in the data transfer document that was discussed earlier. The second stage validation is concerned with the data consistency issues.

7.5.2. Principles of First Stage Validation

The first stage of the validation process can be though of as the check that makes sure the data supplied meets the specification for the data given in the data transfer document. The sort of error that need to be looked for are listed below.
1) **Data framing errors**

If a data record is defined to be a fixed length then it is important that every record in the data set conforms. Each field within the record will be of a defined length. If there is a character omission or addition to the record then it is not worth performing any further validation since there will be a large number of data errors due to the overlap of the record fields with those fields defined by the template given in the data transfer document. These errors are termed 'data frame errors' It is not enough to just check that the records are the correct length for it is possible that two errors may occur and cancel each other out. The only way around this cancelling problem is to check the length of each field that makes up the record and this of course can only be done if there are delimiters defined by the record template. These delimiters must not appear anywhere within the record elements themselves.

2) **Incorrect format**

It is also necessary for each field within the record to be of a specific type. This prevents letters getting into a numeric field and vice versa. Sometimes it is necessary to create combined fields which contain letters and numbers. It is generally true that every field in a database record can be defined as either numeric, alphabetic or mixed. Each position in each of these fields will of course be meaningful. Hence it is vitally important that only correct data types get into the correct data positions. There are generally three field types, these can be termed.

a) **Free format fields**

These can, as their name suggests, take any combination of characters and numerals.

b) **One of many fields**

These fields contain one of a number of choices. These choices are predefined and only previously agreed choices can be present in this field

c) **Structured fields**

These fields contain data in a structured fashion such as a date. It is very important that any data carried in a structured field conforms to the structure
definition. In the case of a date the numerals give must agree with the chosen data format and they must represent a real date.

The IERI database is made up of a combination of these types. In order to be loaded into a database the supplied data must meet all the specifications laid down for the field type given above. It is generally possible during the validation process to find errors, such as miss-keying in the last two field types mentioned above but it is not always possible in a free format field. This is because the miss-keyed character may be a valid character for the position in the field that it takes and it is impossible to know, at the receiving end of the data, what was meant by the typist.

7.5.3. Principles of Second Stage Validation

The second stage of the validation process where interrelationships between the various fields and records are checked.

1) Relations within a single record.

There are many instances where the fields of a single record are related. This relationship need not be direct, indirect relations are also possible. An example of a direct relation is where one field is a calculation based on the contents of another, or many other fields within the record. This is data redundancy and should be avoided if at all possible. However in some instance when trading storage efficiency for speed of operation or clarity makes it necessary to have such compromises. If these sort of relationships are present within the data then it is necessary to check them during this stage of the validation process.

An indirectly related field is one where for example a date of failure has to be after the date of commissioning of a particular equipment. It is important to check that this relationship is true. The IERI database because of its nature has a number of these sort of indirect relationships.
2) Relations between records.

If data is collected throughout a time period then it is quite likely that information about any particular system will occur many times. It is important to check that if this is the case the system information remains the same for each occurrence. For example, if a system fails five times, then in order for the data to be valid data, information about all five failures must be present and they must appear to the database to occur in the correct order. This last will require checking dates of failure as well as failure sequence information. The data to be checked will exist in a large number of record and some may have already been loaded into the database as part of an earlier dataset. While performing this sort of validation it is sometimes necessary to fail a large chain of failures because one of the entries in error.

3) Relations between data files.

The IERI database is constructed as two distinct data types, the failure information and information on the population at risk, or population information. When information about a failure is received it becomes necessary to check that the failed actually occurs on the board that is given in the failure record, that particular board serial number is out in the field and that the failure occurred between the startup dates and end dates for the particular system serial number, ie we must check that the component descriptor is valid for the particular failure time. This requires a large amount of checking between the population and failure tables within the database.

4) Relations between data sets

It is necessary to ensure that there is some consistency between data sets, for example where components are coded. It is important to ensure that different data source code the same component part number in the same manner. Within the IERI database this can only be done where the component part number is known, ie for transistor, IC and hybrids.
7.6. The validation used at IERI

The validation process used at IERI is shown in Figure 14. Data is provided as described in 7.1. This data is loaded onto the HPC and stored in the directory structure described in 5.3.1. The validation programs are a small suite of programs each of which has a distinct purpose. These programs are listed in Table 34.

<table>
<thead>
<tr>
<th>Program Name</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validate</td>
<td>The validation suite driving program, it calls all other programs in the correct order</td>
</tr>
<tr>
<td>Chkdata</td>
<td>First stage validation program</td>
</tr>
<tr>
<td>Loadtemp</td>
<td>Loads the temporary database</td>
</tr>
<tr>
<td>Chkconst</td>
<td>Second stage validation program</td>
</tr>
<tr>
<td>Loadbase</td>
<td>Database loader</td>
</tr>
</tbody>
</table>

Table 34: Program Names and Purposes

The driving program is 'Validate' which supplies the correct parameters to the other programs. When this program is started it ensures that the support files (such as error messages) are up to date and then it calls the first program of the suite. This is called 'ChkData' and it is described in section 7.6.1. This program writes a file containing the data lines that have passed the first stage validation, and a second file that contains the lines that contain errors as well as information about the type of error.

Validate then calls the second program in the suite, called 'Loadtemp', which loads the data in the pass file into a duplicate copy of the database. Again loadtemp writes a file of data that has not been loaded. Validate now calls 'Chkconst' which is described in section 7.6.2. Chkconst acts on the existing main database and the newly loaded data to perform its validation task and then exits leaving a file of data that failed the tests. Validate then calls the database loader which copies the data that has passed the validation and is still in the temporary database table into the main database tables. Validate then calls 'Errfmt' which collects the failed data files and produces a validation report. Finally validate calls the system 'nroff' program to

80
Figure 14: The validation process

produce the actual ASCII output file.

Validate has to ensure that all data files are loaded to the database in the correct order. First the board structure file is loaded, then the equipment structure and equipment identification and finally the failure file. If this order is changed then the second stage validation program will be unable to resolve references to the structure of equipments and will fail the data. Only one validation session can be in progress at any one time but the process is controllable so that any particular data file can be passed through any particular stage. This allows an iterative approach to data validation since there is little point in attempting to second stage validate an dataset that is incomplete due to errors at stage one.

Validate also ensures that no database operations can take place while loading is under way. It does this by setting a flag in one of the database administration tables which is queried by the database logon function provided in the libCORD library and enforced as part of the common programming interface described in section 5.4.5. If users are logged in validate waits until they log out and then prevents them logging in again. Of course this can be circumvented by logging on in an unapproved way but this is discouraged. This is to prevent the database changing while analysis is under way which could dramatically change the analysis results.
Each phase of the validation process is described in more detail below.

### 7.6.1. First stage validation

First stage validation can be thought of as the check that makes sure the data supplied meets the specification for the data given in the data coding document. A flowchart for the tests performed is shown in Figure 15.

![Flow chart for first stage validation](image)

**Figure 15:** Flow chart for first stage validation

The program checks the input file for invalid characters, it checks that field lengths are correct, that one-of-many fields have a valid entry, that dates are real dates and that the fields necessary of the operation of the database and other essential fields are present. These tests are dependent on the format of the input file and are listed in detail in appendix one.

The 'Chkdata' program also does data translation in special circumstances, if a field has a single '?' to denote that the field is not known then this field is filled with '?'. This is done so that the program that loads the temporary database always deals with fields of the exact width specified and by matching a '?' filled field can load a special code into the temporary database to define a database field as being empty. This code is known as the 'NULL' code. The program also checks that the format of the no fault found (NFF) records is correct and removes the equipment level
NFFs from the data by writing them to a separate file. If a field is a NFF and it is entered incorrectly the program will ensure that the correct format is supplied. Details of this format can be found in appendix one.

### 7.6.2. Second stage validation

The second stage of validation is the place where interrelationships between the various fields and records are checked. The flow chart of this process is shown in Figure 16.

![Flow chart for second stage validation](image)

**Figure 16:** Flow chart for second stage validation

The first part of second stage validation is to load the temporary database. This is done by a program known as loadtemp. This program will only load data that will be uniquely identified when within the database. This means that at this stage duplicate records within the input file are removed. The next stage of validation is carried out by the program 'chkconst'. This program first checks that the database descriptor described in section 3 is correct and ensures that each link exists. For example if a failure file is being tested then the program first checks that an entry in the board structure file exists for the component type, then it checks the equipment structure file to ensure that the board is part of an equipment and finally it checks the equipment identification file to check that the equipment is in the field. While doing this it
checks that the time to failure and other times and dates are valid for that descriptor. After checking the descriptor is valid and that the dates and time are correct the next stage is to check the relationships between the current record under scrutiny and other records. For example, if this is a failure record it checks that the component type and the circuit reference position are the same in this case as any previously reported failure with the same descriptor, it checks that the component coding is the same for the same component part number across all data sets and that if this is a simultaneous or sequential failure chain then all other members of that chain exist. A further feature of this cross-checking is that is possible in the case of IC to insert the correct component code at this stage. This allows a data source to provide minimum component coding information as long as they supply the part number and have supplied details of this part number and component code previously. Any coding done by the program is mentioned in the validation report. The IC codings are all stored in a separate data table and have proved valuable for other tasks. The table is automatically updated by the chkconst program if a new IC coding and part number are found.

In order to do this consistency checking the program first looks in the datafiles provided with the currently validating data set within the temporary database, if the links are not resolved here it checks in the main database table before flagging an error. When an error is found then the field in the temporary table is flagged and as the program finishes is copied to an error file.

7.6.3. Error reporting and database loading.

After both validation process have completed, the valid data is loaded into the database by a program called 'LoadBase'. This program searches the duplicate, temporary database structure and copies the unflagged lines into the database proper. It then deletes the data in the temporary table. If an error occurs during the copying process then the line in error is copied into a failure file.
The last IERI written program called by 'validate' is called 'ErrFmt'. ErrFmt takes the failure files generated by the last three programs, if any, and produces a nroff script file that contains information about any failures seen by the previous programs. These errors are grouped into four sections, one for each processing program, and each page contains the errors discovered for that validation stage for a particular record. The error messages are in English and pinpoint the actual character or at least field in error wherever possible. Sometimes a perfectly good record is rejected because another record in the failure sequence of sequential chain was in error but this is flagged as such where possible and a message is printed. Before exiting the errfmt program updates a number of database administration tables which allow the validation history of any particular dataset to be found. This tracking information is written to the front of every report as is any other information contained in the 'soft' information provided with the data set. The versions of the programs to perform the validation are also printed. The flow chart is shown Figure 17

![Flow chart](image)

**Figure 17: Report writing and database loading**

If errors are found then the company supplying the data then resubmits the data and it undergoes the validation loop once more.
7.6.4. The Change Loop

If the number of error is small or the errors obvious to IERI then the changes can be made by IERI staff. If this is done then a change report is generated and returned to the company so that they can update their records. This change loop is implemented by the 'change' program. In order to make a change to the data the database administrator has to edit the data, this can be done in two ways, the actual original data file can be edited or the temporary database can be edited. Where the editing is done will depend on the sort of change that has to be made. If the editing is done in the original file, first a backup file is created by the change program then the change program loads the file into the memacs editor and the user can then make changes. On leaving the editor the change program compares the saved file with the original backup and produces a report listing the changes made on a record by record basis. If the changes are made in the database then the change program takes a backup dump of the database, then it loads an SQL*form program that allows the tables to be edited. After the SQL*form program exits the change program takes another dump of the database. Then it compares the two dumps and again produces the report. While producing the report the program also updates a administration table that keeps track of the number of changes, who made them etc. The flow chart for the change loop is shown in Figure 18.
Eventually the all the data supplied by the company will pass through the validation loop.

7.7. The administration of the validation

As already has been mentioned all the database activity is logged in a number of ways. While the programs are running a audit trail of the process is placed in a journal file. This journal file can be examined for information should any non-obvious problems occur.

All validation and change reports as well as the data sets are stored under SCCS control and are backed up onto disk and tape within IERI. All the reports issued contain the versions of the programs that were used to perform the validation. This allows the tracing of any problems introduced by updating and the like. Finally the error report writer and the change program makes entries in auxiliary tables every time they are run. This allows the history of the validation process of any data set to be produced. Each error or change report has a unique number which is traceable in this database. A number of programs exist that allow IERI staff to access the
administration tables to find out the current status of a data set. Reports, which can be sent back to companies can also be produced.
8. DATABASE DATA ANALYSIS

This section shows many of the forms of analysis that can be performed using the IERI database. This section is not exhaustive as many other avenues of investigation are possible.

8.1. Database totals

These statistics give the overall size of the database in terms of real electronic items. The current statistics are shown in Table 35

<table>
<thead>
<tr>
<th>Description of items</th>
<th>Number of items</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Equipments</strong></td>
<td></td>
</tr>
<tr>
<td>Equipment Types</td>
<td>414</td>
</tr>
<tr>
<td>Total Number of Equipments in the Field</td>
<td>35,506</td>
</tr>
<tr>
<td><strong>Subsystems or circuit boards</strong></td>
<td></td>
</tr>
<tr>
<td>Board Types</td>
<td>1,944</td>
</tr>
<tr>
<td>Total Number of Boards in the Field</td>
<td>413,421</td>
</tr>
<tr>
<td><strong>Components</strong></td>
<td></td>
</tr>
<tr>
<td>Component Types</td>
<td>175</td>
</tr>
<tr>
<td>Total Number of Components in the Field</td>
<td>30x10⁴</td>
</tr>
<tr>
<td>Total Number of Component Hours accrued</td>
<td>221x10⁸</td>
</tr>
<tr>
<td>Total Number of Component Failures</td>
<td>4762</td>
</tr>
<tr>
<td>Total Number of No Fault Founds Reported</td>
<td>2955</td>
</tr>
</tbody>
</table>

Table 35: Current Database Size (1/1/94)

This table highlights the fact that the database is an equipment or system database as well as a component one.
8.2. Raw data extraction.

Raw data can be extracted using the SQL interpreter part of the Oracle package. This allows interactive use of the database for ad-hoc enquiries. An example SQL query which extracts data in a simple form from the database is shown here. Population information is obtained by executing a query like that in Table 36.

```
Set pause off;
title center 'Population Information Quartz Crystals';
break on report;
compute sum of 'Total comps' on report;
select bds.comp_type "Type",
    bds.comp_part_no "Part no",
    bds.comp_add_info "Add. Info",
    bds.co "Company",
    sum (bds.comp_no) * sum(eqs.co_no) "Total comps"
from cord.bds bds, cord.eqs eqs
where bds.comp_type like 'QI%'
and bds.co = eqs.co
and bds.co_design = eqs.co_design
group by bds.comp_type, bds.comp_part_no, bds.comp_add_info, bds.co;
```

**Table 36: SQL query to obtain population information**

This gives an output as shown in Table 37.

<table>
<thead>
<tr>
<th>Type</th>
<th>Part no</th>
<th>Add.</th>
<th>Co</th>
<th>Total comps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1A</td>
<td>&quot;</td>
<td>15</td>
<td>AA</td>
<td>121</td>
</tr>
<tr>
<td>Q1A</td>
<td>&quot;</td>
<td>16</td>
<td>AA</td>
<td>3484</td>
</tr>
<tr>
<td>Q1A</td>
<td>&quot;</td>
<td>16</td>
<td>AC</td>
<td>9</td>
</tr>
<tr>
<td>Q1A</td>
<td>&quot;</td>
<td>2A16</td>
<td>QX</td>
<td>10</td>
</tr>
<tr>
<td>Q1A</td>
<td>&quot;</td>
<td>3A13</td>
<td>QY</td>
<td>4</td>
</tr>
<tr>
<td>Q1A</td>
<td>&quot;</td>
<td>3AP5</td>
<td>HB</td>
<td>8</td>
</tr>
<tr>
<td>Q1A</td>
<td>HC18/U</td>
<td>3AP5</td>
<td>HB</td>
<td>8</td>
</tr>
<tr>
<td>Q1A</td>
<td>HC33/U</td>
<td>3AP5</td>
<td>HB</td>
<td>4</td>
</tr>
<tr>
<td>Q1B</td>
<td>&quot;</td>
<td>1A</td>
<td>AA</td>
<td>39</td>
</tr>
<tr>
<td>Q1B</td>
<td>IQXO-200</td>
<td>16</td>
<td>AC</td>
<td>1</td>
</tr>
</tbody>
</table>

**Table 37: Population information**

To obtain failure information a query like that given in Table 38 would be used.
Set pause off;
title centre 'Failure data Quartz Crystals';
select fail.comp_type "Type",
fail.comp_part_no "Part no",
fail.comp_add_info "Add.",
fail.co "Co",
fail.screen_level "Sc",
fail.environment "En.",
fail.manufacturer "Ma.",
count(fail.co) "Fails"
from cord.fail fail
where fail.comp_type like "QI%"
group by fail.comp_type, fail.comp_part_no, fail.comp_add_info, fail.co ;

table 38: SQL query to obtain failure information

This gives an output similar to that shown in Table 39

<table>
<thead>
<tr>
<th>Type</th>
<th>Part No.</th>
<th>Add</th>
<th>Co</th>
<th>Sc</th>
<th>En</th>
<th>Ma</th>
<th>Fails</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1A</td>
<td>A</td>
<td>1</td>
<td>AA</td>
<td>8</td>
<td>03</td>
<td>&quot;</td>
<td>1</td>
</tr>
<tr>
<td>Q1A</td>
<td>A</td>
<td>2</td>
<td>A6</td>
<td>QX</td>
<td>XMI</td>
<td>01</td>
<td>&quot;</td>
</tr>
<tr>
<td>Q1A</td>
<td>A</td>
<td>2</td>
<td>A6</td>
<td>QX</td>
<td>XMI</td>
<td>01</td>
<td>CC</td>
</tr>
<tr>
<td>Q1A</td>
<td>A</td>
<td>3</td>
<td>A3</td>
<td>QY</td>
<td>8</td>
<td>03</td>
<td>&quot;</td>
</tr>
</tbody>
</table>

| Total | 47 |

Table 39: Result of population query

In general after a query similar to the one above has be created and tested
the SQL will be embedded into a C program to allow more control over the input and
output, for example to write the output to a file directly rather than to the screen.
Alternatively it could be moved into a SQL*ReportWriter report.

8.3. Structural Analysis

The method of analysis described in this section examine the structure and
usage of equipment in the field as well as highlighting the sort of data stored in the
database.

1) Breakdown of equipment environments

This examines the relative numbers of equipments in each environment defined
in the database. These environments are based on the MIL-217 definitions [30] and
details can be found in appendix one. Figure 19 shows the relative number of
equipments in the different environment presently tracked by the IERI database. The codes used for the environment are shown in Table 40.

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Environment Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>GB</td>
<td>Ground Benign</td>
</tr>
<tr>
<td>GM</td>
<td>Ground Mobile</td>
</tr>
<tr>
<td>NS</td>
<td>Naval Sheltered</td>
</tr>
<tr>
<td>AIC</td>
<td>Airborne Inhabited Cargo</td>
</tr>
<tr>
<td>AUT</td>
<td>Airborne Uninhabited Trainer</td>
</tr>
</tbody>
</table>

Figure 19: Breakdown of the relative number of equipments in each environment

Figure 19 shows that most of the data obtained from the database will be for a ground benign environment and this should be taken into account when pooled data, i.e., data derived from all the information in the database, is used.

2) Equipment Complexity

This looks at the relative complexity of equipments in terms of the number of boards that they contain. The more boards that are used in a single instance of an equipment the more complex the equipment is considered to be. If the complexity effects the reliability of a system then there may be an optimum size of an equipment
design, too small and the equipment may not include the required functionality, too large and the equipment may be unreliable.

The sizes of equipments in the field in terms of the number of boards they contain is summarised in Figure 20.

![Figure 20: Breakdown of number of boards per equipment type.](image)

As can be seen from Figure 20 most systems contain between 10 and 100 boards and that 86% of systems have less than 500 boards.

Further investigation of how this complexity affects the time to first failure is shown in Figure 21. This figure shows that there is no relationship between the time to first failure and the complexity of the equipments when measured by the number of boards within the equipment. Since boards can vary in complexity a better measure of equipment complexity may be the number of components present in the system.
Figure 21: Time to first failure versus complexity of equipments

Figure 22 shows the relative numbers of equipments with differing component counts. As can be seen almost 78% of equipments have less than 1000 components while only 0.5% have more than 10,000. Figure 23 shows the time to first failure for equipments versus the number of component that they consist of. As can be seen from the graph there appears to be no relationship between the time to first failure and complexity.

Figure 22: Breakdown of number of components per equipment type.
3) Component analysis

Component analysis involves the breakdown of the contents of the database into generic types and then summing the numbers of each type. The grouping into generic types means that resistors and capacitors are grouped as passive devices, diodes and transistors are grouped as discrete and then all devices except connectors, ICs and hybrids are grouped as others.

a) Failures

Figure 24 shows this breakdown for the number of failures in the database. It should be noted that the no fault found records are included. As can be seen the NFF records account for almost 40% of the total number of reported failures for systems. This trend is also repeated throughout the data sets supplied by different sources and so is a global problem. This is further discussed in 8.6.4. Apart from the no fault founds, integrated circuits have the greatest number of failures.
b) Component Hours

Figure 25 shows the relative numbers of component hours per device type. As can be seen the passive devices account for 47.5% of the components in the field. This means that on average almost 50% of a piece of equipment, in component terms, is made up of passive devices. This makes it vitally important that the reliability of these devices is correct.
8.4. Time Independent Failure Statistics

This describes the simple failure statistics that are independent of time, i.e., constant failure rate statistics the mathematics of which is described in [37]. Failure rate is calculated using the equation in (4) and is expressed in FITS (FIT stands for Failure unITS).

\[
\lambda = \frac{n}{N.t}
\]  

(4)

Where
\[
\lambda = \text{failure rate} \\
N = \text{Number of failures} \\
N.t = \text{Number of component hours}.
\]

Failure rate is an often used measure to compare the likelihood of failure of two differing components. It is felt however that too many assumptions are made when failure rates are calculated. There is little evidence to suggest that the time dependent behaviour of components meets the exponential failure model [37] that is required for the failure rate to be meaningful.

When failure rates are calculated it is useful to calculate some confidence limits that can give some ideas of how good the estimate of failure is. The IERI project uses 95% \( \chi^2 \) confidence limits which can be calculated from (5)

\[
\text{Limit} = \frac{\lambda \chi^2[\frac{(1-y)}{2}:2N]}{2N}
\]  

(5)

Where
\[
N = \text{number of failures observed}, \\
\lambda = \text{failure rate}, \\
y = \text{confidence level}.
\]

The \( \chi^2 \) part of the equation is looked up in tables or can be estimated using (6)

\[
\nu[1 - \frac{2}{9\nu} + \frac{X}{\sigma} \sqrt{\frac{2}{9\nu}}^3]
\]  

(6)
where \( \frac{X}{\sigma} \) is the normal deviate for the confidence level and \( \nu \) is the degree of freedom (equivalent to the number of failures). Either method yields two confidence limits.

In order to compare failure rates a failure rate hierarchy is calculated. Table 43 lists all the components in the database where the failure rate is greater than 100 FITS. The columns upper and lower are the upper and lower 95\% \( \chi^2 \) confidence limits respectively as defined in (5). The rating system [A to F] in the last column is based upon the ratio of the upper confidence limit to the mean of the \( \chi^2 \) distribution. By ignoring the non-symmetrical aspect of the \( \chi^2 \) distribution a statistic \( R \) can be computed

\[
R = \frac{\lambda_u}{\lambda}
\]

where \( \lambda_u \) is the upper confidence limit and \( \lambda \) is the actual failure rate.

This gives Table 41

<table>
<thead>
<tr>
<th>Rating</th>
<th>Value of ratio ( R ) from equation (6)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1.0 to 1.1</td>
</tr>
<tr>
<td>B</td>
<td>1.1 to 1.2</td>
</tr>
<tr>
<td>C</td>
<td>1.2 to 1.3</td>
</tr>
<tr>
<td>D</td>
<td>1.3 to 1.4</td>
</tr>
<tr>
<td>E</td>
<td>1.4 to 1.5</td>
</tr>
<tr>
<td>F</td>
<td>1.5 to 1.6</td>
</tr>
</tbody>
</table>

**Table 41: Value of ratio \( R \) from equation (6)**

Given the nature of the calculation of the ratio, translation to the required number of failures can be performed and is shown in Table 42.
Table 42: The IERI rating system

This rating system is only intended as a crude estimate of the accuracy of the failure rate figures. Recent work within the project [38] has suggested methods for improving this rating system however irrespective of rating system a much better estimate is given by the confidence limits.

The data in Table 43 shows that some of the components with highest fit ratings are based on a small amount of information. However in some cases there is enough information available to be fairly accurate about the expected failure rate of a device. Examples of this are the Digital MOS IC with $10^4$ Gates and the Digital MOS IC with $10^3$ gates.

The detailed nature of the data collected allows various different failure rate comparison to be made. Table 44 shows a comparison of the data available on surface mounted components versus other forms of mounting. The first three entries are where population and failure information for a device type is available for surface mounted versions of the device and the others are where only population information is available because there have been no failures in the surface mounted version as yet. Where failure information is available it noted that the failure rate for the surface mounted components is higher than that of other mountings. This may well be that these devices are unsuitable for surface mounting since other devices types that are surface mounted have accrued a fair amount of component hours without failure. It is also possible that the different manufacturing and production techniques used with surface mounted components are causing the devices to be unreliable.
Table 45 shows a comparison of the data available for plastic packages versus other packages for integrated circuits. As can be seen from the table the failure rate of plastic encapsulated IC is about 0.5 of a fit higher.

Table 46 shows a comparison of the data available for some components under two different environments. As can be seen from the table the failure rate of the ceramic capacitor and the pn-junction diode is higher in the ground mobile (GM) environment than in the ground benign (GB). This is as expected since the GM environment is a harsher environment than GB. However for the integrated circuit the reverse is seen to be the case. There are a number of reasons why this is happening. It should be noted that all the GM equipments are commercially screened equipments while all the GM equipments are military screened equipments. Hence the screening of the component types is having an effect as well as the environment. If the screening of the IC in the GM environment is sufficient that the component can withstand the effects of the environment then the failure rate will be vastly reduced. This is probably the case here. If a commercially screened component was placed in this GM environment then the failure rate would be, as expected, higher than that obtained by the same component in the GB environment. This effect highlights the problems that can occur when failure rates are calculated for components. It is absolutely essential before failure rates can be used effectively to understand exactly what the failure rate is based upon. Misuse of failure rates is one of the greatest problems in reliability engineering at the present time.

Many other types of comparison, between screening methods for example, are possible given the detailed nature of the data in the database.
<table>
<thead>
<tr>
<th>Component type</th>
<th>Falls</th>
<th>FIT</th>
<th>Upper</th>
<th>Lower</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microwave cavity</td>
<td>10</td>
<td>1.96e+05</td>
<td>3.35e+05</td>
<td>9.4e+04</td>
<td>F</td>
</tr>
<tr>
<td>Wire-wound pot.</td>
<td>21</td>
<td>5.06e+03</td>
<td>7.45e+03</td>
<td>3.13e+03</td>
<td>E</td>
</tr>
<tr>
<td>Microwave switch</td>
<td>6</td>
<td>2.07e+03</td>
<td>4.03e+03</td>
<td>760</td>
<td>F</td>
</tr>
<tr>
<td>Metal film pot.</td>
<td>4</td>
<td>1.12e+03</td>
<td>2.46e+03</td>
<td>305</td>
<td>F</td>
</tr>
<tr>
<td>Memory MOS IC (H)</td>
<td>10</td>
<td>853</td>
<td>1.46e+03</td>
<td>409</td>
<td>F</td>
</tr>
<tr>
<td>Digital MOS IC (4)</td>
<td>181</td>
<td>799</td>
<td>920</td>
<td>687</td>
<td>B</td>
</tr>
<tr>
<td>Thyristor</td>
<td>2</td>
<td>435</td>
<td>1.21e+03</td>
<td>52.6</td>
<td>F</td>
</tr>
<tr>
<td>Lumped delay line</td>
<td>5</td>
<td>434</td>
<td>889</td>
<td>141</td>
<td>F</td>
</tr>
<tr>
<td>Microprocessor (3)</td>
<td>70</td>
<td>410</td>
<td>512</td>
<td>320</td>
<td>C</td>
</tr>
<tr>
<td>Push button switch</td>
<td>4</td>
<td>305</td>
<td>669</td>
<td>83.2</td>
<td>F</td>
</tr>
<tr>
<td>Loudspeaker</td>
<td>33</td>
<td>301</td>
<td>413</td>
<td>207</td>
<td>D</td>
</tr>
<tr>
<td>Keyboard switch</td>
<td>32</td>
<td>292</td>
<td>402</td>
<td>200</td>
<td>D</td>
</tr>
<tr>
<td>Schottky barrier diode</td>
<td>19</td>
<td>255</td>
<td>382</td>
<td>154</td>
<td>E</td>
</tr>
<tr>
<td>Digital MOS IC (3)</td>
<td>91</td>
<td>249</td>
<td>303</td>
<td>201</td>
<td>C</td>
</tr>
<tr>
<td>E.T.I</td>
<td>1</td>
<td>243</td>
<td>896</td>
<td>6.14</td>
<td>F</td>
</tr>
<tr>
<td>Pulse transformer</td>
<td>5</td>
<td>241</td>
<td>493</td>
<td>78.2</td>
<td>F</td>
</tr>
<tr>
<td>Memory MOS IC (G)</td>
<td>3</td>
<td>240</td>
<td>578</td>
<td>49.5</td>
<td>F</td>
</tr>
<tr>
<td>Linear JFET IC (2)</td>
<td>11</td>
<td>224</td>
<td>374</td>
<td>112</td>
<td>F</td>
</tr>
<tr>
<td>Microprocessor (4)</td>
<td>24</td>
<td>215</td>
<td>310</td>
<td>138</td>
<td>E</td>
</tr>
<tr>
<td>Digital Bipolar IC (3)</td>
<td>6</td>
<td>211</td>
<td>411</td>
<td>77.5</td>
<td>F</td>
</tr>
<tr>
<td>Thermostat</td>
<td>3</td>
<td>204</td>
<td>491</td>
<td>42.1</td>
<td>F</td>
</tr>
<tr>
<td>Hybrid</td>
<td>284</td>
<td>193</td>
<td>216</td>
<td>171</td>
<td>B</td>
</tr>
<tr>
<td>Crystal oscillator. with logic</td>
<td>1</td>
<td>181</td>
<td>666</td>
<td>4.57</td>
<td>F</td>
</tr>
<tr>
<td>Interface Bipolar IC (4)</td>
<td>63</td>
<td>176</td>
<td>222</td>
<td>135</td>
<td>C</td>
</tr>
<tr>
<td>Fan</td>
<td>1</td>
<td>140</td>
<td>518</td>
<td>3.55</td>
<td>F</td>
</tr>
<tr>
<td>Memory MOS IC (A)</td>
<td>3</td>
<td>134</td>
<td>323</td>
<td>27.7</td>
<td>F</td>
</tr>
<tr>
<td>Circuit breaker</td>
<td>2</td>
<td>131</td>
<td>364</td>
<td>15.8</td>
<td>F</td>
</tr>
<tr>
<td>Rotary switch</td>
<td>9</td>
<td>123</td>
<td>215</td>
<td>56.1</td>
<td>F</td>
</tr>
<tr>
<td>Interface MOS IC (2)</td>
<td>9</td>
<td>121</td>
<td>211</td>
<td>55.2</td>
<td>F</td>
</tr>
<tr>
<td>Memory MOS IC (C)</td>
<td>53</td>
<td>120</td>
<td>154</td>
<td>89.6</td>
<td>C</td>
</tr>
<tr>
<td>Spade Connector</td>
<td>19</td>
<td>119</td>
<td>179</td>
<td>71.8</td>
<td>E</td>
</tr>
<tr>
<td>Optocoupler</td>
<td>24</td>
<td>119</td>
<td>172</td>
<td>76.5</td>
<td>E</td>
</tr>
<tr>
<td>Coil activated relay</td>
<td>1</td>
<td>111</td>
<td>408</td>
<td>2.80</td>
<td>F</td>
</tr>
<tr>
<td>Linear Bipolar IC (2)</td>
<td>84</td>
<td>110</td>
<td>135</td>
<td>87.7</td>
<td>C</td>
</tr>
<tr>
<td>Memory MOS IC (8)</td>
<td>46</td>
<td>105</td>
<td>137</td>
<td>76.7</td>
<td>D</td>
</tr>
<tr>
<td>Memory MOS IC (3)</td>
<td>12</td>
<td>104</td>
<td>176</td>
<td>53.8</td>
<td>F</td>
</tr>
<tr>
<td>Linear Bipolar IC (4)</td>
<td>30</td>
<td>102</td>
<td>141</td>
<td>68.5</td>
<td>D</td>
</tr>
</tbody>
</table>

Table 43: The failure rate hierarchy
<table>
<thead>
<tr>
<th>Component Type</th>
<th>Mounting method</th>
<th>Number of failures</th>
<th>Number of component hours</th>
<th>Failure rate in FITs</th>
<th>Upper 95% $\chi^2$ confidence limit</th>
<th>Lower 95% $\chi^2$ confidence limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multilayer ceramic capacitor</td>
<td>Surface Mount</td>
<td>35</td>
<td>$2.0 \times 10^9$</td>
<td>17.7</td>
<td>23.2</td>
<td>11.9</td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>37</td>
<td>$1.6 \times 10^9$</td>
<td>2.2</td>
<td>3.0</td>
<td>1.5</td>
</tr>
<tr>
<td>Pn-Junction diode</td>
<td>Surface mount</td>
<td>12</td>
<td>$1.6 \times 10^9$</td>
<td>7.5</td>
<td>12.7</td>
<td>3.8</td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>162</td>
<td>$4.9 \times 10^9$</td>
<td>3.3</td>
<td>3.8</td>
<td>2.8</td>
</tr>
<tr>
<td>Schottky Barrier diode</td>
<td>Surface mount</td>
<td>12</td>
<td>$6.0 \times 10^7$</td>
<td>200</td>
<td>338</td>
<td>103</td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>19</td>
<td>$4.4 \times 10^6$</td>
<td>43.2</td>
<td>64.6</td>
<td>26</td>
</tr>
<tr>
<td>Variable tuning capacitor</td>
<td>Surface mount</td>
<td>0</td>
<td>$2.5 \times 10^7$</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>10</td>
<td>$2.3 \times 10^8$</td>
<td>43.3</td>
<td>73</td>
<td>20.0</td>
</tr>
<tr>
<td>Thick film network resistor</td>
<td>Surface mount</td>
<td>0</td>
<td>$2.5 \times 10^8$</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>10</td>
<td>$1.4 \times 10^9$</td>
<td>7.1</td>
<td>12.2</td>
<td>3.4</td>
</tr>
<tr>
<td>Bipolar Digital IC &lt;10 Gates</td>
<td>Surface mount</td>
<td>0</td>
<td>$3.1 \times 10^7$</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>45</td>
<td>$5.5 \times 10^9$</td>
<td>8.0</td>
<td>10.7</td>
<td>5.9</td>
</tr>
<tr>
<td>Bipolar digital IC &lt;100 gates</td>
<td>Surface mount</td>
<td>0</td>
<td>$3.1 \times 10^7$</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>67</td>
<td>$4.5 \times 10^9$</td>
<td>14.8</td>
<td>18.6</td>
<td>11.5</td>
</tr>
<tr>
<td>Metal Oxide resistor</td>
<td>Surface mount</td>
<td>0</td>
<td>$8.8 \times 10^8$</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Other</td>
<td>31</td>
<td>$9.8 \times 10^9$</td>
<td>3.12</td>
<td>4.3</td>
<td>2.12</td>
</tr>
</tbody>
</table>

Table 44: Comparison of failure data for surface mounted components
<table>
<thead>
<tr>
<th>Package Type</th>
<th>Number of failures</th>
<th>Number of component hours</th>
<th>Failure rate in FITs</th>
<th>Upper 95% $\chi^2$ confidence limit</th>
<th>Lower 95% $\chi^2$ confidence limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plastic Transfer moulded</td>
<td>49</td>
<td>$1.050 \times 10^{10}$</td>
<td>2.47</td>
<td>3.51</td>
<td>1.62</td>
</tr>
<tr>
<td>Hermetically sealed -</td>
<td>60</td>
<td>$1.165.6 \times 10^{10}$</td>
<td>1.97</td>
<td>2.86</td>
<td>1.25</td>
</tr>
<tr>
<td>Glass/Ceramic</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 45: Comparison of failure data for plastic packaged components

<table>
<thead>
<tr>
<th>Component Type</th>
<th>Mounting method</th>
<th>Number of failures</th>
<th>Number of component hours</th>
<th>Failure rate in FITs</th>
<th>Upper 95% $\chi^2$ confidence limit</th>
<th>Lower 95% $\chi^2$ confidence limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multilayer ceramic capacitor</td>
<td>Ground Benign</td>
<td>31</td>
<td>$9.31 \times 10^9$</td>
<td>3.33</td>
<td>4.6</td>
<td>2.26</td>
</tr>
<tr>
<td></td>
<td>Ground Mobile</td>
<td>89</td>
<td>$8.63 \times 10^9$</td>
<td>10.3</td>
<td>12.6</td>
<td>8.28</td>
</tr>
<tr>
<td>Bipolar digital IC &lt;100 gates</td>
<td>Ground Benign</td>
<td>50</td>
<td>$1.0 \times 10^9$</td>
<td>69.5</td>
<td>90.0</td>
<td>51.6</td>
</tr>
<tr>
<td></td>
<td>Ground Mobile</td>
<td>36</td>
<td>$5.5 \times 10^8$</td>
<td>65.5</td>
<td>88.6</td>
<td>45.9</td>
</tr>
<tr>
<td>pn-Junction diode</td>
<td>Ground Benign</td>
<td>143</td>
<td>$4.63 \times 10^{10}$</td>
<td>3.09</td>
<td>3.62</td>
<td>2.60</td>
</tr>
<tr>
<td></td>
<td>Ground Mobile</td>
<td>46</td>
<td>$1.27 \times 10^9$</td>
<td>36.3</td>
<td>47.5</td>
<td>26.6</td>
</tr>
</tbody>
</table>

Table 46: Comparison of failure data for different environments
8.5. Time Dependent Failure Statistics

The study of the time dependent failure behaviour of components operating in equipment in the field is important both from (a) a design point of view to determine if the so called 'early failures' are being removed successfully by screening and burn-in and (b) to investigate whether the constant failure rate assumption for so called extrinsic failures is in fact valid.

8.5.1. Failure Intensity

Failure intensity is defined as:

\[ \frac{n}{N \Delta t} \]

where \( n \) is the observed number of failures in a given time period (\( \Delta t \)) and \( N \) is the population at risk during this period. For the purpose of this investigation the \( \Delta t \) interval has been chosen to be 1,000 hours. To illustrate further the means of computation of the failure intensity Figure 26 and Figure 27 show the number of failures of MOS digital integrated circuits with between \( 10^3 \) and \( 10^4 \) gates in 1,000 hour intervals, and the corresponding number of components at risk throughout the same time intervals respectively. The shape of Figure 27 reflects the fact that as the number of systems going into the field increases with time, the populations of the various component types is also increasing.

By dividing the data in Figure 27 by the data in Figure 26 and correcting for the time interval Figure 28 is obtained. The failure intensities are shown as the central solid lines, and the dotted lines above and below are the 95% \( \chi^2 \) confidence limits. These limits are dependent on the number of observed failures and the population at risk. Wide limits are indicative of a lack of failure data. This situation will improve with time as more data on the various component types becomes available. It should be noted that when the failure intensity drops to zero on the time axis, this means no failures have been observed in that particular 1,000 hour period.
Figure 26: Number of failures of MOS digital ICs with between $10^3$ to $10^4$ gates versus time.

Figure 27: Number of MOS digital ICs with between $10^3$ and $10^4$ gates at risk in the field.

Figure 28 shows the failure intensity versus time for CMOS digital devices with $10^3 - 10^4$ gates. Again, the initial values decreases to a lower level over a 4,000 hour period. The values would seem to oscillate around the $5 \times 10^{-7}$ failure intensity value thereafter.
Figure 28: Failure intensity curve for digital MOS ICs with between $10^3$ and $10^4$ gates

Figure 29 shows the failure intensity v. time for rectangular connectors. The failure intensity is observed to decrease within the first 2,000 hours, and thereafter remaining constant within reasonable approximation, although it is appreciated this statement does not take into account the confidence limits associated with the data.

Figure 29: Failure intensity curve for rectangular connectors

More details of this form of analysis can be found in [39],[40],[41],[42] and this form of analysis can be extended to the board and system level.
Figure 30 is the failure intensity curve for a system. The shape of this graph suggests that there is an early failure period that occurs before 5000 hours that has not been successfully removed by board burn in. The failures that occur in this period could well be early life component failures that should have been removed by part screening. By investigating the structure of this board it would be possible to discover which components are responsible for this early life failure by plotting their failure intensity curves.

![Figure 30: Failure intensity curve for a system](image)

The various peaks and troughs in the curve would have to be investigated in detail to discover if they are significant. The operating time after 18000 hours appears to be failure free which may suggest that the earlier fluctuations in the failure intensity are due to early component problems.

8.5.2. M(t) analysis

The theory of the M(t) method is covered in [43]. It is possible using the data in the database to obtain M(t) curves at the component, sub-system and equipment level. M(t) analysis produces a set of parameters which will differ according to the shape of the M(t) function. This function can have three standard shapes, increasing, decreasing and stationary. The following parameters will be obtained in each case,
The steady-state failure intensity \( (I) \) in FITs
The percentage of flawed components \( (P) \)
The uncertainty in the above figure \( (P',P') \)
The steady-state time \( (T_s) \)

Table 47: Parameters in the Decreasing Hazard Rate \( M(t) \) Model

The steady-state failure intensity, \( (I) \)
The failure free life \( (t_0) \)

Table 48: Parameters in the Increasing Hazard Rate \( M(t) \) Model

The steady state failure intensity \( (I) \)

Table 49: Parameters in the Constant Hazard Rate \( M(t) \) Model

These are illustrated in Figure 31 and Figure 32.

Figure 31: Decreasing \( M(t) \) function

\( P \) represents the proportion of (weak) components that will fail early in life: There is a confidence interval around the estimate of \( P \) derived from the confidence interval around the estimate of \( M(t) \). Using this method of analysis, the data in Table 50 may be derived for each component position in a system.
Figure 32: Increasing M(t) function

Table 50: Information Available from M(t) Analysis

| The proportion of components which will fail early in life (together with a confidence interval on the proportion). |
| The length of time over which this proportion will fail. |
| The steady state failure intensity |

An example of a M(t) curve for a system is shown in Figure 33. Examination of this curve allows an estimate of the percentage of weak systems in the population to be determined in a relatively short operational time and corrective action to be taken (This curve shows about 4% weak systems). The M(t) analysis can also be used to predict the eventual steady state failure intensity of a system.

8.6. Engineering analysis

Engineering analysis is analysis that takes into account the basic engineering and physical principles that exist in any equipment. It is more to do with pattern of failure events at the equipment and sub-system level than with the actual components. It can be used to assess the performance of any particular equipment design and can give useful insight into how equipments behave in the field[44].
8.6.1. Design Analysis

Some design faults are recurring faults along similar component descriptors. That is to say repeated failures occurring in the same reference position on a set of circuit boards of the same type. It can be identified by simply counting the occurrences of the same circuit reference positions in the failure table. Figure 34 demonstrates the output of such an analysis technique.

As can be seen from the figure the component in reference position Q26 has failed a larger number of times than the other components. This may suggest that there is a problem with that particular circuit reference position in terms of this board design. Further investigation would now be necessary to discover the cause of this bias towards the Q26 position. This form of analysis can be performed routinely with a field failure database and can help equipment and board designers to avoid similar problems in future designs.

In the time that the database has been active a number of design faults have been identified. Some of these are given in Table 51.
Figure 34: Design fault analysis on a particular circuit board design

<table>
<thead>
<tr>
<th>Component at fault</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quartz Crystals</td>
<td>Contamination inside package causing large amounts of drift.</td>
</tr>
<tr>
<td>Custom Chips</td>
<td>Software fault transferred into hardware</td>
</tr>
<tr>
<td>Wire Wound Potentiometers</td>
<td>Potentiometer wiper forced past end point due to bad control panel design.</td>
</tr>
</tbody>
</table>

Table 51: Some Design Faults Found by the IERI Database

8.6.2. Failure Interrelationships

This form of analysis can be used to determine the relationships between different components within a system. This means that board and equipment failure sequences are considered and any causality is investigated. This is done by looking at a particular board serial number and putting the failures into time sequence. This allows the chain of events to be followed. Analysis can be performed using a number of techniques, including examining the order of failures with respect to component types and the time between subsequent failures. Each will be described in turn.

(1) Consideration of failure order can be used to determine for example if failures in resistors cause failures in any capacitors in the circuit. This can be investigated further since it is possible to look at the boards electrical geography to
see if the secondary failures are close to the original failed component and so subject to heating effects etc.

For example in Figure 35 the failure patterns are shown for a number of circuit boards.

![Figure 35: Failure pattern analysis for a number of board serial numbers](image)

The codes in each box represent the component code as stored in the IERI database. Each box represents one failure event. The code NFF is an abbreviation for no fault found and it is interesting to note that for the first circuit board shown a number of NFFs have occurred. This board has also, in the middle of the NFF chain, has a pair of Hybrid (H) failures. It is possible that these replacements were made while seeking the cause of the NFF and are not true failures since they did not remove the failure problem. Further investigation is obviously necessary in this case. The circuit board with the serial number '300042' also shows the NFF occurring with hybrids. This may suggest that there is problem with the testability of the hybrids since there are appears to be a relationship between the hybrid device, which is a very complex devices, and the NFF problem.

This sort of failure pattern leads to the question of whether or not the board should be removed from the field. It may be more cost effective to replace this board
than to continue to repair it. This would depend upon the nature of the board and the economics of repair which, of course, are company dependent.

(2) Consideration of the time between failures allows the investigation of the 'good as new' or 'bad as old' situations. It also allows the testing and repair cycle to be investigated since if the next failure on a board occurs after a small amount of time (possibly zero hours) then it is possible that either the original testing did not identify the fault or the repair damaged the circuit in some way.

Figure 36: Times to failure for various board serial numbers

Figure 36 shows that for the board with serial number '12' the repair improves the reliability since the time between failure is extended. However this is not the case for the board with serial number 300042 since after the second repair the times to failure accelerate. This situation should be looked at in more detail to assess the viability of the repair process.

These form of analysis can be used as one way of looking at the multiple replacement problem. This occurs when in order to repair a board a large number of components are replaced. These components may or may not have failed. This happens for two main reasons, firstly when the actual failed component cannot be identified amongst a group of similar components so all are replaced, and secondly when the method of sequential replacement is used to repair the board. This is where
the repair engineer replaces components in sequence until the board works. In both cases these replaced components may or may not have failed.

### 8.6.3. Failure Modes Analysis

This form of analysis can be used to predict the most likely failure mode in components.

![Failure Modes for components](image)

**Figure 37: Failure Modes for components**

Figure 37 shows the breakdown of the failure modes for all the failures tracked by the database. As can be seen the most common failure modes is a functional failure. This occurs when the component is no longer able to perform its function and so only applies to active devices. The second largest failure mode observed is open circuit. This failure mode can apply to both passive and active devices.

### 8.6.4. No fault found analysis

There has been much discussion in recent years about the cause of no fault founds (NFF) situations. It has been suggested that the NFF are caused by the numbers of complex components present on a circuit board. Complex components would include hybrid devices, microprocessors, large memories and similar chips. The
reasoning behind this is that these devices will contain untested regions which may contain errors.

![Figure 38: Number of complex components versus nff](image)

These errors would only occur under certain circumstance which testing during maintenance could not hope to duplicate due to time constraints etc. If this is the case then a scatter plot such as that in Figure 38 would show some relationship.

![Figure 39: Number of connectors versus nff](image)

Another reason proposed for the incidence of NFF is the presence of connectors on the board. Connectors can corrode and cause intermittent failures, then
the removal of the board from the equipment, and the insertion of the board in test apparatus may clean the connector contacts and the board functions again. If this was the case then a relationship should be seen in the scattergram in Figure 39. As can be seen from both these figures no relationship appears to exist. There is of course plenty of scope here for further investigation into board complexity, other component combinations etc.

8.6.5. Operation Mode Analysis.

Operating mode analysis examines the relationships between number of failures that occur and the usage of the equipment failing. It is commonly felt that switching an electronic equipment on and off may cause damage to the equipment as electrical transients appear in the circuitry.

![Figure 40: Relative numbers of equipments in the field for each usage band](image)

Figure 40 shows the breakdown of equipments usage throughout the population of equipments that are operating in the field. As can be seen for the equipments tracked by the IERI database 67% have a usage figure of less than 50%.

Figure 41 shows the relative number of failure that occur in each usage band. As can be seen approximately 69% of the failure occur in equipments that have a usage figure less than 50%. This means that 69% of the failures occur in 67% of the population and this is approximately as expected so it appears that usage has very
Figure 41: Relative numbers of failures in each usage band

Little effect on the reliability of the equipments. However it should be kept in mind that there are a number of assumptions made when usage is calculated or used which may not be correct in all cases.

Figure 42: Relative numbers of NFF for each usage band

Figure 42 shows the relative numbers of NFF that are observed in each usage band. As can be seen from the figure 79% of NFF occur in equipments where the usage is less than 50%. This may suggest that there is a relationship between usage and NFF which requires further investigation.
8.7. Reliability Prediction

One of the most controversial techniques used at present in the field of reliability is the use of reliability prediction methodologies based on component failure data for the estimation of system failure rates.

Parts count analysis is one method by which the reliability of a new system can be estimated from previous experience. Data collected on analogous equipment can be used to predict the failure rate of a new system operating in the same environment. There are many reliability prediction models for doing this but care should be taken when using those approaches[45] It is always better to collect data from applications and equipment where the failure rates are going to be directly applicable to the new system.

In order to demonstrate the fallibility of prediction and to exercise one aspect of the database, ie the parts lists, A number of boards were selected from the IERI database and their reliability was predicted and compared with the actual observed performance in the field. The techniques chosen have been MIL-217E[30], HRD4[6], Siemens (SN29500)[46], CNET[47], and Bellcore (TR-TSY-000332)[48]. For each technique the associated published failure rates have been used. Parts count analysis has been performed on a number of equipments and Figure 43 shows the parts count failure rate for a system predicted from the field with the actual observed field failure rate.

The prediction can be seen to differ greatly from the observed field behaviour and from each other. Further analysis of the methods was performed and a widely differing dependence upon physical parameters was observed [45]. This suggests that predictions obtained by differing models can’t be compared and that great care should be taken when selecting a prediction technique since the physical parameters of the system will affect the prediction obtained in possibly unforseen ways.

IERI do not support this form of analysis for predicting system reliability but it is a useful technique when used as part of the design process.
8.8. Other analysis

The structure of the IERI database lends itself to a number of different forms of analysis. All the constant failure rate analysis is based upon an exponential failure model, however it has been suspected within the industry for some time, without any real evidence, that this is not always the case. Exploratory data analysis can be performed to discover the underlying distributions for some of the components in the database. This is described in some detail in [1],[49],[50] and [51].

More advance forms of analysis can also be used, as demonstrated in [1],[52],[53] and [54] where the methods of proportional hazards modelling and proportional intensity modelling are applied.
9. THE RACE-DIRAC DATABASE

Due to the reliability database experience accrued at IERI over many years when a British Telecom lead consortium received support from the EC under the auspices of the RACE program to develop a pan-european reliability database IERI were asked to be involved. The members of the consortium are listed in Table 52.

<table>
<thead>
<tr>
<th>Members</th>
</tr>
</thead>
<tbody>
<tr>
<td>British Telecom (UK) (Project leaders),</td>
</tr>
<tr>
<td>GEC-Marconi (UK),</td>
</tr>
<tr>
<td>Deutsche Budesposte Telekom (Germany),</td>
</tr>
<tr>
<td>Siemens (Germany),</td>
</tr>
<tr>
<td>Bertin et Cie (France),</td>
</tr>
<tr>
<td>CNET French Telecom (France),</td>
</tr>
<tr>
<td>Tecnorolis CSATA Novus Ortu (Italy)</td>
</tr>
<tr>
<td>Danish Engineering Academy (Denmark)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sub-contractors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loughborough University - IERI (UK)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Observers:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thompson-CSF (France)</td>
</tr>
<tr>
<td>Telefonica Spanish Telecom (Spain)</td>
</tr>
</tbody>
</table>

Table 52: List of consortium members

The project was in a number of phases. Phase one was to establish that collection of reliability data was possible and to develop the database for storing the information and to provide the analysis tools necessary to supply useful information to the European electronics industry. Phase two of the project was to fully populate the database. Different partners had different responsibilities, BT were the project managers, Bertin were to supply the software expertise, DIA the data analysis and LUT to host and administer the database.

The consortium met regularly to co-ordinate the project and made a number of decisions as to the form of analysis, the hosts database system and the like. Many documents were produced which fully described the project in term of the user expectations [55], the software methodology to be used [56],[57], the specification of how the reliability of the target systems should be assessed [58], what data should be collected and why [59] and the format of the projects output in terms of a printed
manual [60] and the prediction software [61]. Ideas for exploiting the project were also put forward [62].

The data analysis tool to be used was the M(t) method [63]. To this end three methods of performing the analysis, ie the production of the various parameters of the M(t) model, were developed and so three levels of data supply were defined each corresponding to one of the analysis methodologies.

1) Level One Data Collection

At this data collection level the statements in Table 53 agree true,

a) Each equipment type in the field is identifiable with a part number.
b) Each instance of an equipment in the field is identifiable with some form of serial number
c) Each printed circuit board type in an equipment is identifiable with a part number.
d) Each printed circuit board in an equipment is identifiable with some form of serial number.
e) Each component position on a circuit board is identifiable with a circuit reference position.

Table 53: Level one data description

The above information allows the exact calculation at any time of the number of circuit reference positions at risk in the field and allows the calculation of failure times to be exact.

2) Level Two Data Collection

At this data collection level the statements in Table 54 are true,

a) Each equipment type in the field is identifiable with a part number.
b) Each instance of an equipment in the field is identifiable with some form of serial number
c) Each printed circuit board type in an equipment is identifiable with a part number.
d) The number of components of a particular type on a circuit board is known.

Table 54: Level 2 data description

The above information allows the number of components of a particular component type at risk in the field at any time to be calculated. The time to first failure of a particular component can also be calculated.

3) Level Three Data Collection
At this data collection level the statements in Table 55 are true,

| a) Each equipment type in the field is identifiable with a part number. |
| b) The number of systems of a particular type in the field is known. |
| c) Each printed circuit board type in an equipment type is identifiable with a part number. |
| d) The number of components of a particular component type on a circuit board is known |

**Table 55: Level 3 data description**

The above information allows the number of component of a particular type in the field at any time to be calculated. The cumulative number of failed components up to a certain time can also be calculated.

Based on this three level structure a database was designed [64] and a coding system devised. This coding system was similar to the IERI database scheme and the data collected without further discussion are shown in Table 56, Table 57 and Table 58.
<table>
<thead>
<tr>
<th>Equipment structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturers name</td>
</tr>
<tr>
<td>Equipment Type Identification</td>
</tr>
<tr>
<td>Equipment Serial Number</td>
</tr>
<tr>
<td>Board Type Identification</td>
</tr>
<tr>
<td>Board Serial Number</td>
</tr>
<tr>
<td>Temperature difference from ambient</td>
</tr>
<tr>
<td>Equipment default duty cycle</td>
</tr>
<tr>
<td>Function of equipment</td>
</tr>
<tr>
<td>System default quality level</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Printed circuit board structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturers name</td>
</tr>
<tr>
<td>Board Type Identification</td>
</tr>
<tr>
<td>Component Classification Code</td>
</tr>
<tr>
<td>Component Part Number</td>
</tr>
<tr>
<td>Component Part Number Description</td>
</tr>
<tr>
<td>Circuit Reference Position</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set up information.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturers name</td>
</tr>
<tr>
<td>Equipment Type Identification</td>
</tr>
<tr>
<td>Equipment Serial Number</td>
</tr>
<tr>
<td>Application/Environment classification</td>
</tr>
<tr>
<td>Duty Cycle</td>
</tr>
<tr>
<td>Set up date</td>
</tr>
<tr>
<td>Production Date</td>
</tr>
<tr>
<td>Average Storage Time</td>
</tr>
<tr>
<td>Quality level</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Failure information.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturers name</td>
</tr>
<tr>
<td>Equipment Type Identification.</td>
</tr>
<tr>
<td>Equipment Serial Number.</td>
</tr>
<tr>
<td>Board Type Identification.</td>
</tr>
<tr>
<td>Board Serial Number.</td>
</tr>
<tr>
<td>Failure Date.</td>
</tr>
<tr>
<td>Circuit reference position primary failure</td>
</tr>
<tr>
<td>Circuit reference position secondary failures</td>
</tr>
<tr>
<td>Inserted board serial number</td>
</tr>
<tr>
<td>Failure mechanism.</td>
</tr>
<tr>
<td>Failure mechanism name</td>
</tr>
<tr>
<td>Failure mode.</td>
</tr>
<tr>
<td>Failure mode name</td>
</tr>
</tbody>
</table>

**Table 56: Data for level one analysis**
<table>
<thead>
<tr>
<th>Equipment structure</th>
<th>制造业名称</th>
<th>设备类型识别</th>
<th>设备序列号</th>
<th>板型识别</th>
<th>PCB存储时间</th>
<th>数量</th>
<th>设备默认最大负荷</th>
<th>温度与环境相比较</th>
<th>功能设备</th>
<th>系统默认质量水平</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturers name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equipment Type Identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equipment Serial Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Board Type Identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PCB Storage time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quantity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equipment default duty cycle</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Temperature difference from ambient</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Function of equipment</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>System default quality level</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Printed circuit board structure</th>
<th>制造业名称</th>
<th>板型识别</th>
<th>组件分类代码</th>
<th>组件部分号</th>
<th>组件部分号描述</th>
<th>数量</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturers name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Board Type Identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component Classification Code</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component Part Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component Part Number Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quantity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Set up information</th>
<th>制造业名称</th>
<th>设备类型识别</th>
<th>设备序列号</th>
<th>应用/环境分类</th>
<th>负荷周期</th>
<th>安装日期</th>
<th>生产序列号</th>
<th>平均存储时间</th>
<th>质量水平</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturers name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equipment Type Identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equipment Serial number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Application/Environment classification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Duty Cycle</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Set up date</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Production Code</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average Storage Time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quality level</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Failure information</th>
<th>制造业名称</th>
<th>设备类型识别</th>
<th>设备序列号</th>
<th>生产序列号</th>
<th>板型识别</th>
<th>板型生产编码</th>
<th>失效日期</th>
<th>失效事件</th>
<th>组件分类编码</th>
<th>组件部分号</th>
<th>组件部分号描述</th>
<th>失效机制</th>
<th>失效机制名称</th>
<th>失效模式</th>
<th>失效模式名称</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manufacturers name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equipment Type Identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equipment Serial Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equipment Production code</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Board Type Identification</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Board Production code</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failure Date</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failure event</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component classification code</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component Part Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component Part Number Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failure mechanism</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failure mechanism name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failure mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failure mode name</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 57: Data for Level Two Analysis
This data format and the concepts behind it are described fully in [65]. The component codes used by the RACE-DIRAC project are different to those used by the IERI database. They are split into a number of sections as shown in Table 59.
Section | Description
--- | ---
Technology | The technology describes the process and the sort of materials that make up this process. In order to do this the description is split into multiple levels in a tree-like structure which allows cut off at any point for analysis.
Configuration | The configuration describe the component type in terms of how it is configured. It tries to do this in an unambiguous manner. In order to do this the description is split up into multiple levels in a tree-like structure which allows cut off at any point for analysis purposes.
Complexity | Complexity is in most cases a number with a prefix to indicate what the number applies to, for example for an integrated circuit the complexity may be the number of transistors on a single chip, hence the code should be T2, this indicates that the number of transistors is 10^2. In some other cases a statement as to whether or not a device is a single device or a composite of many devices is all that is necessary. Complexity is not always applicable to a component as some components are monolithic structures.
Package | This describes the type of package that contains the actual component.
Interconnection/Mounting | This describe the means in which a component is attached to the circuit board and is similar for all component types.

Table 59: The structure of the RACE component codings

Further details of the RACE coding system for components can be found in [65] and [66]. A validation system similar to the one used by the IERI database was also designed [67][68]. Data is collected from the partners and validated at IERI where it is loaded into the database. A series of analysis tools are used to produce a set of output data [69] which consists of the various parameters of the M(t) analysis models. This output data is moved to a pc-based file where it can be acted upon by the end user software as described in [70] to predict the reliability of systems. Figure 44 shows a screen shot of the software in action.

Many useful ideas came out of this project which have since been applied to the IERI database. The component coding scheme is considered to be more flexible
than that used by the IERI database and so this database will be adapted to use the
DIRAC codes. This is generally the case for most coded fields, since the DIRAC
database was set up with knowledge of the limitations of the IERI database coding
schemes. In fact it is envisaged that the two databases will merge into one large
database in the near future. There already exists an agreement where the data can be
shared and with the use of DIRAC codes in the IERI database mean that the databases
have moved a step closer together.
10. THE IERI END USER SOFTWARE

Recent work on the IERI database has lead to the development of a PC-based reliability tool. This is known as \( I^2R \), for IERI Information Resource and is described fully in [71],[72],[73] and is summarised below.

The user selects, from a list of available components or equipment, the items that he is interested in. The user then selects the sort or reliability information that he requires. The software then provides the user with that information in a useful form. The user should be able to compare one items reliability against another and be able to group items into equipments for further analysis. When a system has been defined editing of that equipment is possible and the storage of that equipment is persistent. The program provides the correct information according to the item type selected. For single components the reliability information is about that single component and for equipments the information should be about all items in the equipment. This requires the summing of individual component information in a defined way. The user is able to obtain a report of the reliability of an equipment or a component in hard copy or paste the information into other software packages for printing or display. The data available to the user will be updated on a regular basis and some customization is available to allow the user to attach his own information to the system. General reliability information and information about IERI and PCS is also be available. The program is configurable to allow the sharing of data files across a network and to allow the user to fully define the directory structure the program uses. The following information is available from the program.

1) Component alerts

A component alert is a warning to the user that a problem has been observed in the field with the particular component type of interest. The alert could be generated by any of the companies supplying data to IERI or from within IERI after laboratory tests.
2) Failure mechanisms

A failure mechanism report on the sort of failure mechanisms that have been reported in the field or in laboratory tests carried out by the companies supplying data, IERI or a third party. In general the reports will contain information gleaned from a search of the current literature available.

3) Failure modes

A failure mode report on the sort of failure modes that have been reported back to IERI. This will give percentage likelihood of occurrence whenever possible. The information will come from the IERI system.

4) Standards

Information is available on MIL, BS, CECC, and ISO standards for the selected items and the mnemonics and codes used by the major prediction mechanisms for the component type are supplied. This will include MIL-217, HRD4, DIRAC, and IERI.

5) Reliability Information

The constant failure rate figures of a number of prediction systems are supplied and can be used to generate a parts count prediction of a system. The M(t) curve, the failure intensity curve and associated curves are also be available.
Figure 45: Screen shot showing prediction window, M(t) window, system definition window and system/component selection window.

Figure 46: Screen shot showing an alert report, a failure mechanism report, a published paper and a field reliability information bulletin.
11. FUTURE WORK

This section describes the future work that could be performed in terms of development of the actual database and the types of analysis.

11.1. Further development work

There is a large amount of further development that could be performed to convert the database from a research style data base to a commercial product,

1) Complete the merge of the RACE-DIRAC and the IERI database.

2) Move the database to a PC and MS-DOS based system mounted on a Novel netware network to permit local access and loading.

3) Redesign of both end user programs to allow access directly to the databases across the network (client-server architecture).

11.2. Further avenues of research

Due to the detailed nature of the database there is still a large amount of research that can be performed

1) Further no fault found analysis

2) Reliability prediction using Weibul models

3) Reliability prediction using failure intensity analysis

4) Board migration and equipment structure studies.
12. CONCLUSIONS

The IERI database is a useful reliability research tool. It allows the analysis of large amounts of field data to be performed and hence improves the confidence that can be placed in the results. The data structure is well designed and detailed enough for most analysis needs, the implementation is robust and portable to other computer architectures and the procedures used in the administration of the database are well defined and self documenting. The analysis that has to date been performed has yielded a number of useful results, many of which have been touched upon in this document. Future work and further development will make the database a unique resource in the improvement of electronics reliability.
13. ACKNOWLEDGEMENTS

This work has been carried out with the support of Professional Component Services (DRA), and I am grateful for their permission to publish this thesis.
REFERENCES

1. J. Marshall
"The Organisation and Statistical Analysis of an Electronic Component Field failure database"

2. ANSI(1975)
"Interim report of the ANSI/X3/SPARC Stud Group on Database Management Systems"
ACM SIGFIDET

3. K.L. Wong
"What is wrong with the existing reliability prediction methods"

4. Michael Pecht and Wen-Cheng Kang,

5. CODUS User Manual
CODUS LTD 1987

6. British Telecom,
"Handbook of Reliability Data for Components used in Telecommunications Systems",

7. J.L. Hursch and C.J. Hursch
"Working with Oracle"
TAB Professional and Reference Books, 1987

"An Electronic Component Reliability Database"
Proc. 10th ARTS, Bradford 1988

9. J. Hayes
"Field Failure of Electronic Components: A Working Document for the Transfer of Data to LUT"
IERI-CORD Document 1986

10. J. Jones and A.P. Schwarzenberger
"Data Coding Document"
IERI-CORD Document 1992
11. D.M. Ritchie and K. Thompson
"The UNIX time-sharing system"
Bell System Technical Journal Vol 57 No 6
July-August 1978

12. P.Wang
"An Introduction to Berkeley Unix"
Wadsworth publishing Company. 1988

"The C programming language"
Bell System Technical Journal Vol 57 No 6
July-August 1978

14. B.W.Kernighan and D.M.Ritchie
"The C Programming Language",
Second Edition,
Prentice-Hall, 1991

15. ANSI(1988)
"American national Standard for Information Systems - Programming language C"
X3.159-1989

16. ISO(1987)
"Final Text of DIS 9075, Information Processing Systems - Database Language SQL"
Report of TC97/SC21/WG3

17. Oracle Corporation
"Introduction to SQL"
1989

18. D.Vadura
DMAKE Documentation
Available from DVadura@watdragon.uwaterloo.ca

19. D.Lawrence and B.Straight
MEMACS 3.10 Documentation
Available from wuarchive.wustl.edu

20. Hewlett-Packard Company LTD
SCCS Documentation.

21. T Williams et al
Gnuplot Documentation
Available from wuarchive.wustl.edu
22. H.Spencer
   Indian hill coding guide
   Available from University of Toronto

23. E. Allnman, P De Silva, B Appleton
   Parseargs documentation
   Available from Brad@ssd.csd.harris.com

24. J. Ousterhout
   Tcl documentation
   Available from University of California at Berkeley

25. Student Information Processing Board of Massachusetts Institute of Technology
    "A common error description library for UNIX"
    Available from Massachusetts Institute of Technology

26. S.J.Chester and A.Bendell
    "Discussion of Useage information within the LUT Electronic Component
    Database"
    Department of Mathematics, Statistics and Operational Research, Nottingham
    Polytechnic
    IERI-CORD Internal Document 1989

27. IC-Master 1988
    Hearst Business Communications INC

28. J.Jones
    "Division of Integrated Circuits into Categories as Defined by IC-Master"
    IERI-CORD Internal document 1989

29. J.Jones
    "Integrated Circuits Codes for Historic Data"
    IERI-CORD Internal Document 1989

30. US Mil-Hdbk-217,

31. J.Jones
    "Reported Failure Mechanisms in Electrolytic Capacitors"
    IERI-CORD Distributed Document 1989

32. J.Jones
    "Reported Failure Mechanisms in Coil Activated Relays"
    IERI-CORD Distributed Document 1989

33. J.Jones
    "Reported Failure Mechanisms in Separable Connectors"
    IERI-CORD Distributed Document 1990

136
34. J.Jones
"Reported Failure Mechanisms in Wire-Wound Potentiometers"
IERI-CORD Distributed Document 1990

35. J.Jones
"Reported Failure Mechanisms in Piezoelectric Devices"
IERI-CORD Distributed Document 1990

36. J.Jones
"Reported Failure Mechanisms in Schottky Barrier Diodes"
IERI-CORD Distributed Document 1989

37. Shooman, M.L.
"Probabilistic Reliability - An Engineering Approach"
McGraw-Hill 1968

38. D.W.Wightman and S.J.Chester
"Rating Systems for Confidence Levels for Component Data"
Department of Mathematics, Statistics and Operational Research, Nottingham Polytechnic
IERI-CORD Internal Document 1989

"Failure Intensity Analysis of Resistors and Capacitors".

40. J.A.Hayes, J.A.Jones, A.P.Schwarzenberger and D.S.Campbell,
"Failure Intensity Analysis of Electronic Components",

41. D.S.Campbell, J.A.Hayes, J.A.Jones and A.P.Schwarzenberger,
"Reliability Behaviour of Electronic Components as a Function of Time",

42. D.S.Campbell, J.A.Hayes, J.A.Jones and A.P.Schwarzenberger,
"Reliability Behaviour of Electronic Components as a Function of Time",

43. J.Møltoft
"New methods for specification and determination of component reliability characteristics"
CERT 1990
44. J.A.Jones, M.Zahid and J.A.Hayes
"Use of a field failure database from improvement of product reliability".
To be Published June 94.

45. M.Zahid, J.A.Jones and J.A.Hayes
"Evaluation of Reliability Prediction Methodologies"
4th European Symposium on Reliability of Electron Devices (ESREF)
Bordeaux pp 59-64, 4th-7th October 1993.

46. Siemens AG, SN29500,
"Reliability and Quality Specifications Failure Rates of Components",
Siemens Technical Liaison and Standardization 1986.

47. Centre National D'Etudes des Telecommunications,

48. Bellcore Technical Reference TR-TSY-000332,

49. D.W.Wightman and S.J.Chester
"Distributional Analysis of component failures data (Between 10 and 30 failures)"
Department of Mathematics, Statistics and Operational Research, Nottingham Polytechnic
IERI-CORD Internal Document 1992

50. J.M.Marshall, J.A.Hayes, D.S.Campbell and A.Bendell
"The Analysis of Electronic Component Reliability Data"
Proc. 6th EuReDaTa, Siena, 1988

51. J.M.Marshall, A.Bendell, J.A.Hayes and D.S.Campbell
"An Exploratory Approach to the Reliability Analysis of Electronic Component Field Data"
Proc. CERT 90, Crawley, 1990

52. S.J.Chester and D.W.Wightman
"An Exploratory Survey of Component Failure Data for Potential Analysis"
Department of Mathematics, Statistics and Operational Research, Nottingham Polytechnic
IERI-CORD Internal Document 1990

53. D.W.Wightman and S.J.Chester

138
"Illustration of Dataset Formulation and Application of Adapted Proportional Hazards modelling Programmes to D1A Failure Data"
Department of Mathematics, Statistics and Operational Research, Nottingham Polytechnic
IERI-CORD Internal Document 1990

54. D.W.Wightman and S.J.Chester
"Poisson Proportional Intensity Models with Covariates"
Department of Mathematics, Statistics and Operational Research, Nottingham Polytechnic
IERI-CORD Internal Document 1989

55. M.A.Pearce and R.G.Parker
"Software User requirements Document"
92/GEC/000/ID/C/026/a2
RACE-DIRAC Project Document

56. A.Azarian
"Handbook of software procedures and standards"
92/FRB/000/DS/C/019/a1
RACE-DIRAC Project Document

57. A.Azarian and V.Sorel
"Hardware/Software Configuration Specification"
92/FRB/000/DN/C/025/a1
RACE-DIRAC Project Document

58. M.Nyborg
"Common Functional Specification"
92/DIA/000/DR/C/076/a1
RACE-DIRAC Project Document

59. J.Jones
"Data Requirements for Analysis"
92/LUT/000/DN/C/002/A10
RACE-DIRAC Project Document

60. M.Nyborg
"DIRAC Reliability Handbook"
92/DIA/000/DR/C/072/a1
RACE-DIRAC Project Document

61. J.Jones
"End User Software Specification"
92/LUT/000/DN/C/018/A1
RACE-DIRAC Project Document
62. D.S. Campbell and J.A. Hayes
"Dirac Exploitation"
92/LUT/000/DN/C/009/a1
RACE-DIRAC Project Document

63. M. Nyborg
"Data Analysis Techniques and Supporting Tools"
92/DIA/000/DR/C/030/a2
RACE-DIRAC Project Document

64. A. Azarian and V. Sorel
"Full Functional Specification"
92/FRB/000/DS/C/032/a1
RACE-DIRAC Project Document

65. J. Jones
"Data Coding Document for DIRAC"
92/LUT/000/DN/C/013/A10
RACE-DIRAC Project Document

66. J. Jones
"Component Classification for DIRAC"
92/LUT/000/DN/C/004/A10
RACE-DIRAC Project Document

67. P. F. Marteau
"Data Control, Data Validation and Data loading Procedures: User Manual"
92/FRB/000/DS/C/051/a1
RACE-DIRAC Project Document

68. P. F. Marteau
"Definition of Conceptual Scheme for the Database Structure: Detailed Design of the DIRAC Central Database Software"
92/FRB/000/DS/C/048/a1
RACE-DIRAC Project Document

69. J. Jones
"Definition of Dirac Output Data"
92/LUT/000/DN/C/017/A1
RACE-DIRAC Project Document

70. P. F. Marteau, V. Sorel and M. Ho
"User manual for DIRAC End User Software"
92/FRB/000/DS/C/061/a1
RACE-DIRAC Project Document
71. J.Jones
"Functional Specification for CORD Data Access Software"
IERI-CORD Internal Document 1993

72. J.Jones
"Architectural Design of CORD Data Access Software"
IERI-CORD Internal Document 1993

73. J.Jones
"Detailed Design of CORD Data Access Software"
IERI-CORD Internal Document 1993
APPENDIX ONE - DATA CODING DOCUMENT

The appendix contains some of the sections from the data coding document used by the data supplying companies. Sections that duplicate the main text have been removed.

A. DATABASE PROCEDURES

The following protocols need to be strictly observed when coding data.

1. Each field (i.e., coding) is to be separated by a semi-colon delimiter. The final field must be terminated by a semi-colon delimiter. This means that the semi-colon is a reserved character which cannot be used in any other way than that defined here.

2. Where a letter indicator is used as a prefix to a coding a question mark '?' is to be used universally to indicate 'not known'. For 'no fault found's see Section 10 on page 143.

3. When coding non-essential information which is not known, or even likely to be known, the allowed spaces must be filled with '?'s. For 'no fault found's see section 10 on page 143.

4. Dates are to be specified in the following manner:
   Year month day (eight digits allowed)
   e.g. 19860110 is the 10th January, 1986.

5. All times are to be given in real 'calendar' hours which do NOT take usage into account.

6. All numbers specifying a number of items or time should be right justified and leading zeros to be used to fill in allowed spaces not required. e.g. 4 characters are allowed to specify the time a board is in storage, and that field is stored in days, so:
   0001 is 1 day
   0010 is 10 days
   0100 is 100 days
   1000 is 1000 days

7. When specifying serial numbers, part numbers and dates the numbering should be left justified with trailing spaces if all the numbers are not required. e.g. 6 alphanumeric characters preceded by a letter indicator has been allowed to specify equipment serial number. Thus A13760- illustrates a coding requiring only 5 characters. The 0 is part of the coding, the - represents a trailing space.

8. The following definitions of "numeric", "alphanumeric" and "printable" characters are used throughout this document:

   ASCII
   Numeric: Numbers 0 to 9
   Alphanumeric: Upper and lower case letters, and numbers 0 to 9
   Printable: All characters from ASCII 32 (space) to ASCII 126 (tilde), but excluding ASCII 34 (double quotes), ASCII 37 (per cent), ASCII 39 (single quote), ASCII 59 (semi-colon) and ASCII 95 (underscore).

   Note that these are decimal ASCII numbers.

9. All fields are to contain alphanumeric characters unless stated otherwise.

10. No Failure Found (NFF) Records

There are three types of NFF records that can be supplied by a data source:

1. Equipment level NFFs. These occur when an equipment is reported as faulty, but the service engineer switches it on and finds that it works, so no information about the board or the component can be given.

2. Board level NFFs. These occur when an equipment is reported as faulty, the service engineer gets it to work by replacing a board, but no fault can be found on the board when returned for repair. So no information about the component can be given.
3. Component level NFFs. These occur when an equipment is reported as faulty, the service engineer gets it to work by replacing a board, the board is repaired by replacing a component, but when that component is tested it appears to be functioning normally. All information can be given as normal.

For no fault found records at the equipment level it is not possible to work out which specific boards were in that equipment, because only the board types and not the serial numbers are known in the population files. These records are therefore of limited use and will not be loaded into the main database.

For no fault founds at the board level a dummy component type NFF is used, but otherwise the record is similar to a normal failure.

For no fault founds at the component level the component type that was replaced to make the board functional must be given, i.e. the component type must NOT be coded as NFF, but the failure mode of the component should be coded as N. Table 1 summaries the differences from an ordinary failure record for the three types of NFF record:

<table>
<thead>
<tr>
<th>FIELD NAME</th>
<th>Component</th>
<th>Board</th>
<th>Equipment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Company</td>
<td>NFF</td>
<td>NFF</td>
<td>NFF</td>
</tr>
<tr>
<td>Record Number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equipment Type</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Simultaneous component failure identifier</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Equipment serial number</td>
<td></td>
<td>NNNNNN</td>
<td></td>
</tr>
<tr>
<td>Company design/ part number of board or module</td>
<td></td>
<td>NNNNNNNNNNNNNNN</td>
<td></td>
</tr>
<tr>
<td>Circuit board serial number</td>
<td></td>
<td>NNNNNN</td>
<td></td>
</tr>
<tr>
<td>Up and running date of board in system</td>
<td></td>
<td>NNNNNNN</td>
<td></td>
</tr>
<tr>
<td>Manufacturing date of board</td>
<td></td>
<td>NNNNN</td>
<td></td>
</tr>
<tr>
<td>Previous use of board</td>
<td></td>
<td></td>
<td>?0</td>
</tr>
<tr>
<td>Storage</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Component classification code</td>
<td>(not NFF)</td>
<td>NFP</td>
<td>NFP</td>
</tr>
<tr>
<td>Component part number</td>
<td>NNNNNNNNN</td>
<td>NNNNNNNN</td>
<td></td>
</tr>
<tr>
<td>Screening and qualification level of component</td>
<td></td>
<td>NNN</td>
<td>NNN</td>
</tr>
<tr>
<td>Circuit reference conditions</td>
<td>NNNNNNN</td>
<td>NNNNNN</td>
<td></td>
</tr>
<tr>
<td>Component manufacturer</td>
<td>??</td>
<td>?1</td>
<td>??</td>
</tr>
</tbody>
</table>

Table 1: Comparison of NFF Specifications for Component, Board, and Equipment Levels.
### B. FILE I. FAILURE DATA

The required input data on failed components and the method of coding is as follows with the database field name given in brackets.

1. **Company (co)**
   - To be coded by two letters agreed with LUT.
   - 1st letter - company identifier
   - 2nd letter - company data source.

2. **Record Number (report-no)**
   - Coded by six printable characters.
   - e.g. 103479

3. **Equipment Type (equip-type)**
   - To be coded by three alphanumeric characters agreed with LUT which will uniquely define the equipment.
   - e.g. A17

4. **Simultaneous Component Failure Identifier (fail-id)**
   - To be coded by a single digit or letter depending on the system used by the data sources for recording multiple failure.
   - (a) Simultaneous Failures Recorded in One Incident Report
     - A single digit to be used, starting at 1, to uniquely define each failure record.
     - (i) Single failure to be recorded thus

     | Report-no | equip-type       | fail-id |
     |-----------|-----------------|--------|
     | 123456    | (coding for equipment type) | 1      |

     (ii) 3 failures would be recorded by three separate failure records thus

     | Report-no | equip-type       | fail-id |
     |-----------|-----------------|--------|
     | 123456    | (coding for equipment type) | 1      |

---

Table 1 No fault found specification

<table>
<thead>
<tr>
<th>FIELD NAME</th>
<th>NFF Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Component</td>
</tr>
<tr>
<td>Batch Number</td>
<td></td>
</tr>
<tr>
<td>Operating condition of system immediately prior to failure</td>
<td></td>
</tr>
<tr>
<td>Failure mode of equipment</td>
<td></td>
</tr>
<tr>
<td>Operating time of system to failure</td>
<td></td>
</tr>
<tr>
<td>Operating time of board</td>
<td></td>
</tr>
<tr>
<td>Reported failure mode and mechanism of component</td>
<td>N</td>
</tr>
</tbody>
</table>
Simultaneous Failures Reported on Separate Report Forms

A single letter to be used with each failure record number to indicate the number of components failing simultaneously.

The following coding to be used:

- A: Single component failure
- B: Two simultaneously failed components
- C: Three simultaneously failed components etc.

Thus 3 simultaneous failures may be recorded thus:

<table>
<thead>
<tr>
<th>Report-no</th>
<th>equip-type</th>
<th>fail-id</th>
</tr>
</thead>
<tbody>
<tr>
<td>123456</td>
<td>(coding for equipment type)</td>
<td>2</td>
</tr>
<tr>
<td>123456</td>
<td>(coding for equipment type)</td>
<td>3</td>
</tr>
</tbody>
</table>

5. Equipment Serial Number (equip-serial-no)*
To be coded by six printable characters.

6. Company Design/Part Number of Board or Module (co-design)*
Coded by thirteen printable characters, preceded by a letter indicator showing the origin of the part number as follows:

- A: Nato stock number
- B: Company part number
- C: Project specific

The part number should start immediately after the letter indicator and trailing spaces left at the end of the number if all 13 characters are not required.

E.g. B1376940298 __ __

7. Board Serial Number (cb-serial-no)*
The board serial number is to be coded with nine printable characters. Trailing spaces to be used if required.

8. Up and Running Date of Board in System (up-id & cb-up-running)*
To be coded by six digits. This date is preceded by a letter indicator showing how the up and running date of the board in the system was arrived at. The letters to be used are A to I as defined below:

- A: True, recorded up and running date
- B: Calculated from end of commission period
- C: Calculated from start of commission period
- D: Calculated from delivery date
- E: Calculated up and running date for shipboard equipment
- F: True, recorded up and running date (repaired board)
- G: Calculated up and running date (repaired board)
- H: Estimated up and running date (repaired board)
- I: Calculated from first fail

For new equipments the date recorded for the up and running date of the board in the system will be the same as the up and running date of the equipment. If the board is a replacement the date recorded must be the date when the equipment is up and running after the board has been installed.

9. Manufacturing Date of Board (manuf-id & cb-manuf-date)
To be coded by six digits. This date is the final date of testing of the finished assembly. The date is to be preceded by a letter indicating the accuracy of the recorded date thus:

- A: Known accurately from board production records
- B: Estimated from system manufacturer
- C: Estimated from end user

10. Previous Use (prev-id & previous-use)*
Previous use is defined as the sum of the total previous use prior to this time around. Coded by four digits giving the previous use in days preceded by a letter indicating the number of repairs thus:

- A: Undergone no previous repairs
- B: Undergone 1 previous repair
- C: Undergone 2 previous repairs
- and so on alphabetically

A repair to be interpreted as occurring because of component failure or a 'no failure found'. Maintenance is to be reported as soft information. Thus e.g. B0127 would indicate a board with 127 days prior 'active service' and which has undergone one previous repair.

11. Storage (stor-id & storage)
Coded by four digits giving the storage period in days and preceded by a letter indicating storage conditions as follows:

- A: Temperature and humidity control
- B: Temperature control only
- C: No control
- D: Other, special
- ?: Not known

Thus B0520 would indicate a board in storage for 520 days in a temperature controlled environment only. The storage period to be entered is the storage period this time around.

12. Component Type/Generic Description (como-type)*
To be coded by four characters against the generic descriptions listed in Appendix 2 on page 7. Trailing spaces to be left when the four characters are not required.

All IC's are to be coded by a four figure coding. The first two spaces will be coded with reference to the general technology (see page 7) and the next two codings will relate to function and size respectively.

The functional types to be used and their respective codings are:

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>CODING</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital</td>
<td>D</td>
</tr>
<tr>
<td>Microprocessor</td>
<td>U</td>
</tr>
<tr>
<td>Interface</td>
<td>I</td>
</tr>
<tr>
<td>Linear</td>
<td>L</td>
</tr>
<tr>
<td>Memory</td>
<td>M</td>
</tr>
</tbody>
</table>

These categories are based on those used in the 'IC Master' book, and are summarised in Appendix 5 on page 7.

The size of each functional type is to be coded in relation to the following:

Digital: Number of GATES
Microprocessor: Number of BITS
Interface: Number of TRANSISTORS
Linear: Number of TRANSISTORS
Memory: Total number of BITS

The number of GATES (for digital devices) is to be coded by a single digit relating to a power of 10. Thus 100 gates ($10^2$) would be coded by a 2, 1000 gates by a 3 and so on.

The number of BITS (for microprocessor coding) is to be coded by a single digit relating to a power of 2. A 16 bit microprocessor would thus have a code of 4.

The number of TRANSISTORS (for interface and linear devices) is to be coded by a single digit relating to the power 10. Thus a device with 1000 ($10^3$) transistors will be coded by a 3.

The total number of BITS (for memory devices) is to be coded by a single digit relating to the power 2. If the maximum single digit of 9 is insufficient to give the total number of bits, the letters of the alphabet starting with A are to be used sequentially. A look-up table can be found in Appendix 6 on page 7 to help convert letters to numbers.

N.B. If an exact digit cannot be used to express the total numbers of gates, bits or transistors, the next digit UP should be used. Thus:

- $10^0$ means up to 10
- $10^1$ means between 11 and 100
- $10^2$ means between 9 and 16
WHEN AN EXACT DIGIT CANNOT BE USED SOFT INFORMATION SHOULD ACCOMPANY THE DATA SET.

13. **Component Part Numbers (comp-part-no)**
   To be coded by eight printable characters showing the part number of IC’s, hybrid subsystems and transistors. This is essential **ONLY** for IC’s, hybrids and transistors.

14. **Component Encapsulation/Mounting Details (comp-add-info)**
   To be coded by 4 characters from left to right showing in the following order:
   (i) Encapsulation (2 characters)**1**
   (ii) Mounting (1 character)**2**
   (iii) PCB/Substrate (1 character)**2**
   The codings to be used are shown in Appendix 3 on page 7.

15. **Screened Level of Component (screen-level)**
   To be coded with three alphanumeric characters. The first character is the only essential character and will be coded against the following definitions:
   - 9 - BS9000/CECC
   - 8 - Presumed BS9000/CECC (not checked)
   - 7 - E.R. (Established Reliability)
   - X - Other imposed screening
   - Y - Commercial part
   When the component is definitely not a BS9000/CECC component the appropriate coding against the MIL-STD quality levels should be used as given in Tables 1A to 1F. These tables are extracted from DEF. STAN. 00-41.

16. **Circuit Reference Position (circ-ref-position)**
   Coded with six printable characters. The circuit reference position used on the company’s circuit diagrams should be transcribed directly. Number from left to right using trailing spaces if necessary.

17. **Component Manufacturer (comp-manuf)**
   Coded with two alphanumeric characters.

<table>
<thead>
<tr>
<th>Component manufacturer</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMD</td>
<td>AD</td>
</tr>
<tr>
<td>AMP</td>
<td>AP</td>
</tr>
<tr>
<td>Bourns</td>
<td>B</td>
</tr>
<tr>
<td>Cathodeon Crystals</td>
<td>CC</td>
</tr>
<tr>
<td>Ferranti</td>
<td>F</td>
</tr>
<tr>
<td>Hamlin</td>
<td>HA</td>
</tr>
<tr>
<td>Hitachi</td>
<td>HI</td>
</tr>
<tr>
<td>ICL</td>
<td>IC</td>
</tr>
<tr>
<td>Intel</td>
<td>IN</td>
</tr>
<tr>
<td>ITT</td>
<td>I</td>
</tr>
<tr>
<td>Motorola</td>
<td>MO</td>
</tr>
<tr>
<td>Mullard</td>
<td>MR</td>
</tr>
</tbody>
</table>

---

1 Not all character positions in this field are essential for all component types.

2 This is essential for all component types.

3 Only the first character of this field is essential.
Component manufacturers codes

<table>
<thead>
<tr>
<th>Component manufacturer</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEC</td>
<td>NC</td>
</tr>
<tr>
<td>IMO</td>
<td>OM</td>
</tr>
<tr>
<td>Phillips</td>
<td>PH</td>
</tr>
<tr>
<td>Plessey</td>
<td>PL</td>
</tr>
<tr>
<td>Semitron</td>
<td>C</td>
</tr>
<tr>
<td>SGS</td>
<td>SG</td>
</tr>
<tr>
<td>Sinclair</td>
<td>SI</td>
</tr>
<tr>
<td>STC</td>
<td>ST</td>
</tr>
<tr>
<td>Texas</td>
<td>TE</td>
</tr>
<tr>
<td>Toshiba</td>
<td>TO</td>
</tr>
</tbody>
</table>

Table 6: Component manufacturers codes

18. **Batch Number** (batch-no)
   Coded with six alphanumeric characters.

19. **Operating Condition of System Immediately Prior to Failure** (equip-operat-cond)
   To be coded by three alphanumeric characters, the first of which will be a letter indicator defining a common category of system operating condition immediately prior to failure.
   - A - Operational
   - B - Self test
   - C - Stand by
   - D - Other

   The second two characters will be system specific and coded to provide unique definitions of system failure. The codings will be defined individually by the companies and LUT.

20. **Failure Mode of Equipment** (equip-fail-mode)
   Coded by two alphanumeric characters. The failure modes will be equipment specific. LUT will define these individually with the companies and attempt to determine a commonality of failure mode descriptors.

21. **Operating Time of System to Failure** (equip-op-time-id & equip-operat-time)*
   Coded with five digits giving the operating time of system to failure in hours. The coding will be preceded by a letter indicator showing how the time was arrived at.
   - A - Known accurate time. ETI adjusted - time zero known.
   - B - Calculated from ETI reading - time zero not known.
   - C - Estimated, no ETI reading available.

   N.B.: The ETI reading must not be taken as a true operating time of system to failure unless an adjustment has been made for 'time zero'.

22. **Operating Time of Board** (this time around) (bd-op-time-id & bd-operat-time)*
    To be coded with five digits giving the time in hours and preceded by a letter identifier showing how the time was arrived at as defined below:
    - A - Known accurate time. ETI adjusted - time zero known.
    - B - Calculated from ETI reading - time zero not known.
    - C - Estimated, no ETI reading available.

    NOTE: The ETI reading must not be taken as a true operating time of system to failure unless an adjustment has been made for 'time zero'.

148
23. **Reported Failure Mode of Component (co-fail-mode)**

To be coded by one letter followed by five printable characters. The letters will be defined to indicate failure modes as shown below:

- **O** - Open circuit
- **S** - Short circuit
- **F** - Functional; mainly applicable to logic devices e.g. incorrect output for a given input.
- **P** - Parametric drift
- **D** - Frequency degradation category
- **Q** - Solder joint failure
- **M** - Mechanical failure of component
- **A** - Electronic failure associated with a non-track failure e.g. software or mechanical

If a failure record is subsequently identified as a systematic failure, then a coding X may be used on updates only. The 5 spaces following the letter have been left to accommodate a failure report number. When no failure analysis is carried out, these five spaces should be filled by 5 "P's.

C. **FILE 2. EQUIPMENT STRUCTURE**

Coding from left to right, following the protocols outlined in Section 2, the data required in the following order is:

1. **Company (co)**
   - To be coded by two letters agreed with LUT.
   - 1st letter: company identifier
   - 2nd letter: company data source

2. **Equipment Type (equip-type)**
   - To be coded by three alphanumeric characters agreed with LUT which will uniquely define the equipment e.g. A17

3. **Company Design/Part Number of Board or Module (co-design)**
   - Coded by thirteen printable characters, preceded by a letter indicator showing the origin of the part number as follows:
     - **A** - Nato stock number
     - **B** - Company part number
     - **C** - Project specific
     - The part number should start immediately after the letter indicator and trailing spaces left at the end of the number if all 13 characters are not required.
     - e.g. B1376940598 _ _ _

4. **Number of Boards of a Particular Design Number (co-no)**
   - To be coded with three digits giving the total number of boards for each design number present in the particular equipment type.

5. **Primary Function of Board/Module (co-prim-func)**
   - To be coded by two alphanumeric characters. The first character in the coding to indicate the application frequency, the second the primary function. The codes to be used are as follows:

<table>
<thead>
<tr>
<th>Application Frequency</th>
<th>Primary Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 DC</td>
<td>A General (used if others not applicable)</td>
</tr>
<tr>
<td>2 RF</td>
<td>B Filters</td>
</tr>
<tr>
<td>3 Microwave</td>
<td>C Amplifiers</td>
</tr>
<tr>
<td>4 Optoelectronic</td>
<td>D Signal sources</td>
</tr>
<tr>
<td>5 Digital</td>
<td>E Converters</td>
</tr>
<tr>
<td>6 Electromechanical</td>
<td>F Storage</td>
</tr>
</tbody>
</table>

   e.g. 1C indicates a DC amplifier

   If necessary, other definitions of primary function will be added if too many boards fall into the 'General' category.
The following common modules found in equipments should be designated thus:

- Control panel 6E
- Display panel 4D
- Power Supply 1D
- Backplane 6A

D. FILE 3. BOARD STRUCTURE

Coding from left to right the data required is listed in order below:

1. **Company (co)**
   To be coded by two letters agreed with LUT.
   - 1st letter - company identifier
   - 2nd letter - company data source.

2. **Company Design/Part Number of Board or Module (co-design)**
   Coded by thirteen printable characters, preceded by a letter indicator showing the origin of the part number as follows:
   - A - Nato stock number
   - B - Company part number
   - C - Project specific
   The part number should start immediately after the letter indicator and trailing spaces left at the end of the number if all 13 characters are not required.
   - e.g. B1376940298

3. **Component Type (comp-type)**
   To be coded by four characters against the generic descriptions listed in Appendix 2. Trailing spaces to be left when the four characters are not required.
   All IC's are to be coded by a four figure coding. The first two spaces will be coded with reference to the general technology (see page ?) and the next two codings will relate to function and size respectively.
   The functional types to be used and their respective codings are:

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>CODING</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital</td>
<td>D</td>
</tr>
<tr>
<td>Microprocessor</td>
<td>U</td>
</tr>
<tr>
<td>Interface</td>
<td>I</td>
</tr>
<tr>
<td>Linear</td>
<td>L</td>
</tr>
<tr>
<td>Memory</td>
<td>M</td>
</tr>
</tbody>
</table>

These categories are based on those used in the 'IC Master' book, and are summarised in Appendix 5 on page ?.

The SIZE of each functional type is to be coded in relation to the following:

- Digital Number of GATES
- Microprocessor Number of BITS
- Interface Number of TRANSISTORS
- Linear Number of TRANSISTORS
- Memory Total number of BITS

The number of GATES (for digital devices) is to be coded by a single digit relating to a power of 10. Thus 100 gates (10^2) would be coded by a 2, 1000 gates by a 3 and so on.

The number of BITS (for microprocessor coding) is to be coded by a single digit relating to a power of 2. A 16 bit microprocessor would thus have a code of 4.

The number of TRANSISTORS (for interface and linear devices) is to be coded by a single digit relating to the power 10. Thus a device with a 1000 (10^3) transistors will be coded by a 3.

The total number of BITS (for memory devices) is to be coded by a single digit relating to the power 2. If the maximum single digit of 9 is insufficient to give the total number of bits, the letters of the alphabet starting with A are to be used sequentially. A look-up table can be found in Appendix 6 on page ? to help convert letters to numbers.

N.B. If an exact digit cannot be used to express the total numbers of gates, bits or transistors, the next digit UP should be used. Thus:

- 10^0 means up to 10
- 10^1 means between 11 and 100

150
2^ means between 9 and 16

WHEN AN EXACT DIGIT CANNOT BE USED SOFT INFORMATION SHOULD ACCOMPANY THE DATA SET.

4. Component Part Numbers (comp-part-no)*
   To be coded by eight printable characters showing the part number of IC's, hybrid subsystems and transistors.

5. Component Encapsulation/Mounting Details (comp-add-info)*
   To be coded by 4 characters from left to right showing in the following order:
   (i) Encapsulation (2 characters)*
   (ii) Mounting (1 character)*
   (iii) PCB/Substrate (1 character)*

   The codings to be used are shown in Appendix 3 on page 7.

6. Number of Components (comp-no)*
   To be coded by 3 digits giving the number of components of a given type present on a particular board design number.

E. FILE 4. EQUIPMENT IDENTIFICATION

   Coding from left to right the data required is listed in order below:

1. Company (eo)*
   To be coded by two letters agreed with LUT.
   1st letter - company identifier
   2nd letter - company data source.

2. Equipment Type (equip-type)*
   To be coded by three alphanumeric characters agreed with LUT which will uniquely define the equipment.

3. Equipment Serial Number (equip-serial-no)*
   To be coded by six printable characters.

4. Up and Running Date of Equipment (up-id & up-running)*
   Coded by six digits. The up and running date is the date on which the equipment is 'switched on' and in service as a unit. A letter indicator preceding the date is required to show how the up and running date was calculated.
   A - True, recorded up and running date
   B - Calculated from end of commissioning period
   C - Calculated from start of commissioning period
   D - Calculated from delivery date
   E - Calculated up and running date for shipboard equipment
   F - Calculated from first failure date

5. End Date (end-date)*
   To be coded with six digits. The end date is defined as the date when the period of observation is finished or the date when a particular piece of equipment is withdrawn from service.

6. Usage (usage)*
   To be coded by three digits indicating the percentage of possible usage time the equipment was operating.
   e.g. 010 indicates 10%

7. Environment (env)*
   Coded by two numbers as shown in Appendix 1 on page 7.
   e.g. 03 indicates 'ground mobile'

F. Actual first stage validation checks

---

4 This is essential ONLY for IC's, hybrids and transistors.

5 Not all character positions in this field are essential for all component types.

6 This is essential for all component types.
This is a list of the validation tests carried out at LUT on data being loaded into the database.

1. Equipment identification file

The first three fields must uniquely identify a line in this data file.

Company
   Field width 2, one of a list of allowed codes.

Equipment type
   Field width 25, characters alphanumeric with ? not allowed.

Equipment serial number
   Field width 15, characters printable with ? not allowed.

Up and running date of equipment
   First character : Field width 1, one of a list of allowed codes.
   Rest of field : Field width 8, valid date with ? not allowed.

End date
   Field width 8, valid date with ? not allowed.

Useage
   Field width 3, characters numeric with ? not allowed.

Environment
   Field width 20, one of a list of allowed codes.

2. Equipment structure file

The first three fields must uniquely define a field in this data file.

Company
   Field width 2, one of a list of allowed codes.

Equipment type
   Field width 25, characters alphanumeric with ? not allowed.

Company design/Part number of board or module
   Field width 16, characters printable with ? not allowed. The first character of this field is tested against a list of allowed codes.

Number of boards of a particular design number
   Field width 3, characters numeric with ? not allowed.

Primary function of board or module
   Field width 2, one of a list of allowed codes.

3. Board Structure File

The first five fields must uniquely define a line in this data file.

Company
   Field width 2, one of a list of allowed codes.

Company design/Part number of board or module
   Field width 16, characters printable with ? not allowed. The first character of this field must be one of a list of allowed codes.

Component classification code
   Field width 30, one of a list of allowed codes.

Component part number
   Field width 50, characters printable with ? allowed. If the first character of the comp type field is H, I or T then the characters must be printable with ? not allowed.

Number of components

152
4. Failure File

The first four fields must uniquely define a line in this data file.

Company
Field width 2, one of a list of allowed codes.

Failure report number
Field width 6, characters printable with ? not allowed.

Equipment type
Field width 25, characters alphanumeric with ? not allowed.

Simultaneous component failure identifier
Field width 1, characters alphanumeric with ? not allowed.

Equipment serial number
Field width 15, characters printable with ? not allowed.

Company design/Part number of board or module
Field width 16, characters printable with ? not allowed. The first character of this field must be one of a list of allowed codes.

Board serial number
Field width 15, characters printable with ? not allowed.

Up and running date of board in system
First character : Field width 1, one of a list of allowed codes.
Rest of record : Field width 8, valid date with ? not allowed.

Manufacturing date of board
First character : Field width 1, one of a list of allowed codes.
Rest of record : Field width 8, valid date with ? not allowed.

Previous use
First character : Field width 1, one of a list of allowed codes.
Rest of record : Field width 4, characters numeric with ? not allowed.

Storage
First character : Field width 1, one of a list of allowed codes.
Rest of record : Field width 4, characters numeric with ? allowed.

Component classification code
Field width 30, one of a list of allowed codes.

Component part number
Field width 50, characters printable with ? allowed. If the first character of the comp_type field is H, I or T then the characters must be printable with ? not allowed.

Screening and qualification level of component
Field width 4, one of a list of allowed codes.

Circuit reference position
Field width 6, characters printable with ? not allowed.

Component manufacture
Field width 2, characters alphanumeric with ? allowed.

Component batch number
Field width 6, characters printable with ? allowed.

Operating condition of system immediately prior to failure
Field width 3, characters alphanumeric with ? allowed. The first character of this field must be one of a list of allowed codes.
Failure mode of equipment
Field width 2, characters alphanumeric with ? allowed.

Operating time of equipment to failure
First Character : Field width 1, one of a list of allowed codes.
Rest of record : Field width 5, characters numeric with ? not allowed.

Operating time of board
First character : Field width 1, one of a list of allowed codes.
Rest of record : Field width 5, characters numeric with ? not allowed.

Component failure mode and mechanism
Field width 6, characters alphanumeric with ? allowed. The first character of this field must be one of a list of allowed codes.

G. Actual second stage validation checks

5. Equipment identification file
1) Check that the end date is more recent than the up and running date of equipment

6. Equipment structure file
1) Check that this equipment is found in the equipment identification file.

7. Board structure file
1) Check that this board is found in the equipment structure file.
2) Check for consistency of component classification code coding for each component part number, but don't do the test when there is no component part number.
   a) Compare with other records in this board structure file.
   b) Compare with data in the main database.
   c) Compare with data in separate IC part number database

8. Failure file
1) Check that this equipment is found in the equipment identification file.
2) Check that this board is found in the equipment structure file for this equipment.
3) Check that this component is found in board structure file for this board?
4) Check that the up and running date of board date is not older than the up and running date of equipment.
5) Check component simultaneous failure identifier for simultaneous failures and check them. Two failures are regarded as simultaneous if they occur on the same serial number board at the same equipment operating time.

   The component simultaneous failure identifier can either be a digit or a letter. The two cases are as follows:

   DIGIT: component simultaneous failure identifier incrementing digits, failure report numbers all the same.

   LETTER: component simultaneous failure identifier all the same, the letter corresponding to the number of simultaneous failures, the failure report numbers should all be different.

Any error in any one record causes rejection of all simultaneous failures to ensure consistency within the database.

6) Deal with component failure mode and mechanism 'X' as per the section ? document.

7) Check previous use for trackability and check the track back. This is done by selecting out all the failures on a particular board serial number and ordering them by previous use. The first character previous use fields should be in sequence starting at A with none missing. The first previous use figure in the series must be zero. The calculation of previous use figure is checked. It is expected that:

   where
   TPU = this previous use
   LBOT = board op time of last failure record

154
LPU = previous use of last failure record

Any error in any record in a series causes all other records in the series to be rejected.

H. Possible validation error codes and their meanings.

Non ASCII character found.
A non ASCII characters exists in one of the fields. - see section ?

Field too narrow or ";" in middle of field.
The field length is incorrect and may be too short. Alternatively the reserved character ";" may have been used in a field. See section 1

Field too wide or missing ";" at end of field.
The field length is incorrect and is too wide. There may be a missing ";" at the end of a record.

Field not one in list of allowed values.
The field contains an illegal value which has not been defined. In order to use a new value IERI must be informed before data is supplied using this new value.

Field contains character(s) other than alphanumeric or ?.
A character has been found which is illegal for this field type.

Field contains character(s) other than alphanumeric.
A character has been found which is illegal for this field type.

Field contains character(s) other than numeric or ?.
A character has been found which is illegal for this field type.

Field contains character(s) other than numeric.
A character has been found which is illegal for this field type.

Date field has year earlier than 1980.
The data is earlier than that expected by the database. This check can be relaxed on request providing the reason for the relaxation is valid.

Date field has month not in range 1-12.
A date field contains an invalid date.

Date field has illegal day for that month.
A date field contains an invalid date.

Character not in list of allowed values.
A character has been found which is illegal for this field type.

Old style IC coding which could not be updated from part number database.
The old style IC coding supplied could not be updated from the database information.

Fourth character of component classification code not alphanumeric.
Invalid component classification code.

Fourth character of component classification code not numeric.
Invalid component classification code.

Part number is null for Hybrids, ICs or Transistor component type.
A part number was not supplied for the component classification code for hybrids, transistors or ICs

Field is incorrect for a no-fault-found record.
The no fault found field entry is incorrect. See section ?

Date field all spaces or ?s not allowed here.
The format of a date field is incorrect.
A character has been found which is illegal for this field type.

Field contains character(s) other than printable or.

A character has been found which is illegal for this field type.

Failed to load into database.

The record failed to load into the database because the key fields are non-unique. This may be because the field is

Up running date of equipment more recent than end of observation.

The up and running date of the equipment id after the end date of the same equipment

Equipment type does not occur in equipment ID file.

The equipment identification file contains information about an equipment type that is unknown to the database

Board type does not occur in board structure file.

The equipment structure file contains information about a board type that is unknown to the database

Component classification code and component part number fields not coded consistently within new data set.

The data supplied does not match that supplied previously

Component classification code and component part number fields coded differently from data in IC part number database.

This equipment does not match any equipment in equipment id file.

This board does not match any board in equipment structure file.

This component does not match any component in board structure file.

More than one component type found for this circuit reference position.

The failure data supplied to IERI contains a component classification code/Circuit reference position pair that does not match the previous pair.

Numeric fail id codes out of sequence.

The failure identifier sequence does not match the actual failure date sequence.

Alphabetic fail id code does not match the number of simultaneous failures.

The simultaneous identifier does not match the actual number of simultaneous failures.

First failure does not have previous use set to 00000.

The first failure supplied for a board does not have a previous use of zero.

Previous use id code out of sequence.

The previous use identifier order does not match the previous use time sequence.

Previous use figure incorrect or out of sequence.

The previous use identifier order does not match the previous use time sequence.

Fail mode X is only allowed on updates, not in original file.

The failure mode code 'X' can only be used to update records as design faults. It cannot be used to update the current record.

One error in previous fail sequence causes all to be rejected.

A failure sequence has been rejected because of an error in one of the failure records that make up the sequence.

One error in simultaneous fail group causes all to be rejected.

A failure sequence has been rejected because of an error in one of the failure records that make up the sequence.

The equipment type does not occur in the equipment structure file.
The equipment identification file contains information about an equipment type that is unknown to the database.

Up running date of board more recent than end of observation.
   The up and running date of a board is after the end date of the including system.

Up running date of board LESS recent than up running date of the equipment.
   The up and running date of a board is before the up and running date of the including system.

Board's up running date is earlier than equipment's up running date.
   The up and running date of a board is before the up and running date of the including system.

Board up running date and board operating time give failure date after end date of observation.
   The calculated failure date of a board is after the end date of the including equipment.

Board's up running date is before the date of the previous failure on this board.
   The calculated failure date of a board is before the previous date of failure the including equipment.

The previous failure on this board is a simultaneous failure and gives a different failure date for each of the simultaneous records.
   The simultaneous failures did not occur at the same time.

Failed during load to main table.
   The data failed to load into the database.

WARNING - old style Ixxx comp_type updated from part number database.
   Unable to update the old style IC coding from the part number database.
APPENDIX TWO - SOFT INFORMATION QUESTIONNAIRE

This appendix contains the soft information questionnaire. It is in the form of a number of multiple choice questions and a number of descriptive questions. Details about why these questions are present can be found in section 6.31.

A. Equipment Type and Design

1. What is the equipment's market?
   - Military
   - Commercial
   - Other - please specify

2. What is the equipment type?
   - Communications - Wire based
   - Communications - Radio based
   - Computer - Main frame
   - Computer - Mini
   - Computer - Micro
   - Weapon system - Attack type (e.g., missile)
   - Weapon system - Defensive type (e.g., RADAR)
   - Commercial hardware (e.g., control system)
   - Test equipment
   - Consumer goods
   - Other - Please specify

3. Was the equipment designed for the customer?
   - Yes, to the customer's own complete specification
   - Yes, only the purpose of the equipment was specified
   - No, System available off the shelf

4. When was the equipment originally designed?
   - Within the last two years
   - 2 - 5 Years ago
   - 5 - 10 Years ago
   - Longer than ten years ago

5. Has the equipment undergone any design changes in its lifetime?
   - Yes, more than one major redesign
   - Yes, one major redesign
   - Yes, only minor design changes
   - No

6. If there have been design changes, why did they occur?
   - Original design did not meet spec.
   - Customer changed specification
   - Improvements to equipment

7. When did the design changes occur?
   - During design
   - During prototype construction
   - During manufacture
   - After shipping some items

8. Were previously sold items recalled for the same design changes?
   - Yes
   - No

9. Does the data supplied to IERI contain any previous designs?
10. Are any design changes likely to be made in the future
   Yes
   No
   Minor changes only

B. Maintenance policy and repair strategy

1. Is routine preventative/corrective maintenance carried out on the system
   Yes
   No

2. Does this maintenance involve modification of the system if faults are discovered across all systems
   Yes
   No

3. Are system modifications made with maintenance or only when a failure is corrected
   With failure
   During maintenance

4. Who repairs the equipment
   - The customer
   - Agents of the manufacturer
   - The manufacturer
   - Other - Please specify

5. Is the equipment repaired on site or shipped elsewhere
   - On site
   - Shipped elsewhere

6. Are all repairs reported to the manufacturer
   - All
   - Some

7. When a system fails
   - Is the system replaced with a working one, the non-working system then taken for repair
   - Are boards replaced until the system works, the boards then being taken for repair
   - Testing and specific repair carried out immediately
   - Other - please specify

C. System construction

1. How large is the system
   - Handheld
   - Laptop
   - Desktop
   - Free standing
   - Other - Please specify

2. While the system is operating is it
   - Portable by a single man

159
3. How is the system constructed

- Single board
- Mother-daughter board system
- Bus based system
- Backplane system
- Other - please specify

4. Is the system

- A single unit
- Multiple units in local groupings
- Multiple remote units
- Other - please specify

5. How is power supplied to the system

- Internal power supply
- External power supply
- Other - Please specify

6. Is the system supplied with power from

- 3-Phase mains supply
- Standard 240 v AC supply
- Externally derived DC supply
- Internal batteries
- Other - Please specify

7. Can the system operate without external power

- Yes - Full functionality
- Yes - Partial functionality
- Yes - Full functionality for short time
- Yes - Partial functionality for short time
- Yes - Shutdown system only
- No

8. Is the system cooled

- No Cooling / Natural cooling
- Fan assisted cooling
- Water cooling
- Other

9. Is the equipment exposed to any unusual hazards outside its design specification

- No
- Yes - please specify

D. Data collection mechanism

1. Who collects the failure information.

- The customer
- The manufacturer
- Appointed agents of the manufacturer.
- Other - please specify

2. Is a standard form used.

- No
3. Who fills in the form

A qualified engineer
A service technician
The operator
Other - Please specify

E. General

1. Is second sourcing used for components in this system

   Yes
   No

2. Is there project wide screening on components

   Yes
   No

3. If there is a failure on a particular circuit reference position, can we assume that all other components that are in that position have the same screening

   Yes
   No

4. If there is a failure on a particular circuit reference position, can we assume that all other components that are in that position have the same manufacturer

   Yes
   No

5. Is IERI supplied with all failure reports

   Yes
   No

   If No what is filtered out

   Failures during production/testing
   Failures during burn in
   Equipment no fault founds
   Non-Electrical failures
   Other - Please specify

6. If failure records are screened out are adjustments made to the supplied records to 'hide' this screening from LUT

   Yes
   No

   If Yes Then please explain how these adjustments are made.

7. Is there board movement between equipment types

   Yes
   No

8. Are failed components tested/analyzed

   Yes
   No

   If Yes can IERI get the reports

   Yes
9. Are failed components available for analysis at IERI

Yes
No

F. Database specific information

1. Please state any general assumptions that are being made about this data set

2. Specify the method of calculation/estimation of the up and running date and state the accuracy

3. Specify the method of calculation/estimation of the manufacturing date of board and state the accuracy

4. Specify the method of calculation/estimation of the total previous use and state the accuracy

5. Specify further details of the storage environment
   - Temperature (nominal and accuracy)
   - Humidity (nominal and accuracy)

6. Specify details of the previous storage environment.
   - Temperature (nominal and accuracy)
   - Humidity (nominal and accuracy)

7. Specify the assumptions made in presuming that the component is BS9000/CECC without checking

8. Specify the assumptions made in presuming that the component is other imposed screening (X) without checking.

9. Specify what is meant by the 'D - other' system operating conditions immediately prior to failure.

10. Specify the method of calculation/estimation of the operating time of system to failure, and the accuracy of the results obtained

11. Specify the method of calculation/estimation of the operating time of board (this time around) and the accuracy of the results obtained

12. Is the end date
   - End of observation
   - Withdrawal from service

13. Will further data be supplied after this end date

   Yes
   No

14. Is there any other information that could prove useful while analysing the failure data from this system
This appendix contains a number of papers published on the IERI database where the author has been a major contributor.


J.A.Jones, M.Zahid and J.A.Hayes "Use of a field failure database from improvement of product reliability". 9Th European Safety and Reliability conference. 30th May - 3rd June 1994. To be Published.
An Electronic Component Reliability Database has been established at Loughborough University of Technology and data collection has been under way for a period of three years. This work is sponsored by the Electronic Component Group (ECG) of RSRE and involves a collaborative effort with several major British and Danish equipment manufacturers and the Danish Engineering Academy.

The workshop will identify the concepts behind the collection of dependable data at the component level. This, by necessity, involves the tracking of equipments and circuit boards (or sub-systems), ideally, throughout their operational lives. It is only by such a long term study that the failure behaviour throughout the component working life can be modelled.

The methodology behind the data reporting format, the confidentiality aspects involved in dealing with data from different companies who are competitors in the market place, and other considerations which resulted in the final database design, are discussed.

The detailed nature of the final design will eventually allow statistically meaningful comparisons to be made between environments, interconnection technologies and screening levels.

Finally, a practical demonstration of the database will be given. This will highlight the ease of data retrieval at the equipment, circuit board and component levels.
FAILURE INTENSITY ANALYSIS OF RESISTORS AND CAPACITORS

J. A. HAYES, J. A. JONES, A. P. SCHWARZENBERGER & D. S. CAMPBELL

International Electronics Reliability Institute,  
Department of Electronic & Electrical Engineering,  
University of Technology,  
Loughborough, Leicestershire LE11 3TU,  
United Kingdom.

Tel.: 0509-222851; Fax: 0509-222854

SUMMARY

The International Electronics Reliability Institute (IERI) at Loughborough University of Technology in the U.K. has been collecting and analyzing reliability information on electronic components and systems over a period of seven years. This has involved a collaborative exercise with several major U.K. and European electronic equipment manufacturers. This exercise, by nature of its detailed data collection at the component, circuit board and system level allows IERI to undertake in depth analytical studies into the field failure behaviour of the whole range of electronic components. At its most simple level this allows the computation of hierarchical failure rates based on dependable field failure data. The production and use for system level prediction of such constant failure rate figures is a source of discussion and controversy in reliability journals.

The present paper undertakes to examine the validity of constant failure rate for electronic components in general, with references to the results on the data available so far on resistors and capacitors. This has been performed by means of failure intensity analysis. This technique involves division of component failures into small discrete time intervals throughout system operating periods and the calculation of instantaneous failure rates within these time intervals. Such analysis, based on actual field behaviour, enables the early, useful and so-called "wear out" periods of component behaviour to be investigated.

BACKGROUND

The collection of electronic component reliability data from the field has been under way at IERI for a period of four years or so. Data has been collected at the system and subsystem level in order to obtain detailed data at the component level. This has involved several major British and Danish companies. The organisation and some preliminary results have been discussed in several papers.

The detailed nature of the data collected has meant that as more data becomes available for each component type, it is possible to investigate the time dependent reliability behaviour of a range of electronic components based on field failure information. This method of presenting component behaviour examines the traditional constant failure rate assumption for electronic components as used in existing models for reliability prediction.

The observed failure rate of electronic components can be considered to be made up of an intrinsic failure rate and an extrinsic failure rate. The intrinsic failure rate is dependent on the physical processes inherent within the device. Those physical processes which result in failure have such long time scales that they do not contribute significantly, if at all, to the failure rate of the device during its useful life. In fact, CECC/WG-Reliability have defined the intrinsic value during the working life as zero. Wear out at the end of the useful working life is attributed to an increasing intrinsic failure rate.

Extrinsic failure rates are attributable to random events such as electrical overstress, ESD etc. If the assumptions in [7] above are valid, then the extrinsic failures are the only ones observed during the so called useful life of the component. This assumes that all early or residual failures have been removed during burn in.

Reliability prediction using constant failure rate models has come under much criticism recently. However, in practice there does appear to be some commonality of published failure data from a variety of sources collected on similar equipments operating in similar environments.
The study of the time dependent failure behaviour of components operating in equipment in the field is important both from (a) a design point of view to determine if the so called 'early failures' are being removed successfully by screening and burn in and (b) to investigate whether the constant failure rate assumption for so called extrinsic failures is in fact valid.

**RESULTS**

The results presented show the variation of failure intensity of selected components with time. Failure intensity is defined as:

\[
\frac{n}{N \Delta t}
\]

where \(n\) is the observed number of failures in a given time period (\(\Delta t\)) and \(N\) is the population at risk during this period. For the purpose of this investigation the \(\Delta t\) interval has been chosen to be 1,000 hours. To illustrate further the means of computation of the failure intensity Figs. 1 and 2 show the number of failures of MOS digital integrated circuits with between \(10^3\) and \(10^4\) gates in 1,000 hour intervals, and the corresponding number of components at risk throughout the same time intervals.

![Figure 1: Number of failures of MOS digital ICs with between \(10^3\) to \(10^4\) gates versus time](image1)

![Figure 2: Number of MOS digital ICs with between \(10^3\) and \(10^4\) gates at risk in the field](image2)

Figure 3 shows the failure intensity versus time plot for this type of device.

![Figure 3: Failure intensity curve for digital MOS ICs with between \(10^3\) and \(10^4\) gates](image3)

The failure intensity is shown as the central solid line, and the dotted lines above and below are the 95% \(\chi^2\) confidence limits. These limits are dependent on the number of observed failures and the population at risk. Wide limits are indicative of a lack of failure data. This situation will improve with time as more data on the various component types becomes available. It should be noted that when the failure intensity drops to zero on the time axis, this means no failures have been observed in that particular 1,000 hour period.

Figure 3 shows the initial failure intensity decreases to a lower level over a 4,000 hour period. The values would seem to oscillate around the \(5 \times 10^7\) failure intensity value thereafter.

Another example of an active device is shown in Fig. 4 where the failure intensity versus time plot for a pn-junction diode is shown. The failure intensity...
is observed to decrease within the first 6,000 hours, after which the value becomes low i.e. less than $2.5 \times 10^9$.

![Figure 4: Failure intensity curve for pn junction diodes](image)

Unfortunately, the data available on the various resistor and capacitor types does not allow the confidence limits to be narrowly defined. This situation will be rectified as more failure data becomes available on these particular types. Figures 5, 6, 7 and 8 show the failure intensity plots versus time and the associated confidence bands for wirewound resistors, carbon film potentiometers, ceramic multilayer capacitors and tantalum sintered solid electrolytics respectively.

Figure 5, for wirewound resistors, shows after an early failure period, zero failure intensities i.e. no failures are observed after 10,000 hours.

![Figure 5: Failure intensity curve for wirewound resistors](image)

In Fig. 6 for carbon film potentiometers, observed zero failure intensities are punctuated by occasional peaks which are representative of very small numbers of failures as demonstrated by the wide confidence limits. It may be postulated that these are extrinsic failures due to the nature of this device.

![Figure 6: Failure intensity curve for carbon film potentiometers.](image)

Very low failure intensities, generally less than $5 \times 10^9$ are observed in the cases of ceramic multilayers and tantalum sintered solid electrolytics, shown in Figs. 7 and 8 respectively. Each peak corresponds to a very small number of observed failures in any 1,000 hour interval.

![Figure 7: Failure intensity curve for ceramic multilayer capacitors](image)

![Figure 8: Failure intensity curve for tantalum sintered solid electrolyte capacitors](image)

The data presented in Figs. 5-8 is pooled data comprising of data from a variety of sources and environments. It has been presented in this manner as there is insufficient data as yet in the database in
CONCLUSIONS

It has been demonstrated from the results presented that:

(a) Early life failures are still occurring in the field even after burn in for the components presented. These effects are being observed to date on some components up to the first 6,000 hours of field operation. This effect is more apparent on the results present for active devices than passives. This should alert system designers to re-assess their burn in and screening procedures.

(b) If constant failure rates are to be used for prediction purposes, then more realistic values can be obtained in general after the early life failure data has been removed. It is felt that such extrinsic values have more use to the system designer than the assumption of zero failure attributable to intrinsic failure.

(c) The type of data presented could not be derived by any accelerated test programme, and it is only through the study of the field behaviour of systems and subsystems that the real reliability behaviour of components can be established.

ACKNOWLEDGEMENTS

The work is being carried out with the support of the Procurement Executive, Ministry of Defence, and we are grateful for permission to publish this paper.

REFERENCES


ABSTRACT

Considerable attention has been paid recently to the behaviour of the hazard rate function $h(t)$ as a function of time. This discussion has led to a lot of speculation as to the shape of the traditional 'bath-tub' curve showing failure or hazard rate as a function of time. Studies at the International Electronics Reliability Institute (IERI) at Loughborough University of Technology, in the United Kingdom, have recently been examining in detail the database created from field failure returns on a wide spectrum of electronic components. These components are in equipments subject to a spread of environmental conditions. The database created has been exercised with particular reference to the behaviour of MOS Ics, rectangular connectors, bipolar transistors and pn-junction diodes. Data has been analyzed using pooled information from a wide variety of sources and also from two specific environmental conditions, ground benign and ground mobile.

The results of this analysis are presented and the failure intensities as a function of time are given at 1,000 hour intervals up to a total time of 21,000 hours. Confidence limits at the 95% $\chi^2$ level are also given. The results show a rapidly falling failure intensity for the first few thousand hours and after this time the failure intensity appears to be relatively constant given the accuracy obtainable with the data available.

BACKGROUND

The collection of electronic component reliability data from the field has been under way at IERI for a period of four years or so. Data has been collected at the system and subsystem level in order to obtain detailed data at the component level. This has involved several major British and Danish companies. The organisation and some preliminary results have been discussed in several papers[^1[^2[^3]]].
The detailed nature of the data collected has meant that as more data becomes available for each component type, it is possible to investigate the time dependent reliability behaviour of a range of electronic components based on field failure information. This method of presenting component behaviour examines the traditional constant failure rate assumption for electronic components as used in existing models for reliability prediction.4-6

The observed failure rate of electronic components can be considered to be made up of an intrinsic failure rate and an extrinsic failure rate. The intrinsic failure rate is dependent on the physical processes inherent within the device. Those physical processes which result in failure have such long time scales that they do not contribute significantly, if at all, to the failure rate of the device during its useful life. In fact, CECC/WG-Reliability have defined the intrinsic value during the working life as zero.7 Wear out at the end of the useful working life is attributed to an increasing intrinsic failure rate.

Extrinsic failure rates are attributable to random events such as electrical overstress, ESD etc. If the assumptions in [7] above are valid, then the extrinsic failures are the only ones observed during the so called useful life of the component. This assumes that all early or residual failures have been removed during burn in.

Reliability prediction using constant failure rate models has come under much criticism recently.8-9 However, in practice there does appear to be some commonality of published failure data from a variety of sources collected on similar equipments operating in similar environments.5-6

The study of the time dependent failure behaviour of components operating in equipment in the field is important both from (a) a design point of view to determine if the so called 'early failures' are being removed successfully by screening a burn in and (b) to investigate whether the constant failure rate assumption for so called extrinsic failures is in fact valid.

RESULTS

The results presented show the variation of failure intensity of selected components with time. Failure intensity is defined as:

\[
\frac{n}{N \Delta t}
\]

where n is the observed number of failures in a given time period (\( \Delta t \)) and N is the population at risk during this period. For the purpose of this investigation the \( \Delta t \) interval has been chosen to be 1,000 hours. To illustrate further the means of computation of the failure intensity Figs. 1 and 2 show the number of failures of MOS digital integrated circuits with between \( 10^3 \) and \( 10^4 \) gates in 1,000 hour intervals, and the corresponding number of components at risk throughout the same time intervals. The shape of Fig. 2 reflects the fact that as the number of systems going into the field increases with time, the populations of the various component types is also increasing.
Figure 1: Number of failures of MOS digital ICs with between $10^3$ to $10^4$ gates versus time.

Figure 2: Number of MOS digital ICs with between $10^3$ and $10^6$ gates at risk in the field.

Figures 3 to 6 show failure intensity versus time for the following components: rectangular connectors, digital CMOS with $10^3 - 10^4$ gates, pn-junction diode and bipolar transistor decreases.

The failure intensities are shown as the central solid lines, and the dotted lines above and below are the 95% $\chi^2$ confidence limits. These limits are dependent on the number of observed failures and the population at risk. Wide limits are indicative of a lack of failure data. This situation will improve with time as more data on the various component types becomes available. It should be noted that when the failure intensity
drops to zero on the time axis, this means no failures have been observed in that particular 1,000 hour period.

Figure 3 shows the failure intensity v. time for rectangular connectors. The failure intensity is observed to decrease within the first 2,000 hours, and thereafter remaining constant within reasonable approximation, although it is appreciated this statement does not take into account the confidence limits associated with the data.

![Figure 3: Failure intensity curve for rectangular connectors](image)

Figure 4 shows the failure intensity versus time for CMOS digital devices with $10^3 - 10^4$ gates. Again, the initial values decreases to a lower level over a 4,000 hour period. The values would seem to oscillate around the $5 \times 10^{-7}$ failure intensity value thereafter.

![Figure 4: Failure intensity curve for digital MOS ICs with between $10^3$ and $10^4$ gates](image)
Figures 5 and 6 show the failure intensity versus time graphs for pn-junction diodes and bipolar transistors respectively. The failure intensity for the diodes does not appear to reach a steady state until approximately 6,000 hours when the value becomes low (i.e. less than $2.5 \times 10^{-9}$). The bipolar transistor exhibits the initial failure intensity decrease within 2,000 hours.

The data presented in Figs. 3-6 is pooled data comprising of data from a variety of sources and environments. It has been presented in this manner as there is insufficient data as yet in the database in every environment for every component to present meaningful results. However, at this moment it is possible to present failure intensity behaviour with time for bipolar transistors in ground benign and ground mobile
environments. These are shown in Figs. 7 and 8.

![Figure 7: Failure intensity curve for bipolar transistors in a ground mobile environment](image1)

![Figure 8: Failure intensity curve for bipolar transistors in a ground benign environment](image2)

A continuing decrease to low values of failure intensity is observed in the ground mobile environment. However, in the ground benign environment the wide confidence limit does not allow of any significant statement to be made at this stage, although it does appear to have a similar shape to the pooled data.
CONCLUSIONS

It has been demonstrated from the results presented that:

(a) Early life failures are still occurring in the field even after burn in for the components presented. These effects are being observed to date on some components up to the first 6,000 hours of field operation. This should alert system designers to re-assess their burn in and screening procedures.

(b) If constant failure rates are to be used for prediction purposes, then more realistic values can be obtained after the early life failure data has been removed. It is felt that such extrinsic values have more use to the system designer than the assumption of zero failure attributable to intrinsic failure.

(c) The type of data presented could not be derived by any accelerated test programme, and it is only through the study of the field behaviour of systems and subsystems that the real reliability behaviour of components can be established.

ACKNOWLEDGEMENTS

The work is being carried out with the support of the Procurement Executive, Ministry of Defence, and we are grateful for permission to publish this paper.

REFERENCES


FAILURE INTENSITY ANALYSIS OF ELECTRONIC COMPONENTS

J. A. Hayes, J. A. Jones, A. P. Schwarzenberger & D. S. Campbell

International Electronics Reliability Institute,
Department of Electronic & Electrical Engineering,
University of Technology,
Loughborough, Leicestershire LE11 3TU,
United Kingdom.

Tel.: 0509-222811; Fax: 0509-222854

ABSTRACT

The International Electronics Reliability Institute (IERI) at Loughborough
University of Technology in the U.K. has been collecting and analyzing reliability
information on electronic components and systems over a period of seven years. This has
involved a collaborative exercise with several major U.K. and European electronic
equipment manufacturers. This exercise, by nature of its detailed data collection at the
circuit board and system level allows IERI to undertake in depth analytical
studies into the field failure behaviour of the whole range of electronic components. At
its most simple level this allows the computation of hierarchical failure rates based on
dependable field failure data. The production and use for system level prediction of such
constant failure rate figures is a source of discussion and controversy in reliability journals.

The present paper undertakes to examine the validity of constant failure rate
assumptions with particular regard to a number of different component types. This has
been performed by means of failure intensity analysis. This technique involves division
of component failures into small discrete time intervals throughout system operating
periods and the calculation of instantaneous failure rates within these time intervals. Such
analysis, based on actual field behaviour, enables the early, useful and so-called "wear out"
periods of component behaviour to be investigated.
INTRODUCTION

The collection of electronic component reliability data from the field has been under way at IERI for a period of four years or so. Data has been collected at the system and subsystem level in order to obtain detailed data at the component level. This has involved several major British and Danish companies. The organisation and some preliminary results have been discussed in several papers [1,2,3].

The detailed nature of the data collected has meant that as more data becomes available for each component type, it is possible to investigate the time dependent reliability behaviour of a range of electronic components based on field failure information. This method of presenting component behaviour examines the traditional constant failure rate assumption for electronic components as used in existing models for reliability prediction [4,5,6].

The observed failure rate of electronic components can be considered to be made up of an intrinsic failure rate and an extrinsic failure rate. The intrinsic failure rate is dependent on the physical processes inherent within the device. Those physical processes which result in failure have such long time scales that they do not contribute significantly, if at all, to the failure rate of the device during its useful life. In fact, CECC/WG-Reliability have defined the intrinsic value during the working life as zero [7]. Wear out at the end of the useful working life is attributed to an increasing intrinsic failure rate.

Extrinsic failure rates are attributable to random events such as electrical overstress, ESD etc. These can also cover latent defects in the devices which manifest themselves throughout the useful life of the device. If the assumptions in [7] above are valid, then the extrinsic failures are the only ones observed during the so called useful life of the component. This assumes that all early or residual failures have been removed during burn-in.

Reliability prediction using constant failure rate models has come under much criticism recently [8,9]. However, in practice there does appear to be some commonality of published failure data from a variety of sources collected on similar equipments operating in similar environments [5,6].
The study of the time dependent failure behaviour of components operating in equipment in the field is important both from (a) a design point of view to determine if the so called 'early failures' are being removed successfully by screening or burn-in and (b) to investigate whether the constant failure rate assumption for so called extrinsic failures is in fact valid.

RESULTS

The results presented show the variation of failure intensity of selected components with time. Failure intensity is defined as:

$$\frac{n}{N \Delta t}$$  \hspace{1cm} (1)

where n is the observed number of failures in a given time period (\(\Delta t\)) and N is the population at risk during this period. For the purpose of this investigation the \(\Delta t\) interval has been chosen to be 1,000 hours. To illustrate further the means of computation of the failure intensity Figure 1 and Figure 2 show the number of failures of MOS digital integrated circuits with between \(10^3\) and \(10^4\) gates in 1,000 hour intervals, and the corresponding number of components at risk throughout the same time intervals. The shape of Figure 2 reflects the fact that as the number of systems going into the field increases with time, the populations of the various component types is also increasing.

The failure intensities are shown as central solid lines, and the dotted lines above and below are the 95% \(x^2\) confidence limits. These limits are dependent on the number of observed failures and the population at risk. Wide limits are indicative of a lack of failure data. This situation will improve with time as more data on the various component types becomes available. It should be noted that when the failure intensity drops to zero on the time axis, this means no failures have been observed in that particular 1,000 hour period.

Figure 3 shows the failure intensity versus time for CMOS digital devices with
Figure 1. Number of failures of MOS digital ICs with between $10^3$ to $10^4$ gates versus time

Figure 2. Number of MOS digital ICs with between $10^3$ and $10^4$ gates at risk in the field.

$10^3$–$10^4$ gates. The initial value decreases to a lower level over a 4,000 hour period. The values would seem to oscillate around the $5 \times 10^{-7}$ failure intensity value thereafter.

Figure 4 shows the failure intensity v. time for rectangular connectors. The failure intensity is observed to decrease within the first 2,000 hours, and thereafter remaining...
constant within reasonable approximation, although it is appreciated this statement does not take into account the confidence limits associated with the data.

Figure 4. Failure intensity curve for rectangular connectors

Figure 5 and Figure 6 show the failure intensity versus time graphs for pn-junction diodes and bipolar transistors respectively. The failure intensity for the diodes does not
appear to reach a steady state until approximately 6,000 hours when the value becomes low (i.e. less than $2.5 \times 10^{-9}$). The bipolar transistor exhibits the initial failure intensity decrease within 2,000 hours.

![Figure 5](image)

**Figure 5.** Failure intensity curve for pn-junction diodes

![Figure 6](image)

**Figure 6.** Failure intensity curve for bipolar transistors

Figure 7, for wirewound resistors, shows after an early failure period, zero failure intensities i.e. no failures are observed after 10,000 hours.
Figure 7. Failure intensity curve for wirewound resistors

In Figure 8 for carbon film potentiometers, observed zero failure intensities are punctuated by occasional peaks which are representative of very small numbers of failures as demonstrated by the wide confidence limits. It may be postulated that these are extrinsic failures due to the nature of this device.

Figure 8. Failure intensity curve for carbon film potentiometers.
Very low failure intensities, generally less than $5 \times 10^{-9}$ are observed in the cases of ceramic multilayers and tantalum sintered solid electrolytics, shown in Figure 9 and Figure 10 respectively. Each peak corresponds to a very small number of observed failures in any 1,000 hour interval.

Figure 9. Failure intensity curve for ceramic multilayer capacitors

Figure 10. Failure intensity curve for tantalum sintered solid electrolyte capacitors
The data presented in Figure 3 to Figure 10 is pooled data comprising of data from a variety of sources and environments. It has been presented in this manner as there is insufficient data as yet in the database in every environment for every component to present meaningful results. However, at this moment it is possible to present failure intensity behaviour with time for bipolar transistors in ground benign and ground mobile environments. These are shown in Figure 11 and Figure 12.

![Graph](image)

**Figure 11.** Failure intensity curve for bipolar transistors in a ground mobile environment

A continuing decrease to low values of failure intensity is observed in the ground mobile environment. However, in the ground benign environment the wide confidence limit does not allow of any significant statement to be made at this stage, although it does appear to have a similar shape to the pooled data.
CONCLUSIONS

It has been demonstrated from the results presented that:

(a) Early life failures are still occurring in the field even after burn-in for the components presented. These effects are being observed to date on some components up to the first 6,000 hours of field operation. This should alert system designers to re-assess their burn-in and screening procedures.

(b) If constant failure rates are to be used for prediction purposes, then more realistic values can be obtained after the early life failure data has been removed. It is felt that such extrinsic values have more use to the system designer than the assumption of zero failure attributable to intrinsic failure.

(c) The type of data presented could not be derived by any accelerated test programme, and it is only through the study of the field behaviour of systems and subsystems that the real reliability behaviour of components can be established.

ACKNOWLEDGEMENTS

The work is being carried out with the support of the Procurement Executive, Ministry of Defence, and we are grateful for permission to publish this paper.
REFERENCES


RELIABILITY BEHAVIOUR OF ELECTRONIC COMPONENTS

AS A FUNCTION OF TIME

D. S. Campbell, J. A. Hayes, J. A. Jones & A. P. Schwarzenberger

International Electronics Reliability Institute,
Department of Electronic & Electrical Engineering,
University of Technology,
Loughborough, Leicestershire LE11 3TU,
United Kingdom.

Tel.: 0509-222811; Fax: 0509-222854

ABSTRACT

Considerable attention has been paid recently to the behaviour of the hazard rate function h(t) as a function of time. This discussion has led to a lot of speculation as to the shape of the traditional 'bath-tub' curve showing failure or hazard rate as a function of time. Studies at the International Electronics Reliability Institute (IERI) at Loughborough University of Technology, in the United Kingdom, have recently been examining in detail the database created from field failure returns on a wide spectrum of electronic components. These components are in equipments subject to a spread of environmental conditions. The database created has been exercised with particular reference to the behaviour of MOS Ics, rectangular connectors, bipolar transistors and pn-junction diodes. Data has been analyzed using pooled information from a wide variety of sources and also from two specific environmental conditions, ground benign and ground mobile.

The results of this analysis are presented and the failure intensities as a function of time are given at 1,000 hour intervals up to a total time of 21,000 hours. Confidence limits at the 95% $\chi^2$ level are also given. The results show a rapidly falling failure intensity for the first few thousand hours and after this time the failure intensity appears to be relatively constant given the accuracy obtainable with the data available.

BACKGROUND

The collection of electronic component reliability data from the field has been under way at IERI for a period of four years or so. Data has been collected at the system and subsystem level in order to obtain detailed data at the component level. This has involved several major British and Danish companies. The organisation and some preliminary results have been discussed in several papers.1,2,3

The detailed nature of the data collected has meant that as more data becomes available for each component type, it is possible to investigate the time dependent reliability behaviour of a range of electronic components based on field failure
information. This method of presenting component behaviour examines the traditional constant failure rate assumption for electronic components as used in existing models for reliability prediction\(^4\).\(^5\)\(^6\).

The observed failure rate of electronic components can be considered to be made up of an intrinsic failure rate and an extrinsic failure rate. The intrinsic failure rate is dependent on the physical processes inherent within the device. Those physical processes which result in failure have such long time scales that they do not contribute significantly, if at all, to the failure rate of the device during its useful life. In fact, CECC/WG-Reliability have defined the intrinsic value during the working life as zero\(^7\). Wear out at the end of the useful working life is attributed to an increasing intrinsic failure rate.

Extrinsic failure rates are attributable to random events such as electrical overstress, ESD etc. If the assumptions in [7] above are valid, then the extrinsic failures are the only ones observed during the so called useful life of the component. This assumes that all early or residual failures have been removed during burn in.

Reliability prediction using constant failure rate models has come under much criticism recently\(^8\)\(^9\). However, in practice there does appear to be some commonality of published failure data from a variety of sources collected on similar equipments operating in similar environments\(^5\)\(^6\).

The study of the time dependent failure behaviour of components operating in equipment in the field is important both from (a) a design point of view to determine if the so called 'early failures' are being removed successfully by screening a burn in and (b) to investigate whether the constant failure rate assumption for so called extrinsic failures is in fact valid.

RESULTS

The results presented show the variation of failure intensity of selected components with time. Failure intensity is defined as:

\[
\frac{n}{N \Delta t}
\]

where \(n\) is the observed number of failures in a given time period (\(\Delta t\)) and \(N\) is the population at risk during this period. For the purpose of this investigation the \(\Delta t\) interval has been chosen to be 1,000 hours. To illustrate further the means of computation of the failure intensity Figs. 1 and 2 show the number of failures of MOS digital integrated circuits with between 10\(^3\) and 10\(^4\) gates in 1,000 hour intervals, and the corresponding number of components at risk throughout the same time intervals. The shape of Fig. 2 reflects the fact that as the number of systems going into the field increases with time, the populations of the various component types is also increasing.

Figures 3 to 6 show failure intensity versus time for the following components rectangular connectors, digital CMOS with 10\(^3\) – 10\(^4\) gates, pn-junction diode and bipolar
Component operating time [hours]

Figure 1: Number of failures of MOS digital ICs with between $10^3$ to $10^4$ gates versus time

Component operating time [hours]

Figure 2: Number of MOS digital ICs with between $10^3$ and $10^6$ gates at risk in the field.

transistor decreases.

The failure intensities are shown as the central solid lines, and the dotted lines above and below are the 95% $\chi^2$ confidence limits. These limits are dependent on the number of observed failures and the population at risk. Wide limits are indicative of a lack of failure data. This situation will improve with time as more data on the various component types becomes available. It should be noted that when the failure intensity drops to zero on the time axis, this means no failures have been observed in that particular 1,000 hour period.

Figure 3 shows the failure intensity v. time for rectangular connectors. The failure
intensity is observed to decrease within the first 2,000 hours, and thereafter remaining constant within reasonable approximation, although it is appreciated this statement does not take into account the confidence limits associated with the data.

Figure 3: Failure intensity curve for rectangular connectors

Figure 4 shows the failure intensity versus time for CMOS digital devices with $10^3 - 10^4$ gates. Again, the initial values decrease to a lower level over a 4,000 hour period. The values would seem to oscillate around the $5 \times 10^7$ failure intensity value thereafter.

Figure 4: Failure intensity curve for digital MOS ICs with between $10^3$ and $10^4$ gates

Figures 5 and 6 show the failure intensity versus time graphs for pn-junction diodes and bipolar transistors respectively. The failure intensity for the diodes does not appear to reach a steady state until approximately 6,000 hours when the value becomes low (i.e.
less than $2.5 \times 10^9$). The bipolar transistor exhibits the initial failure intensity decrease within 2,000 hours.

![Figure 5: Failure intensity curve for pn-junction diodes](image)

![Figure 6: Failure intensity curve for bipolar transistors](image)

The data presented in Figs. 3-6 is pooled data comprising of data from a variety of sources and environments. It has been presented in this manner as there is insufficient data as yet in the database in every environment for every component to present meaningful results. However, at this moment it is possible to present failure intensity behaviour with time for bipolar transistors in ground benign and ground mobile environments. These are shown in Figs. 7 and 8.
A continuing decrease to low values of failure intensity is observed in the ground mobile environment. However, in the ground benign environment the wide confidence limit does not allow of any significant statement to be made at this stage, although it does appear to have a similar shape to the pooled data.

CONCLUSIONS
It has been demonstrated from the results presented that:

(a) Early life failures are still occurring in the field even after burn in for the components presented. These effects are being observed to date on some components up to the first 6,000 hours of field operation. This should alert system designers to re-assess their burn in and screening procedures.

(b) If constant failure rates are to be used for prediction purposes, then more realistic values can be obtained after the early life failure data has been removed. It is felt that such extrinsic values have more use to the system designer than the assumption of zero failure attributable to intrinsic failure.

(c) The type of data presented could not be derived by any accelerated test programme, and it is only through the study of the field behaviour of systems and subsystems that the real reliability behaviour of components can be established.

ACKNOWLEDGEMENTS

The work is being carried out with the support of the Procurement Executive, Ministry of Defence, and we are grateful for permission to publish this paper.

REFERENCES


EVALUATION OF RELIABILITY PREDICTION METHODOLOGIES

M. Zahid, J.A. Jones and J.A. Hayes

International Electronics Reliability Institute
Department of Electronic and Electrical Engineering
Loughborough University of Technology
Leicestershire, LE11 3TU, United Kingdom

Tel: 0509-222849 Fax: 0509-222854

ABSTRACT

One of the most controversial techniques used at present in the field of reliability is the use of reliability prediction methodologies based on component failure data for the estimation of system failure rates. This paper is an attempt to investigate some of the available prediction models. A number of boards were selected from the International Electronics Reliability Institute's (IERI) electronic components field failure database and their reliability was predicted and compared with the actual observed performance in the field.

The techniques chosen have been MIL-217E, HRD4, Siemens (SN29500), CNET, and Bellcore (TR-TSY-000332). For each technique the associated published failure rates have been used. Parts count analysis has been performed on a number of boards from the aforementioned database and this is compared with the failure rate observed in the field. The prediction values were seen to differ greatly from the observed field behaviour and from each other. Further analysis of the methods was performed and a widely differing dependence upon physical parameters was observed.

This suggests that predictions obtained by differing models can't be compared and that great care should be taken when selecting a prediction technique since the physical parameters of the system will affect the prediction obtained in possibly unforeseen ways.

BACKGROUND

IERI at Loughborough University have been collecting a large amount of failure information over the past several years from leading British and Danish electronic manufacturing companies. This data is of high quality so IERI are able to perform reliability analysis and compare the predicted with the field failure rate.

The concept of electronic failure prediction methodology (EFPM) often affects major decisions in system design. EFPM is based on the assumption that systems fail as a result of failures of component parts, and those parts fail partly as a result of exposure to application stress. This means that by some consideration of the structure of such a piece of equipment and by further consideration of its usage it is possible to obtain an estimate of the systems reliability in that particular application.

There are many reasons why this task may be necessary. These include feasibility evaluation where the compatibility of a design concept is weighed against the reliability requirements for acceptance, and where different parts of a system can be compared and any necessary trade off of cost, reliability, weight etc. can be made. Further uses are for the identification of potential reliability problems and as reliability input into other tasks such as maintainability analysis, testability evaluations and FMECA.

There are two slightly different approaches to EFPM involving differing amounts of information about the system. The first is known as parts count analysis and requires comparatively little information. This takes the parts list for a particular design and bases the reliability estimate on the number of components used in it without any reference to operating conditions of these components. This method is generally used early in the design phase to obtain a simple estimate of the system reliability. The second method is known as parts stress analysis and involves knowledge of a wealth of information about the system but is assumed to provide more realistic estimate of the reliability. This second method tends to be used towards the end of the design cycle when actual circuit parameters have been established.

Reliability prediction methods are widely accepted throughout the electronics industry (this in itself is the source of much debate!). This enables it to be used as a general yardstick which allows comparison between different equipments to be made.
However many manufacturers have commented that the models can be wildly inaccurate when compared with the performance in the field, particularly in the case of modern microelectronic devices, and their use can lead to increased costs and complexity while deluding engineers into following a flawed set of perceptions and leaving truly effective reliability improvement measures unrecongnised.

This paper is concerned only with the parts count approach since information about circuit conditions is not available within the IERI database. In effect what is stored is the parts lists for a large number of boards that are operating out in the field. Information on the failure of these boards is also stored which allows the comparison of the predicted reliability with the actual observed reliability.

OVERVIEW OF PREDICTION METHODOLOGIES

Each prediction method chosen is described briefly below.

A) BELLCORE

This method is defined in Bellcore TR-TSY-000332 and was developed by Bell communications research for use by the electronics industry so that they would be aware of Bellcore’s view of generic requirements for reliability prediction procedures of electronic equipments. The data presented is based upon field data, laboratory tests, MIL-HDBK-217E, device manufacturers data, unit suppliers data or engineering analysis.

The failure rate of a system is assumed to be equal to the sum of the device failure rates. Modifiers are included to account for variations in equipment operating environment, quality, and device applications conditions such as temperature and electrical stress.

B) CNET (SIMPLIFIED)

This method is defined in Recueil de Donnees de Fiabilite du CNET and was developed by French National Centre of Telecommunications. The data used comes from the exploitation of breakdowns of the equipment used by the military and civil administration, and other equipment and component manufacturers. This handbook forms a common base intended to make reliability predictions uniform in France.

The failure rate of a system is assumed to be equal to the sum of the device failure rates when modified by various II factors.

This method is known as the simplified method since it is taken from the English translation of the full CNET document which is only available in French. The translated document was supplied by British Telecom.

C) HRD4

This method is defined in Handbook of Reliability data from components used in telecommunications and was developed by British Telecom materials and components centre for use by designers and users of electronic equipments so that there exists a common basis for system reliability prediction.

The generic failure rates given in the handbook are estimates of the upper 60% confidence levels based upon, wherever possible, data collected from the in-service performance of the equipment installed in the U.K inland telecommunications network. Where such data is not available for particular components, alternative sources or estimated values have been employed and the status of the source indicated by a letter code. The failure rate of a system is assumed to be equal to the sum of the device failure rates when modified by various II factors.

D) MIL-217

Mil-Hdbk-217 was developed by the U.S department of defence with the assistance of the military departments, federal agencies, and industry for use by the electronic manufacturers supplying to the military. The handbook describes two methods, namely parts count, and parts stress, which are used to predict the reliability of electronic components, systems, or subsystems in different stages of the design. The failure rates given in the handbook have in the main been derived from test bed and accelerated life studies. The failure rate of a system is assumed to be equal to the sum of the device failure rates when modified by various II factors.

E) SIEMENS

This method is defined in SN29500 and was developed by Siemens AG for the use of Siemens and Siemens associates as a uniform basis for reliability prediction.
The standard presented in the document is based on failure rates under specified conditions. These failure rates are determined from applications and testing experience taking external sources (e.g., MIL-217) into consideration. This includes failure rates and conditions to which the statement of component failure rate is referred (i.e., reference conditions). Components are categorized into many different groups each of which has a slightly different reliability model.

**ANALYSIS**

Six different board designs were selected from the IERI-CORD database. These board designs were chosen to have a large coverage of component types and are from different applications. Table 1 describes the environments and applications of the boards.

<table>
<thead>
<tr>
<th>Board</th>
<th>Environment</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Ground Mobile</td>
<td>Radio System</td>
</tr>
<tr>
<td>2</td>
<td>Ground Mobile</td>
<td>Radio System</td>
</tr>
<tr>
<td>3</td>
<td>Ground Benign</td>
<td>Telephone Exchange</td>
</tr>
<tr>
<td>4</td>
<td>Ground Benign</td>
<td>Telephone Exchange</td>
</tr>
<tr>
<td>5</td>
<td>Ground Mobile</td>
<td>Command System</td>
</tr>
<tr>
<td>6</td>
<td>Ground Mobile</td>
<td>Command System</td>
</tr>
</tbody>
</table>

Table 1: Brief descriptions of the boards

The reliability of the boards was calculated using all the aforementioned models. Table 2 shows the percentage deviation of the predicted values from the observed field failure rate for one of the circuit boards analyzed. For a matter of brevity, only the results of one board are shown.

<table>
<thead>
<tr>
<th>Model</th>
<th>Failure Rate (Fits)</th>
<th>Deviation (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bellcore</td>
<td>23735</td>
<td>455</td>
</tr>
<tr>
<td>CNET</td>
<td>26050</td>
<td>509</td>
</tr>
<tr>
<td>HRD4</td>
<td>9408</td>
<td>120</td>
</tr>
<tr>
<td>Mil-hdbk-217</td>
<td>3849</td>
<td>-10</td>
</tr>
<tr>
<td>Siemens</td>
<td>2345</td>
<td>-45</td>
</tr>
</tbody>
</table>

Table 2: Failure rates of boards according to various models

However, similar analyses were performed for all six boards. Figure 1 shows the percentage deviation from the observed field failure rate for six board designs selected from the IERI-CORD database.

The predictions not only differ widely between the various models but they also differ greatly from the observed field failure rate. The models are not always consistent in the deviation from this observed field value, they can be optimistic in some cases while pessimistic in others. This suggests that there are some underlying factors that are causing divergence of the models.

In order to investigate this apparent inconsistency in the various models, it is necessary to look at each model carefully and analyze the way in which the predicted failure rate is influenced by the various parameters that influence system performance.

In order to examine the variations observed in Figure 1, the sensitivity of the system predicted failure rate to various parameters is investigated. This is done by varying temperature, quality, stress, and environment in turn while keeping the others at typical or nominal values.

The results are presented graphically and show percentage variation of predicted failure rate from the nominal value while a single parameter is varied within the limits that the models allow. The largest spread shows the parameter that has the greatest effect on the model's prediction. Care should be taken in some cases where the effect is accentuated by a highly non-linear variation in one of the parameters' II factors. This is particularly true when the parameters are discrete, as in the case of environment, where selection of a particular environment can cause a large change in the associated II factor. The degree of non-linearity can be...
demonstrated using graphs of normalised stress versus deviation where normalised stress is defined as the ratio between members of the range of available IT factors and the nominal IT factor value for the parameter under investigation.

A) BELLCORE MODEL

The percentage deviation in the board failure rate from the nominal with respect to different stress levels using the BELLCORE methodology is shown in Figure 2.

![Figure 2: Deviation of board failure rate in the BELLCORE model.]

As can be seen from the figure the allowed variation in the electrical stress makes the largest difference to the calculated failure rate.

The non-linear nature of the effect of varying electrical stress and temperature are shown in Figure 3. This shows that the BELLCORE model is based upon an Arrhenius style acceleration formula which is reflected in this non-linearity.

B) CNET MODEL

Figure 4 shows the percentage deviation in the board failure rates with respect to different stress levels using the CNET(simplified) methodology. As can be seen from the figure, the range in quality factor has the largest influence on the calculated failure rates.

![Figure 4: Deviation of board failure rate in the CNET model.]

The quality factor used in this model is based upon a set of discrete quality bands over which the IT factor is defined. This means that the quality factor is a stepwise non-linear function of the device quality.

C) HRD4 MODEL

The percentage deviation in the board failure rates with respect to different stress levels using the HRD4 methodology is shown in Figure 5.

![Figure 5: Deviation of board failure rate in the HRD4 model.]

As can be seen from the figure, it is the allowed range of quality factors that makes the largest difference to the calculated failure rates. The quality factor used in this model is based upon a set of discrete quality bands over which the IT factor is defined. Again this means that the quality factor is a stepwise non-linear function of the device quality.
D) MIL-217 MODEL

Figure 6 shows the percentage deviation in the board failure rates with respect to different stress levels using the MIL-217E methodology. The figure shows that it is the allowed variation in environment factors that causes the largest difference in the calculated failure rates.

The non-linear nature of the electrical stress and temperature effects are shown in Figure 9. This means that the Siemens model is based upon an Arrhenius style acceleration formula which is reflected by this non-linearity.

E) SIEMENS MODEL

The percentage deviation in the board failure rates with respect to different stress levels using the Siemens methodology is shown in Figure 8. As can be seen from the figure it is the range of temperature factor that causes the largest difference in the calculated failure rates.

CONCLUSIONS

The results are summarised in Table 3. It is evident that some of the models are more sensitive to a II factor that varies according to an Arrhenius model, such as temperature and electrical stress, while others are more sensitive to the discrete II factors used to model environment and quality.
I'll talk more about the models' sensitivity.

<table>
<thead>
<tr>
<th>Prediction Model</th>
<th>Greatest sensitivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bellcore</td>
<td>Electrical Stress</td>
</tr>
<tr>
<td>CNET(Simplified)</td>
<td>Quality</td>
</tr>
<tr>
<td>HRD4</td>
<td>Quality</td>
</tr>
<tr>
<td>MIL-217E</td>
<td>Environment, Quality</td>
</tr>
<tr>
<td>Siemens</td>
<td>Temperature</td>
</tr>
</tbody>
</table>

Table 3: Models' Sensitivity

It is not surprising that direct comparisons of the models result in wide variations. Although the models are based upon the same criteria there is disagreement about the effects the different parameters have on the failure rate.

Also by carefully examining the models, it was observed that although the quality levels are clearly defined within each procedure, it is extremely difficult to find a quality level description that is compatible across all models. In addition every organisation has developed their model according to the experience they have obtained in the field and have tailored the model to meet their specific needs.

It is inadvisable to compare the models' prediction with field performance unless the systems used are manufactured according to the guidelines and procedures that are specified by the model designers. Under such circumstances the system manufacturers would argue that their models are suitable for reliability prediction.

Care must also be taken when comparing the predicted reliability with that observed in the field as prediction models assume a constant hazard rate. It has been observed however that early life failures do occur in the field even after system burn-in.

Even if the above considerations are taken into account, there is no guarantee that the field reliability will be the same as that predicted. This is due to the underlying reason that models are generally simple empirical approximations. Indeed it is postulated that because of this, their use should actively be discouraged. Moreover, they do not take into account many of the other critical factors such as vibrations, mechanical shock, etc.

ACKNOWLEDGEMENTS

This work has been carried out with the support of Professional Component Services (DRA), and we are grateful for their permission to publish this paper.

REFERENCES


ABSTRACT - Before any reliability improvement can be made to a product it is necessary to assess the current reliability in the field. This can be achieved is by collecting field failure information throughout a products life and performing analysis on this data. This reliability analysis enables the manufacturer to assess and improve the dependability of the product as well as improving the performance of the next generation of products. The aim of this paper is to guide the would-be data collector into the correct methodology for data collection and to give some examples of the kind of analysis possible when sufficient data has been collected. A description of the equipment behaviour model used at International Electronics Reliability Institute (IERI) is given and examples of some forms of analysis and the sort of conclusions that can be drawn are presented.

1. INTRODUCTION

For a number of years the IERI have been collecting field reliability information from a number of British and Danish electronic equipment manufacturers and have accrued a database that contains information on a large cross-section of electronics systems and components. The database contains information about the number of components currently in use in the field. Using this information it is possible to calculate the actual operating time for any component, board or equipment. When failures occur all relevant information is used in conjunction with the population information to allow IERI to build a complete reliability picture of each equipment studied. By analysing this wealth of information it is possible for IERI to provide valuable insight into the reliability of the of electronic systems, sub-systems and components.

This reliability study is performed using a number of analysis techniques which can highlight potential reliability or operational problems. These analysis procedures allow the assessment of manufacturing practises, the comparison of system reliability with past experience, and the identification of possible future reliability problems. The companies involved in this study have found the data analysis a useful input into their own in-house reliability work.
2. DATA MODEL

In order to design a database based upon electronic equipment operating in the field, a model of how equipments are structured is required. The simplest model is to assume a hierarchical structure. Equipment can be considered to contain a number of circuit boards. These circuit boards contain a number of components. This means that in order to understand the way a component behaves information about the circuit board the component is mounted on is required. Then in order to fully understand the circuit board information about the equipment the board is operating in also required. This is illustrated in Fig 1.

![Diagram of equipment structure](image)

**Fig 1:** A model of the structure of electronic equipment

This gives us a simple model of equipment which is flexible enough to deal with most forms of equipment. The boundaries between what is meant by equipment, board and component are vague and the terms can sometimes be interchanged. Specifically it is possible to consider a board to be an equipment or a board to be a component if this is what is necessary to model the actual equipment.

Another consideration is how an equipment behaves when operating in the field. Most modern equipments can be repaired by exchanging circuit boards. This means that the engineer will remove a particular board from a system, insert a replacement, and return the faulty board to the workshop for testing and repair. The repaired board will then be placed in storage and used at some later date. It may not go back into the system it came from. This is illustrated in Fig 2.

Board trackability, the ability to follow a board between systems, is important, particularly from an analysis point of view, since it allows much more detailed investigations into the behaviour of systems.

In order to achieve full board trackability, and hence component trackability, it is necessary to be able to uniquely identify every component existing in the field. In order to do this a component descriptor is defined. The most general case is given in Table 1.
Throughout the life of a component, the first three elements of this descriptor may change as the circuit board the component is mounted on migrates through a number of systems. The second set of three are constant throughout a component's operating life until it fails or the circuit board is scrapped. Since component are non-repairable items and hence when a component fails it is of no more interest except as to what the failure modes and mechanisms were.

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equipment manufacturer</td>
<td>This must uniquely identify the manufacturer within the database system</td>
</tr>
<tr>
<td>Equipment type</td>
<td>This must be unique for this equipment type for this manufacturer.</td>
</tr>
<tr>
<td>Equipment serial number</td>
<td>This must be unique for this instance of an equipment of this type supplied by this manufacturer</td>
</tr>
<tr>
<td>Circuit board manufacturer</td>
<td>This must uniquely identify this manufacturer within the database.</td>
</tr>
<tr>
<td>Circuit board type</td>
<td>This must be unique for this circuit board type for this manufacturer.</td>
</tr>
<tr>
<td>Circuit board serial number</td>
<td>This must be unique for this instance of an circuit board of this type supplied by this manufacturer</td>
</tr>
<tr>
<td>Component reference position</td>
<td>This must be unique for a particular circuit board type.</td>
</tr>
</tbody>
</table>

Table 1: Definition of component descriptor
Further information on the data structure and analysis techniques can be found in [Haye 90],[Bryd 90] and [Mars 90].

3. DATA ANALYSIS

This section outlines the analyses that can be performed with the data stored within the IERI database. It should be noted that this is not an exhaustive list and there are many other avenues of analysis which are possible. The techniques presented have been chosen to highlight the qualitative nature of the analysis and their application to engineering design.

3.1. Design fault analysis

Some design faults are recurring faults along similar component descriptors. That is to say repeated failures occurring in the same reference position on a set of circuit boards of the same type. It can be identified by simply counting the occurrences of the same circuit reference positions in the failure table. Fig 3 demonstrates the output of such an analysis technique.

![Design fault analysis on a particular circuit board design](image)

As can be seen from the figure the component in reference position Q26 has failed a larger number of times than the other components. This may suggest that there is a problem with that particular circuit reference position in terms of this board design. Further investigation is now necessary to discover the cause of this bias towards the Q26 position. This form of analysis can be performed routinely with a field failure database and can help designers, when the cause of a problem is found.

3.2. Parts count analysis

Parts count analysis is one method by which the reliability of a new system can be estimated from previous experience. Data collected on analogous equipment can be used to predict the failure rate of a new system operating in the same environment. There are many reliability prediction models for doing this but care should be taken when using those approaches [Zahi 93]. It is always better to collect data from applications and equipment where the failure rates are going to be directly applicable to the new system. Fig 4 shows the parts
count failure rate for a system predicted from the field with the actual observed field failure rate. Also shown are the predictions made using a number of the different models.

IERI do not support this form of analysis for predicting system reliability but it is a useful technique when used as part of the design process.

3.3. Sub-system and no fault found analysis

This technique examines the way in which systems or circuit boards fail. The order in which failure occurs may be important in terms of the relationships between components on a board.
For example in Fig 5 the failure patterns are shown for a number of circuit boards. The codes in each box represent the component code as stored in the IERI database. Each box represents one failure event. The code NFF is an abbreviation for no fault found and it is interesting to note that for the first circuit board shown a large number of NFFs have occurred. This board has also, in the middle of the NFF chain, has a pair of Hybrid (H) failures. It is possible that these replacements were made while seeking the cause of the NFF and are not true failures since they did not remove the failure problem. Further investigation is obviously necessary in this case. The circuit board with the serial number '300042' also shows an interesting pattern, again the NFF occur with hybrids. This may suggest that there is problem with the testability of the hybrids since it suggests that there is a relationship between the hybrid device, which is a very complex devices, and the NFF problem.

This sort of failure pattern leads to the question of whether or not the board should be removed from the field. It may be more cost effective to replace this board than to continue to repair it. This would depend upon the nature of the board and the economics of repair which, of course, are company dependent.

![Graph showing times to failure for various board serial numbers](image)

Fig 6: Times to failure for various board serial numbers

It is possible that all the failures after the first have been induced by the repair process and to analyze whether this is the case the failure times are examined and are shown in Fig 6. This figure shows that for the board with serial number '12' the repair improves the reliability since the time between failure is extended. However this is not the case for the board with serial number 300042 since after the second repair the times to failure accelerate. This situation should be looked at in more detail to assess the viability of the repair process.

### 3.4. Failure intensity analysis

The theory and methodology behind this technique was described in [Camb 92]. Fig 7 shows the observed failure intensity for a circuit board. The graph shows the calculated failure
intensity (The solid line) and the upper and lower confidence limits (The dashed and dotted line respectively).

The shape of this graph suggests that there is an early failure period that occurs before 5000 hours that has not been successfully removed by board burn in. The failures that occur in this period could well be early life component failures that should have been removed by part screening. By investigating the structure of this board it would be possible to discover which components are responsible for this early life failure by plotting their failure intensity curve.

![Failure intensity curve for a system with predicted curve](image)

Fig 7: Failure intensity curve for a system with predicted curve

The various peaks and troughs in the curve would have to be investigated in detail to discover if they are significant. The operating time after 18000 hours appears to be failure free which may suggest that the earlier fluctuations in the failure intensity are due to early component problems.

3.5. M(t) Analysis

The technique of M(t) analysis is described in [Mart 90]. An example m(t) curve is given in Fig 8

Examination of this curve allows an estimate of the percentage of weak systems in the population to be determined in a relatively short operational time and corrective action to be taken. The M(t) analysis can also be used to predict the eventual steady state failure intensity of a system.

3.6. Other analysis techniques.

The data collected from field reliability studies can be used for many other forms of analysis, such as proportional hazards and intensity analysis, component reliability model analysis and equipment structural analysis.
4. CONCLUSION

The ultimate aim of collecting field reliability data must be to assess the reliability performance of a particular equipment design. Problems in manufacturing, design or screening can be identified in the collected data and corrective action can be taken.

This paper has attempted to show the value of collecting field data. It has demonstrated a simple electronic equipment model and has shown some of the different kinds of analysis that are possible.

5. ACKNOWLEDGEMENTS

This work has been carried out with the support of Professional Component Services (DRA), and we are grateful for their permission to publish this paper.

6. REFERENCES


