ON THE DETERMINATION OF FUNCTIONAL REQUIREMENTS IN A MAINTENANCE ENVIRONMENT

Harry H. Martin
On the determination of functional requirements in a maintenance environment

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de Rector Magnificus, prof. dr. J.H. van Lint, voor een commissie aangewezen door het College van Dekanen in het openbaar te verdedigen op 4 november 1994 om 16.00 uur

door

HARRY H. MARTIN
geboren te Winterswijk
Subject headings: maintenance management / standard software packages.
Table of contents

CHAPTER 1: INTRODUCTION
1.1. The research subject in this study 1
1.2. The complexity of the maintenance function 1
1.3. Information systems in maintenance
   1.3.1 General developments 3
   1.3.2 Information system development from computer scientists perspective 5
1.4. Choices made in this study 7
1.5. The results of this study 8
1.6. The structure of the thesis 8

CHAPTER 2: ANALYSIS OF CURRENT SOFTWARE EVALUATION PRACTICES
2.1. Introduction 13
   2.1.1 Make or buy ? 13
2.2. The evaluation framework 15
   2.2.1 Nomenclature 15
   2.2.2 The parties involved 17
   2.2.3 Types of relationships 19
   2.2.4 Consequences of party-relationships 21
2.3. Evaluation and selection phases 22
2.4. The evaluation principle 25
   2.4.1 Introduction 25
   2.4.2 Evaluation steps 26
2.5. Evaluation methods 28
   2.5.1 Introduction 28
   2.5.2 The certification method 28
   2.5.3 The checklist 29
   2.5.4 The ranking approach 31
   2.5.5 Questionnaires 31
   2.5.6 Conclusion 32
2.6. Executing an evaluation in practice 33
2.7. Conclusion 35

CHAPTER 3: THIS STUDY'S METHODOLOGY
3.1. Introduction 39
### Contents

3.2. Ambiguity of functional requirements .............................................. 39
3.3. Evaluation techniques ................................................................. 41
3.4. Methodology .................................................................................
   3.4.1 Objective of this thesis ......................................................... 42
   3.4.2 Research methodology ......................................................... 45
3.5. On industrial engineering design methodology ................................. 46
3.6. The status of this study ...................................................................
   3.6.1 Design criteria and constraints ............................................. 49
   3.6.2 Synthesis of applied sciences (prototype design) ..................... 51
   3.6.3 The case study ............................................................... 52
   3.6.4 The prototype acceptance evaluation .................................. 53
3.7. The determination of functional requirements in an industrial engineering context ......................................................... 54
   3.7.1 Functional requirements in maintenance ............................. 54
3.8. Developing a functional analysis and design procedure ..................... 57

### CHAPTER 4: MONO-DISCIPLINARY ASPECTS OF FUNCTIONAL ANALYSIS FOR MAINTENANCE

4.1. Introduction .................................................................................... 59
4.2. Design requirements from a organisation function specialist perspective 59
4.3. The personnel system design requirements .................................... 60
   4.3.1 The integrated organisation redesign approach .................... 61
   4.3.2 The organisation function as a cohesion factor .................... 62
4.4. The information system analysis and design requirements ............... 63
   4.4.1 The information system infrastructure ................................ 63
   4.4.2 Formalisation of specifications ............................................ 70
   4.4.3 Individual and user group preferences ................................. 72
   4.4.4 Conclusions on the information systems development and selection aspect ......................................................... 72
4.5. The organisation design processes ................................................. 73

### CHAPTER 5: THE PRINCIPLES OF FUNCTIONAL ANALYSIS AND DESIGN

5.1. Introduction ................................................................................... 77
5.2. The definition of functional requirements ....................................... 77
   5.2.1 A "mechanistic" approach .................................................... 77
   5.2.2 The "function" definition ..................................................... 79
On the determination of functional requirements in a maintenance environment

5.3. Elements of an FAD-process
   5.3.1 Justification of functions
   5.3.2 The validity of a function
   5.3.3 The completeness of functions
   5.3.4 The decomposition principle
   5.3.5 The decomposition criterion
   5.3.6 The standard function
   5.3.7 The evaluation function

5.4. The functional analysis and design process
   5.4.1 The distinction between structural and detailed analysis and design

5.5. The structural analysis and design in relation to the personnel system analysis and design

5.6. The detailed functional analysis and design in relation to the information systems analysis and design

CHAPTER 6: STRUCTURAL DECOMPOSITION CRITERIA FOR MAINTENANCE

6.1. Introduction
6.2. A maintenance literature review
6.3. Elementary maintenance situations
6.4. Decoupling as a means to decompose maintenance functions
   6.4.1 Basic considerations with respect to decoupling
   6.4.2 Basic considerations with respect to the maintenance concept
   6.4.3 Basic considerations with respect to maintenance capacity
   6.4.4 Considerations with respect to the evaluation function
6.5. Compound maintenance situations
6.6. Conclusion

CHAPTER 7: DESIGN AND DEMONSTRATION OF A FUNCTIONAL ANALYSIS AND DESIGN FRAMEWORK FOR MAINTENANCE

7.1. Introduction
7.2. The personnel system analysis and design objective: the creation of semi-autonomous groups
7.3. Using structural functional analysis and design decomposition criteria.
   7.3.1. Determining guidelines for the personnel system analysis and design process

III
## Contents

7.3.2. The formation of operational groups 118  
7.3.3. Temporary group structures 119  
7.4. Determination of structural analysis and design functions per group 119  
  7.4.1. The initiation of the detailed functional analysis and design process 119  
7.5. The case-study: The PTT Post Maintenance service 122  
  7.5.1. Objective of the case study 122  
7.6. The 's-Hertogenbosch hub maintenance unit 123  
  7.6.1. The general PTT structure 123  
7.7. The production function of a main sorting centre 125  
7.8. Information sources used for the functional analysis and design process 126  
  7.8.1. The maintenance concepts 126  
  7.8.2. The BISTOD information system 127  
7.9. Determining functional units 128  
7.10. The determination of structural functional analysis and design functions 131  
7.11. Options to improve hub maintenance 132  
7.12. Conclusion 136  

CHAPTER 8: THE APPLICATION OF FUNCTIONAL REQUIREMENTS IN EVALUATION PRACTICES  
8.1. Introduction 139  
8.2. The completeness criterion 139  
8.3. The original evaluation approach 140  
8.4. The enhanced evaluation approach 142  
  8.4.1 Compensating functional mismatches 144  
  8.4.2 The enhanced evaluation procedure 146  

CHAPTER 9: CONCLUSIONS  
9.1. Multi-disciplinary insights 149  
  9.1.1 The objective of this study 149  
  9.1.2 The practicality of the framework 151  
9.2. Mono-disciplinary insights 154  
  9.2.1 Maintenance management 154  
  9.2.2 Sociotechnology 155  
  9.2.3 Computer science 156  

IV
On the determination of functional requirements in a maintenance environment

REFERENCES 159

APPENDIX I: WORK LOAD DISTRIBUTION GRAPHICS 167

APPENDIX II: EXAMPLE OF A CHECKLIST FORM 169

APPENDIX III: THE EUT MAINTENANCE MODEL 170

SUMMARY 173

SAMENVATTING (SUMMARY IN DUTCH) 175

CURRICULUM VITAE 177
1 Introduction

1.1 The research subject in this study

With the advent of computers as a potential tool for decision making in various business application areas much scientific attention has been devoted to the design of information systems. In this study we will concentrate on the assessment of the specification of requirements for management information systems for the internal maintenance function of organisations.

The technique for evaluating information systems is, in general, a fairly straightforward process. However, the determination of valid criteria for an evaluation is a difficult process for a complex organisational function such as the maintenance function. The criteria should be capable of reflecting the consequences of current or improved procedures and practices within a maintenance function of an actual organisation. The task of determining valid maintenance procedures and practices, so called functional requirements, specifically for the evaluation of information systems is, from a scientific point of view, still not solved. The primary objective of this study is to develop a method in the form of a framework of design rules which facilitate the determination of functional requirements for the internal maintenance function of an organisation.

1.2 The complexity of the maintenance function

The problem of evaluating the effectiveness of information systems support for the maintenance function lies inherently within the complexity of the maintenance function. Administrative functions such as bookkeeping are general and formal up to a high extent and are readily standard within a software package in such a situation. However, in maintenance, complexity is high because of the high degree of situation specific activities and because of a high demand for flexibility in maintenance management, which has to cope with the usually unpredictable occurrence of demand for maintenance activities with varying levels of urgency.
On the determination of functional requirements in a maintenance environment

Many factors combine to produce a complexity of tasks unique to the maintenance function where situation specific solutions are required. These include the types of technical systems and the way in which they are used by the production function, the actual maintenance staff with its skills, experience, personal background and their organisational responsibilities, the state-of-the-art in maintenance management knowledge and the information infrastructure. They all influence to a varying degree what procedures and practices are actually required at a certain moment in time. The diversity in factors also illustrates that improving maintenance performance can hardly be achieved within an academic mono-discipline.

In recent years there seems to be a growing scientific interest in maintenance issues on an international scale. Several approaches aiming at the improvement of maintenance strategies and practices have emerged [36]. The growing awareness to pay attention to maintenance problems on a scientific level seems justifiable if the level of expenditures in maintenance are taken into account. In table 1.1 the expenditures in maintenance, which total to about 14% of the national income of the Netherlands, are expressed in relation to some major technical system categories. The table shows that roughly 35 % of all maintenance expenditures have been spent on industrial technical systems.

<table>
<thead>
<tr>
<th>Maintenance cost (1979) in the Netherlands, 14% of national income (dfl. 227.310*10⁹)</th>
<th>dfl. *10⁸</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>industrial maintenance</td>
<td>11,500</td>
<td>35</td>
</tr>
<tr>
<td>transportation</td>
<td>6,000</td>
<td>19</td>
</tr>
<tr>
<td>building facilities</td>
<td>12,500</td>
<td>40</td>
</tr>
<tr>
<td>other</td>
<td>2,000</td>
<td>6</td>
</tr>
<tr>
<td>Total</td>
<td>32,000</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 1.1. The maintenance expenditures in absolute figures and in percentages.
1.3 Information systems in maintenance

1.3.1 General developments

Over the last ten years computer hardware technology has achieved major advances. Computers have become a mass item, produced cheaply in large quantities, with increased reliability and with an ever increasing computing power. A striking example of this development is the Personal Computer.

In order to utilise this computer power computer software is needed. Unfortunately, although hardware is relatively cheap and available in large numbers, application software development still tends to be cumbersome, costly and full of risks. In contrast with hardware, which can be regarded as a generally applicable tool, software has up to a high extent to be developed specifically for each task. Thus, the complexity, i.e. the number of the required tasks and their interrelationships, in combination with the uncertainty with respect to their validation, largely determine the effort needed for the development of software. It goes without saying that development efforts increase costs and uncertainty increase risks. The higher the risks and the costs, the more anxious an organisation will be to achieve a information system performing as requested and expected.

Historically, two non-exclusive strategies can be pursued to acquire information systems:

a) in-house development;
b) using standard software packages.

ad a) The first strategy concerns the traditional path of developing software. The situation at hand is considered to be unique. That means, no one else is supposed to have similar requirements and therefore no common solutions are available. Consequently, the software to be developed will be unique as well. Companies which adopt this strategy do accept the potentially high cost and high risks of in-house developed software.

Possibly, costs for in-house development may be justified if the software contributes significantly to the performance of operational organisational functions. The ability to fully control the development process is considered crucial to reduce costs and risks involved with respect to in-house development.
ad b) The second approach benefits from the possibility to spread development costs over users in more, different, organisations. Commercial developers of software in particular aim at this strategy. They develop standard software for a chosen market of users. Logically, the more the number of users which are expected to buy the same software without substantial modifications, the lower the intended sales price can be, and consequently, the higher the profits on the developers side. It is the developers responsibility and major aim to develop the software in such a way, that as many different clients as possible will evaluate the software to be applicable in their situation. In the end, the decisions made by the developer in the generalisation of the functions the software is to perform, reflect the developers view on the way in which certain operations have to be carried out in a particular maintenance situation. We will refer to this type of software as standard software packages. General purpose packages such as word processors, spreadsheets, etc. owe much of their success to this strategy. Because there are many potential users who have similar requirements, high development costs can be easily justified and still substantial profits can be made by developers.

Figure 1.1. Annual sales of standard software packages in Dutch industry (Source: CBS report "Statistics in software automation 1990 - 1992)
Companies are increasingly tempted to concentrate on the second strategy. Encouraged by the successful application of a number of general purpose software packages, packages intended for specific complex organisational functions many users and vendors expect them to be applied with similar success. Even large companies with highly specific organisational functions, which could afford in-house development seem to prefer the first strategy. The shorter time required before a system is operational may contribute heavily to that choice. Not surprisingly, developers offering organisation function specific packages market their packages aggressively. In most offerings a panacea to virtually all problems and requirements is promised. In figure 1.1 the annual sales volume in Dutch industry for standard software packages are shown ([19]). The high volume of organisation function specific packages demonstrates the large financial importance this market has for developers. Many developers suggested unrealistic high expectations in favour of their packages. In developers/suppliers advertising and in numerous publications by management consultants it is also strongly suggested that these packages can be applied easily after installation and with some basic training of the users. Unfortunately, due to this marketing boom, many companies which have only little experience with regard to standard software or who are primarily interested in a quick solution are inclined to pay little attention to a thorough evaluation and selection process, which is required to identify whether a package is suitable indeed ("Digging deeper is of no use; why do we have to reinvent the wheel, and besides, we don't have time for this"). It is not surprising that as a result many have failed completely in their attempt to realise an operational information system by using a standard software package.

1.3.2 Information system development from computer scientists perspective

Historically most software development projects are dominated by software design in its own right. The user is expected to understand his problem area perfectly and to present precise requirements at all times. In the context of this study, computer scientists would consider two major outside factors to be relevant in today's failures in software development for organisational functions, namely deficiencies in communication with the user and a slow development cycle.
Deficiencies in communication

Today, computer technology and computing science have advanced so rapidly that user needs are seldom restrictive when it comes to the efficiency of a computer application, at least in theory. Encouraged by the increased computer power promising sophisticated information systems much more ambitious information system development projects have been started. Unfortunately, in the majority of cases things did not work out as expected. In the USA a government report (dated 1989, [18]) it is stated that only 2% of all software development projects in business have been completed successfully, i.e. the software runs as intended without modifications. In the Netherlands similar findings have been reported; only 25% of the standard software packages run satisfactory from scratch [19]. This CBS report states that 56% of all sold standard software packages do not run satisfactory due to functional drawbacks. With respect to packages aimed at the internal maintenance function similar observations have been reported by Christer and Whitelaw [20]. These figures are of concern and justify an analysis for the cause of this unsatisfactory performance. In view of the magnitude of the problem, it seems unlikely that deficiencies in communication with the user would be the only explanation for this phenomenon. From a computer science perspective, development projects can be shortened by the application of high level programming tools, aimed at increasing the information system developer's productivity. Faster development would reveal mistakes earlier, which then could be corrected sooner. Existing information systems would benefit from faster development capabilities as well. Some computer scientists (e.g. [1]) suggest that a substantial backlog in development and redesign of existing information systems could be reduced by a faster development process as well. It is clear that shorter development times provide the potential to build information systems in a shorter time and therefore contribute to the in-house development strategy.

The discussion presented above leads to the conclusion that from a developers point of view major problems start at the beginning of an information system development cycle, i.e. during the information analysis stage. Mistakes which become apparent later during practical testing with end users can only be corrected by redeveloping the erroneous sections. This can be a time consuming and costly undertaking.
1.4 Choices made in this study

Multi-disciplinary approach
The previous section illustrates that today's problems with respect to the development of software and the evaluation of (commercial) software packages for organisational functions cannot be expected to be solved by computer science alone. If the overall objective is to improve maintenance performance, it is evident that other disciplines are required as well. In this study we assume that it is accepted that different knowledge areas, connected to applied sciences, are needed to contribute to solving our problem of determining valid functional requirements for a maintenance function. However, to make effective use of these different knowledge areas they need to be interfaced appropriately, i.e. we shall pursue a multi-disciplinary approach rather than a pluri-disciplinary approach. In the maintenance field we can identify four essential knowledge areas which have a strong impact on how a maintenance function is designed and, consequently, which procedures and practices will be appropriate. These areas are technology oriented disciplines, maintenance management, sociotechnology and computer science. We shall concentrate on the knowledge areas that can be characterised as industrial engineering disciplines. Technology aspects such the design and modification of technical systems are too diverse to be dealt with in one study. Therefore we take these technological aspects for granted, and we shall limit ourselves to the technical characteristics relevant for the choices to be made in maintenance management.

An (end)user perspective
In this study we shall not attempt to design an information system suitable for all kinds of maintenance situations. Instead, we look at evaluation of information systems from an end user's perspective. This means that individual users, or several users together with a common responsibility, in maintenance, are the primary benefactors of an information system for maintenance management. Therefore, functional requirements should not only be valid with respect to the overall organisational perspective, but these requirements should also represent a direct translation of the intended procedures and practices determined by the users themselves. Other, efficiency oriented aspects, such as the required computer processing power and so on, are of secondary importance and will be left out.
1.5 The results of this study

The primary objective of this study is to develop a framework of design rules to be used in the determination of the functional requirements for an individual maintenance function in an organisation. The development of such a framework requires an industrial engineering design approach. Knowledge elements from maintenance management, sociotechnology and computer science will be used to construct the framework. Although it is not intended here to pursue basic research in any of the listed discipline areas, perceived missing elements of knowledge will be explicitly identified. It is especially true that in the area of maintenance management, a single universal type of planning and control is unfeasible. Moreover, a differentiation in planning and control types is needed. Each type should reflect a most preferred way of planning and control in a given maintenance situation. In addition, evaluation and selection techniques will be enhanced to make use of our definition of functional requirements.

It is important to note that the application of our framework has a much greater impact than our design objective would suggest. Due to the holistic nature of our framework of design rules, application of this framework in an individual situation may result in a (re)design of other elements within the operational maintenance function.

Due to the fundamental nature of this study, the reasoning in establishing our framework is more or less based on theoretical considerations. The time required to fully test this framework in a practice would necessitate extensive introductory assistance to the maintenance staff involved before observations can be made, and largely exceeds the time frame available for this PhD study. Therefore, the scope of this study has been limited to the development of a first prototype model of the framework.

1.6 The structure of the thesis

Essentially, in developing the framework the following steps have been taken:

1) Analysis of current evaluation practices (chapter 2)

2) Determination of the design methodology (chapter 3);

3) Derivation of the design criteria and constraints (chapter 4);

4) Design of the principle prototype framework (chapter 5);

5) Design of a prototype framework for maintenance (chapter 6 and 7);
6) Demonstration of the maintenance specific framework (chapter 7);
7) Enhancement of current evaluation techniques (chapter 8);
8) Conclusions.

Analysis of current evaluation practices
In this section of the thesis the current problems with the evaluation of information systems in general and standard packages in particular will be analysed. It will be demonstrated that some evaluation techniques should be used with great care, because they provide means for a global judgement only. It will become clear that the success of an evaluation process primarily depends on the suitability of the criteria that will be used. In our context, we are interested in functional criteria as the essential basis for the evaluation of information systems for maintenance. Observations in practice and literature research indicate that in most situations too global functional requirements have been used.

Determination of design methodology
In the context of this research, the result of the application of the framework, i.e. the functional requirements in a maintenance situation, will be defined. It will be explained that functional requirements are actually a product of several design steps, each requiring contributions from different knowledge areas. The core approach of the concept will be based on the philosophy of the PBI-model described by Bemelmans [7]. In this study we shall focus especially on the "B"-part. The "B"-part stands for the functional aspect, which covers the procedures and practices and the personnel required for an effective and efficient control of all organisational processes.

Derivation of design criteria and constraints
For the development of our concept we have to combine three scientific design oriented disciplines, i.e. the maintenance theory providing the core procedural logic for the maintenance function, sociotechnology for allocating functions as areas of responsibility to personnel, and finally, computer science for the information systems part. Each discipline has its own design objectives and constraints, which have to be matched. Maintenance management knowledge is needed for the design of justified, complete and valid functional requirements in a maintenance situation. Sociotechnology uses functional
requirements in order to maximise self-control of each group distinguished. Finally, computer science requires detailed unambiguous formal functional requirements in order to be able to develop efficiency criteria for selection of a standard package or to develop an in-house information system.

Design of the principal prototype framework
We can incorporate standard socio-technical and computer science design approaches in our framework consisting of two steps: the Structural Functional Analysis and Design (=SFAD-process) and the Detailed Functional Analysis and Design (=DFAD-process) steps. The SFAD process is a preparation to the socio-technical design step (the PSAD process). General characteristics of an organisational function are used to determine structural functional areas which can be allocated to groups easily. Then, after task allocation the personnel takes the responsibility to carry out the detailed functional analysis. A maximum of situational knowledge and personal and group preferences determine which detailed formal functions are required for information system evaluation or (re)design.

A key approach used in the prototype framework is the decomposition principle. It provides a suitable basis for detailing global organisational functions via the intermediate structural functional areas, until feasible detailed functions have been determined.

Design of a prototype framework for maintenance
The principle prototype framework provides only a skeleton for our framework, which has to be extended with organisation function specific knowledge. In this study we are in particular interested in the internal maintenance function. Subsequent decomposition levels will be determined based on the EUT maintenance model, which represents a universal formal representation of the maintenance function, until the SFAD-process has been completed. Because of the sheer amount of alternatives in the decomposition process we shall concentrate on the core subfunction depicted in the EUT maintenance model, i.e. the planning and control subfunction. It will be demonstrated that due to the lack of a maintenance planning and control theory, an explorative approach has to be used, which compensates in part for this omission. Alternative options for structural functions will be developed in addition to guidelines for the maintenance function, which will be used in the following design step, i.e. the personnel system analysis and design process.
Introduction

Demonstration of the maintenance specific framework
An actual maintenance situation will be analysed by applying our framework. This case situation will illustrate the principles of our newly developed framework in view of its suitability in practice. Because the detailed functional analysis process is very elaborate, only specific suggestions for improving maintenance performance will be discussed. This discussion will also illustrate the dynamics of an actual maintenance situation.

Enhancement of current evaluation techniques for using functional requirements
The initial problem of evaluation and selection of standard software packages can be solved in theory, provided that our framework is used in order to determine sufficiently detailed functional requirements. However, because of using detailed requirements chances are small that suitable packages can be found at all. To solve this problem an enhanced evaluation and selection process will be developed. In this enhanced process a "black or white" approach in evaluation is replaced by a listing of functional mismatches. During the selection stage the possibilities to compensate for this mismatches will be assessed. The most preferred package will have the least costs and efforts required with respect to compensating measures. Still, if all packages have functional flaws which cannot be compensated, in-house development is the only other option.

Conclusions
In the final section the outcome of the study will be evaluated and summarised. The role of the framework will be discussed with respect to the spin-off it may have to other industrial engineering design projects and to initiate research in the contributing knowledge areas in this study.
On the determination of functional requirements in a maintenance environment
Analysis of current software package evaluation practices

2.1 Introduction

Due to the enormous decline in the unit price of computer equipment as a result of technological advancements, the computer has become increasingly available for many kinds of users. Especially for smaller firms and departments, personal computers become an attractive proposition. Together with the large assortment of software packages becoming available, the applicability of the computer has increased considerably.

In most cases the personal computer is operated by the user himself, making him independent of centralised processing facilities and of the direct assistance by computer specialists. As a consequence the user becomes more directly involved in the decision process whether to buy and adapt standard software, or to develop software in-house. In this chapter we shall concentrate on the suitability of standard packages in organisational functions such as the maintenance function. The suitability aspect of packages will be addressed from the user perspective, who is primarily interested in finding a package that fits best the requirements of the individual functional area at hand.

2.1.1 Make or buy?

The advantages versus the disadvantages of buying standard software packages have been discussed by several authors (i.e. Wortmann [77] and Stahlknecht & Nordhauß [74]). In general, the following reasons in support of the application of standard packages are put forward:

1) *Standard packages are cheaper*

   There is no need for programming by the user, which saves a lot of time. One may also expect that standard packages have lesser bugs which would require additional programming effort to remove them.

2) *Standard packages can be implemented faster*

   Since the programming and debugging stages in the development cycle can be omitted
almost entirely the throughput time for a complete implementation can be reduced significantly.

3) **Standard packages are professional**

Software for complex organisation functions is expensive to develop in an individual situation, because some specialists are to be hired for a long time to reach professional quality.

In practice the arguments seem to be correct with respect to the application of general purpose packages such as word processing and spreadsheet programs. Nowadays, this type of software satisfy most functional requirements users have at substantially lower cost compared to in-house development. However, the situation is different with organisation function specific software. In chapter 1 already some serious problems have been identified. In some business sectors some serious doubts have risen against the application of standard packages for organisational functions. A research report from the International Iron and Steel Institute in 1989 [50] states that many companies believe that packages cannot offer the same basic but essential functional requirements provided by their own in-house developed information systems. Others believe that within a certain type of organisation the commonalties are large enough to justify standard software package development just for that particular sector. E.g. the Danish Hospital Institute [25] followed this approach to develop functional requirements for the maintenance functions of hospitals.

In order to be able to judge if packages meet the requirements of a client a **measurement instrument** is needed which measures the deviation between predefined required properties, and the properties of the available packages considered.

In this chapter a framework will be presented in which some of the major aspects of the software evaluation process for maintenance functions will be discussed. The framework describes the main evaluation activities and the sequence in which they should be carried out. The evaluation approaches most frequently encountered in literature and in practical situations will be subjected to a critical analysis. In addition some practical problems frequently encountered with respect to a software package evaluation will be discussed.
Finally, as a conclusion, it will be pointed out what research is necessary to improve the evaluation process.

2.2 The evaluation framework

2.2.1 Nomenclature

The (re)design of an organisational function, known as application area in computer science terminology, requires detailed knowledge of the function concerned. In our case, general knowledge of maintenance theory in combination with situation aspects is a necessity in defining the requirements of the maintenance function.

Software is, like any other type of information processing tool, the outcome of several decisions made in different design stages such as the organisational function and the personnel system design stage. Each decision requires a certain type of knowledge in its own right. Therefore an evaluation and selection process should be based on sets of criteria with respect to each relevant area of knowledge. In order to prevent ambiguity the main criteria and definitions relevant to the explanation of the evaluation principle will be presented and discussed in this section. McCall [60] typifies these criteria as quality attributes of software.

For now, we shall categorise McCall's attributes into three distinct groups, which are relevant from a users perspective, i.e.:

1) effectiveness
2) efficiency
3) flexibility

(1) Effectiveness

Whether a package meets the functional requirements of an organisational function is the major question in an evaluation. Effectiveness (or functionality) measures the number of functions that match the functions that are required, i.e. what a package is capable of doing in view of its perceived application area.

To measure effectiveness a list of required formal functions, i.e. the functional requirements, is needed, which serves as the standard to which a package is compared. We shall denote
measuring effectiveness by means of actual functional requirements as standard, as absolute measurement or measurement on an absolute scale.

(2) Efficiency

Efficiency concerns the amount of effort and costs that are needed to achieve the desired level of effectiveness of a standard software package. Effort should be interpreted in a broad sense. The effort is not only determined in terms of hardware or software expertise but also of, e.g. the required skills and education level of the users in addition to the cost obtaining, using and adaptations of a standard package. In fact, many evaluation aspects which do not clearly relate to the effectiveness criterion will be related to efficiency criteria. Due to the diversity of possible relevant efficiency aspects a multitude of specialist disciplines are involved in the assessment of the efficiency.

(3) Flexibility

Flexibility concerns the degree to which the effectiveness/efficiency ratio of a standard package will stay within acceptable limits in time, or in different situations but for the same type of organisational function. This criterion addresses the dynamics of an operational organisational function. The flexibility criterion is defined as a function of time and type of situation, and therefore must certainly be considered the most complex of the three criteria. In practice, functional requirements to be used for an evaluation can change due to a growing understanding of users to improve to procedures and practices, or simply because the group, in which users operate, has to cope with changing requirements from outside. Similarly, available means for realising functional requirements advance as well. E.g. newer computers facilitate larger and more sophisticated algorithms, potentially providing the users more options.

The flexibility of a package can be influenced positively by:

a) the ease in which functions (external modules) can be added to or deleted from a package. Some packages provide some flexibility by activating alternative functions already available in the package (so called "parameterised software") when required. Then, no additional programming effort is needed if minor changes in functional requirements are desired.
b) The ease in which a package can be (re)designed. If non negligible changes in functional requirements cannot be met by the options readily provided by the package under consideration, the option of redesigning the software is a possibility to be considered. Specific attention of the developers for the implementation process (in terms of computer science: the way the programming-code is achieved) with respect to the development means or tools available and the information infrastructure (see also chapter 4) is essential here.

It is important to note that the effectiveness, efficiency and flexibility are based on the functional requirements and therefore they are situation-dependent criteria. Consequently, "more" doesn't necessarily mean "better". E.g. the effectiveness of a standard software package in a typical maintenance situation is not increased by adding functions which are not required in that situation.

2.2.2 The parties involved

Unlike complete in-house development, the actual potential of packages is determined by the availability in the market. The driving force behind what properties packages must have largely depends on individual marketing policy of each vendor. It is clear that the commercial interest of a vendor is different from the buyer's interest in maximising his information system effectiveness. In this paragraph we shall discuss the consequences of using standard software packages, which result from different interests of participants in this market.

In the decision process of evaluating and selecting software four stakeholders can be distinguished. On the one hand the developer and seller, on the other hand the procurer and actual user.

The developer is the "engineer" of a package on hand. Because software is a multi-disciplinary product, more than one developer with different skills may be involved in the development process. The dominant question here is what the origin is of the functional requirements on which the package is based. Developers often design software for a specific client. In that case the software is tailored to meet the specific requirements of that client. As
both the client and the developers are faced with the high costs of this development of software, they seek for possibilities to spread these high costs. It is common practice for such specific software to be presented as "generalised" and offered in the standard software market in a later stage. Stahlknecht [74], for instance, found that the majority of packages offered for administrative functions in the printing industry were, originally, designed for a single specific situation.

The seller is interested in maximising his profits by selling packages. It is clear that, in order to maximise profits, the investment in development should be minimised, whilst the general applicability should be maximised. The software developers perception of the market for a specific and complex application area will influence the level of investment in development work. Solid research of the varying situations in the application area itself can only be justified if the market potential is believed to be sufficiently large. In the sellers marketing approach, it is very important to address both the user, who is expected to assess effectiveness, and the procurer party, who mainly sets the limits with respect to the efficiency.

The user is considered to be the actual "problem-owner". The user, being strongly involved in his day-to-day activities, must define the essential functional requirements which are expected to improve his performance, but would require some degree of automation. In general, it can be stated that the users have knowledge of specific and detailed aspects of his situation not known by others, outside the users scope.

The procurer is the ultimate decision-maker. Based on mainly economical considerations (cost benefit estimates) he decides about the possible investment in packages. In general, if the procurer is not the same person as the user, we can assume the procurer has no in-depth knowledge of the application area on hand. In larger organisations, the procurer is normally situated at the higher levels of management. Observations in practice show that at higher levels of management in particular there is little understanding of lower level requirements and sometimes "keeping up with the Jones" is the primary motivation to choose a standard software package.
2.2.3 Types of relationships

On the basis of the interests of these four parties, the characteristics of their relationships can be explained. In table 2.1 the basic relationships are shown. Each relationship will be discussed according to its reference given in table 2.1

<table>
<thead>
<tr>
<th>Developer</th>
<th>Seller</th>
<th>procurer</th>
<th>user</th>
</tr>
</thead>
<tbody>
<tr>
<td>developer</td>
<td>C</td>
<td>F</td>
<td>D</td>
</tr>
<tr>
<td>seller</td>
<td>B</td>
<td>E</td>
<td></td>
</tr>
<tr>
<td>procurer</td>
<td></td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>user</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2.1 The types of relationships of the interested parties

Type A. The procurer-user relationship

In general the user, as the primary problem-owner, is the initiator of the package evaluation and selection process. Once the user knows what functions are needed two basic approaches are possible.

The functional requirements have to be discussed with the procurer, who is responsible for the organisation and financial consequences of the functional changes proposed by the user. Their relationship is largely based on establishing an acceptable cost benefit trade-off, i.e. costs of realisation versus user effectiveness. In-house development, the first approach, potentially provides all required functionality whereas standard packages, the second approach, at best can result in an acceptable functionality in view of the cost benefit trade-off.

Type B. The seller-procurer relationship

Normally, this relationship becomes important at the end of the evaluation and selection process. If the user has arrived at a recommendation of a package with respect to its functional requirements and the "efficiency"-experts support that recommendation, the procurer starts the negotiations with the seller regarding pricing and delivery conditions (One should not think only of cost and delivery date, but also of additional services such as
consulting, conversion of data files, training, etc.). In this relationship the bargaining process is the most characteristic aspect.

Type C. The developer-seller relationship
In some cases the developer of a package and the seller are not identical parties. Organisation function specific packages, such as maintenance management information systems, are mostly sold by consultants and by software houses established in the market area in which they find their potential clients.
Again, this relationship is dominated by a cost benefit trade-off. However, unlike in the user-procurer relationship, observations in practice during at least 20 Msc projects in Industrial Engineering show that actual users have a relatively passive attitude and they are seldom taking the initiative in the defining requirements concerning the effectiveness of packages. Here, developers must rely more or less on a seller-defined virtual user, as an abstract representative of the entire market.
The seller is more or less dominant in this relationship. Because of his direct contact with the (potential) clients, he determines the price of a package and the upgrading policy (future versions) concerning functionality. The developer acts in this relationship as a contractor responsible for the implementation of a chosen package (he is an efficiency expert).

Type D. The developer-user relationship
Whether in-house development is preferred or whether a suitable package has been selected for use, in both cases this relationship is of utmost importance. The developers main goal is to design an information system which satisfies a maximum of the functional requirements determined by the user. The better the developer understands the needs of the user, the more likely will the resulting information system satisfy the user needs. Nowadays, the developer is not an expert in organisational functions and, consequently, relies primarily on his communicative abilities. In case the developer belongs to the same organisation as the user, direct communication can be much better established as compared with external developers. In the optimal situation, users would not only define their own functionality, but would also be able to develop the information system themselves (= end user computing).
Type E. The seller-user relationship
The user's primary interest is to know where to find accurate and reliable information on the standard software package's functionality designed by the developer. Experience suggests it is difficult for the user to get in touch with the (original) developers. In that case, the seller party constitutes an interface between developer and user, thus adding an additional communication problem.

Type F. The developer-procurer relationship
Although this relationship is a theoretical option, in most cases the (original) developer and the procurer do not communicate directly. It is the seller who will establish direct contact with the procurer. As in the seller-user relationship, this relationship has little significance, unless the user and procurer are identical, or alternatively, the seller and developer are identical.

2.2.4 Consequences of party-relationships

It is evident and vital that in particular the user is aware of the difference in interests of the parties involved and of the dangers of the resulting unreliability and incompleteness of the information he will receive, as compared with what he needs to arrive at the correct choice for his specific situation.

Before entering into the evaluation and selection process, management should be aware of the consequences of using standard packages. Accepting standard packages, especially in complex application area's as in maintenance, requires a long term commitment of management and users to the packages which were chosen, and inevitably also to the seller of the packages, at least, if they can still be approached when problems turn up. Apart from the sellers creditability and his experience in the application area, the sellers upgrading policy is of major interest. The effectiveness, efficiency and flexibility of a package can be improved by the developers of packages, by making new versions available. However, there is always a certain risk that future versions presented are trivial if not undesirable or even invalid for particular clients, who in this respect are faced with, and are
dependent on, the changes determined by the policy of the developer/seller. This risk may enforce management to choose for in-house development only. Especially, if the user is faced with a dynamic environment resulting in sometimes unpredictable functional requirements, the use of standard software package may not be a valid option.

2.3 Evaluation and selection phases

In general, to streamline an evaluation and selection process a stepwise approach is advocated by various authors. E.g. Brownstein & Lerner [16] recommend a phased approach in software evaluation and selection, referred to as "the evaluation process". Their approach consists of 21 sequential steps which have been summarised in figure 2.1 without conditional bypasses.

In the phase of study initiation an orientation is carried out to identify what packages are available in the market. Based on global criteria a preliminary rejection will take place of packages which obviously don't meet these global requirements. The remaining packages are subjected to an evaluation in which detailed criteria will be used. The package which performs best in view of these criteria will be the first choice from a user point of view. Finally, in an operational performance test the most preferred package will be used for a limited time for its intended purpose routinely or if this isn't possible in a closely simulated environment to check whether the package conforms to efficiency standards.
Although their approach seems to be a logical way to carry out an evaluation and selection of standard software packages there is a serious drawback in their approach. They assume that there is always, definitely, at least one suitable package available for selection. Thus implicitly, the "buy" decision has been made already before the start. In our view, the suitability of a package is expressed in terms of its effectiveness in with respect to the actual situation in which a package would be used. An evaluation is not a matter of ranking packages according to their performance score and selecting the package with the best score (relative measuring). Since the ultimate aim is to achieve a 100% effectiveness, an evaluation should reveal what deficiencies a package has, i.e. what functions are missing, and must be compensated for in order to get 100% effectiveness. If compensation measures are ineffective or inefficient, the result of an evaluation and selection is that no package is suitable.

Furthermore, Brownstein and Lemer assume that functional requirements are more or less implicitly available and a list of user needs will be sufficient for evaluation purposes. They describe their approach in global terms and their primary interest is focused on the way in which the evaluation is to be organised. Because of the, suggested, general nature of their approach they consider it suitable for the evaluation of packages for any field of application, including the field of maintenance information systems.

Basically, from a procedural point of view the sequence in which each step can be carried out in an evaluation process is rather straightforward, as Brownstein and Lemer have demonstrated. However, we have to adapt their approach and introduce the possibility for complete in-house development.

We advocate the following steps to be carried out in an evaluation and selection process:

1) define the domain of the maintenance function;
2) identify potentially suitable packages;
3) determine the functional requirements;
4) establish and evaluate packages;
5) select most efficient package of all effective options, or develop in-house.

The third step, determination of the functional requirements, is certainly the most important.
On the determination of functional requirements in a maintenance environment

Figure 2.2. shows the steps (the single bordered blocks) of the evaluation and selection process and their relationships. The double bordered blocks represent the methods or aids to be used in each step.

First, the domain of the maintenance situation is established. In this step the main subfunctions belonging to the maintenance function are determined. E.g., it can be decided to include the function inventory control of maintenance items in the maintenance function. This type of decision requires a general functional maintenance model on which to base the considerations for that decision. This general model should be detailed more specifically according to the situation at hand. As a starting point the EUT-maintenance model can be used [37]. A sound understanding of all EUT maintenance model aspects in combination with situational circumstances provides the basis for the definition of the global (high level) maintenance functions in a specific case.

Once the combination of subfunctions has been defined a preliminary selection of packages can be made. The purpose of this step is to reject those packages which are obviously unsuited for supporting the specified functions. In order to be able to make this first selection
not only the required set of global maintenance subfunctions should be defined but also the set of packages which should be subjected to this quick first selection. There is however no procedure for discovering the complete set of packages available. Practically, users are advised to scan maintenance journals for vendor advertisements, attend vendor demonstrations, contact colleagues active in the maintenance department in other firms, etc. to learn about standard software package offerings.

At this point it will become clear if there are no or only partially suitable package alternatives left. If no package meets the demands completely a choice must be made between adaptation and in-house development.

If there are packages left after this initial filtering process, a detailed functional requirements analysis is necessary. This detailed analysis of the functional requirements must provide, essentially, the effectiveness criteria for the evaluation. In the following section the principles of such an evaluation will be discussed more in depth.

Once the (functional) evaluation has been carried out it has become definitely clear if there are still candidates left for selection. Eventually, the final possible selection is to be based on the efficiency criteria. In case the selection step does not result in finding a package considered to be suitable, in-house development, or continue with the the status quo will be the only options left.

2.4 The evaluation principle

2.4.1 Introduction

In the foregoing section it was illustrated that the eventual evaluation step (step 4) is only part of the total evaluation and selection process of standard packages. Because of the sequential relationships the correctness of the evaluation depends on the accuracy with which each of these preceding steps are carried out. In this section the evaluation itself will be discussed. Special attention will be paid to the basic steps of an evaluation and to the major constraints of each basic step. Starting point in the evaluation phase is the effectiveness, determined in the detailed functional requirements phase.
In order to carry out an evaluation, effectiveness criteria must be defined beforehand. Many criteria can be established, reflecting the relative importance of the functional requirements. E.g., sometimes it is desirable to limit the number of work orders on hand in an operational work order control function in order to prevent losing the overview. In that case a volume restriction for the number of work orders on hand is a criterion. Nielen [67] defines also accuracy of the information provided by an information system and the correctness of the information system algorithms as vital effectiveness criteria. Definitely, these criteria will receive extra attention in this study at a later stage. At this stage of the study we shall concentrate only on the completeness criterion.

Basically, the functional completeness criterion is a measure to determine the number of functions available in a package that match the required functions determined in a functional analysis. As a result, a comparison of functions can give certain types of outcome (i.e. a required function is seemingly identical to the function available in a standard software package; a function has a different internal operational mode, but uses identical input and provides identical output data meeting the requirements as defined, and so on). A clear definition of requirements of all possible outcomes should be agreed upon by the responsible person(s) before for the evaluation is started.

Other effectiveness criteria can be defined if considered necessary.

2.4.2 Evaluation steps

In an evaluation three basic aspects can be distinguished:

a) **Criteria.** Establishment of the criteria based on the functional requirements analysis.

b) **Properties.** Determination of the properties of the package which are relevant in view of the criteria.

c) **Comparison.** The assessment of the differences between functional requirements on one hand and the properties of the package on the other hand.

ad a) **Criteria**

The comparison of the properties of the package and the requirements that have been established cannot be carried out in all cases. Sometimes it is undesirable, or even
impossible, to automate all defined functions. In general, functions that change often due to the dynamics of the environment are difficult to determine formally. In particular functions at higher organisation levels can be defined in a less formal manner only.

ad b) Properties

The criteria established in step a) determine how and what to look for in a package. As has been demonstrated earlier, it would be practical that the format to present the functional properties of the package is similar (compatible) to that which has been used for the functional requirements. In assessing the required information on a package one is usually dependent on information provided by the sellers and by registered users. Because there are no accepted standards on the way in which the functional properties of packages should be presented to potential clients, there is only a small chance that a client will receive detailed information about the packages indeed, unless he is known beforehand to buy the package anyhow. To test the sellers ability to provide detailed information for evaluation purposes more than 20 sellers were asked to send in depth information on the packages they are offering. At best, some advertisement-like sheets were obtained from the sellers, giving only global promises and global black box illustrations of the main functions provided. To be sure about the functional properties of standard software packages in most cases a laboursome hands-on trial by potential users becomes necessary in which case special arrangements about the temporary use of a package must be made.

ad c) Comparison

A suitable measuring scale is required in order to be able to compare the values of the potential properties of the package with the criteria. The comparison of the potential of a package with the functional requirements normally is carried out by means of a relative scale. A relative scale implies that one package is taken as the standard to which all others are compared. Usually, it is unclear what properties are perceived to be relevant to serve as criteria. Inevitably, relative scale comparison favours packages which provide more functions, e.g. the package which can store more workorders, etc. (the "more-is-better-syndrome"). On an absolute scale a package providing much more functions than a "smaller" package is less effective (completeness criterion!) if fewer functions match the functional requirements.
However, in order to be sure that the evaluation is correct indeed, it is necessary to define the functional requirements and consequently the evaluation criteria on an absolute scale. Using an absolute scale requires to establish the exact functional requirements in the first place. Then, these requirements are used as criteria to assess packages.

In conclusion, with a relative scale the options are ordered and the best, or least worst, is selected. This assures no minimal requirements have been set. However, an absolute scale can provide the answer concerning the actual suitability, which permits the possible conclusion that none of the packages is suitable.

2.5 Evaluation methods

2.5.1 Introduction

The need for practical evaluation methods has been put forward by many authors over the years. Some methods concern a general approach, others are directed at specific branches of industry or specific departmental functions. In this section the diversity of existing evaluation methods is categorised in four standard approaches. Taking into account the evaluation principles presented in paragraph 2.3, some critical remarks will be made about these approaches.

The following approaches will be discussed:
1) The certification method.
2) The checklist.
3) The ranking approach.
4) The questionnaire.

2.5.2 The certification method

In this approach a committee of acknowledged specialists is formed in order to present their opinion about a set of standard software packages. The members of the committee are
expected to consider whether in their opinion the packages do have a certain minimum set of functions which they regard to be essential. In case a package supports that minimum set of required functions it passes the evaluation. Thus, an accepted package is a candidate recommended by the committee.

Usually, a committee is used if the specific expert knowledge considered essential, is taken to be not available with one person, because of the diversity of relevant aspects. The essential question then is how, exactly, the committee arrives at its judgement. The criteria which were used remain unclear in most cases (see e.g. [26], [27]). As has been argued before, criteria must be based on the results of an functional requirements analysis for each individual situation. An accurate definition of the maintenance situation is a prerequisite for this functional analysis. In general, certification appears to lack sufficient specification of the maintenance situation considered. Therefore the opinion of a committee is, in most cases, unproven.

2.5.3 The checklist

The most frequently encountered approach to evaluating software packages is the checklist method (see figure 2.3). The checklist consists of a table with the functions which have been

<table>
<thead>
<tr>
<th></th>
<th>WORK ORDER</th>
<th>SHUTDOWN PL</th>
<th>BUDGETING</th>
<th>PROCUREMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>PACKAGE X</td>
<td>YES</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>PACKAGE Y</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>PACKAGE Z</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
<td>NO</td>
</tr>
</tbody>
</table>

Figure 2.3. The checklist
On the determination of functional requirements in a maintenance environment

distinguished, on one axis, and the packages which support the intended application, on the other axis. A marker is placed in the appropriate box if a package contains the function mentioned (e.g. see Armstrong [6] and Parkes [68]), limiting the answer to "yes" or "no". Theoretically, it is unclear what a "yes" or a "no" means, since most of the attributes are specified in global terms. More detailed specifications would allow a more precise judgement.

Typically, only the criterion of completeness is considered in this approach. Again, the authors do not explicitly present the maintenance requirement which they used to justify the global or detailed functions they took up in the checklist. In addition, not always do authors justify the selection of packages they evaluate.

Some authors don't make a clear distinction in effectiveness and efficiency criteria, which results in a checklist mixing up functional and efficiency aspects. In that case the maintenance specialist, who is inclined to be preoccupied with effectiveness, rather than efficiency criteria, is confronted with features which can be judged only by a specialist such as computer specialists or an ergonomist. Usually authors fail to specify at what stage in the evaluation and selection process they want to apply the checklist: in the global maintenance domain specification or in the detailed functional requirements analysis. Apparently, in literature the use of checklists in an attempt to assist users in making their preliminary filtering of package candidates dominates.

<table>
<thead>
<tr>
<th>PACKAGE X</th>
<th>PACKAGE Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>P R O P E R T I E S</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>100%</td>
</tr>
<tr>
<td>2</td>
<td>70%</td>
</tr>
<tr>
<td>n</td>
<td></td>
</tr>
<tr>
<td>TOTAL SCORE =</td>
<td>315</td>
</tr>
</tbody>
</table>

LEGEND:
A: relative importance of a property
B: the degree to which a property is available
Total score: e.g. the sum of A*B over n properties

Figure 2.4. An example of the ranking approach
2.5.4 The ranking approach

The ranking approach is similar to the checklist approach in most aspects. The presence of attributes in standard software packages, covered in this approach, is balanced against the relative importance of the required properties distinguished (e.g. see Frank [35], Raouf [70] and Brownstein and Lerner [16]).

An example of a frequently used ranking approach is presented in figure 2.4. In essence, the remarks which were made with respect to the checklist approach apply here as well. In addition to this comment it can be stated that in view of our intention of an evaluation the scores of packages are not important. Only the effectiveness measured on an absolute scale is important. A possible selection must distinguish how 100% effectiveness can be achieved, if this is possible at all, purely on efficiency reasons.

2.5.5 Questionnaires

Questionnaires are very popular in literature as well (see for example COMGE [23] and Kelly [51]). Figure 2.5 illustrates an example of a part of a questionnaire published and recommended by the COMGE-group [23]. Normally, the users/procurers jointly determine, e.g. during brainstorm sessions, what they think is important with regard to using standard software packages. Typically, questionnaires produced by other users or special interest groups such as the COMGE-group provide blue print questionnaires which could serve as a guideline for users/procurers in producing their own questionnaires.

The purpose of a questionnaire is twofold. Firstly, it should assist in assessing all perceivable aspects related to buying, implementing and using standard packages, some of which otherwise might be forgotten. Secondly, questionnaires are used to question the seller of the package, in an effort to discover the completeness, the weak, and the strong points of the seller and of his package(s).

Typical issues raised in questions to the sellers cover topics such as the likelihood of the seller to survive as a company, financial disposition, the number of packages sold, the number of actual users, etc.. In most cases questionnaires tend to be very general, emphasising the efficiency aspects. Therefore questionnaires can be useful to collect
additional information needed in the final selection stage when the specific functional evaluation has already been carried out.

*The developer
Name, address of developer
Nature of the firm
- Hardware vendor?
- Software house?
- Systems developer?

*The package
Name of package
Available version (release date)
Is a new version currently under development?

etc.

Figure 2.5. An example of a questionnaire (COMGE[23]).

2.5.6 Conclusion

The categorisation of evaluation approaches presented above lists only the approaches most frequently described in the literature. Most striking is the fact that no, or only very little, attention has been paid to the development of sound detailed functional maintenance requirements. The omission of such requirements makes any outcome of an evaluation and selection process questionable. If functional maintenance requirements are being used at all, they are presented as very general maintenance models. However, not all functions are described at the same level of detail, a drawback which is probably due to the specialist technical background of some authors. E.g. practitioners active as mechanical maintenance engineers appear inclined to pay much attention to the function "determine lubrication schedules", in contrast to a much more complex, but less detailed, function such as "spare
parts inventory control", which usually is limited to stock administration. Some authors make no clear distinction between functions and data in their models, thereby confusing the reader. Generally, maintenance models presented are suitable for the description of maintenance on a high level of aggregation only. However, only a detailed functional requirements analysis can provide sufficient possibilities to eventually discriminate between the packages under consideration. Finally, in case an absolute scale is used, still the true effectiveness of a package remains undetermined. Usually, the outcome of an evaluation will be a sorted list of evaluated packages.

2.6 Executing an evaluation in practice

The foregoing discussion focused on some considerations about evaluations in literature. In this paragraph we shall present some practical considerations indicating drawbacks in the evaluation of software. In general most limitations, as appeared in the discussion of existing methods, presented in the foregoing chapters do apply here as well.

Some problems turning up frequently are:

a) **No clearly structured maintenance model is used.**
   
   Without structured high level maintenance models it is virtually impossible to assess which global functions have to be distinguished or need detailing. At the beginning of an evaluation and selection process at least the main functions that are regarded as a part of the maintenance function should be defined (the domain of the maintenance function). In this stage the EUT- maintenance model can provide assistance, since all possible high level maintenance functions are defined. In the evaluation and selection approach presented in paragraph 2.2 it was shown that subsequent stages then will be questionable, resulting in a doubtful outcome of this process.

b) **The functional requirements analysis stops as soon as global, and only global requirements have been determined.**

   The stage of functional requirements analysis is basically the most vital step in the evaluation and selection process. The selection of an functional requirements analysis
method, the level of detail, and the accuracy of the analysis, determine the quality of the eventual evaluation.

c) *The criteria are based on relative, not on absolute measuring scales.*
Relative criteria can discriminate only between the existing packages which were, somehow, chosen for evaluation. The resulting sacrifice with regard to the functionality cannot be prevented if relative criteria or too global functional requirements are used. In the end this results in selecting the "least unsuitable package", which may very well be very poor in respect of the - not assessed - true requirements of the situation at hand. Frequently, a low level of attention for functional requirements analysis tends to accept the use of relative criteria, expecting that it still will lead to correct results.

d) *No clear distinction is made between effectiveness and efficiency aspects.*
As has been stated before the software phenomenon involves many aspects involving several disciplines. Theoretical knowledge of maintenance in view of specific situational maintenance aspects is needed to define the evaluation criteria. Other disciplines cover other valid but not maintenance-specific, i.e. efficiency considerations. Therefore, in order to be complete, the execution of the evaluation and selection process requires a team off all specialists concerned, in which the maintenance specialist has a vital position because of his specific knowledge of the functional requirements.

e) *The information available about the functionality of the packages is global only and insufficiently adequate to allow of judgement of the functionality.*
In the functional requirements analysis stage a certain method and a certain format are used. In order to be able to use the predefined criteria, derived from this analysis, it is necessary to describe the properties of the package to be compared in a similar format. Even if software sellers are willing to give information about the functionality of their packages, it is necessary to convert that information into the format required, which may can be hampered by ambiguity due to incompatibility.
2.7 Conclusion

Considering the problems explained in this chapter the question arises whether complete in-house development is to be preferred to buying some available standard software packages.

To answer that question it is necessary to examine if the general recommendations in support of buying packages are still valid, because most statements on the potential of packages refer to the status quo at the time that the package was designed, which may be up to a decade earlier.

Proponents of the standard software package-buying-strategy primarily state that in-house developments are much more time-consuming, more costly and more risky, resulting - if positive at all -, in more expensive, delayed and usually inferior product.

The assumption of general suitability of packages, including situations which they were not specifically designed for, is based on the assumption that different maintenance situations still can be considered to consist of sufficiently identical maintenance functions. However, the large variety of environments in which the maintenance function is embedded makes this assumption more than questionable.

In order to by-pass the problem of differences in situations it is assumed frequently that somehow the primary process (i.e. the production function) is directly related to the way in which the maintenance function is realised. The primary process utilises certain types of technical systems, which are supposed to be - usually - arranged and handled in a type-of-industry-specific way (see [25]). Ideally, the realisation of the maintenance function, in terms of functional requirements, is supposed to be very similar in different firms within the same type of industry. Individual differences are considered to be sufficiently small that they can be solved by the inherent flexibility of the package.

On the other hand some developers assume that different maintenance functions required of different types of industry will consequently lead to different results with respect to the functional requirements of the maintenance function.

However, only a detailed decomposition of the maintenance function provides sufficient unambiguous evaluation criteria which permit to make conclusions if maintenance functions can be considered to be identical indeed. For the time being, therefore, we assume that each
specific maintenance situation is unique. In particular at the lower levels of decomposition non-negligible differences will show up.

The standard software package buying strategy may be appropriate for general purpose software, like word-processing packages where the user is prepared to adopt his way of working to the requirements of the package. But in cases where individual specific functional requirements dominate and where generalisations remain unproved - as expected in the area of the maintenance function - very often in-house developments turn out to be the only feasible alternative. In such complex application areas the principal question whether to buy standard packages or to develop dedicated software in-house is not an issue to be determined beforehand on the basis of some global simple statements (see paragraph 2.1). As already stated, only a detailed functional requirements analysis can provide the tool to arrive at a correct judgement of the effectiveness of packages.

In complex application areas, such as the maintenance function, a detailed functional requirements analysis represents a considerable "investment" in terms of effort spent by scarce organisation function specialists, who are familiar with the maintenance function. If the only purpose of this analysis would be to enable users to define their evaluation criteria, with the potential risk to find no suitable packages in the end, the "investment" could be unjustified. However, in the ideal situation, the functional requirements analysis is performed to serve both alternatives: the evaluation of existing packages and, if the result is negative, the option of in-house development.

Further research is required to find a satisfactory way in which an functional requirements analysis should be carried out, i.e. methodology, and in what terms analysis results should be defined, i.e. format definition. So called information analysis approaches tend to aim only at the registering of functions and data definitions in a systematic way. Because of the general nature of the functional requirements analysis, problems identified in this study may also turn up in other application areas, e.g. production planning and control, inventory control, and such like. Fundamental multi-disciplinary research is necessary in order to identify generally valid principles for the methodological way in which an functional requirements analysis should be carried out.
Research in a specific area, such as maintenance, will contribute to establish such a fundamental framework.
On the determination of functional requirements in a maintenance environment
3 This study's methodology

3.1 Introduction

The discussion presented in the previous chapter demonstrated that existing methods of software evaluation should be used with great care. The major drawback is not caused by the methods as such, but they lack sound effectiveness evaluation criteria.

In this chapter we shall analyse this drawback in order to revise the original research objective. It will be clear that due to ambiguous evaluation criteria based on unjustified global functional requirements questionable evaluation outcomes are being produced. A more fundamental research approach is required to develop a procedure for the determination of detailed functional requirements.

In addition the methodology which has been applied in this study will be discussed. Basically, the choice of a research methodology depends on a clear definition and analysis of the problem area which has been chosen as the main theme of the study. It will be demonstrated that this study requires an engineering design approach to deal with the problem area of detailing functional requirements for a maintenance information system. A design cycle approach will be developed, as a valid methodology.

3.2 Ambiguity of functional requirements

Apparently, functional requirements are formulated very often too global. This raises the question of how detailed functional requirements must be defined to be of use for an evaluation. The reason that we cannot accept too global defined functional requirements is the ambiguity that go with global formulations. To analyse the problem of ambiguity we must realise that ambiguity largely depends on how individuals interpret the issues at hand. Amongst other factors the capability to interpret issues unambiguously is determined by the background, the experience and knowledge, of each individual. In our context, it makes a lot of difference whether evaluation criteria have to be developed and used by individuals who are operationally involved in the organisational function at hand, or by computer specialists. One can expect that individuals who are actually responsible for an organisation function require less detailed functional requirements to understand the typical ins and outs of
organisation function specific working procedures than someone who is not. In other words, non-organisation function specialists like computer programmers, have to rely in part on almost trivial functional requirements on an elementary level, and they have to take them for granted and, perhaps, without question.

We shall assume that in our context, organisation function specific knowledge has two major components:

a) general (theoretical) organisation function knowledge, and
b) (practical) situation dependent knowledge.

ad a) We shall characterise organisation specialist knowledge as a function of terminology (definition of organisation function specific entities), frameworks (universal procedures) and "recipes" (constrained algorithmic procedures and practices) which are unique for an organisation function type.

ad b) Situational knowledge consists of all characteristics which are unique for the individual organisation at hand in which the organisation function is embedded. These characteristics originate from the organisational objectives chosen by the management and supported directly or indirectly by all organisation functions, the history of an organisation, the staff, etc.

Both knowledge components can not be dealt with in isolation, since the theoretical knowledge components only become meaningful in the areas where they are applied. E.g. theoretically precise terminology sometimes needs the enhancement of organisation specific terms, frameworks are just startingpoints for the development of organisation specific procedures and practices, constraints have to verified in the actual situation before known "recipes" can be applied. It is clear, that the need for communication and, therefore, the risk for misinterpretations is minimised if both types of knowledge are combined in individuals. We can conclude that potential for ambiguity in functional requirements is bound to individuals background and therefore, the level of detail in which functional requirements are formulated is situation dependant. In addition, we conclude that the individuals who are responsible for an organisation function are qualified best to determine (detailed) functional requirements for that particular organisation function.
3.3 Evaluation techniques

It is a common practice in today's evaluation techniques for maintenance to use functional requirements on a high level of abstraction (little detail), i.e. (sub)functions such as workorder planning, stock control, etc. without further references or more detailed descriptions. All evaluation techniques discussed in chapter 2 except for the checklist approach have structural drawbacks which are not acceptable with respect to an evaluation. Essentially, the checklist approach seems to be acceptable for evaluation purposes, provided that the criteria used for the checklist are reconsidered addressing the following issues (see 2.5.3):

1) non-effectiveness criteria;
2) level of detail;
3) origin of functional requirements;
4) outcome of checklist.

ad 1) In the context of this study we are only interested in effectiveness of packages. Other types of criteria will not be taken up in our checklist approach.

ad 2) Essentially, the level of detail in which functional requirements are stated determines the precision of the evaluation process. Apart from the ambiguity aspect discussed in the previous paragraph, an evaluation also depends on the level of detail in which software package documentation is available. Usually, the available documentation will not be changed by the supplier to individual potential customer requirements. Therefore, the level of detail used in package documentation is mandatory for the level of detail in which functional requirements from the user-organisation will be determined.

ad 3) Having functional requirements presumes that there is a procedure for the determination of functional requirements in each individual organisation which is interested in designing and using information systems. Because lack of sound detailed functional requirements is a major drawback in current evaluation practices, we must assume that there is no satisfactory approach or procedure yet aimed at the determination of detailed functional requirements.
On the determination of functional requirements in a maintenance environment

ad 4) The purpose of a checklist is not to arrive at a recommendation of the most effective package. It merely provides insight of the possible shortcomings of the packages submitted to an evaluation. A package is considered to provide a maximum of effectiveness if all functional requirements are met. Operationally, a checklist facilitates only a "binary" judgement, i.e. a criterion is available in a package or not.

In case no effective packages are available, in-house extensions or even complete development projects will be a next option. Since functional requirements should reflect the "best" solution in terms of working procedures for a given organisation function irrespective of the way these requirements will be realised, they can equally well serve as functional requirements for (complete) in-house development projects. Then, as a matter of course, previous decisions concerning the level of detail in which functional requirements are formulated have to be reconsidered, and a maximum of detail is required. In the previous chapters we already observed a general lack of suitability of organisation function specific packages in practice. As a consequence, we may expect that in-house development or extensions to available packages will be necessary in most situations and thus we must address the determination of functional requirements for in-house development as well. We shall explicitly extend our research objective to the determination of functional requirements for the maintenance function in general, serving as a basis for both, evaluation of standard software packages and in-house development.

3.4 Methodology

3.4.1 Objective of this thesis

Definition of the problem area

In the introductory chapter we already made clear that the unsatisfactory performance of information systems in maintenance is our main concern. In particular, we are interested in a special type of information systems, i.e. the so-called standard software packages for maintenance. Currently, numerous packages for the maintenance function are commercially available. Organisations with an internal maintenance function are interested in the suitability of these packages as an alternative for in-house development. Very often, in vendors package...
advertising universal applicability of their packages is suggested. In addition, an increase in maintenance function effectiveness and efficiency is promised. However, literature, observations and discussions with actual users during numerous Msc research projects carried out at the Faculty of Industrial Engineering and Management Science (Eindhoven University of Technology) demonstrate that vendors suggestions and promises are not realised in many situations.

In this study, we shall analyse this problem from a users point of view, i.e. we shall not attempt to develop an universal applicable software package, because of the expected inherent complexities of potentially controversial functional requirements that must be met. Users, having responsibility for a subset of all organisational functions, are primarily interested in the assessment of the potential of an available package in their situation. We shall define the process in which a package is assessed in respect of its applicability for a subset of organisational functions in an actual organisation as an evaluation process.

Current conventional evaluation techniques are seriously hampered due to the fact that they make use of a relative scale, i.e. the actual maintenance situation is not used as the standard to which a package is measured, and due to a lack of detail in the specification of the functional requirements (chapter 2). In most cases, the result of an evaluation is a ranking of packages according to the number of features available in each package. Lack of sufficient detail implies that functional requirements cannot be interpreted unambiguously. On the other hand, if an absolute scale is used, i.e. properties are directly derived from actual functional requirements in an individual maintenance situation, a checklist-like technique can be applied.

Functional requirements must be valid, complete and justified to allow their use for in-house development and evaluation purposes. The determination of valid, complete and justified functional requirements is a critical factor for success in performing an evaluation of an information system, with respect to packages as well as to in-house developed system. The majority of current approaches to determine functional requirements originate from computer science. Because computer science isn't aimed at particular organisational functions, i.e. no organisational function specific elements are taken up, these approaches can at best register user descriptions formally and consistently in terms of algorithms and conceptual database specifications. Computer science oriented approaches (chapter 4) do not provide a guarantee for complete, justified and valid functional requirements assessment.
The objective in this study
Given these findings, our primary objective in this thesis is to develop a framework, which specifies what steps are to be taken in order to determine valid, complete and justified functional requirements, and which can be used as the basis for the evaluation of information systems, aimed at the internal maintenance function in particular. In addition, the required conditions in terms of when the procedure should be carried out and who should be responsible for this procedure should be specified. In all, the "product" of this study can be typified as a framework for design.

A multi-disciplined approach
The determination of justified, valid and complete functional requirements is a complex task which cannot be solved within only one type of applied science. Functional requirements, in our context, are a function of both organisation function specific theory, actual responsibilities of personnel and computer science, being sub-areas in industrial engineering.

The first basic question in the evaluation of software packages or in developing in-house is what knowledge is required. Depending on the organisational function for which an information system is needed, a wide spectrum of design oriented scientific knowledge areas may be involved. In this study, we focus on the evaluation and development of information systems for the maintenance function. Consequently, the main knowledge area will be maintenance as a process to be managed. However, it is unrealistic to assume that maintenance management will provide the complete "input" for functional requirements.

The second relevant type of applied science we have to consider in our study is sociotechnology. In sociotechnology [33] a systematic approach is advocated in which complex formal control mechanisms are minimised by the use of personnel group structures. In fact, sociotechnology aims to make better use of human ingenuity and recovery capabilities needed in complex organisational functions, which can never be compensated for by strictly formal theoretical approaches employed by automation. This inherent human flexibility is regarded as a vital part of functional requirements.

In practice we may expect that chances are small that a standard package for a maintenance function will have 100 % effectiveness. Therefore, except for the assessment of a packages efficiency, also compensation measures ranging from simple adaptations to a package to
complete in-house development must be considered as well. To facilitate in-house development and to assess the efficiency of functionally approved packages, computer science knowledge is needed. In the context of this study we shall refer to computer science as the key applied science area for the realisation (or implementation) of information systems.

3.4.2 Research methodology

From a methodological point of view different approaches have been advocated on how research has to be carried out in various scientific disciplines. The research methodology strongly depends on the objectives of the researcher. At first, researchers may be interested in finding explanations for some observations, which are interesting to them in view of their particular field of research (e.g. archaeology). Validation of the research on the explanation of phenomena is carried out on strictly empirical hermeneutical basis. Hypothesis are formulated (=theory), then observations are made to assess "life" historical or simulated data, which then may lead to a potential falsification of the original hypothesis (= empirical research, [44], [81], [55]). A more ambitious objective in research is to predict the behaviour of the research artefact [72] in certain specified conditions, which is quite common in e.g. applied sciences.

Yet a different approach in research is to actively influence behaviour towards a more preferable predefined condition. We shall refer to this approach as design oriented research. In a scientific note from the "Instituut Vervolgopleidingen" (the EUT-IVO [11]) the following characteristics of a design oriented approach are highlighted:

- the result of design oriented research is an artefact, which is imperative, normative and prescriptive;
- the emphasis is on synthesis of knowledge elements which may originate from different scientific disciplines;
- the process of designing is a creative, ingenuitive process where the not-yet-existing comes into being.

In this particular note these, and more, characteristics are compared to traditional research methods which are primarily interested in increasing the "truth" and generalisation of
On the determination of functional requirements in a maintenance environment

findings. In addition, traditional design oriented research is aimed at the design of physical artefacts. However, in this study our design objective has an abstract nature, i.e. the design of a prototype framework for a functional analysis and design procedure. Therefore, it would be more correct not to refer to this project as being a research project but as a (prototype) design project of a framework.

Design of abstract tools is typical in industrial engineering science. There is a growing interest from methodologists ([82], [83]) to define industrial engineering design methodology. Further research into, in particular, the interrelationships of the participating scientific disciplines in industrial engineering design methodology is required.

3.5 On industrial engineering design methodology

In this paragraph the methodology used in this PhD design project will be discussed. Due to the lack of consensus in industrial engineering design methodology, we have to introduce our own solution to this problem.

The justification of our methodology is based on the consideration that a systematic approach in design is much more efficient and faster than a pure trial and error approach. Essentially, this means that as much information as possible is specified with respect to the purpose of the artefact to be designed and its "operating conditions", i.e. the design specifications, beforehand in order to direct the design efforts and thus to reduce the set of alternative solutions. Our methodology distinguishes five consecutive steps which have to be carried out. The consecutive steps are illustrated in figure 3.1.

We shall incorporate Van Engelen's and Van Der Zwaan's [83] idea of the industrial engineering design cycle in our design methodology. They distinguish five stages in the design cycle; design objective - design specifications - generate alternative designs - selection of suitable design - presentation of findings. Most of these stages can be translated into one of our design steps visualised in figure 3.1.

Step 1: determination of the design objective ("design objective")
First of all the design objective has to be defined. Various observations stemming from practical experience in actual organisations in addition to literature research may lead to the definition of a problem area. In case the problem area requires an "instrument" to facilitate a
change in the circumstances that lead to the problems identified in the problem area, a design oriented approach is needed.

Due to missing or too global functional requirements, risks are high that an unsuitable package will be selected or inadequate information systems for the maintenance function will be developed. A procedure is needed describing how valid, justified and complete functional requirements are determined. The development of a procedure including a specification of responsibility areas and the conditions to be met for an appropriate application of the procedure requires a design oriented approach. Existing industrial engineering knowledge areas (maintenance management, sociotechnology and computer science) provide the construction elements for this procedure. The development of such a procedure itself requires industrial engineering design knowledge.

**Step 2: determination of design criteria and constraints of the artefact ("design specification")**

The next step in the design cycle is the determination of the design specifications. Whereas the design objective only defines the criterion for ending the design cycle, the specifications
formally define the direction in which the design should go, i.e. the design criteria, and what boundaries are to be taken into account, i.e. the design constraints. In our situation, criteria and constraints originate from the three chosen applied sciences, maintenance management, sociotechnology and computer science.

Step 3: Prototype design ("generate alternative designs")
The prototype design process is the creative element in the design cycle. The prototype framework is constructed by combining knowledge elements originating from the three chosen applied sciences (=synthesis). The prototype is complete if all design criteria are met and no constraints have been violated. Each aspect of the designed prototype framework must be explicitly correlated to a design criterion. In addition, for each constraint a logical explanation must be presented in order to demonstrate that the prototype is not in violation of a constraint. An important characteristic of this design cycle stage is, that if it isn't possible to define at least a satisficing prototype in the first place, the entire design project already fails before the design cycle can be complete. In other words, at least one complete prototype must be produced. E.g. an incomplete prototype may be caused by missing knowledge elements from the applied sciences.

Step 4: Field testing
Testing a design alternative, i.e. a prototype, in an actual or at least in a simulated environment is required to validate the design objective, criteria and constraints. In addition, if at least one design specification exists for which during design no 100% guarantee can be given that the "designed-in" specification functions as intended, field testing is required. Since the practical usability is an important claim of industrial engineering design studies one can expect that at least the practicality criterion requires testing. Several tests may be required to ensure that the performance of the design corresponds with its specifications in all perceived conditions.

Step 5: Prototype acceptance evaluation ("select suitable design")
Finally, in a prototype acceptance assessment, essentially, two aspects are assessed:
1) the prototype performance with respect to the original specifications of the objective, and
2) the validity and completeness of the design specifications (=criteria and constraints).

As a result a prototype may live up to expectation and, hence, can be approved as the final design. Alternatively, depending on the nature of the prototype shortcomings any previous design stage is candidate to be reviewed again. Generally, the higher a design cycle stage at the top of figure 3.1, the more radical the change will be in the next prototype.

3.6 The status of this study

The four steps of design which have been carried out in this study are also presented in figure 3.1 in black. Step 5 and the design feedback (in grey) represent design steps which were not carried out, but nevertheless should be carried out if the ultimate objective is to go beyond a prototype design and to accomplish final accepted design.

The first activity in the design cycle is the formulation of the design objective. A problem area is defined with respect to the current condition which was presumably considered to be unsatisfactory in some way. In a preliminary analysis of the problem area it is concluded that to change the condition an instrument (an artefact according to Simon [72]), or a framework in our case, is needed. Since the design objective has been already discussed in the previous section we shall concentrate in this paragraph on the design specifications, the prototype design, the selection of a case situation and the prototype acceptance assessment.

3.6.1 Design criteria and constraints

In holistic organisation (function) design many aspects are tightly intertwined, making it difficult or even impossible for applied sciences to deal with each aspect in isolation. In figure 3.2 this situation has been visualised. The circled area marked as "unknown area" represents a possible area that doesn't fit in any known organisational function knowledge area or simply because it has not been identified yet. Only aspects within the context of each applied science are identified and analysed within its particular science context leading to different interpretations of the actual application area. Each type of applied science has its own limited perception on how a part of an organisation, i.e. the application area, is functioning. The relations with other aspects of an organisation are sometimes vague and
mostly taken for granted, because they "belong" to a different type of applied science. As a consequence, possible changes in one single organisational aspect are made without considering possible consequences with respect to other organisational aspects. A typical situation of suboptimisation (see also [28])- even risking the introduction of counterproductive measures - is created. This phenomenon is also referred to as the "piecemeal syndrome" (Skinner [73]).

Although, completeness, validity and justification are the basic design criteria which must be addressed by maintenance management science, these criteria have only a meaning within the span of control of the actual personnel groups involved in the actual situation. Therefore, socio-technical design criteria are valid in our situation as well. A major contribution to the costs of realisation is brought about by the effort required to develop the information system based on the specified functional requirements. Cost constraints of this type may limit the potential benefits of new or revised functional specifications in the short as well as in the long term.
3.6.2 Synthesis of applied sciences (prototype design)

In our approach we shall convert individual contributions of isolated applied sciences by combining them to an overall industrial engineering design approach (see figure 3.3). In our view, the applied sciences must be tightly interfaced to achieve an integral organisation.

**INTEGRAL DESIGN OF THE MAINTENANCE FUNCTION**

![Diagram showing interfaced applied sciences.](unknown area)

*Figure 3.3 "interfaced" applied sciences.*

function design without suboptimisation, inconsistancy or even contradictory design options. In this study a first attempt will be made to develop such an integral approach with special reference to the internal maintenance function even if it is only one of many different organisational functions. As a result of this approach we are not only determining functional requirements, which is our original objective; in doing so we are also (re)designing maintenance procedures and the areas or responsibilities of personnel groups in order to attain a better fitting to the organisational goal.

In fact, as a consequence of this integral approach, functional requirements are just one product of our framework. Above all, the justification, validity and completeness should lead to improved maintenance practices specified and applied by the organisational function experts in the field.
On the determination of functional requirements in a maintenance environment

Since the prototype design is achieved primarily by a synthesis of existing knowledge in the field of the three applied sciences, completeness and validation of the resulting functional requirements are essentially limited by the current "state-of-the-art" of each knowledge area. It will be demonstrated that especially in the area of maintenance management several enhancements are needed.

In the area of maintenance management we are forced to develop additional maintenance theory with respect to the structuring of maintenance planning and control situations.

In sociotechnology we enhanced the Integrated Organisation (Re)design (=IOR) approach with respect to the parallelisation and segmentation principles in case semi-autonomous maintenance groups are considered to be a valid option.

These enhancements cannot be regarded as available knowledge, because they are hypothetical and haven't been tested, and therefore they must be regarded as assumptions. Depending on their nature a different methodological approach, e.g. empirical research may be necessary for validation. In this study this additional research will not be carried out.

Due to our specific focus on the maintenance function, other organisational functions, such as production management, financial management, purchasing management, are not equally dealt with. However, since a substantial number of considerations with respect to the prototype design stage (chapter 5) are not specifically related to the maintenance function, other conceptual models to be used for other organisational functions may benefit from our approach as well.

3.6.3 The case study

In this study an actual maintenance situation is used as a case study to demonstrate the prototype framework. Although, ultimately the framework is intended to be used mainly by the personnel responsible for the maintenance function, in this explorative case the framework was used by the researcher himself. In addition, the outcome of the application of the framework is hypothetical, since suggested changes in functional requirements were discussed with the actual maintenance group, but at the time of writing they have not been approved for realisation by the management yet.
This study's methodology

To be able to demonstrate the framework realistically, special care has been taken to select an actual maintenance situation for a case study. The following criteria have been used for selection:

1) limited maintenance planning and control complexity;
2) valid maintenance concepts, i.e. concepts that have consequently been used in planning and control of all maintenance activities, are used for at least a year;
3) availability of historical work order data for the period in which valid maintenance concepts have been used;
4) willingness/interest of maintenance personnel to comment on the application of our new design framework.

The maintenance department of the Dutch PTT located in a main sorting centre in 's-Hertogenbosch met all criteria.

3.6.4 The prototype acceptance evaluation

A true prototype acceptance assessment is only possible if the prototype framework has been applied under "normal" conditions. In our case study, not all elements of the framework have been applied completely. E.g. the detailed functional analysis and design (=DFAD) process was only used to illustrate how one topic (the share of user inflicted down time in the total account of maintenance activities) relates to a change in functional requirements.

In addition, an important operational condition of the framework is that actual users perform their own DFAD-process, which wasn't the case here. As a consequence further assessment by real users for a longer period of time (let's say for at least one year) of the practicality of the framework is necessary.

The other design specifications, i.e. the justifiability, validity and completeness of the resulting functional specifications by applying the framework, were examined on the basis of primarily theoretical considerations; they are more or less indifferent to actual testing in practice.
3.7 The determination of functional requirements in an industrial engineering context

In this study, we shall assume that an organisation consists of three major elements:
1) sets of interrelated functions (= functional requirements);
2) design processes, and
3) actual means of realisation.

3.7.1 Functional requirements in maintenance

Since we are interested in maintenance carried out within organisations we have to define what is meant by "maintenance in an organisation". In this study we shall refer to maintenance as an organisational function.

An organisational function is:

*a high level area of responsibility defined by a scientific or professional speciality, and contributes to the objectives of an organisation*

Maintenance consists of all activities intended to keep technical systems in, or restore them to the condition considered necessary to fulfil their intended function.

The maintenance function is an organisational function, where maintenance is the scientific and professional speciality.

Functions facilitate a description of phenomena in terms of activities having a direction or goal rather than pure identification, which would be equivalent to black box modelling. Because we are interested in the procedural mechanisms of maintenance a functional representation of activities is needed. The term function originates from systems theory, which is based on "thinking in terms of processes or functions". According to Blumenthal [15] an organisation is a functional unit, described as an ordered set of subfunctions. The functional unit includes all procedures required for decision making and personnel responsible for the functional unit.
In this study we shall further refine Blumenthal's concept of functional units by separating functional units into distinct components each designed with different design objectives. Blumenthal's definition of a functional unit can be translated into our context as:

*A functional unit is a subset of an organisational function, which has been allocated to a group of personnel (the personnel system)*

Geraers [37] developed the so-called EUT-maintenance model, which defines all relevant main functions and relations of the maintenance function from an industrial engineering point of view. The model is used as an aid to show students, researchers and practitioners what aspects are defined in maintenance as an in-house activity. The model represents also a structure in which the completeness of current knowledge in the maintenance area can be assessed. In our case this EUT-maintenance model is of particular interest, because it provides a complete and universal definition of all main functions of an internal maintenance function. In this thesis we shall accept the definitions made in the EUT-maintenance model.

Functional requirements can be interpreted as formalisations of what personnel in an organisation function "is supposed to do", i.e. as procedures and practices. From an industrial engineering perspective, the process aimed at the determination of what functional requirements are valid in an individual maintenance situation is typically a design activity. Theoretical and contextual knowledge and constraints are used creatively to design procedures and practices for the organisational function at hand.

Essentially, our research objective will be the determination of a framework of a functional design procedure for the maintenance function. Normally, this design procedure cannot assume a green field situation. Current (maintenance) procedures and practices must be verified for suitability first and possibly changed to accommodate up-to-date (maintenance) practices. An analysis of the current procedures and practices should be an essential element in the functional design procedure. We shall refer to this procedure as the functional analysis and design procedure.

We already discussed the relationship between the level of detail in which functional requirements are defined and the skills and theoretical background of personnel in the
On the determination of functional requirements in a maintenance environment

previous section. Basically, this means that functional requirements depend on the actual organisational personnel structure and vice versa. Similarly, some formal functional requirements have to be realised by information systems, but in case information systems technology is restrictive, functional requirements have to be revised. Because of the relationships of the functional analysis and design (=FAD) process with the personnel structure and information systems technology, we cannot develop a FAD framework in isolation. The FAD-process is related to the personnel system analysis and design (=PSAD) and the information system analysis and design (=ISAD).

In general, this view of the relationship between FAD-, PSAD-, and ISAD-processes is supported by authors from all three disciplines, i.e. organisation function specialists, socio-technology and computer & management science. Geraerds [38] and Nielen [67] emphasise that a thorough understanding of organisational functions is required in the first place. v. Eijnatten and Loeffen [32] state that a control structure only makes sense in relation to socio-technically designed groups. The control structure reflects the responsibilities groups have within an organisation. They emphasise that socio-technically designed groups with their "allocated" control structure should serve as a

![Diagram of PBI model](image)

Figure 3.4 The PBI model \( I=F(P,B) \).

basis for designing information systems. Bemelmans [7] formalises the general idea into the so called PBI model (figure 3.4).

This model refers to essential aspects of systems theory. The P-aspect represents the systems in an organisation which have to be controlled. The B-aspect represents the system which controls the P-system, i.e. all organisation function logic comprises the B-system. The I-aspect represents the information which supports the B-system. The arrows in the PBI model indicate in what direction the three systems depend on each other. The B-system can
be designed only when the P-systems has been designed already. In turn, the information system can only be designed after the B-system has been completed.

Both the personnel system design and the information system design processes use functional requirements but in a different way. Due to the different design goals different functional requirements, i.e. the output of the functional analysis process, are needed as design input.

3.8 Developing a functional analysis and design procedure

The considerations presented in the previous paragraphs, demonstrate that the original research objective, namely to develop a suitable evaluation technique, is too limited in scope. Improving current evaluation approaches is not simply a matter of developing a new evaluation technique. A more fundamental approach is required. We have to extend our research objective to the development of a functional analysis and design procedure aimed at the determination of functional requirements, considering the context outlined in the previous paragraphs. These requirements should be used to determine, in an individual maintenance situation, the effectiveness of a standard software package or to serve as a basis for in-house development. The functional analysis and design process tries to meet the design objectives derived from the general organisation objectives by defining (terminology) functions, structuring functions (framework) and by determining valid unambiguous procedures ("recipes"). The outcome of a functional analysis imposes requirements on both, the selection and knowledge of the personnel responsible for the realisation and execution of the functional requirements.

In fact, a functional analysis and design procedure has to fit seamlessly into the other organisation design processes, which all together have the overall objective to design a complete operational functional unit.

We shall define the process which has the objective to determine the skills and numbers of personnel including the areas of individual and group responsibilities as the personnel system analysis and design process.

As a result of the personnel system analysis and design, functional requirements are enhanced to "fit" within the organisational structure and to meet detailed personal requirements. Then, functional requirements may result in an evaluation & selection process or an in-house development to realise an information system as has been already outlined in chapter 2.
On the determination of functional requirements in a maintenance environment
4 Mono-disciplinary aspects of functional analysis for maintenance

4.1 Introduction

Before a framework for functional analysis and design can be developed we have to know what criteria for the design of such a procedural framework are valid. At first, we shall clarify the dependency of the functional analysis and design (=FAD), the personnel system analysis and design (=PSAD) and the information system analysis and design (=ISAD) processes in general. We will do this by analysing the dominant and widely accepted design approaches with respect to how they "interface". Then, based on the nature of these "interfaces" we shall determine design requirements for our framework for functional analysis and design.

4.2 Design requirements from a organisation function specialist perspective

Each organisational function has its own goals. E.g. the maintenance function tries to minimise the consequences of failures occurring when technical systems are used. The degree to which an organisation function achieves its goal is referred to as the organisation function effectiveness. This effectiveness is not only determined by the technical quality of performing physical activities but also by the way in which the activities at shop floor level are managed.

It is clear that the available knowledge of proven theory with respect to how a organisational function should be designed, in comparison with the complexity of an organisational function in a particular situation, eventually limits an effective design of the functional unit [38, 67]. As organisation function theory evolves, a need may arise to revise current procedures and practices. In other words, functional requirements have to be adaptable. Currently, conventional organisation theory and research have primarily a hermeneutical and not an instrumental, design oriented nature. This makes the conventional organisation theory less suitable for our design oriented framework.
At first sight, it seems desirable to formalise as much as possible managerial procedures. Especially, automated information system's effectiveness seems to benefit from maximised formal functional requirements. Experience in the field doesn't support this opinion. Kramer [53] reports large difficulties in the effort to automate 100% of all functions by developing large complex information systems for production planning and control in a steel mill. On the other hand, he experienced having little difficulty in automating the 90% which concerns mainly well definable routine functions. The automation of all kinds of exceptional situations which do occur from time to time unexpectedly, has failed completely. In addition, those exceptions may have a negative effect on the implemented routine procedures. Ideally, to automate exceptional situations successfully all exceptions must be known beforehand, and, for every one of them a formal model has to be developed and implemented in a form of software. Apart from excessive implementation costs -Kramer reports up to 5 times the implementation costs of the routine functions are reported- it is virtually impossible to have total knowledge of all potential extremely varying exceptions beforehand in complex organisation functions. Human creativity and flexible data-retrieval possibilities are believed to be of vital importance in dealing with exceptions in the steel mill and other complex situations.

Wijngaard [79] also warns for too much formalisation. Especially, formalising procedures into every detail restricts users in making more suitable non-routine decisions. In our framework it is required that special attention is paid to avoid rigid formalisation.

The most important aspect from a organisation function specialist point of view is the effectiveness of functional requirements. Essentially, a designed function must be correct in view of the desired output. In other words, a designed function must be existent and valid in order to be sure that it provides the complete desired output in all its known and relevant variations. Our framework will address this issue as the justification and validity aspect of functional requirements.

4.3 The personnel system design requirements

The objective of this analysis and design process is to define an organisational structure in terms of areas of functional responsibility and to allocate individual staff members to these areas of responsibility.
This theme has received much attention in literature and many approaches and opinions are advanced. In traditional organisation theory very little attention has been paid to methodological approaches how an organisation design should be carried out. By contrast, in sociotechnology design principles have been developed to support an organisation design in a practical way.

Of the design approaches that have been developed in sociotechnology [33] we shall concentrate on the IOR (=Integrated organisation (re)design) approach described in the following section.

4.3.1 The integrated organisation redesign approach

This approach enhances the general socio-technical design principle, which emerged in the early days of sociotechnology and is the basic goal of socio-technical design approach (Trist et al. [76]). The IOR approach has been described extensively in [34], [3], and [48]. The basic philosophy of socio-technology is to reduce centralised and functionally specialised staff and to bring as much control power as low as possible in the organisational hierarchy. In its optimal form organisations would consist of semi-autonomous task groups and multi-disciplinary operational groups. High level business-unit-like structures decrease the need of centralised specialised staff support.

The integrated organisation (re)design approach can be characterised as holistic for two reasons:

1) The personnel systems design is explicitly interfaced with other organisation design processes, such as the (physical) production process design, the information system design, and more implicitly, the functional analysis and design.

2) All hierarchical organisation levels are covered (micro-, meso- and macro level).

In view of our objective to identify what functional requirements are needed, two design steps in the IOR approach are of interest in particular, i.e. the design of the control structure and the substep named "listing of control activities". The low level control activities are allocated to task groups in support of operational groups in such a way that the activities, or functions in our terminology, have a greater cohesion within a group than between groups. If applied in a production organisation this allocation principle results in paralleled and
segmented production flows. The important factor of cohesion is related to a wide area of organisation dependent (functional) aspects. Technical design of production process, production planning and control structures, maintenance planning and control structures are all examples of cohesion factors. Essentially, the cohesion factor must ensure that the need for formal co-ordination outside the groups is minimised, whilst the internal co-ordination between group members takes place in a natural and preferably informal way.

In the IOR approach a bottom up design is advocated. Task groups on shopfloor level are to be formed first (micro level), then operational groups (meso level) and further up to plant or business unit level (macro level).

4.3.2 The organisation function as a cohesion factor

Most socio-technical organisation design theory has devoted attention towards a flexible organisation design centred approach in which the production function is regarded as the core, the primary function of an organisation, whereas other functions are supportive, i.e. secondary, to the production function. Other, secondary organisation activities are simply taken up in the (production) group as an additional responsibility or are to be allocated to specialist groups.

Although in sociotechnology no explicit directives have been proposed with respect to secondary organisation activities, we shall assume socio-technological design principles in general and the IOR approach in particular do apply here as well.

E.g. this would mean that some more complex maintenance activities, which cannot be allocated to production oriented task groups, have to be allocated to specialist maintenance task groups.

Even specialist operational groups may be needed in situations where complex specialist control activities are identified. E.g. in the chemical industry large shutdowns require careful planning in advance and quick response control actions during execution. The high demand for specialist knowledge and experience would require more specialised operational groups having a more narrowed scope than would normally expected in regular operational groups that have to deal with a multitude of organisational functions.

In sociotechnology a cohesion factor represents a grouping criterion to form task groups. Basically, a cohesion factor defines a commonality that may distinguish one group from
another and builds a bond between group members. An organisational function can be an important cohesion factor. If subsets of organisational functions contain feedback loops, then the personnel responsible for carrying out activities is able up to a high extent to evaluate and correct their actions. We shall denote these functions as structural analysis and design functions.

In our terminology structural analysis and design functions can be interpreted as a set of interdependent unallocated (specialist) control activities. In socio-technical design (see Van Amelsvoort [3]), control activities are the basis to form task groups. In addition, the capacity required to carry out control activities is essential in determining group composition and size. To accommodate socio-technical design we distinguish the following functional input information must be provided by a functional analysis and design process:

1) The structural functions (activities) to be allocated;
2) Guidelines on the way in which functions can be paralleled, and segmented, if necessary without sacrificing too much cohesion.

4.4 Information system analysis and design requirements

4.4.1 The information system infrastructure

A trend exists towards more integration of information systems, originally intended for individual organisational functions, and an increased need for flexibility (Bemelmans [7]). In an computer science context, integration can be interpreted as the coupling of different organisational functions. The output of an organisational function invariably constitutes the input for other organisational functions. Coupling of functions is not a technical issue only. When viewed within this context, the trend towards more integration means that organisational functions will tend to become more tightly coupled. Tight coupling manifests itself by an increased number of shared data-elements and an increase in data-exchange. The need for more flexibility can be largely attributed to an increased number of users requiring the possibility of rapid changes in functional requirements. The need for quick
adaptation of organisation practices and procedures conflicts with the long times required for the adaptation of existing software.

There is a price to be paid for increased integration and flexibility. Most realisations of information systems require a relative high investment of organisational resources, moneywise as well as specialist capacity-wise. In view of these high investments, it seems unattractive to depreciate these investments completely each time an information system has been realised, and to start all over again as in a green field situation. Doing so would result in a low efficiency of the information system analysis and design process and possibly in lengthy development cycles. To counteract potential inefficiency in development and use of information systems we must distinguish the long lasting "investment" elements of information system realisation from the elements more susceptible to changes. The long lasting elements are named infrastructural elements, because they are supposed to be application independent and, hence, applicable or valid multiple times. In other words, the development of information systems for individual organisational functions must recognise information infrastructural constraints. Basically, two primary approaches for an infrastructure are advocated:

1) realisation of infrastructural elements by applying means and methods with an intrinsic flexibility, and
2) standardisation of infrastructure elements.

ad 1) The principle of this approach relies on maximising the use of multi-purpose means and methods and therefore minimising the need of adaptation for each new project.

ad 2) Standardisation is in fact a measure to reduce diversity in infrastructural elements. Essentially, standardisation questions the statement that each individual requirement would require its individual solution. Standardisation assumes that the same solution can be applied over time or in different but comparable situations.

Bemelmans [7] distinguishes four types of infrastructural elements:
- the technical infrastructure;
- the data and knowledge infrastructure;
- the application infrastructure;
- the organisational structure of information system management.

Each of the four elements will be discussed with regard to the consequences if the two basic approaches of realisation are applied.

The technical infrastructure

The technical infrastructure element refers to all components for common and shared use (such as large computers, peripheral equipment, communication devices, systems software, etc.). Clearly, an increase in cross-organisation data traffic will require investments in hardware to increase capacity in terms of computation power and data communication means. Since data traffic intensity is insensitive with respect to the contents of the digitised messages, the need for flexibility concerns primarily the volume aspect of data traffic. Practically, an increase in flexibility can be achieved if computer power and communication speed is large enough to support the estimated performance requirements in a foreseeable future. Additional (intrinsic) flexibility must be provided as a design feature of most computer and communications hardware components and therefore, in most situations, increasing flexibility is pure a matter of pushing the technological limits.

Standardisation of hardware can be an approach worth of consideration. Sometimes, standardised hardware can reduce the need of highly trained information system analysis and design (=ISAD) specialists who have to deal with a variety of technical equipment. Apart from employing expensive specialists, who must keep up with technical advancements, it will be a difficult and time consuming undertaking in a situation where a large variety of technical infrastructural means are used and technology is still advancing rapidly.

In this study we shall assume that sufficient inherent technical flexibility is available, or will be made available as the need arises.

The data and knowledge infrastructure.

This infrastructure element consists of all shared data definitions in an organisations. This element represents a rather complex issue, because we are confronted with a complicated and sometimes ambiguous trade-off in flexibility versus standardisation. In assessing their objectives, organisational functions can not be realised as completely isolated entities.

Normally, organisational functions are up to some degree dependent on data-input from other

\[ \text{e.g. the number of shared data-elements times the frequency of shared use.} \]
organisational functions and in turn, provide output which is input for other organisational functions. In addition to the problem of determining the functional requirements to meet the objectives of an organisational function - which is our objective in this study - we have to decide what to do if more than one function uses the same data elements. Clearly, this problem is valid in view of an efficient realisation of functional requirements. Basically, this problem can be solved in two extreme ways. The one extreme is that each organisational function produces its own data, i.e. functions will be duplicated in each organisational function. The other extreme is that one organisational function is the sole creator, whereas any other function can only passively use those data-elements (see Martin [59]). The first extreme guarantees a minimum of mutual dependency, but potentially, duplicates functions and can cause inconsistent data. The second extreme conforms to a well advocated computer science paradigm. The objective of this paradigm is to input, or create and store, an occurrence of a data element only once, and from then on make it available to all other functions needing this data element without recreating it. In addition, the likelihood of faulty input and inconsistent data will be reduced as well.

In practice, a consequence of this paradigm is that an authority will be assigned to act as the guardian with respect to unauthorised changes of data elements which otherwise could be introduced by various different organisational functions. A frequently encountered metaphor for this form of arbitration is based on the effect of eliminating local independent information systems, the so called "islands of automation". Islands of automation have to be connected by making them completely transparent to each other, thus, creating a new single "mainland" with no barriers in between. Central arbitration is required in case conflicting requests for change emerge. Unfortunately, if this paradigm is realised too rigidly some severe problems may arise.

The most important consequence is an increase in complexity, because an important part of functional analysis and design of multiple organisational functions has to be co-ordinated on a central level. In our context, arbitration in conflicting data requirements requires an assessment of all functional design options of all organisational functions being involved. In practice this process may prove to be too difficult to solve due the sheer number functions and data elements involved, which sometimes are not even understood completely in advance.
Another consequence of centralised arbitration is the additional amount of time and effort needed to carry out the arbitration process. Clearly, centralised arbitration has a counterproductive effect on the increased need of users being able to change functional requirements fast. In addition, functions from other organisational functions which rely on the output of these functions create an obligation for the provider/creator of the output with respect to continuity.

A possible alternative to central co-ordination and arbitration is to allocate the responsibility to decide upon whether output from other organisational functions will be used to the local organisational function designers, i.e. the actual users. This responsibility would require that each change of data-types is reported and made available to every user. Designers of functions, or professional users must have the disciplin to verify their functional and data specifications for duplicates by using the list of data elements. At a later stage, local users and developers together can make the trade-off between less effort, reduced chances of faulty data and more consistancy versus more dependency. E.g. a trade-off can be based on:

- the essentiality of a particular data element;
- the effort to input/update/delete a particular data element, and
- the data input/update/delete frequency.

In terms of the "island" metaphor, the "islands" decide if it is desirable to build bridges to other "islands". In this approach the complexity of central arbitration is reduced, but it may take some time before efficient cross function connections are determined. Because in this study we are interested in the functional requirements aspect of information systems only, we shall concentrate on the verification of data-element as soon as new data-elements in a functional analysis and design process will be determined. In the context of this thesis we shall regard the trade-off issue as an efficiency aspect.

The application infrastructure

This infrastructure element refers to all common information system components in use or under development. An application can be regarded as a module which can be analysed and designed mainly in its own right, with regard to its functions. In computer science the application infrastructure is of particular importance with respect to the speed of
development. In the introductory chapter to this thesis we already discussed the importance of reducing development times. The demand for flexibility in changing software to dynamic functional requirements is one of the key problems in application development. It goes without saying that one way to speed up the development of applications is to utilise higher level programming languages and specialised development tools. In addition, a more selective development process may increase development speed as well. It is assumed that any application for an organisational function consists of more or less standard or generic parts and highly specific parts requiring custom design. In an individual situation it is hoped that sufficient standard or relatively stable functions can be identified and only a small portion would require customisation. Modern standard software packages typically aim at the potential of selective development. In the next section we shall discuss both application design factors more in depth.

Custom development

The flexibility of applications, in terms of the degree of ease in which an application can be altered or increased, is determined by many factors, e.g. the software tools and the programming language used, the complexity of the application itself, the productivity of programmers, and so on. However, although there is a trend towards software tools aimed at enlarging development productivity, mainly the available capacity of the developer proves to be the ultimate bottleneck. A recent approach to ease this problem is to encourage end-users to take care of the realisation of their own functional requirements within their personal scope themselves (end-user computing).

Standard application components

Standardisation of application components represents in some way a controversial approach, since standardisation implies that dynamics or individual differences can be solved by adding the appropriate software components just like building blocks are put on top of each other to construct a building. In this study we deliberately chose the individual maintenance situation as the primary base on which functional requirements depend. Practically, standardisation assumes that, for all possible maintenance situations a complete set of valid standard procedures and practices exist. In computer science the idea of constructing complete information systems by linking up standard modules has evolved into constructing
standard blue prints called reference models [42]. However, the approach prescribing how valid reference models in maintenance can be developed and how they should be used successfully remains unclear.

Wortmann [10] demonstrates how standard MRP\(^2\) functions in different elementary production management situations are related to characteristic conceptual data models. He emphasises the growing importance of data sharing between applications, operating in different functional areas (see previous section). Therefore all information systems should concentrate on achieving some sort of standardisation on a conceptual data level. In addition, developers of software packages should realise that data structures may have different "life expectancies". E.g. Wortmann distinguishes state-independent data from state-dependent data. State-independent data constitute the data related to the long term aspects in an organisation, such as the production layout, the products and their bill of material, etc.

State-dependent data structures are related to frequently changing aspects, such as updating the inventory levels of products, the status of a production order, etc. The longer the "life expectancy" of data structures the more likely different organisation decision functions will share these basic data and therefore the more difficult it will be to change them quickly. Consequently, evaluation of information systems should concentrate on whether a system supports these long term data structures in the first place.

However, using data structures imperatively raises some questions. From an organisational function specialist view, in this study the ultimate problem-owner, the procedures and practices in order to maximise his performance are imperative. Therefore, we shall regard formalised representations of procedures and practices, i.e. the functional requirements, as the standard to which information systems are measured in an evaluation. The data-structure (the data types and constraints) is regarded a derivation of the functional requirements. So to speak, the derived data-structure can be considered a secondary standard in an evaluation.

Data structures are in fact justified, valid and complete if the functional requirements are justified, valid and complete, not the other way around. The dynamics in an organisational function are caused by a multitude of factors, some of which may be highly unpredictable. Sometimes the resulting changes in functional requirements may call for an adaptation of the data structure. Since data structure located in the inner layers are more difficult to change, because most likely they are relevant for multiple organisational functions making a redesign

---

\(^2\) MRP = Materials Requirements Planning, a frequently advocated production planning and control technique
On the determination of functional requirements in a maintenance environment

a necessity as well, high costs will be the primary consequence. Consequently, we can conclude that if the user, being the primary problem-owner, is served best if evaluations are based on functional requirements then data structures are best regarded as an efficiency criterion relevant during the selection stage (see evaluation and selection approach presented by figure 22).

The organisational infrastructure

The organisational infrastructure refers to the group structure of development specialists. Essentially, such a specialist group structure can be regarded as a functional unit in its own right. Consequently, socio-technical principles on the allocation of tasks and responsibilities do apply here as well. Since we are interested in functional units in maintenance we shall consider the development and redesign of information systems as a separate organisational function requiring its own framework.

4.4.2 Formalisation of specifications

Computer software and database design requires unambiguous formal specifications. Procedures have to be specified with mathematical precision and the required information needs must be defined in terms of datatypes and constraints. In chapter 3 we discussed the context sensitivity of ambiguity in specifications. The individuals knowledge and background experience largely determine what can be taken for granted. From a computer science point of view, there is little margin with respect to organisation function specific knowledge and background experience with computer scientists in most situations. Consequently, functional requirements have to be formatted into rather universal mathematically oriented specifications to exclude potential misinterpretations (E.g. see v. Hee [46]). However, most users may feel quite uncomfortable in mathematical formatting, a skill which many computer scientists have developed as a part of their basic training. In the traditional approach of information system development much emphasis is laid on the formal specification aspect and relatively little on the user domain. Normally, a computer scientist would interview a (future) user on his activities in an attempt to assess his information needs and to identify possible inconsistencies [29]. If the computer scientist believes to have sufficient knowledge of the user's wishes he will develop an information
system according to his formal specifications. It comes as no surprise that many information system development projects fail, because of the superficiality of this approach [8]. Improving communication by primarily better training of communication skills of computer scientists would reduce the problem.

Despite the short history of information management science a diversity of methods have been published in literature (e.g. ISAC, SADT, SDM, IE [13]) aimed at the analysis and formalisation of user requirements. Some of these advocate a topdown analysis, comparable to the approach presented in this study, others prefer a bottom-up approach. Some pretend to cover only certain management area's and only one hierarchical level within an organisation or department. The majority of the requirements analysis methods make use of graphical - more user friendly- representations of data and of processes in an attempt to enable users to influence the formal specification process. In general, these approaches are geared towards narrowing the communication gap between users and computer scientists. Some approaches concentrate on determining the required data elements whilst others focus on a more function-oriented approach. There are also approaches which advocate both function and data orientation in subsequent analysis stages. These approaches are designed to be universally applicable, i.e. irrespective of any (typical) organisation specialist knowledge area. They are dependent on the input of organisation function specialists and on the communication capabilities of the development team and the actual users.

However, from a methodological point of view apart from consistency checking, the completeness, validity and justification issue is, grosso modo, left at the users discretion. In this study we cannot eliminate communication problems, but the need for communication can be reduced by a user driven approach. Instead of trying to make the computer scientist an expert in any subject and all the disciplines involved he could be confronted with, which is utterly impossible, we could ask users - being the ultimate experts - to analyse and format their functional requirements relevant for information system design more formally. In addition, the functional requirements to be used as information system specifications for custom development should be specified in terms which are not organisation function specific by the users themselves in order to enable interpretation by non-organisational function specialists. Then, in an ideal situation computer science specialists would only need to communicate with users on unrealistic (too expensive) or conflicting specifications. This
requires that, our framework for functional analysis and design should be used by (end)users as much as possible.

4.4.3 Individual and user group preferences

It is clear, that individual preferences may differ substantially. Preferences may range from anything like procedures and practices to timing and user interfaces. Although many of these personal preferences have no relationship with our research objective, timing aspects may relate indirectly to functional requirements. Timing, i.e. the time a user is prepared to wait for a computer response after he initiates a change, depends on the complexity and computational requirements of the information system, imposed by functional requirements. E.g. a work order planner is trying to determine acceptable levels of workload. Functionally, in scheduling work orders he looks at a workload bar chart to determine the schedule of a new work order. In case he is dissatisfied with the resulting workload he will reschedule the work order until he has reached an acceptable solution. Frequent "what-if" type of computations will benefit from a quick information system response, because otherwise the order planner will get demotivated after a while. Preferences may also appear as a formal requirement. E.g. maintenance planners have a meeting with production supervisors each morning to decide what maintenance jobs should be carried out that day. For every meeting a list of planned work orders must be prepared to serve as input information to the planners and supervisors. In this situation the compilation of such a list each morning is a formal functional requirement. (Formal) functional user preferences also have to be taken up in our framework.

4.4.4 Conclusions on the information systems development and selection aspect

In our functional analysis and design process we have to address information infrastructure elements. The application and the data and knowledge infrastructure are of special interest in this study. To reduce development efforts both in time and cost special attention must be paid to the way in which functional requirements affect the infrastructure. Since the validity of the application and data infrastructure are based on functional requirements, their period of validity (in other words the "life expectancy") indirectly also
determines the period of validity of these infrastructure elements. If this property is added to the format of functional requirements, developers can relate development effort and costs of maintaining the infrastructure elements that are subjected for change. E.g. in case a revised functional requirement necessitating costly adjustments to the infrastructure, is tagged "experimental" and therefore has only a very limited life expectancy, it is unlikely that the infrastructure actually will be considered unless the experimental change has some extraordinary benefits for the users.

4.5 The organisation design processes

Now that we know what organisation design processes are relevant in our research context we can define their relationship formally. We shall develop the basic idea of the PBI model further to demonstrate how the three design processes are related. Essentially, the functional analysis and design (=FAD) process provides functional requirements to the personnel system analysis and design (=PSAD) process, which in turn allocates functional requirements to individuals and groups who, in the end, decide what functional requirements should be automated and how it should be done by the information system analysis and design process.

![Figure 4.1. The relationship of the functional analysis and design process with other organisational design processes.](image-url)
However, the FAD-process cannot be a single process, which can be carried out in one go. In case the entire FAD-process would be regarded as input to the PSAD-process, the human background would be excluded totally. On the other hand, if the personnel system analysis and design (=PSAD) process would be regarded as an input-source to the functional analysis and design (=FAD) process, the PSAD-process would miss essential elements such as a listing of the control activities required as input for a socio-technical IOR design.

Consequently, we have to separate the FAD-process into two subprocesses. The first FAD-subprocess (the SFAD = structural functional analysis and design) has to provide the structural functional requirements, which are valid theoretically, and which are situationally sound but independent of the human involvement. Naturally, one may expect that this type of functional requirements is not unambiguous for information system analysis and design (=ISAD) designers, but still is suitable for both the PSAD design and is logical for the individuals in groups as a startingpoint to detail the structural functional requirements of their own detailed requirements. We shall denote this second FAD-type subprocess as the detailed functional analysis and design (=DFAD) process. The DFAD-process starts when the PSAD-process has been completed. The group individuals themselves are responsible for the DFAD-process. In figure 4.1. the basic relations of the organisation design processes have been visualised.

The processes and their interrelationships shown in figure 4.1 represent only an elementary view of basic dependencies between the processes. Special design input for the PSAD and ISAD-processes has been omitted, because we are only interested in the relationships with respect to the FAD-process. In addition, numerous feedback relationships have been omitted as well. Since any organisation design process can be confronted with requirements from a previous design step which cannot be met, a feedback signal from any step to an earlier step may result.

The SFAD receives input from theoretical knowledge and requirements from other organisation design processes which may very well be located outside the organisation at hand. E.g. a production function may require maintenance to be carried out only in tightly restricted maintenance windows. Such a restriction is essential for the design of a maintenance (planning and control) function.

The SFAD-process produces functional requirements in terms of definitions and functions purely based on human-independent requirements. Normally, most functions can only be
defined at a high level of abstraction. The objective of the SFAD-process is to define those functions in such a way that a maximum of cohesion within each defined function will be reached.

The personnel system analysis and design (=PSAD) process, in turn, uses these structural analysis and design (=SFAD) functions to form various socio-technical group structures. The IOR-approach may use a large variety of cohesion factors, although in the context of this study we only refer to the control-factor. In addition, the PSAD-process may also receive SFAD-functions from other SFAD-processes. Practically, different organisation functions can be designed in parallel, each of them providing SFAD-functions to the PSAD-process. In other words, in an organisation several SFAD-processes may exist at the same time, but there is only one organisation wide PSAD-process. For the sake of simplicity different SFAD processes are also not shown in figure 4.1. E.g. A task group can be formed which is responsible for the planning, control and execution of production activities (production function) and some operators maintenance (maintenance function).

The detailed functional analysis and design (=DFAD) process continues with the determination of the more detailed functional requirements at the point where the SFAD-process has stopped. This process is performed by the actual individuals who are responsible for those particular SFAD-functions. They determine the functional requirements at their appropriate level of detail. If functional requirements are used for software packages evaluation a more abstract level of detail may be used than the level of detail required for the information system analysis and design (=ISAD) process. If package's documentation is sufficiently clear and detailed, the evaluation of packages can be carried out by the actual individuals (i.e. the group members).

Finally, the ISAD-process produces the actual information system. If the package evaluation was carried out and effective packages have been identified the selection is performed on efficiency criteria, which partly are determined by ISAD-specialists. Software as a product is the eventual result of the contributions of several disciplines. E.g. the coding of the program requires programming skills in the field of computer science, the design of the layout of the screen display requires specific knowledge of ergonomics, etc. Therefore, the ISAD-process itself shouldn't be regarded as computer science territory only.

In case no or insufficiently effective packages have been found or if the package's strategy is not considered a valid option, in-house development by the ISAD-process is required.
On the determination of functional requirements in a maintenance environment

In the next chapter we shall discuss the of the SFAD- and the DFAD-process more in detail.
5 The principles of functional analysis and design

5.1 Introduction

In chapter 4 basic design criteria and constraints from the functional analysis and design, personnel system analysis and design and information system analysis and design processes have been presented. In this chapter we shall focus on a precise translation of the criteria and constraints into a functional analysis design approach, i.e. our prototype framework. To determine a suitable procedure for a functional analysis and design process, we first need to define what is meant by functional requirements more precisely. The component elements of this procedure can then be clarified and, finally, the entire functional analysis and design process itself can be specified. Each, prototype design step will be discussed in the following sections.

5.2 The definition of functional requirements

5.2.1 A "mechanistic" approach

In chapter 4 we have chosen the user as the central source for "expertise" for his area of responsibility, which makes him the ideal "functional designer". Communication problems with the computer science specialists and other information specialists are reduced by making sure that the users functional requirements\(^3\) are as unambiguous as possible with respect to the purpose for which functional requirements will be used (see chapter 3). From a developers point of view, Aerts, Alblas and van Hee (= v. Hee in short) [2] conclude that in many schematic conventions found in the computer science literature, some sort of input-transformation-output modelling is advocated. However, in most situations the formal specifications are semantically weak. They present a framework in which formal criteria are presented to construct semantically correct specifications. Such specifications are assumed complete with regard to information system development if a data model, a process model

\(^3\) The term functional requirements is used from user perspective or context. From the information system developers point of view it is more appropriate to use the term functional specifications.
and a functional specification are available. The data model represents the static part of an information system, i.e. the definition of the types of data that need to be stored for later (perhaps repetitive) use, and the constraints used to increase data consistency. In practice, the data model will be used to build a data base that acts as permanent memory. The functional specification and the process model, which describes the relationship of the functional elements, are the dynamic part of a specification. Several schema techniques can be used to model processes which can be understood easily by non computer experts, since in most situations natural language can be used instead of strict mathematical conventions. However, a strict mathematical notation is preferred because it avoids ambiguity. On the other hand, defining a data model can be much more difficult for users, because mathematical precision is required even though data model specifications can be specified in natural language elements. Certain mathematical skills are required which cannot be compensated for easily. It would go beyond the scope of this project to study educational and personnel selection options in order to solve this problem, but possibly additional elements can be taken up in the formal process specification which would ease the data modelling task of computer scientists without overloading the user.

v. Hee's [2] reasoning is based on an information systems developers point of view. He advocates a data oriented approach, since due to the perceived long term requirement of a data model great care is needed for its definition, whereas processes by nature can be changed easily. In short, data modelling comes before process modelling. By contrast, we are interested in justification, completeness and validity of functional requirements, which are important criteria from a users point of view. It is notable that the justification criterion would put a process oriented approach before data modelling. Data elements can only be justified if valid functions exist which need or produce these data elements. Thus, data elements can only be specified and justified if the processes are known. In the next sections we shall present the definition of functional specifications. We shall conform as much as possible to existing definitions known in computer science.

In the beginning of this thesis we defined functional requirements as a formal representation of procedures and practices in an organisational function. In the following paragraphs we shall discuss the elements of functional requirements and how they are related to each other.
5.2.2 The "function" definition

The key issue in formalising a process is the modelling of input-process-output relationships. Mathematically, an input-process-output description can be formalised by functions. A formal representation of a function is \( f(x) = y \), where

- \( x \) corresponds to the input type data,
- \( y \) corresponds to the output type data, and
- \( f(x) \) corresponds to the transformation mechanism that turns an occurrence \( x \) into an occurrence \( y \).

It may not always be convenient to resort to a strict mathematical notation. Natural language can be used as well, albeit that type \( x \), type \( y \) and \( f(x) \) need to be specified to be complete. A process or procedure can be modelled by connecting individual functions by means of defining output of one function as input for another function. In doing so, a representation of "chained" functions would look like a "network". We can use this property to visualise functions and how they relate to each other. In computer science many dataflow-like schema techniques have been defined. We shall use a well known schema technique known as ISAC (Information System Work and Analysis of Change)\(^4\) [57]. In figure 5.1 the graphic elements

\[\begin{align*}
\text{SYMBOLS USED} & \quad \text{SYMBOLS NOT USED} \\
\text{Data set (type)} & \quad \text{Set of physical items} \\
\text{Activity} & \quad \text{set of both physical and data set items} \\
\text{Decision} & \quad \text{Flow of physical items} \\
\text{data stream (occurrences) or signals} & \\
\end{align*}\]

\(\text{Activity properties:}\)
- transformation function
- trigger
- period of validity

\(\text{Decision properties:}\)
- selection criterion
- period of validity

\textbf{Figure 5.1. A formal representation of the modified ISAC-schema conventions.}

\(^4\) Other process schema conventions could be chosen equally well from a theoretical point of view.
On the determination of functional requirements in a maintenance environment

and their meaning are shown. To facilitate data model development and to solve the omission of conditional branching at lower level of detail, some enhancements are made. We will discuss next the enhancements to the standard ISAC schema conventions.

Unused symbols
Since physical items have no meaning in information systems all symbols relating to physical items have been omitted.

Data type symbols
This symbol defines the type of data element that is needed or produced by a function. The level of detail in which these elements are specified may vary.

Activity symbols
This symbol represents the transformation process that actually turns input data into output data. The level of detail in which functions are specified may vary. At its lowest level elementary functions are specified (see also v. Hee [2]), i.e. functions which have no internal permanent memory. In view of data modelling and the development of algorithms the following types of elementary functions are of particular interest:

a) data creation function;
b) selection function;
c) accumulation function;
d) sorting function.

ad a) The data creation function produces output-information by using certain input data. Typically, these types of transformation conform to strict mathematical formulas of the type y=f(x). E.g. the optimum (re)order quantity of parts could be described by the Wilson formula.

ad b) The selection function checks and selects data from an input-dataset according to a criterion. All other elements from the input dataset are discarded. E.g. in determining what maintenance jobs have to start today, a selection can be made of all jobs with highest priority to be released first.

However, the ISAC conventions are widespread and well known for many years.

80
ad c) The accumulation function can be interpreted as the reverse of the selection function. Data arriving to the function is accumulated and passed on if the accumulation process is triggered. Accumulation of data is essential because it serves as memory to other transformation processes.

ad d) The sorting function is the only function that passes the input data to the next function without creating or discarding data. The only operation performed is that an input data set in a stream will be reordered according to a sorting criterion into a sorted dataset. E.g. all outstanding maintenance jobs have to be sorted according to their priority before any job release selection function can be activated.

An important aspect in defining a function is triggering. Functions are activated whenever a particular event occurs. E.g. The priority of a new maintenance work order is determined as soon as this work order is identified and known to the planner. Essentially, any activity must be triggered to start. A function can be regarded as a (formal) metaphor of a "real-life" activity and therefore a triggering event should be an equally important property of our function definition.

We shall distinguish three types of triggers:
1) activation whenever output-information is requested;
2) activation whenever input-data is offered;
3) activation due to other external stimuli.

Triggers of type 1) and 2) speak for themselves. In situation 2 some more specialised functions may even produce only a trigger-signal as output information. In other words, in that situation the only information content would be: "start function now".

Activation by other stimuli occurs whenever a function can be activated irrespective of triggering signals sent by other functions, such as timely routines which have been set for reasons of convenience or practicality, etc. E.g. in a company every morning a co-ordination meeting takes place between maintenance and production planners to agree about the maintenance to be carried out during that day. Lists have to be produced showing planned and uncompleted maintenance jobs and their urgency. The function "produce job list" is activated every morning.
In chapter 4 we discussed the importance of the period of validity of functions to computer specialists. If functions are changing rapidly, we may indicate a function as being temporary or experimental, in case one isn't sure about the function's effectiveness. Computer specialists may recommend end-user computing tools such as database query and report generators, spreadsheets, etc. for such functions. However, if available end-user tools may not provide the level of efficiency to satisfy certain response times determined by some triggers, in which core computer specialists are needed. This is especially true for time-critical applications such as airline reservation systems. Efficient implementation usually means high investments in hardware and computer specialist capacity. In addition, high expenditures for an efficient realisation of functions may also be justified where a long period of validity has been specified. In other words, the longer the period of validity (the "life expectancy") of a formal function the easier it will be to justify the expenditure involved with its realisation. Some functions may relate directly to an organisation's information infrastructure and are shared with different organisational functions. Computer specialists will need to have an overview of all infrastructure functions to assess the most efficient way for realisation. It is a core activity within the information systems analysis and design process to monitor these different functional requirements to decide what investments could be justified in terms of infrastructure and more provisional solutions.

Since it may be hard for the users, being the functional designers, to estimate accurately a period of validity, computer specialists could assist in e.g. providing distinct time-frame options, associated with different information system development approaches and consequently with different realisation cost levels. At this point it is important to stress that some co-ordination between users and developers can be beneficial, especially in case users come up with different functional alternatives, each having different periods of validity.

The decision activity

The decision type of function represents a special type of function, i.e. the process part is a Boolean expression that checks certain properties of the input data stream. Depending on the values of these properties, the data will be branched into a different, and by nature a more specialised network of functions. Because input data must be present at the decision function, triggering can only take place whenever input data arrives at the decision function. Subsequent decision functions provide a possibility to model multi-channel branching.
The data stream

The data stream is a representation of the path by means of which functions are linked by data, being "sent" from one function as an output to another function as input data. Data streams usually carry data, but they can also act as triggers, or both at the same time.

5.3 Elements of a functional analysis and design process

Since we have defined functional requirements, we can discuss how to incorporate the criteria justification, validity and completeness into our framework. All three criteria can be met by applying the principle of decomposition. Decomposition originates from systems theory (e.g. Simon [71], 1952). We shall demonstrate how the general decomposition principle can be enhanced so that these criteria are incorporated.

5.3.1 Justification of functions

Nielen [67] makes a distinction in data versus information. Only if data has a meaning and contributes in decision making it is information. In concordance to Nielen's perception we can typify the output-data as information because the data do have a meaning or utility as to what should be accomplished needing this data. In other words, another function needs this data for its process which is perceived as useful or valuable. Essentially, the effect of providing output-information to other functions, which in turn uses this information as input-data, implies functions may be connected via a data stream.

In general, a function is justified if a related function exists that needs input data and/or signals and no other function exists which is capable of providing these data and/or signals. Consequently, the process of justification follows the opposite direction of the data stream, until the data stream passes the boundaries of an organisational function. In turn, the organisational function that provides the data stream continues the justification process, and so on.

Functions need not necessarily originate all within one single organisational function. E.g. production planning functions may need information on what maintenance activities need to be carried out, in let's say a week's time, in order to make a valid production plan. A maintenance planning function is providing a, preliminary, schedule for the production
planning function. Also, not every demand for data types and/or signals needs to be satisfied within a single organisational function. E.g. the determination of the value of spare parts stocks requires information on the price of each spare part. This information may be provided by the supplier of a spare part, or by the financial function within the organisation. In general, it is a matter of definition and co-ordination to decide what particular organisational function should be responsible for the provision of certain types of data and/or signals.

In some situations functions are justified because a physical process needs control signals to guarantee a proper execution of the physical activities. E.g., the primary justification of a maintenance function is that physical maintenance activities will be carried out in an orderly way.

5.3.2 The validity of a function

Usually, output information must conform to certain minimum standards in terms of completeness, accuracy, timing, correctness, etc. We shall refer to these aspects as the effectiveness of a function. The effectiveness of a function can be measured in terms of the degree to which properties of the output, when the function is used, match the ideal output requirements. In the functional design stage the level of effectiveness reflects the state of the art in both situational and theoretical knowledge that is available and the knowledge that is required. Normally, a certain minimum effectiveness is required for a function to be considered as an effective alternative. Clearly, also the efficiency of a function plays a role, i.e. the amount of efforts or costs in realising and using a function. The efficiency of realisation of a function may even invalidate an otherwise perfectly effective function. E.g. many combinatorial problems require a simple function which would check every possible combination to find the most favourable combination. This function is fully effective, but not feasible for realisation in case the number of combinations to be checked exceeds available resources. Other, potentially less effective functions may be required to balance the available resources. This example illustrates that the efficiency of a function is not determined during functional analysis and design but is a result from personnel and information system analysis and design. To assess the validity of a function the perceived value of its effectiveness is related to its efficiency. In case more alternative functions are considered, the best
effectiveness/efficiency ratio will determine the choice. To arrive at a best choice, especially in complex organisational functions such as maintenance, functional design, personnel system and information system design (in reality or in a simulated way) and back to functional design have to be cycled through to make a complete assessment of all potentially feasible alternatives.

Although measuring and assessing the validity of functions as described above may seem rather straightforward, the real difficulty becomes evident when used in practice. In many cases the perceived value of effectiveness cannot be measured in quantitative terms, whereas the efficiency could be measured quantitatively in many cases, but sometimes only in terms of rough indicators. It could be that only a subjective assessment is possible, or necessary, to get an idea of the validity of functions.

Due to the dynamics of the environment and of the changes in validity of other functions, a function's validity is constrained in time, i.e. the period of validity is limited. To what extent this period can be estimated will depend on the actual situation. Sometimes, a change in one function may lead to invalidation of other functions due to the justification criterion. This may happen unexpectedly, and therefore the validity of a function must be monitored.

The validity of a function is in the first place strongly dependent on the personal background of the functional designer on the one hand, i.e. organisation function specific knowledge and experience and ingenuity, and on the inherent complexity of the function in its actual context on the other hand.

5.3.3 The completeness of functions

The completeness criterion is closely related to the validity criterion. If input data streams are specified for which no valid function provides the required information as output, the design is incomplete. Therefore, incompleteness implies invalidity of the overall functional requirements. When invalid functions occur other functions "upward" of the data stream can be invalidated as well. In practice we must demand 100% completeness, otherwise not all information and signalling that is required can be provided. In terms of our schema technique completeness implies no "loose" arrows are allowed.
5.3.4 The decomposition principle

In chapter 4 we discussed the relationship between ambiguity and formalisation of functional requirements. Usually, if ambiguity is high in specifying functional requirements then the level of formalisation will be low. Since, an evaluation of information systems will be less uncertain and more accurate if criteria have been determined in a more formal way. Hence, ambiguity in functional specifications must be minimised.

Minimising ambiguity also requires a high degree of detail in which specifications are to be determined. In practice it may be very difficult to determine complete sets of detailed valid functions for complex organisational functions in one step. This problem can methodologically be solved by the decomposition of high level ambiguous functions into more detailed less ambiguous and more formal functions.

The principle of decomposition is well known in systems theory and it has been advocated in different applied sciences, as a tool to reduce complexity by separating the problem area into smaller problem areas. An important potential of decomposition is that those areas within the
main problem area can be identified, which have -predominantly- more internal than external relationships.

We shall enhance the general decomposition principle as the main tool in our functional analysis and design process method. This means that a decomposition criterion must be determined and that provisions will be made to satisfy completeness, justification and validity criteria. In figure 5.2 the process of decomposition is depicted. For each function, which has been assessed already, it must be decided if further decomposition is required.

5.3.5 The decomposition criterion

The purpose of a decomposition criterion is to provide a formal standard or norm to assess whether an acceptable level of detail in functional specifications has been reached, and to provide guidelines and constraints for a next decomposition step if it is necessary to continue to decomposition. In this study the organisational functions are the artefacts which will be decomposed. Determining a decomposition criterion for a function requires specific knowledge on the part of the functional designer with respect to that function. The specificity of the knowledge increases at lower levels of decomposition, whereas the scope of a lower level function decreases. Due to that specificity in required knowledge, decomposition criteria depend on the type of organisational function and on the decomposition levels which have already been determined.

Practically, decomposition criteria may have to be revised in time because of changes in other organisational functions related to the organisational function considered, or simply because functional designers gain a better understanding of the effects of their design. Given this variable nature of decomposition criteria it is inappropriate to define standard decomposition criteria beforehand. Decomposition criteria for a possible next step of decomposition should be determined as soon as a valid function has been specified. The principle of decomposition will be demonstrated in the next chapter, where we shall analyse the maintenance function.

In addition to guiding and constraining functional designers the decomposition criterion should also explicitly state the minimum and, if not evident also the maximum level of performance that decomposed functions should provide as a part of the validity.

Simons decomposition rule [71].
On the determination of functional requirements in a maintenance environment

(Effectiveness) aspect. E.g. a planning function of maintenance work orders may allow lateness for a maximum of 5% of all planned work orders as an effectiveness criterion.

5.3.6 The standard function

In theory, the decomposition process stops when all functions have been formalised sufficiently according to the functional designers personal judgement, or if all determined functions are elementary. Practically, there is also a third possibility to stop the decomposition process. This possibility is strongly related the organisation function specialist background. Decomposition may also stop as soon as a standard function has been determined.

Standard functions are functions which have an unambiguous meaning within the scope of the functional designers responsible for the design of a standard function, and to the personnel that is responsible for the realisation of that function. In other words, irrespective of the individual responsible for the design of a standard function, acting as a functional designer, a decomposition would result always into an exactly identical set of more detailed functions. E.g. the hazard plotting technique developed by Nelson [65] for estimating failure rates and mean times to failure is a technique which is well known to maintenance function specialists. In addition, functional designers familiar with a certain maintenance situation would know that only certain failures are that important to justify failure data collection and a formal hazard plot. The Nelson hazard plotting technique can be regarded as a standard function, not only because it is a well established way of analysing failures, but also because the constrained validity in this maintenance situation is known to every functional designer. Determining standard functions can be important in view of evaluating standard software packages (see chapter 7), because decomposition until elementary functions have been determined may not be necessary. On the other hand, if a functional analysis and design process is carried out particularly for the purpose of in-house development, then the background of developer specialists conditions when decomposition can stop.
5.3.7 The evaluation function

Evaluation as a means to monitor validity

In organisation theory the "control loop", originating from systems theory, has gained wide acceptance. Anthony [5] suggested that numerous control loops may exist in actual organisations. He also suggested a relationship between the span of control and the time interval that control loops need to take effect. The classical distinction in operational, tactical and strategic control refers to the difference in that interval. In this study, the decomposition of an organisational function is aimed at the operational level only. We regard the operational level as the level that controls the physical execution of activities. An evaluation function uses feedback signals, which consist of data elements with respect to relevant properties of the output stream produced by the function to be monitored. The evaluation function assesses if and how the monitored function can be adjusted. We distinguish two types of feedback. Some control loops have a pure operational nature and are determined during decomposition. E.g. a capacity loading function must feedback to the scheduling function in order to check if the schedule would not exceed capacity availability. In this example, the decomposition process, and thus the functional analysis and design process shows that feedback is produced to another function on the same operational level (horizontal feedback).

In our view, evaluation is considered to be a secondary control loop providing feedback on the validity of the functions that have been determined during decomposition (vertical feedback). Thus, special functions are needed that monitor the performance of the functions on the operational level.

The determination of evaluation functions

Functions on an operational level may get invalidated for several reasons. First of all unsatisfactory performance can trigger a functional redesign. The decomposition of functions requires the determination of a decomposition criterion for each function that is to be decomposed. The decomposition criterion itself is an important factor with respect to the validity of functions that result from a decomposition step, since it contains the design guidelines and constraints for each decomposition step.

Secondly, the temporary validity of functions is set by the period of validity property which is estimated by the functional designer. Thirdly, the output of a function may have become irrelevant purely because the function served by the output has become invalid. Invalidation
On the determination of functional requirements in a maintenance environment

Figure 5.3. The evaluation function prototype.

of the function that produces the output as well due to the justification criterion. The same is true if the "parent" function on a higher level of decomposition is invalidated due to the justification criterion.

Depending on the nature of a decomposition criterion, actual output of the decomposed function may be collected to be used as a part of the decomposition criterion. In figure 5.3. a prototype of an evaluation function is shown. Redesign is triggered as soon as one criterion invalidates the decomposed function. This figure illustrates how a function "F1" is evaluated. Output "Data O" is collected and a decision function decides if a redesign is necessary, i.e. redesign function "R1" is activated or simply a "do nothing" recommendation is produced. If "R1" is activated the decomposition criterion can be revised and/or a redesigned prototype function "Redesign Data" is determined.

5.4 The functional analysis and design process

5.4.1 The distinction between structural and detailed functional analysis and design

Normally, at the organisation function level it is clear that further decomposition is required. At the first levels of decomposition rather general decomposition criteria are used and only
few of the various possible situational conditions are relevant. The determined functions will be global, which manifests itself in the primarily indicative formulation of function properties. E.g. at high levels of decomposition it is not possible yet to indicate what exact input-data is required for each function.

But at the following decomposition level personnel structure requirements become relevant as a decomposition criterion or as a situational condition. At that level further decomposition must stop once the personnel system analysis and design process has determined what groups are distinguished and what functions, determined so far, have to be allocated as a part the responsibility of those groups. Further decomposition would require the initiative, situational knowledge and experience of group members.

In figure 5.4 the distinction between structural and detailed functional analysis and design (named SFAD and DFAD respectively), i.e. functional decomposition when groups are distinguished is shown schematically.

The characteristics of a structural analysis and design process

The structural analysis and design process starts with the lowest level of detail, i.e. the organisational function. At this level the input and output data streams will be specified in global terms. At this point, and potentially at some lower levels of decomposition, functions
can be justified only globally, simply because required input data streams are also defined in
global terms. The primary justification of functions at this high level will be primarily based
on the existence of a set of (physical) activities to be controlled. Guaranteeing validity and
completeness of functions is equally difficult, and will primarily depend on definitions used
as decomposition criteria, rather than situational characteristics. Identification of valid main
functional areas is the objective in this stage. Sound theoretical knowledge and analytical
skills of the designer with respect to the organisational function at hand is a key factor of
success here.

The characteristics of a detailed functional analysis and design process
In the detailed functional analysis and design process decomposition criteria will be
predominantly based on situational characteristics and on individual and group preferences.
At this stage the level of detail should be sufficient to allow for the determination of the
functions that can provide output data streams for functions from related organisational
functions. In turn, requests for input data streams to be provided by other (external)
organisational functions should be unambiguous enough to trigger a detailed functional
analysis and design process for those types of organisational functions. Practical experience
in an organisational function in combination with a good sense of carefulness by designers is
expected to be a prerequisite.

5.5 The structural functional analysis and design in relation to the personnel system analysis
and design
Each function that has been specified at the lowest level of decomposition in the structural
functional analysis and design process (=SFAD-functions) must be allocated to personnel
groups who will be responsible for that function. We shall denote the set of SFAD-functions
a group is responsible for as a functional unit.
In preparation of the socio-technical fundamentals of personnel system and design, in which
group autonomy should be maximised, (re)design and execution of functions should both be
part of the groups responsibility. This means that groups will determine and carry out the
more abstract functions as well as the physical activities. In addition, all evaluation functions
that resulted from their decomposition efforts are part of their shared responsibility as well.
Since we developed the theoretical framework for the structural analysis and design process, it is possible to describe how the results of this design process should be interpreted as a basis for the integrated organisation (re)design (=IOR) approach. The following aspects are of particular interest:

a) The number of activities, including control activities, to be carried out;
b) the span of control of socio-technical groups;
c) the complexity of the functions.

The number of activities
Socio-technical design and the IOR approach starts with a complete listing of activities which have to be carried out by the personnel of an organisation. The structural analysis and design (=SFAD) process is meant to provide a complete list of valid and justified functions that must be dealt with. The list can also be used to assess what skills are required by group members to actually carry out the functions they will be responsible for. In this context, skill should not only be associated with physical skills, but also with (re)design skills. In addition, to obtain some idea of the required group size if functions are added to their responsibility, the workload of a group should be estimated. It depends on the actual organisational functions at hand as to how this can be achieved (see chapter 7).

The span of control of groups
The capability of groups to design, execute and redesign functions altogether is in modern sociotechnology considered to be a major contribution to semi-autonomous groups leading to a high level of work satisfaction and of a high level of motivation. In sociotechnology, so called cohesion factors constitute the determining factors for the formation of groups (see chapter 4). The SFAD-function is a prime candidate for cohesion, because design, execution and redesign are inherent properties of a function and the decomposition process as we defined it. Consequently, cohesion is maximised if sets of entire SFAD-functions are allocated to groups.

Speaking in terms of classical organisation theory, if the number of SFAD-functions which are allocated to a group is increased, the span of control of a group is increased as well.
The complexity of functions

Some SFAD-functions can be complex, i.e. they tend to require many levels of decomposition allowing short validity periods and therefore require a substantial (re)design effort. If some specialisation cannot be realised within a group, it is necessary to transfer those functions from the group to specialist staff groups.

5.6 The detailed functional analysis and design in relation to the information systems analysis and design

It has been outlined in chapter 4 that a maximum level of detail is required, i.e. functional specifications down to an elementary level, to be of use for the development of information systems for organisational functions. Information system analysis and design (=ISAD) processes are needed in case standard software packages cannot provide the functionality required, or in situations where packages are not considered as a valid option at all. In chapter 4 we also discussed how the information infrastructure relates to the period of validity, which we introduced as a property of a function. Therefore, we shall concentrate on the development of (partial) information systems for organisational functions - the application shell in Wortmann's terms [10]. In the beginning of this chapter we defined the components of a functional specification.

The coding of functional specifications

Elementary functions are a prerequisite for coding. If they have been determined during detailed functional analysis and design the interpretation by ISAD specialists is rather straightforward. The transformation part of a function can be translated into a programming code directly. Additional function properties are helpful in a different way. The trigger property can be used in several ways. Requirements with respect to the required performance can be derived. E.g. if the rate of arrival of data stream occurrences is known for a certain function, and if this function is triggered by the arrival of each data occurrence, the performance of the information system with respect to that function must be sufficient to guarantee the possibility to accept a new data occurrence well within the time frame of a new occurrence. The period of validity can also be used to determine whether a very efficient, and therefore laborious and time consuming, coding process is worthwhile.
Determination of a data model

The determination of a data model is a much less straightforward process. A data model consists of a data entity type definition, attribute type definition and a definition of constraints, including subset requirements.

The entity and the attribute type definitions are closely linked to the decomposition steps that have been performed in the detailed functional analysis and design stage. At higher levels of decomposition data streams will be defined at best in terms that can be suitable for entity type definition. At lower levels the data streams may very well be specified in terms that correspond with the attribute type definitions.

Constraints and subset requirements cannot be derived directly from data stream types. Constraints can be determined by identifying elementary functions of the type selection and sorting. Subset requirements can be identified by tracing data streams belonging to different entity types at the functions where they join as required input-data.

In most situations the definitions resulting from an approach such as the one described here can only be used as a first try-out. Verification with actual users will still be necessary because they still have to agree with the developer specialists interpretation.
On the determination of functional requirements in a maintenance environment
6 Structural decomposition criteria for maintenance

6.1 Introduction

In the previous chapter a general approach has been discussed of the functional analysis and design process. In this chapter we shall define the maintenance function by using a maintenance model developed at the Eindhoven University of Technology; abbreviated as the EUT maintenance model. Because the functional analysis and design process leads to potentially many additional decomposition levels and even more decomposition criteria, the sheer number of options in an actual situation makes it virtually impossible to develop all potential decompositions without detailed knowledge of a specific application situation of maintenance.

Furthermore we shall concentrate on the maintenance planning and control subfunction. In accordance with the general framework developed in chapter 5, decomposition criteria will be developed for the structural functional analysis and design process, but only with regard to the maintenance planning and control function.

6.2 A maintenance literature review

*The EUT maintenance model*

An analysis of the maintenance function in an actual situation requires a clear definition of the main subfunctions to be found in any actual maintenance situation, irrespective of the industrial sector, the scale and other situation-dependent aspects. The EUT maintenance model as such is a universal model [37]. The EUT maintenance model distinguishes 14 different functional areas. In appendix 3 a visual representation of EUT-model and its functional areas is presented. To cover all 14 functional areas in depth would greatly exceed the scope of this PhD study. Therefore we shall concentrate on functional analysis and design process with respect to the planning and control function, one of the core functions within the EUT maintenance model.

Because the planning and control function cannot be dealt with in isolation, we have to take into account the other functions within the maintenance management block. The functions to be considered are:
On the determination of functional requirements in a maintenance environment

1) the maintenance planning and control of rotables (repairables);
2) the spare parts management (consumables);
3) maintenance evaluation.

ad 1) The management of rotables can be regarded as a special form of planning and control, as we will illustrate in the following paragraphs.

ad 2) The management of consumable items requires studying in its own right. As far as planning and control issues are concerned we assume that only a maintenance activity can be released if the correct items are available. Furthermore, we shall assume that the execution of a maintenance activity can be restricted by consumables which have to be ordered for this activity only. In that situation a maintenance activity must be planned for release at the current date (order entry date) taking into account the suppliers leadtime for any consumables (in case more consumables are ordered, the longest leadtime).

ad 3) The evaluation function is a compound function. Evaluation means that a lot of properties are monitored and used to assess a diversity of aspects of the maintenance performance. We shall concentrate on those aspects which have direct relevance to the planning and control function.

The maintenance management literature in general

Review could tend to suggest that the majority of operations research oriented maintenance literature concentrates on reliability centred issues (e.g. Anderson and Neri [4]). However this doesn't concern maintenance management but elements of the maintenance concept, referred to as maintenance policies or strategies, aimed at finding optimal maintenance intervals for a limited number of failures (usually one or two) in a static environment (e.g. Kelly [52], Küpper [54]). Christer [21] presents an extensive overview of operational research in maintenance. The preoccupation with statistical methods neglected the strong potentials of the control mechanisms in maintenance management on an operational level and in condition based maintenance. Gits [41] developed a framework in which maintenance concepts are designed in a systematic way. The literature on maintenance management is scarce and not based on a fundamental analysis leading to a consistent set of maintenance planning and
control methods. One of the early publications are Morrow [62] and Swärd [75] who mentioned some elements but they concentrate on primarily technological aspects (more recent, see also [69]). Blanchard [12] who published the Life-Cycle-Costing approach actually treats primarily investment criteria for new technical systems taking maintenance into account, but for granted. The Japanese TPM (=Total Productive Maintenance) approach ([63] and [64]), is actually a combination of manufacturing, setting up and maintenance, but in particular is restricted to the possibilities for continuous improvement (Kaizen) by mobilisation of the work force on shop floor level.

Maintenance planning and control
In maintenance planning and control literature fundamental analysis are absent. Most authors (e.g. [56], [47], [66], [22], [24], [49], [43]) concentrate on describing administrative procedures, design of forms, maintenance office layout and other practicalities. Others (e.g. [45], [61]) focus on budgeting issues in maintenance. However, all those approaches are unsuitable for planning and control on an operational level. In general, the authors lack any considerations with respect to a formal justification of budgets, because they are mostly based on historic financial reports only. Application of budgeting seems to be limited to provide some means for the evaluation of maintenance performance. Only seldom planning and control situations are addressed directly. Network planning techniques are the only basic issues discussed by some authors. However, they limit themselves to the explanation of the general theoretical principles. Mann [58] is one of the few authors explaining maintenance planning of shopfloor maintenance activities based on a simple priority system. Sometimes authors intuitively categorise maintenance activities into distinct types, such as breakdown (emergency) maintenance, preventive maintenance, routine maintenance, each requiring its own maintenance capacity. Blegen [14] made an effort to develop a typology on the basis of theoretical considerations on several factors, assumed to be relevant. However, it did not lead to an eventual framework for the maintenance function. Geraerds [40] stressed, already in the early seventies, that maintenance theory needs to be developed in the area of maintenance concepts, maintenance planning and control and maintenance inventory control. He also clearly depicted the
relationship between production and maintenance as interrelated functions, in which production should be interpreted as "use" or "operation". Apart from the relationship between production and maintenance as organisational functions, there are also similarities in planning and control of activities. Some planning and control methods can be easily translated from their original use in a production environment to a maintenance environment. However, many structures in maintenance require a different approach.

![Diagram of elementary production planning and control types](image)

**Figure 6.1. The elementary production planning and control types (see also [9]).**

### 6.3 Elementary maintenance situations

In Production planning and control basically five different elementary types can be distinguished (see figure 6.1).

Elementary production planning and control types are essentially different with respect to the mix of applicable planning and control methods. E.g. production in batches (close to the series production in figure 6.1) addresses key elements such as the optimal batch size and batch sequencing, whereas project management ("network" production in figure 6.1) concentrates on control of critical path activities.
In a production organisation, usually, a combination of elementary production and control situations will be found. At a higher level of control than the elementary level this introduces the need to pay attention to the interrelationships to be achieved by co-ordination of the non similar but connected planning and control situations of which each has its own type of elementary method.

When comparing elementary maintenance planning and control types with the elementary production planning and control types, no analogy in maintenance with the process or mass manufacturing situations can be discovered. The other three situations, in principle, show analogy with respect to repetition of demand and the number of activities in process simultaneously. We shall discuss the jobshop ("lot for lot" production), project management ("network" production) and "series" planning and control with regard to their relevance in maintenance in the next sections.

**Job shop**

Maintenance activities with a relatively low repetitiveness are scheduled either deterministically, e.g. by barcharting methods, or on a priority basis. In the production planning and control literature (e.g. [10]) great emphasis is put on the development of formal priority rules based on statistical considerations for mono's. A mono is a work order of which the operations are carried out in a sequence. However, in most maintenance situations the number of orders simultaneously in process at shop floor level simultaneously, is compared to normal production situations, still very low. Consequently, statistical considerations are not relevant in that case. A central general maintenance workshop will have more work orders on hand to be controlled simultaneously. Also in that case, great emphasis must be put on sound human judgement in short term planning, frequently on an opportunistic basis, making a maximum use of human recovery capabilities in dealing with control issues.

**Project management**

Individual maintenance clusters containing activities which have mutual dependencies with respect to their execution (activities have a network structure) fall in this category. Maintenance clusters containing mutually dependent EMR's (= Elementary Maintenance Rules) are usually managed this way. Priorities in planning and control of individual activities in a network are based on the consequences of the individual (time)schedule of activities on
the total throughput time of the entire network, i.e. whether activities are on the critical path or not. In contrast with the production environment, maintenance networks are seldom complete in terms of the number of activities known at the start of the project. Usually, diagnostic activities are a part of a network, featuring as activities to be carried out. However, the eventual result of a diagnostic activity is not known until a diagnostic activity has been completed (see Wubben [78]). The typical network techniques are unable to deal with this uncertainty. In practice, high flexibility in maintenance resources (in volume and mix) is needed. Short term uncertainties require a great deal of human experience and ingenuity to solve these short term high urgency problems. In case multiple projects are to be managed simultaneously, a special type of planning and control is required known as multi-project planning and control. In this highly complex situation activities are primarily scheduled on jobshop-like priority basis and the network structure is roughly translated into a routing. In (specialised) workshops multi-project planning and control is quite common.

"Series" planning and control
In a limited number of cases series planning and control is possible. Sometimes in specialised central workshops the assortment of technical systems to be maintained is limited, but the demand of each technical systems is more or less continous. The major planning issue here is to determine a suitable sequence in which the identical series of orders will be carried out to minimise set-up losses and to reduce throughput time.

6.4 Decoupling as a means to decompose maintenance functions

6.4.1 Basic considerations with respect to decoupling
In the previous paragraph types of elementary maintenance planning and control have been introduced analogous to production planning and control types. Depending on the work order characteristics different types are valid. However, in maintenance planning and control there is more than distinguishing one or more elementary types. In production planning and control elementary situations can usually be attributed to individual production departments. The co-ordination between different production departments is referred to as goods flow control.
The production planning and control approach named MRP (Materials Requirements planning) is popular in assembly of discrete products and addresses the timely manufacturing of the parts going into assembly, seen as an essentially deterministic process. Maintenance lacks the uniformity to make MRP feasible. Unfortunately, in maintenance there can be only very little or no analogy found to goods flow control. However, compound maintenance situations requiring more or less co-ordination do exist in maintenance as well.

The primary objective of maintenance - and all its required subfunctions - is to reduce the consequences of failures. As a consequence of using technical systems deterioration will occur and eventually maintenance is needed. In figure 6.2 this basic relationship between the production function and the maintenance function has been visualised (according to Geraerds [40]).

A major factor in which consequences of failures with respect to the user of the technical systems concerned is the degree to which the production and the maintenance function can be decoupled. The primary purpose of decoupling is that maintenance activities are carried out without hindering the production function.

It is clear that more decoupling not only means more potential production capacity, but also a reduction in planning and control complexity. Ideally, instead of establishing a joint production and maintenance planning and control function, both functions can be activated.
more or less independently with only a small need for co-ordination. In that case a maximum of decoupling has been achieved, because all maintenance activities can be carried out without any disturbance or hindrance to the production function (= running maintenance).

The basic idea to achieve decoupling is to isolate planning and control functions, which can be decomposed separately without the inherent danger of suboptimisation. Basically, decoupling is achieved if some sort of redundancy of technical systems capacity or use is available. Redundancy can be achieved partly by technical design determined by the engineering function and partly by the technical system operation characteristics set by the production function. Decoupling can also be applied within the maintenance function in case certain conditions can be met. We shall discuss both decoupling possibilities in the next sections.

**Technical system design and decoupling**

Basically, technical system design can contribute to decoupling in two ways: via build-in flexible maintenance friendly operation modes and by making technical systems moveable and therefore replaceable. In a maintenance friendly design maintenance can be carried out on technical systems in standby mode or in running condition or simply because little maintenance is required, e.g. by quick repair by replacement. In that context maintenance friendliness means providing a high degree of maintainability. It is clear, that the degree of maintainability is an engineering and an investment issue. Within the context of this study we assume that this aspect can change and therefore can influence the way maintenance is planned and controlled on the long term. However, structural consequences in the way maintenance activities are planned and controlled are difficult to assess beforehand, because they depend on each actual change in the technical properties.

The moveability of technical systems or subsystems can be used effectively to enlarge the decoupling between maintenance and production. Technical systems designed to be highly moveable can be replaced quickly by a serviceable maintained identical type of technical system or subsystem. Actual downtime can be limited to the time needed for carrying out the replacement activity. Maintenance on the replaced part or technical systems can be carried out, potentially at a different location, independently from operation requirements set by the production function. However, utilising this possibility of quick replacement requires an additional investment in redundant technical systems or subsystems. In literature this
decoupling principle is known as the rotable system. We shall discuss the structural consequences of this option more deeply in paragraph 6.5.

Another possibility to realise decoupling in technical system design is to "built in" redundant subsystems. Then, unexpected maintenance becomes postponable and can be carried out without increase in overall system downtime, since a failed subsystems function can be taken over quickly by the other built in subsystems (= passive redundancy).

Production process design

Similar to building in redundancy at a technical system level, the same design principle can be applied for the entire production processes to various degrees. The level of redundancy is solely determined by the trade-off between the required investments and the criticality of the production function. E.g. in manufacturing production lines anything between installing only some critical redundant technical systems up to the installation of completely identical parallel production lines can be considered.

Technical system operation characteristics.

Applying redundancy here means to use functional redundancy of the technical systems. Technical systems must have an overcapacity in relation to the production demand, i.e. these technical systems need no 100% uptime to satisfy 100% product demand (no lost sales). The time a technical system is not needed for production, i.e. the maintenance window, can be used for scheduled maintenance. Clearly, this situation is more complicated than the "built-in" redundancy, since it requires co-ordination between maintenance and production to decide the allocation of a technical system over a specified period. In military situations this co-ordinated planning is known as "planned flying - planned servicing".

6.4.2 Basic considerations with respect to the maintenance concept

Gits [41] described in his dissertation a systematic approach to determine for a technical systems what maintenance should be carried out and when it should be initiated, taking into account production requirements, technical system characteristics and failure behaviour of technical systems. The result of this approach, named "the design of maintenance concepts", are clusters of elementary maintenance rules. Maintenance clusters are the smallest entities to
be planned and controlled individually on an operational level. For the planning and control of maintenance activities it is imperative that all maintenance activities are defined in the maintenance concept. The following characteristics, sometimes reflecting certain types of clusters, are important with respect to the maintenance planning and control structure:

1) a cluster contains one elementary maintenance rule (=EMR);
2) each EMR in a cluster is different;
3) the EMR's in a cluster are all identical, i.e. the same EMR's are carried out for different technical systems of the same type;
4) the EMR's in a cluster are all different, but they are mutually dependent on each other in sequence and timing of their execution.

ad 1) In case cluster contains only one EMR, the cluster is considered a mono. A single (set of) activities aimed to reduce the consequences of one individual failure. This type of cluster has least complexity in planning and control.

ad 2) More but also mutually independent EMR's in a cluster can be regarded as a set of mono's. Although these clusters are scheduled and released as a group, the sequence - and within certain limits their timing- are flexible, thus creating flexibility for the maintenance task group assigned to this type of cluster.

ad 3) Identical EMR's in cluster represent a special situation, known in literature as group replacement or group repair. Task groups frequently assigned to these types of clusters require usually little multi-skill capabilities. The repetitiveness of EMR's within a cluster facilitates building routine skills within a task group, and it is hoped this routine will pay off in shorter throughput times or alternatively requiring less capacity.

ad 4) If EMR's in a cluster are mutually dependent with regard to their execution, the EMR's have a network structure.

Clusters defined in the maintenance concept having regular intervals are not the only clusters that exist in planning and control. In many situations a lot of clusters exist with one EMR initiated on the occurrence of a failure (Failure based maintenance). Then, taking into account other similar timed clusters in addition to the regular clusters, it can be efficient to plan and
release groups of clusters as a whole opportunistically. We can assume that these opportunistic clusters can be formed relatively often and, due to their nature, can be treated as sets of mono's. In practice at the planning stage the content of identical clusters may vary due to adding postponed maintenance activities and modifications.

6.4.3 Basic considerations with respect to maintenance capacity

Another major factor in structuring a maintenance planning and control function is the structure of the maintenance capacity. We shall focus on the two most relevant factors:

1) the degree of skill specialisation;
2) operating restrictions;

ad 1) In most cases carrying out maintenance involves the use of some specialised capacity, i.e. personnel and tools. In general, multi-skilled capacity is more flexible with regard to planning and control and consequently we can assume that multi-skilled capacity can be utilised more effectively. On the other hand, multi-skilled capacity may not have all the necessary skills to perform more specialised activities. The costs of having certain skills available on a long term basis, assuring short response times, have to be traded-off against hiring in skills when needed at a higher hourly rate and usually with longer response times. In situations where capacity constraints have to be taken into account, the workload of these critical capacities must be monitored by the planning and control function. Consequently, planning and control requires a more formal approach in that situation.

ad 2) Operating conditions of types of capacity refer to technical properties of the capacity groups in combination with physical environmental conditions within which a capacity group has to operate. E.g. sometimes, special facilities are required to carry out certain activities. Therefore, technical systems must be moveable and brought to the workshop (or "bay" or "dock") where these facilities are available. Sometimes, environmental conditions are harsh making it impossible for maintenance personnel to repair failures in situ.

Mostly, these operation conditions can be attributed as set-ups in maintenance
On the determination of functional requirements in a maintenance environment

concepts. However, planning and control must take them into account as an extra (set of) activities to be planned and controlled.

6.4.4 Considerations with respect to the evaluation function

The evaluation subfunction defined in the EUT maintenance model covers the dynamic aspect of a functional analysis and design. In chapter 5 we outlined the general concept of decomposition by using decomposition criteria. Each function which is decomposed requires a decomposition criterion. Essentially, evaluation comes down to monitoring the validity of the decomposition criteria which have been used for the current functional analysis and design. In turn, an invalid decomposition criterion invalidates all functions on lower decomposition levels. The EUT maintenance model defines the evaluation subfunction as a separate function. In our view the evaluation function is in fact a collection of functions aimed at monitoring the validity of each individual decomposition criterion. Therefore, the decomposition of a maintenance (sub)function automatically leads to the relevant evaluation functions.

Since we are interested only in the core maintenance planning and control function, evaluation will also be restricted to the maintenance planning and control function, and from there on all subsequent levels "downwards" of the decomposition tree.

6.5 Compound maintenance situations

The rotatable system

The principle of decoupling can be used as an effective measure to minimise pressure and constraints from the production function. A frequently used way for decoupling is the application of the maintenance rotatable system (see Geraers [39]). In essence, this system relies on having more technical systems of a certain type available than are directly required for the production function only. Those technical systems in addition to the number required for production are used as a buffer for the maintenance process. As soon as a technical system in use needs maintenance, that particular TS is replaced by a serviceable maintained identical technical system from the buffer (see the "valves" and buffers in figure 6.1). The technical system requiring maintenance is simply put in the buffer waiting for maintenance.
The rotatable system requires two different planning and control functions. One planning and control function deals with activities of replacing the rotatable technical systems quickly, whilst the other is concerned with the actual maintenance activities and the balancing of the two buffer stocks. Characteristically, the first type of planning and control is active close to the production function and therefore is much more decentralised than the workshop, which usually has the capability to maintain multiple technical systems from different production functions.

Echelonised maintenance
The rotatable system as a measure to decouple a production function and a maintenance function can also be used to decouple different maintenance situations. Sometimes, not all required maintenance activities can be carried out efficiently within one single maintenance unit. In case, the required skill to perform certain maintenance activities is outside the responsibility area of a maintenance unit, this skill must be contracted out to another more specialised maintenance unit, possibly even to a maintenance unit from outside the organisation. If technical systems are moveable, they can be moved to this specialised unit in a similar buffered system as has been described for the rotatable system. In turn, that specialised maintenance unit can apply the same principle to contract out activities of an even more specialised nature. The approach to channel technical systems through subsequent more specialised maintenance units is called echelonised maintenance.

If a technical system is not moveable, specialised maintenance groups have to move to the appropriate location. In this situation, these specialised maintenance groups may call in even more specialised maintenance groups if needed. In many respects, this situation may also referred to as echelonised maintenance, although maintenance capacity cannot be buffered, but has to be available if called in.

6.6 Conclusion
In the previous paragraphs several important aspects have been discussed. These aspects determine how maintenance units can be organised in an organisation. The characteristics of an elementary maintenance situation determines whether this type of planning and control is effective. In addition, it can be more efficient to separate these elementary situations further taking into account the basic considerations with respect to the technical system structure, the
On the determination of functional requirements in a maintenance environment

maintenance concepts and the capacity characteristics outlined in this chapter. This process of separation into smaller elementary units should be guided by socio-technical design principles. In the next chapter we shall discuss how the basic considerations can be used to guide the personnel system analysis and design process towards an efficient and socio-technical sound design.
7 Design and demonstration of a functional analysis and design framework for maintenance

7.1 Introduction

In this chapter we shall construct a framework for design based on the fundamental considerations which have been presented in chapter 6. We shall do so by performing the decomposition steps of the structural analysis and design process of a maintenance function. In figure 7.1 the three subsequent decomposition steps are shown.

Legend:
MC= maintenance concept, SFAD= structural functional analysis and design, DFAD= detailed functional analysis and design, PSAD= personnel system analysis and design.

Figure 7.1 The decomposition of the maintenance function.
The first decomposition step is the definition of the internal maintenance function itself. In our study the maintenance function is regarded as the highest decomposition level and therefore it is determined by definition rather than a result of decomposition of even higher levels.

In the second decomposition level the EUT-maintenance model is used as the decomposition criterion to get from the general maintenance function to its main functions. In chapter 6 we already discussed the EUT-maintenance model. For our study we shall concentrate exclusively on the maintenance planning and control function within the EUT framework.

The basic considerations discussed in chapter 6 actually serve a dual purpose. First, they provide the decomposition criterion for the planning and control function, and secondly, they can be operationalised into guidelines for the personnel system and design (PSAD) process. The PSAD-process uses the structural functional analysis and design functions and allocates them to task and operational group.

In the detailed functional analysis and design process each structural analysis and design function within a functional unit will be carried out by the group members according to their own decomposition criteria.

To illustrate the principles of decomposition in a maintenance environment an actual case situation will be discussed in view of the functional analysis and design process principles. In addition, suggestions for improving maintenance performance put forward by group members will be discussed to demonstrate the influence of a dynamic environment upon the functional analysis and design decisions.

7.2 The personnel system analysis and design design objective: the creation of semi-autonomous groups

Before we go into discussing PSAD-guidelines to create maintenance groups, we have to make clear that we shall exclude those maintenance activities which can be easily allocated to production operators, i.e. operators maintenance. Maintenance activities which can be carried out by operators should require no advanced maintenance skills, extensive work preparation or conflict with other regular operator duties. Normally, operators will carry out this type of maintenance on a more or less opportunistic basis without any formal planning, control and co-ordination involved. Further, we shall assume that a valid option for allocating
maintenance tasks is to contract out entire maintenance activities altogether with the accompanying planning and control functions. However, this latter option involves long term organisational and financial issues, which are not covered in this study.

For the development of personnel system analysis and design (=PSAD) guidelines it is vital to interpret the integrated organisation (re)design (=IOR) approach with respect to the way in which semiautonomous groups are formed. In chapter 4 we already discussed the principle of parallelisation and segmentation as means to create groups in a production situation. As a result of a structural functional analysis and design process functional maintenance units can be identified, which can be realised by either allocating them to production groups or by creating separate specialised maintenance groups. To realise the first option personnel having maintenance skills are allocated to production task groups. The second options sees maintenance as a specialised function requiring its own socio-technical group structure. The socio-technical principles parallelisation and segmentation must be applied here as well. Parallelisation in a production situation means that parallel production lines can be created where each line produces its own assortment of products. Autonomy of production groups allocated to a line is achieved because in this production process, task groups require only little or negligible co-ordination. In maintenance situations no direct analogy to the production line situation exists.

Segmentation of groups is achieved by separating production lines into smaller parts, with the result of more specialised groups. Segmentation is sometimes needed for efficiency reasons, but in sociotechnology segmentation should be avoided as much as possible because it creates an increased need for intergroup co-ordination. The purpose of segmentation is to group capacity types according to their skill in order to achieve a higher utilisation.

7.3 Using structural functional analysis and design decomposition criteria.

7.3.1 Determining guidelines for the personnel system analysis and design process

Our aim for the second level decomposition will be to determine the functional units required for the planning and control of all maintenance clusters. Compound maintenance situations
are in fact interrelated elementary maintenance situations, which may coexist at the same time. Eventually, the allocation of maintenance activities to task groups will define what maintenance units are to be dealt with in the detailed functional analysis and design process. The allocation of functions to personnel, for which the socio-technical approach is used, is not a matter of mathematical optimisation. Instead, different options for group parallelisation and segmentation have to be evaluated with respect to their effectiveness and efficiency and in direct participation with the individuals. In this study our guidelines can contribute primarily to the assessment of the efficiency aspect of task allocation. If it is known what clusters have to be carried out and with what frequency, then it is possible to estimate the workload for a longer time interval. All types of clusters are defined in the maintenance concept. The frequency for activation of clusters depends on the operation intensity, which is a function of time. As we know from chapter 6 not any combination of clusters is valid for consideration. Certain characteristics of maintenance clusters themselves, technical systems installed, and the available maintenance capacities, have to be considered. In this paragraph we shall focus on the transformation of these characteristics into design guidelines. This transformation can be carried out in two stages, which will be discussed in the next sections.

In maintenance functional units are identified on the basis of decoupling, which is based on the physical design and operation intensity of technical systems. In chapter 6 it was seen discussed how the decoupling potential of technical systems can be used in two ways, i.e. for decoupling the maintenance function and the production function, and to identify maintenance units. Decoupling is synonymous to a minimum of co-ordination since functional units, production units and/or maintenance units, can operate independently. Segmentation in maintenance situations occurs - after the decoupling potentials have been fully used - if there is insufficient volume of clusters requiring certain capacity types. The effect of segmentation is more centralised capacity oriented groups (e.g. specialised central workshop).

The reasoning in the previous section can be summarised in a slightly adapted integrated organisation (re)design approach to establish task groups:

1) determine the list of the maintenance clusters ("the activities to be controlled")
2) group clusters into functional units to identify decoupling potentials (the parallelisation equivalent)
3) estimate the workload for each functional unit
4) allocate functional units to prototype task groups
5) Apply segmentation if needed
6) Evaluate prototype task groups according to socio-technical criteria.

Task groups may be responsible for multiple functional units originating from different organisational functions. Steps 1 to 3 can be regarded as preparatory steps. Then, in socio-technical design the potential of various mixes of functional units can be assessed in step 4 to 6. Finally, if prototype task groups have been created and as a consequence the functional units have been allocated, higher level organisational groups can be formed. Operational groups carry out vertical co-ordination activities between task groups. In the end, possibly new business unit structures and other group structures can be formed, but these will not be discussed here. When all prototype group settings on all hierarchical levels meet socio-technical criteria and when they are agreed upon by the individuals, the prototype groups become the active groups.

In the following paragraphs the socio-technical adaptions and the final structural functional analysis and design step will be discussed in detail.

**Stage 1: determination of maintenance clusters**
Each cluster is defined in the maintenance concept and consists of a unique set of elementary maintenance rules (=EMR's) having a common set-up. In addition a cluster is relevant with respect to those technical system types on which the EMR's have impact and they require a certain amount of a mix of capacities. Clusters should contain information on what EMR's are carried out, the set-up required, the capacity types needed, and the workload per capacity type, with respect to the technical system types involved.

**Stage 2: grouping of clusters into functional units to identify decoupling potentials**
A technical system type is a set of physical components having a unique set of production functions (or modes of use). Because more than one specimen of a certain technical system type can be used in an organisation, clusters can be initiated not only as a function of the operation intensity, but also because of multiple occurrences of a technical system type. Technical system types and their realisations, the actual installed base of technical systems,
determine the potential of decoupling in a certain situation. Clusters can be grouped according to the scheme presented in figure 7.2. The qualification "decoupled" means that clusters related to those technical systems require no or only very little co-ordination with respect to the replacement of the technical systems. Technical systems with semi-decoupling usually don't require co-ordination on a short term basis, e.g. a monthly agreement between the maintenance function and the production function will be sufficient. Coupled technical system require most co-ordination, because any unexpected maintenance activity leading to a change in the current or planned condition of operation require to consult production control. The mobility property can be used as a criterion to decouple maintenance groups. In case technical systems are immobile, their location can be used as a criterion for the creation of separate maintenance groups. In practice, this is known as area maintenance.

**Stage 3: estimation of the workload for each functional unit**

Basically, there are two ways, which can be used in combination, to estimate the workload of groups of clusters. The first approach is, to use the maintenance concept in combination with
the expected (or planned) operation intensity to extrapolate cluster maintenance demand in
time. The second approach is to rely entirely on historical data of cluster maintenance carried
out. If a maintenance concept is in use for some time this approach may be preferred, since it
is algorithmically simple.

Stage 4: the allocation of functional units to prototype task groups

Individuals could be allocated based on the workload per (human) capacity group per cluster
group. Normally, there will be a preference for "integer allocation", i.e. only full time
individuals, which may result in considerable losses in efficiency for certain groups of
clusters. The creation of a first version of a group allocation will be difficult in situations
where the workload distribution over time will be highly irregular. Then, simply allocating
individual workforce size according to the mean workload will not always be desirable.
Incidental requirements for special skills can be brought in "on call".

Stage 5: segmentation of capacity if needed

If the efficiency of the first group allocation is unsatisfactory, i.e. workload variations are
high, segmentation of capacity types can be advantageous. The effect of this action is that
capacity types will be concentrated.

Stage 6: Evaluation of prototype task groups according to socio-technical criteria

At this stage all prototype task groups have been created, being responsible for maintenance
planning and control functional units, and possibly other functional units, with an acceptable
trade-off in required co-ordination and efficiency. However, in sociotechnology additional
cohesion-factors such as cultural background and age, may impose a regrouping of task
groups. In addition, other group structures, such as operational groups must be created
before the quality of the entire socio-technical design can be evaluated. Van Amelsvoort [3]
described extensively how a socio-technical design is achieved and evaluated. In our context,
maintenance is primarily relevant to task groups with regard to planning, control and
execution of the maintenance activities, and operational groups with respect to the required
co-ordination between groups.
7.3.2 The formation of operational groups

In the previous paragraph we focused on the formation of task groups. The basis for grouping was largely determined by the physical maintenance activities. To establish a complete planning and control structure, the requirements for co-ordination of the individual task groups must be addressed as well. In sociotechnology, so called operational groups, staffed with individuals being responsible for different functional areas, decide upon operational and mid term aspects which cannot be taken by individual task groups, because of the inherent danger of sub-optimisation. The formation of operational groups is not primarily an efficiency constrained issue as in the task group formation stage. Amongst other factors not discussed here are how the co-ordination requirements in terms of complexity and the variety of functions involved, rather than volume, determine the size and composition of an operational group. Although, the formation of operational groups is strictly a socio-technological design issue, we shall discuss some primary issues with respect to the influence of co-ordination requirements of maintenance task groups.

**Horizontal co-ordination**

The principle of decoupling is a structural measure to reduce the need for co-ordination in the first place. However, complete decoupling cannot be achieved in all situations. In case there is some degree of coupling, the first and most natural way of avoiding the need of big operational groups is to assess the possibilities of direct co-ordination between task groups, which might extend beyond day-to-day co-ordination issues. However, exploiting this option will be difficult and requires a relative stable environment. Task groups, having a limited scope of control, would need "prefabricated" co-ordination decision schemes valid for and agreed upon by all task groups involved. E.g. a production task group signals unacceptable downtimes for rotatable technical systems due to long maintenance leadtimes. A standard co-ordination agreement, authorised by higher management for say two years, could be to authorise additional purchasing of those rotatable technical systems by a maintenance taskgroup until the desired availability of technical systems has been reached.

**Specialised operational groups.**

The paradoxical term "specialised operational groups" is used to depict special types of operational groups which co-ordinate task groups operating within the same organisational
function. In maintenance, more than one task group may be needed requiring "mid-term" co-ordination, such as temporary re-allocation of maintenance clusters, because a task group is facing an unexpected temporary overload in activities.

7.3.3 Temporary group structures

Sometimes in maintenance, clusters themselves are substantial in workload but occur infrequently. E.g. major shut downs in chemical plants constitute a substantial workload, i.e. they exceed the regular workload extremely, but only during a brief period of time. Practically, a large portion of the workload must be contracted out, but parts of the regular maintenance task groups may be involved in the preparation and in the execution of a shut-down, which in fact is a large cluster of elementary maintenance rules. During shut-down normal plant operation is interrupted and therefore the regular task group structure is invalidated.

This example indicates the need for temporary maintenance task groups geared towards a largely planned, but exceptional, maintenance cluster. Depending on the impact of this temporary phenomenon it may also be required to regroup the regular operational groups. It would lead too far to discuss all options of temporary group structures in this study. Still, we must assume that in some maintenance situations a static task and operational group structure may not always be the best choice.

7.4 Determination of structural functional analysis and design functions per group

Before each group can start with the detailed functional analysis and design process the structural functional analysis and design functions must be determined clearly, since these functions determine where to start the decomposition.

In case pure maintenance task groups have been formed, cluster characteristics determine what type of elementary planning and control situations are appropriate for a task group. The set of clusters, a group is responsible for can be categorised by the properties presented in figure 7.3. In its simplest form one maintenance task group is "connected" to a single elementary maintenance situation. However, in most actual situations combinations of elementary situations, i.e. compound maintenance situations, are more likely. In this
paragraph we shall discuss how valid planning and control structural analysis and design functions are determined for individual task groups. We do this by discussing the potential types of task groups each dealing with a characteristic set of activities.

**Task groups allocated to multiple types of clusters.**
A single task group may be responsible for multiple elementary maintenance situations and therefore they are responsible for multiple planning and control functions. In this compound maintenance situation an additional structural functional analysis and design function is required for internal group capacity allocation, because each elementary situation requires its own allocated capacity.

**Specialised task groups in maintenance performing other activities.**
Sometimes maintenance task groups receive additional responsibilities in the personnel system analysis and design process for which other task groups lack the technical skills, or because a specific task group would be inefficient. E.g. maintenance task groups sometimes perform other technical duties such as setting up technical systems for different production runs, install new technical systems, or perform small engineering and modification tasks.
Additional tasks always bear a risk that they do not comply with the cluster characteristics presented in figure 7.3. Theoretically, each misfit may create an additional planning and control situation and additional co-ordination. On the other hand, these additional tasks can easily be carried out by a maintenance task group if these tasks fit with the characteristics of the clusters the task group is already responsible for. In practice, most additional tasks will fit as a mono or as a collection of mono's.

7.4.1 The initiation of the detailed functional analysis and design process

In this study we assumed a situation where only planning and control functions are allocated to maintenance task and operational groups. In reality, however, the structural analysis and design functions of spare parts management and the maintenance concept function have to be included as well for a valid personnel system analysis and design process. In addition, each decomposition step creates an additional evaluation function, which has to be allocated as well. For the planning and control function three decomposition steps were needed. The first two decomposition steps concerned the definition of maintenance and the planning and control function making a separate evaluation function superfluous. On the other hand, the criterion for the third decomposition step, the technical system and the cluster characteristics, justify an evaluation function. Changes in the technical system assortment or in operation intensity or in product assortment will influence directly the personnel system analysis and design process and consequently the detailed functional analysis and design process and, possibly, the information system analysis and design process in turn.

Additional information requirements

It is important to note that the structural functional analysis and design process concerned only the structuring of the maintenance function with respect to the achievement of maximum maintenance performance, traded off against the available means for realisation. During a detailed functional analysis and design process, possible additional information requirements from other organisational functions have to be satisfied too, which may result in extra functions in the decomposition process.
Formalisation of functions

Although each maintenance task and operational group should carry out the decomposition themselves, and make their own decisions how formal a function should be, one can expect that the degree of formalisation, i.e. the number of decomposition levels, increases in three cases:

1) if external information requirements are strictly formal;
2) if functions are activated frequently, so called routine functions\(^6\) (see chapter 3);
3) if performing a function would require too much (human) capacity\(^7\).

The information system analysis and design process can influence up to a certain extent the third reason by lowering development and running costs of information systems.

7.5 The case-study: The PTT Post Maintenance service

7.5.1 Objective of the case study

In this chapter we shall illustrate the functional analysis concept developed in this study. We shall do this by analysing an actual maintenance situation namely the the maintenance unit of the main sorting centre of the Dutch PTT, located in 's-Hertogenbosch. The major steps of a functional analysis and design process will be carried out for this illustrative case situation. Choices made in the current maintenance situation will be commented. In addition, some actual questions the maintenance task group has with regard to the improvement of maintenance performance on a detailed level will be discussed as well. It will be demonstrated how these questions relate to the current detailed functional analysis and design functions and to the personnel system and information system design. Special interest will be paid to the detailed functional analysis and design redesign options in solving these questions.

\(^6\) provided that those functions can be decomposed to a unambiguous level.
\(^7\) the same remark as in footnote 6.
7.6 The 's-Hertogenbosch hub maintenance unit

7.6.1 The general PTT structure

The business objective Dutch PTT Post b.v. is to collect, sort and distribute postal items of various kinds. In order to support this objective the company has twelve main sorting centres in five different regions in the Netherlands. In addition, the company uses 168 so called pre-sorting centres throughout the country. The annual capacity in distribution totals to about 21 million items (94% is strictly business mail). Recently, the Dutch PTT Post has been transferred from a state owned subsidised company to a commercial company operating in an open market. This transition process manifests itself in several ways. One important aspect is that the company now must be financially self supporting, which increased the motivation to continuously improve service levels and efficiency. A direct consequence of this new "drive" is the reduction from twelve main sorting centres into six regional distribution hubs.

The PTT Post company service unit is responsible for all internal service activities, which include purchasing of all consumable goods, installation of new and modified equipment and maintenance of technical systems. The service unit employs about 550 people in total and about 375 of them are responsible for maintenance.

In 1991 the decentralised maintenance departments, which are located in twelve regional main sorting centres, have been joined hierarchically by a central maintenance organisation. This central maintenance organisation, located in The Hague, is hierarchically responsible for all maintenance operations. Their primary responsibilities concerns co-ordination of super main sorting centre level maintenance activities and to provide managerial and technical support, and to intermediate between the user of the technical systems (the client of the service organisation) and the maintenance departments. The maintenance organisation, in the new PTT company strategy, offers maintenance services to all internal PTT clients on a commercial contract basis. In this contract the national maintenance organisation guarantees a minimum of 95% availability of all technical systems located in the main sorting centres. In figure 7.4 the national maintenance organisation is shown.

The hub maintenance section is responsible for the maintenance of all technical systems within the main sorting centres. Currently 12 main sorting centres are distinguished and at the five regional main centres centralised support groups are located, with responsibilities for
purchasing spare parts, work preparation and administrative tasks for all maintenance activities within their region. E.g. the support group of the South-east region is located in the 's-Hertogenbosch regional main sorting centre and supports three main sorting centres ('s-Hertogenbosch, Sittard and Arnhem). This support group employs currently six administrators, three purchasers annex work preparators, one chief hub maintenance, one chief external maintenance services and one regional maintenance director. Each hub maintenance workforce is located on the main sorting centre site, close to the technical systems to be maintained. The external or field service maintenance section is responsible for all technical systems which are located at the post offices and the smaller pre-sorting stations. Since the external maintenance sections are concentrated at the regional main sorting centres, performing maintenance "on site" demands a high degree of mobility of the field service technicians. They rely on service cars which are also used to store spare parts.

The Staff bureau is responsible for internal support, such as calling down of parts (in co-ordination with the central purchasing department), computer support & system services, warehouse handling and administration. The TPI section is responsible for the installation of PTT specific technical systems, and GBI for all maintenance of buildings and the mechanical and electrical equipment which is directly attached to the buildings.
7.7 The production function of a main sorting centre

The aim of a main sorting centre is to sort various types of incoming mail (letters and items) to the appropriate main sorting centre closest to the receiver of the mail. During this process improper addressed mail will be corrected whenever this is possible. Sometimes, also some pre-sorting of mail into districts and streets can be done to ease the sorting job of the delivery mailman. In Figure 7.5 the characteristic of the mail sorting process including the major types of technical systems, is shown.

The business mail primarily originates from institutional senders and is normally suitable for automatic identification and sorting. The ALIMA (optical character recognition and indexing machine) scans the address of a mail package automatically and prints a bar-code on the package. Improper addressed or unreadable mail items have to be identified manually by using HIA's (manual indexing desks). Individually mailed items are normally more diverse in size, shape and weight requiring some shifting and positioning. The SOSMA (heading-facing-canceling machine) rejects mail which cannot be handled physically by the other sorting technical systems, in addition to positioning the mail in an upright position (stamps in the upper right corner) and stamping them if necessary. The physically "unfit" items have to be sorted completely manually. Physically "fit" mail can be sorted on the
SORMA (high volume sorting machine) in multiple sorting runs into separate sections by reading the bar-code. The incoming mail from other main sorting areas is also sorted on this technical system. Finally, the sectioned mail will be packaged in large mail bags and trays in the expedition department and distributed to other main sorting centres (outgoing mail) or to the smaller receiving stations (incoming mail).

7.8 Information sources used for the functional analysis and design process

For the functional analysis and design process several information sources have been used. In addition to interviews with the central maintenance organisation and the local maintenance staff in 's-Hertogenbosch. Formal material such as work order history records have been used also. The maintenance concept which is used at the 's-Hertogenbosch maintenance unit is an important formal source of information which is available as an internal publication. All internal maintenance is based on the maintenance rules stated in the maintenance concept (see chapter 6). Other important information is the work order history file, which is generated by the current information system. Both information sources will be discussed in this paragraph.

7.8.1 The maintenance concepts

The 12 main sorting centres all maintain technical systems which are both, unique for the type of business the PTT is in, and technically very similar or even completely identical. One of the first initiatives of the newly formed centralised maintenance organisation was the introduction of nation wide maintenance concepts for all PTT technical systems. The maintenance concepts have three purposes:
- To serve as a basis for establishing maintenance contracts (definition of the service maintenance is offering,
- to estimate the required maintenance resources, and
- to be used as standard work orders in the operational planning and control function.

The PTT maintenance concepts contain repetitive work items only (used based maintenance and condition based maintenance rules) grouped to the major technical system installations.
In addition, a history log is maintained on any work order carried out, which also includes failure based maintenance rules and modifications, set ups etc. At the time this research has been carried out the maintenance concept has been used for about one year and about 8200 work orders have been recorded. We can use therefore this maintenance history as a valid source for our analysis.

7.8.2 The BISTOD information system

The BISTOD ("Bestuurlijk InformatieSysteem Technische OnderhoudsDienst", Eng: Information management system for the Technical Maintenance Department) information system is a customised version of a commercially available standard software package named RAPIER developed and distributed by SQL Systems. Originally, RAPIER was introduced at the main sorting centre in Rotterdam and has been customised to their particular needs at a time before all maintenance activities have been joined in the reorganisation process. At that time maintenance groups in a main sorting centre operated completely independently. As a result, each group developed its own working procedures and practices. Sometimes, large differences in the way maintenance was managed turned up. Some groups, such as in the Rotterdam internal maintenance unit, were relatively advanced in their approach towards an efficient management of maintenance activities, whereas at other locations maintenance was carried out more or less intuitively and only little attention has been paid to establish management information in order to sustain a minimum of control.

In view of this situation and the company objective to improve efficiency and service levels, the new central management groups first priority was to get all hub maintenance groups on an equal level of management. On the procedural level, all hub maintenance groups must base their repetitive maintenance work orders on the centrally defined and approved maintenance concepts. A new information system for the maintenance function must be realised that could be used, preferably unaltered, by all maintenance units. This new information systems should provide:
- support for administrative functions, i.e. all maintenance jobs and spare parts transactions have to be registered properly;
On the determination of functional requirements in a maintenance environment

- basic planning and control functions, i.e. repetitive activities and postponable breakdown maintenance should be scheduled formally in addition to the registration of high priority breakdown jobs;
- facilitate a basic structure for the evaluation of the maintenance function.

The central maintenance group looked for a quick but workable solution. Global criteria concerning organisational, financial and adaptability issues were used to evaluate and select a standard software package. The RAPIER software package, which was already in use for some time at the Rotterdam maintenance hub has been chosen as the PTT maintenance information system after some modifications have been carried out resulting in the new PTT maintenance information system BISTOD.

For the hub maintenance group in 's-Hertogenbosch the introduction of BISTOD resulted in a discontinuation of their own information system. The 's Hertogenbosch hub maintenance group was also urged to adopt their working procedures to make them fit for BISTOD. Now, BISTOD has been used at the 's-Hertogenbosch hub for about a year.

7.9 Determining functional units

Because we narrowed the scope of our research to the maintenance planning and control function, we can begin at the third decomposition level. In the first level the maintenance function itself is identified. In the second level only the planning and control function are relevant to us. Before the final structural functional analysis and design functions on a task group level can be determined, personnel system analysis and design has to be carried out. We shall step through the personnel system analysis and design guidelines presented in paragraph 7.3 in order to identify suitable functional units. Then we shall evaluate the current group structure in view of these functional units. In the end, appropriate structural functional analysis and design functions will be determined.

Stage 1: determine maintenance clusters

The maintenance clusters have been determined about a year ago. The clusters are grouped on primary technical system level (see description of available technical systems in paragraph 7.7), e.g. in terms of HIA, SORMA, etc.
Stage 2: grouping clusters into functional units

At the 's-Hertogenbosch main sorting centre 104 technical systems (12 technological different technical system types) are used in the collection, sorting, grouping and distribution process. The technical systems are located in a two story building of about 8000 m² in total. All technical systems including their technical subsystems are regarded as immobile, which means that maintenance must be carried out at the actual location. Maintenance capacity should be mobile within the range of the locations of technical systems they are assigned to maintain. Although most technical systems are not of a singular type, all technical systems are needed for operation during the production window, i.e. there is no redundancy and interchangeability within a technical system type is limited.

All technical systems are required for operation (in operation or at least in standby mode) every day of the week in the periods of 18.00 to 23.00 hours, 23.30 to 6.00 hours and sometimes 8.00 to 10.00 hours. The primary criterion for effective maintenance is the availability of the technical systems during these production periods. Any downtime between these periods is regarded irrelevant for the production function.

Decoupling potentials are limited to the scheduled non-availability times. The complete set of clusters connected to a major TS can be identified as a functional unit during scheduled operation hours and decoupled at all other times.

Figure 7.6 The workload distribution (in hours of work) of all maintenance activities
Stage 3: estimation of workload per functional unit

Figure 7.6 shows the total workload in time of historic maintenance work orders. In appendix 1 the workload has been grouped into technical system types which are grouped together in areas within the building. The mean daily workload is about 6.6 hours with a standard deviation of about 10.4, which leads to the conclusion that the workload is highly irregular. These figures provide only a very rough indication to the ideal maintenance crew size. The crew size will be primarily determined by the required service level the main sorting centre users are willing to pay for. In other words, a larger crew size can be justified if the availability of the TS's will increase too. Some basic historic analysis is possible, because all down and uptimes have been recorded in addition to the actual maintenance activity times. The relation between crew size and downtime can be simulated by using this data assuming a simple priority rule. However, the effect of the repetitive maintenance during non production times on the reduction of failure occurrences during production times cannot be simulated. Clearly, the determination of the ideal crew size would require some experimentation in practice (satisficing approach). E.g. the maintenance performance can be evaluated for a certain period of, let's say 2 months, during which crew size has been reduced with one man. At the time of this writing in some main sorting centres experiments have started with a bare minimum crew on-site in combination with service technicians on standby.

Another important aspect in determining the crew size (and ditto the maintenance task group size) is the technical system production pattern itself, which requires a three shift operation.

Stage 4 to 6: PSAD design

Because only one functional unit with respect to maintenance planning and control has been identified and hub maintenance in other main sorting sites is highly impractical, the largest possible task group consists of all maintenance men located at a single main sorting centre site.

Segmentation in skills is unnecessary since all crewman are regarded multi skilled. Segmentation in locations is highly inefficient as can be seen in the figures 7.6 and in the workload barcharts per technical system type presented in appendix 1. It seems unlikely that the extreme workload peaks calculated from the maintenance work history file reflect the actual work order history. The maintenance director suspects the history entry procedure in BISTOD causes the maintenance activities to appear in groups, because of being entered into
the system groupwise at regular time intervals. Therefore, the peaks exaggerate the actual situation. In addition, the production task groups working during the night, whereas the majority of maintenance personnel works in a day shift, make a combination of production and maintenance units unfeasible. Therefore, the current choice of one specialised maintenance task group seems to be justifiable.

Although, operational groups in a socio-technological context are not known in 's-Hertogenbosch main sorting centre, regular joint meetings between the maintenance director, and his staff, and the production director, with his staff, have a similar function. In fact, the suggestions to improve maintenance performance discussed in paragraph 7.11 spawn from these meetings.

7.10 The determination of structural functional analysis and design functions

The types of planning and control functions are determined on the basis of the maintenance clusters within the functional units. The clusters defined in the maintenance concept have been grouped into intervals of:

- one day (daily maintenance);
- every second day;
- every week;
- every second week;
- every fifth week (one month);
- every 13th week (one quarter of a year);
- every 26th week (half year);
- every 52nd week (one year).

Every cluster contains at least one EMR (=elementary maintenance rule). In general the EMR's in clusters have no or only evident dependencies with respect to the sequence in which they have to be carried out. Therefore, the clusters which contain only one EMR and those clusters containing multiple but independent EMR's can be regarded as mono's and as sets of mono's respectively.
On the determination of functional requirements in a maintenance environment

**TS operation conditions during maintenance execution**
Although some condition based maintenance rules are currently under consideration which require a running condition of the technical systems at hand, all identified EMR's require a shutdown condition of the technical systems in case maintenance has to be carried out.

**Maintenance capacity skills and mobility**
The hub maintenance workforce is generally specialised for PTT typical technical systems, which is achieved by a two year training period immediately after the intake of new maintenance crewman. Within the main sorting centre location all men are multi-skilled, i.e. they can carry out any maintenance job for any technical system.
Due to the static nature of the technical systems a mobile maintenance crew is a prerequisite.

There is only one effective planning and control type required in this situation, i.e. the job shop on a shop floor level. This means that primarily maintenance clusters can be scheduled to a simple priority system (e.g. First-Come-First-Served priority). Due to the nature of the operation requirements, it seems unwise to schedule regular repetitive maintenance clusters requiring shutdown conditions during the operation periods. In the time intervals in which the production function wants to maximise availability, only high priority failure based type clusters are allowed (failure based maintenance to get technical systems back in running condition). All repetitive and low priority postponable failure based maintenance type clusters are to be carried out preferably when the technical systems are not used.

**7.11 Options to improve hub maintenance**
In an environment in which personnel is motivated, there is a lively interest for continuous improvement. At the end of each working day a discussion takes place in which the performance is evaluated, operational information is exchanged and suggestions between shift groups as a part of the daily routine at 's-Hertogenbosch hub maintenance unit.
Suggestions for improvement may address very different areas, and consequently some may eventually influence the functional requirements for the BiSTOD maintenance information system. It is the maintenance group, on a detailed functional level that is responsible for

---

*There is, however, an exception for some rare maintenance activities requiring special immobile tools only available in workshop located within the main sorting centre.*
evaluating each suggestion for improvement. To illustrate the difficulty of evaluating suggestions, and in particular for their realisation consequences, we shall discuss some actual options for improving maintenance at the 's-Hertogenbosch hub maintenance unit.

**Reduction of operator induced failures**

The maintenance group suspects that a lot of calls for maintenance during the intervals in which the technical systems are in operation are caused by an improper handling by the operators. However, the main sorting centre users (the production function) don't pay any particular attention to this suspicion. Actually, this problem doesn't decrease the maintenance units performance, but due to the contract in which the user has to pay for all maintenance, additional potentially unnecessary costs are made.

A first step to address this problem is to identify those maintenance work orders that are caused by operators induced failures objectively. This is possible by counting those types of work orders in addition to the contribution these work orders make in terms of the required maintenance capacity, spare parts and costs. Then, this information can be presented to the

![Diagram](image)

**Figure 7.7 The ISAC diagram for the determination of the work order causes.**
On the determination of functional requirements in a maintenance environment

users of the technical systems to decide what they want and can do about it. From a functional point of view this would mean that:

1) A detailed functional analysis and design function is needed temporarily (let's say for a year) to register the reason for maintenance at a work order level.

2) A detailed functional analysis and design function is needed (to be used only once!) that summarises all work orders for the criterion "operator induced failure" in terms of capacity and materials used in volume and money. In addition, the total maintenance efforts should be presented to assess the relative importance of the problem area.

Both functions are shown according to the diagram conventions defined in chapter 5 in figure 7.7. Both functions are not decomposed to an universal unambiguous level, because all parties involved understood unambiguously the functional requirement in this form.

Function A represents "Register cause of work order and store each registration" and is triggered for each completed work order. Function B represents "determine the score of operator induced failure, the score for other causes and the total number of work orders (size of the set) for each technical system during the entire observation period" and is triggered once-after a one year period.

After an evaluation of the BISTOD information system it was concluded that these functions are not available. However, partly this omission can be compensated by using so called action code fields in the work order table specifically for registering the cause of a work order. Each maintenance crewman must be instructed to use this field only for this registration purpose. He also must fill in "user failure" or "regular maintenance" in this field. Other entries must be considered invalid.

Since over 8000 work order entries are expected for a year period, manual scoring of work order causes is considered to be too cumbersome. Because of the one-time-only characteristic of this question a regular adaptation of the BISTOD system is also unfeasible. The best way would be to ask an BISTOD-development specialist to provide a download of work order history fields "work order entry date", "object", "capacity group" "amount of man-hours", "spare parts used" and "action codes". Then this file can be imported in a regular spreadsheet program to determine the scores and present them graphically to enhance the understanding of the production operators.
Design and demonstration of a functional analysis and design framework for maintenance

Evaluation of downtimes and response times.

To enable an evaluation of the hub maintenance unit performance during evening and night shifts, a special report is needed. The report should have the following format:

<table>
<thead>
<tr>
<th>Work order number</th>
<th>TS time of breakdown</th>
<th>time of arrival of main. crew</th>
<th>downtime</th>
<th>response time</th>
<th>names of main. crew</th>
<th>jobs carried out</th>
</tr>
</thead>
</table>

Functionally, this report can be interpreted as a single detailed functional analysis and design function that provides the types of information requested and sorted by each individual technical system listed in this report specification.

An evaluation of the BISTOD system with respect to the presence of this DFAD-function proved to be disappointing. However, this shortcoming can be compensated quite easily, since:

1) the required input data can be provided by other functions already in use within BISTOD;
2) BISTOD data base management system provides a report generator in which reports can be defined by ISAD-specialists on the fly.

Automatic time logging

Before the introduction of the BISTOD information system a special operators button was operational, which was pressed by the operator as soon as a failure at the technical systems at hand occurred. Using the button caused a signal to go off calling for a maintenance man to go directly to the failing technical system. The button also triggered an automatic entry for the time the technical system was going down. As soon as the failure has been resolved, the button is pressed again and the system can be restarted and another automatic data entry is made for the time the technical system is operational again. The maintenance group had to give up this special device because it wasn't a part of the standard specification of the BISTOD information system. The current version of BISTOD still allows, and actually requires, only manual time entries. Because the maintenance group and the management of
On the determination of functional requirements in a maintenance environment

the main sorting centre are interested in an accurate by the minute reporting of the downtimes and the maintenance response times, these manual entries require a high level of discipline by the maintenance man being under pressure to make the entries accurately. It is suggested that a button triggered data entry to register technical downtime and maintenance crew response times can provide very accurate data entries, without disturbing the maintenance process to much.

Clearly, this issue addresses the realisation aspect of functional requirements. The original functional requirement defined in the detailed functional analysis and design functions prescribing to register the time the technical system went down, the time of arrival of the maintenance capacity and the time the technical system is "up" again, still is valid as it stands. The automatic time logging would provide extra convenience to the maintenance group and increased accuracy in data entry by a physical device (hardware) directly interfaced with the information system (software). Installing such a device including the required interface software is clearly in the domain of information system analysis and design specialists and technicians. Eventually, the cost for installing this system has to be traded off against the benefits of automatic data logging. At the time of this writing, this system has already been installed and tested.

7.12 Conclusion

The case description presented in previous sections demonstrated the functional analysis and design process in relation to the personnel and information system analysis and design processes. As a first impression, the whole design procedure may seem overwhelming in view of the findings. The personnel system analysis and design and the structural functional analysis and design functions are left untouched, because the original reasoning which has been effectively used for some time seemed justifiable and therefore can be approved. The true benefits of our design approach however, will become evident when complex maintenance situations are subjected to an analysis. The suggested improvements in the PTT case could be related to the appropriate design consequences quickly, because all functional analysis and design, personnel system analysis and design and information system analysis and design decisions are known beforehand.

Yet assessing the practicality of the design framework is not possible. The core design process, i.e. the detailed functional analysis and design process, has been omitted in this
initial practical test. The detailed functional analysis and design process should be carried out by the actual maintenance task group members, which would take quite some time due to the level of detail involved. A long term commitment of the maintenance task groups is required to fully carry out a detailed functional analysis. Further long term experimentation in more complex situations is desirable.
On the determination of functional requirements in a maintenance environment
8 The application of functional requirements in evaluation practices

8.1 Introduction
In chapter 2 two different strategies with respect to the realisation of information systems have been presented. In-house development is the traditional approach which potentially guarantees an exact realisation of all functional requirements. Practically, differences between functional requirements and in-house information systems occur only if the organisational unit responsible for the development of information systems has a backlog in development activities. The other strategy, which concerns the application of standard packages aimed at avoiding substantial development efforts, requires an evaluation and selection process.
In this chapter we shall demonstrate how the principle of functional analysis and design can be used to facilitate a package evaluation process.

8.2 The completeness criterion
It goes without saying that the completeness of information systems is of primary importance. It depends on the importance of functions if an omission is a serious matter or not. In our view any formal function specified in the functional analysis and design process is required for carrying out tasks which contribute to organisational goals. Therefore, a maximum of completeness of an information system will be achieved if all formal functions are encoded in an information system.
In order to check for completeness, functions have to be defined formally. The precision of the evaluation is increased if more detailed lower level functions are used as evaluation criteria. The precision is maximised if elementary or standard functions are used. However, not all functions can be detailed into standard and elementary functions, nor would it be desirable to do so always.

---

9 The detailed functional analysis and design process determines what functions are suitable for further decomposition into standard and elementary functions.
On the determination of functional requirements in a maintenance environment

Since professional developers of standard packages strive at maximising sales volume whilst minimising development effort, a tendency exists to "generalise" functionality. They hope that only minor modifications to their packages will be required to make them fit in a particular situation. Application areas such as the maintenance function are perceived as complex because a decomposition process reveals a considerable number of decomposition levels, a large number of functions and data streams in most situations. On top of that, a considerable number of functions cannot be decomposed into elementary functions due to lack of understanding. Consequently, chances are small to find at least one package that matches all functional requirements perfectly or to be sure about ones preference. To assess the potential for adaptation on both sides, i.e. the developer and the user, the purpose of an evaluation process has to be enhanced.

The original sole purpose of an evaluation process is to identify what differences exist between functional requirements of an individual situation and the functionality provided by an package. We shall enhance this purpose of an evaluation process to: the assessment of functional differences aimed at the determination of functional mismatches and to whether these mismatches can be solved.

Both, the original and the enhanced evaluation types will be described separately in two subsequent paragraphs. Firstly, the original evaluation process (see also chapter 2) as a pre-selection instrument will be revised in view the way in which functional analysis and design results can be used. Secondly, an enhanced version of the original evaluation process will be discussed, which facilitates the assessment of functional mismatches.

8.3 The original evaluation approach

In chapter 2 a basic procedure for evaluating standard packages has been presented (see figure 2.1). Now, having developed an functional analysis and design framework, the steps in the evaluation procedure can be described more precisely.

Essentially, the determination of differences in functionality is a straightforward process. The basis of the evaluation process is the decomposition tree\(^{10}\) which is determined for the individual situation at hand and the "standard" decomposition tree of a package to be evaluated. In our view it is absolutely essential that a decomposition tree is available of all

\(^{10}\) We shall denote a (sub)set of functions located on various decomposition levels as a decomposition tree.
The application of functional requirements in evaluation practices

packages which are to be evaluated. Since the determination of a decomposition tree is
coupled to the personnel, i.e. the developers in this case, which (partly) carry out the
decomposition themselves, only the developers can provide such a decomposition tree.
However, if in practice a decomposition tree of a package is not available, a thorough
examination of available user documentation in addition to a tedious "hands on" trial and
error process with an actual installed version of the package becomes necessary to construct
the decomposition tree. This construction process itself may be restricted in terms of the
(number of) identified functions, if the documentation of a package is limited in describing
the basic functions available, or if a different or possibly non-consistent terminology has
been used. It is necessary to have a consistent terminology in which functions have been
documented to allow a direct comparison at all.
The evaluation with respect to completeness is performed by walking through the
decomposition tree top down, whilst comparing each function property with the available
functions on the same level of the package at hand. The top down approach is very efficient,
because if at a high decomposition level differences are identified, the package will be
rejected without spending more time in comparing functions on lower levels.
Essentially, this general top down approach has been already propagated in the general
evaluation approach presented in chapter 2. Because of the modular concept of defining
functions on various decomposition levels the distinction of where the "rough" pre-evaluation
ends and where the detailed evaluation begins can be made in a flexible manner; we consider
an exact distinction not necessary.
High in the decomposition tree functions are identified only roughly, thus allowing a quick
identification of a possible lacunae in a package's basic functionality. Down at the lower ends
of the decomposition tree, the increased precision in which functions are defined facilitates
an equally precise assessment of mismatches and their consequences. The traditional idea of
performing a pre-evaluation prior to a detailed evaluation is to decrease the effort of carrying
out a detailed evaluation by rejecting rather incomplete package offerings in an early stage.
The top down approach presented in this chapter actually has the same effect. As early as
possible unsuitable packages will be rejected. This revised evaluation procedure is
schematically shown in figure 8.1.
A comparison of figure 2.1 and figure 8.1 immediately reveals the two basic differences
between traditional and the revised evaluation approach. The determination of functional
Figure 8.1. The revised evaluation procedure.

requirements is now regarded as an integral part of designing fully operational organisational functions and it is not specifically performed for the evaluation purpose. Also, there is no distinction between a global pre-evaluation and a detailed evaluation. The final selection of effective packages is based on the efficiency (see chapter 2) requirements to implement a running package. Still, the revised evaluation procedure is aimed at the primary purpose of evaluation.

In the next section the enhanced evaluation process will be discussed with respect to the determination of potential compensation measures.

8.4 The enhanced evaluation approach

If the tasking of a function which will be used as a starting point for an evaluation is large, the chances that a package will meet all functional properties are rapidly diminishing. If we take the maintenance function as a starting point of a functional analysis and design process, that due to decomposition typical group and individual preferences are being taken into account.
The application of functional requirements in evaluation practices

Figure 8.2 The enhanced evaluation procedure (Note: for clarity functional and personnel system analysis design steps have been left out, since they are identical to figure 8.1, i.e. they "feed" the survey step at which the procedure starts).

apart from other situational circumstances, we must assume unless proven otherwise that a maintenance function will be unique in each individual situation. This means that in practice it will be virtually impossible to find, or to develop a package which meets exactly all functional requirements in an individual situation, despite numerous offerings in the market. Therefore, the assessment of functional mismatches becomes the appropriate option in view of the application of packages for complex functions such as the maintenance function. In this paragraph we shall discuss if and how functional mismatches can be resolved. We shall refer to this process as the enhanced evaluation process.

Figure 8.2 shows the procedural scheme which can be used for an enhanced evaluation. Due to the decomposition tree structure the evaluation process progresses in steps top-down recursively, assessing each individual function.

The key enhancement in this approach is that the evaluation process builds a list of potential compensation measures for each function mismatch (in case compensation is regarded
appropriate). In the end, when all potential packages have been evaluated, only those packages remain which are distinguished as being effective, considering the list of required compensating measures identified for each effective package.

Before going into enhanced evaluation process details we shall discuss what is meant by compensating measures.

8.4.1 Compensating functional mismatches

Basically, if a functional mismatch is discovered between functional requirements on one hand and the functions provided by a package on the other hand, the following three distinct options are available for correcting a mismatch:

a) Modify the standard package at hand.

b) Assess the possibility of an alternative decomposition criterion.

c) A mix of (a) and (b).

ad a) Essentially, modifying a standard package in any direction (e.g. enhancing functionality, improving user friendliness, increase data storage- and retrieval speed, reduce system support, etc.) is a development effort, provided that it is precisely known what direction a modification must take. In our case, the specified functions from the users side are used as a basis for "patching" a package. This option is convenient for the organisation evaluating a package at hand, but may be costly or simply unacceptable for the vendor of the package. In case the vendor doesn't allow alterations, in-house development or starting an additional evaluation process for only the mismatched functions involved as a "startingpoint" may be the only alternative.

ad b) Assessing the potential of alternative functional requirements necessitates a (partial) functional analysis and design process. This process differs from the "regular" functional analysis and design process with respect to its objective. The original objective of a functional analysis and design process is to define functions which are regarded most effective in view of realising organisational objectives, i.e. to maximise the validity of a function. In this option, the objective is to achieve an acceptable compromise between required effectiveness and the available effectiveness of a package. In the best case a new procedure or a changed habit or practice can be
developed without sacrificing effectiveness too much ("It is just another alternative"). The worst situation occurs if the package allows only one predefined procedure. The only assessment a functional analysis and design process can make in such a situation is to verify whether this provided package function is acceptable or not ("take it or leave it"). Any suggestions for a functional redesign are not carried out immediately, because full implementation is only valid if that particular package is selected for use. The problems of resolving a functional mismatch are bigger if the mismatch is already discovered in the early stages of decomposition. E.g. it may prove to be difficult if structural functional analysis and design functions are submitted for redesign, because the personnel system analysis and design process must first resolve the theoretical consequences for the chosen socio-technical group structures. Consequently, the detailed functional analysis and design process will be hampered by the non-existence of "fully-operational" socio-technical groups. Therefore, this option is only recommended if low-level detailed functional analysis and design functions are assessed.

ad c) The potentially most complicated situation occurs when neither a (partial) functional analysis and design process nor a package modification alone can resolve functional mismatches. Depending on the severity of the restrictions for modifying the package, the constraints for a (partial) functional analysis and design process will be set. It may be necessary to re-evaluate the standard package and its modifications iteratively to arrive at the best solution. Consequently, this option is much more tedious in comparison with the pure option (a) and (b) and should be avoided if possible.

Unfortunately, the options described above cannot be applied in isolation for the function currently at hand. Since a functional analysis and design process follows a "top-down, left-to-right" pattern of decomposition, the same is true for the partial functional analysis and design process (option b). Therefore, it is vital that functions are also evaluated in an "top-down, left-to-right" order. Then, whenever a change in a function is considered, other functions, to the left of the redesigned function, have to be redesigned immediately as well. These newly redesigned functions replace the "old" functions as evaluation criteria in the remainder of the evaluation process.
8.4.2 The enhanced evaluation procedure

In figure 8.3 a detailed scheme is presented of the evaluation process (distinguished as a single step in figure 8.2) itself. In the following sections we shall discuss each subsequent part of the process.

*Step 1: define the function from which to depart*

The decomposition tree consists of functions located on various levels of decomposition. Each decomposition step introduces a new, more detailed, decomposition level with
functions, each having its own unique decomposition tree. Basically, this approach results in relative modular functions (see chapter 5, the scope of functions in relation to the decomposition level), since each function and its decomposition tree can be evaluated in relative isolation, i.e. functions are modular only in a top-down direction.

In theory, each individual function can be chosen as a starting point. In case, a package for the entire maintenance function will be evaluated, the maintenance function itself is the starting point. If just a package for spare parts management is required, the starting point will be the inventory control function.

**Step 2: Identify the socio-technical groups responsible for the relevant functions**

As a result of the personnel system analysis and design process socio-technical groups are formed who are responsible for the detailed functional analysis and design process of the functions. Each group decides when a decomposition stops and consequently, how formal detailed functional analysis and design functions will be. Only group members can interpret informal functions and define what properties are relevant for an evaluation.

If the "starting point" function is a structural functional analysis and design function, more than one socio-technical group may get involved, while detailed functional analysis and design functions can only be allocated to one group.

Although the enhanced evaluation procedure as such is independent of socio-technical group structures, in practice, attention must be paid to the co-ordination of groups. This is especially important in case compensating measures will be defined by more groups together.

**Step 3: Determine when to stop an evaluation**

Two reasons can be distinguished for the decision to stop the evaluation process:

a) all functions have been evaluated successfully;

b) the package at hand does not provide a standard function;

ad a) The evaluation process starts at the starting point function, which is normally not a standard function. Each branch of the decomposition tree will be followed top-down until (pseudo-) formal functions have been reached which are suitable for evaluation. The recursive mechanism makes sure that we end at the function where we started.

ad b) As soon as a standard or elementary function has been reached which is unavailable in the package at hand, or the deficiency of which cannot be compensated for, inevitably

---

**Not shown in figure 8.3**
the evaluation process must stop to avoid further evaluation efforts, since at this point we already know for sure the package at hand is unacceptable.

**Step 4: compare the functions provided by the package with own list of functions**

The comparison of package functions and the specified functions is carried out for all functions on each decomposition level. Naturally, a comparison of function properties comes down to answering the question: "Are the specified functions provided by the package at hand?" (for an example see appendix 2). If the answer is a "yes" without reservations, then the next function on the same decomposition level can be evaluated, or, alternatively, the first function on the next higher decomposition level in case all lower level functions have been evaluated.

Expectantly, the answer will not be always an unreserved "yes", but on the other hand, that doesn't always mean a strict "no". Further analysis of the cause of the mismatch can be undertaken to find out if an alternative action can resolve the mismatch (see 7.4.1). If a clear mismatch is found which can not be compensated, the function on the next higher decomposition level can not be carried out as intended by the package at hand and therefore the package must be regarded as incomplete with respect to that particular function. Then, continuing the evaluation any further will be of no use and consequently the package at hand must be rejected for the particular situation at hand.

**Step 5: prepare for package selection and potential modifications**

As a result of the evaluation process a package may be suitable but seldom without compensation measures. In theory any function may be the cause of compensation measures with the result that there are several individual adaptations required to customise the package properly. In case the functional analysis and design cannot provide justifiable new functions meeting the suggested adaptations from the evaluation process, the package at hand still must be rejected.

If, however, the adaptations prove to have only minor consequences for a functional redesign process, the package at hand can be submitted to the final selection process where efficiency criteria are used to make the final selection. As a matter of fact, in the end, if a package has been selected the compensating measures must be realised.
9 Conclusion

The original objective of this PhD research was to determine how standard software packages for the internal maintenance function can be evaluated. Close examination of existing evaluation techniques revealed shortcomings, but most importantly explicit attention to the determination of evaluation criteria with respect to the maintenance function was largely missing. The more fundamental issue of the way in which to determine functional requirements, providing the evaluation criteria, broadened the scope of this research drastically. As a result a functional analysis and design framework has been developed based on a multi-disciplinary organisational function design approach. The framework has been developed for the maintenance function only, because of the original objective of this thesis. In this final chapter we shall perform a first evaluation of the prototype framework in view of its orginal and the revised objective of this study. Because the entire development of the framework had a multi-disciplinary nature we refer to this evaluation as new multi-disciplinary insights. The insights gained in the development of the new framework for design with respect to the original mono-disciplinary knowledge areas maintenance management, sociotechnology and computer science will be discussed as new mono-disciplinary insights.

9.1 Multi-disciplinary insights

9.1.1 The objective of this study

The original objective of this study was to solve the problem of inadequate evaluation methods or their inappropriate application in order to prevent users selecting an unsuitable standard software package. Soon it became clear that simply developing a new evaluation method would not solve the problem. In fact, the true problem lies much deeper, namely in the way in which a maintenance function as an organisational function is designed and realised. The revised objective of this study is aimed at the development of framework for design of the in-house maintenance function. In a sense this revised objective comes down to the development of a holistic maintenance function design framework. It may come as no
On the determination of functional requirements in a maintenance environment

surprise that the resulting prototype framework in its first version still is a "skinny" framework, since some aspects could not be addressed in depth, which nevertheless need scientific attention to solidify the framework. Next, we shall discuss some of these more important aspects.

The technical system design stage

In designing a maintenance function, technology plays an important role in at least two ways. Firstly, technological skills determine the potential effectiveness of maintenance rules (maintenance technology), and secondly, technology as a skill to design and produce technical systems for use (design and production technology). The required level of skills in maintenance technology are implicitly addressed in the socio-technical design stage. The technical system design stage, however, has been left out. Nevertheless, its relevance in our framework is clear. Technical system design can be considered as a organisation design process as well, albeit that in most actual situations this process will not be an in-house activity. Since the functional analysis and design stage depends almost entirely on the decisions made during the design of technical systems, the technical system design stage can be considered as the true first design stage (the "P" aspect in Bemelman's PBI model [7]). Therefore, it would be logical not only to include the current technical design specifications as input to the functional analysis and design process, but also the feedback resulting from all subsequent design processes to the technical design process. In the EUT-maintenance model the feedback loop is defined as the possible result of the maintenance evaluation function, i.e. modification of technical systems still in use, terotechnological feedback and specifications for selection of new technical systems. Further research is recommended to include the technical design stage into the framework.

Vendor information with respect to the functionality of standard software packages

In our enhanced evaluation approach presented in chapter 8 we assumed that detailed lists of available functions are readily available for every standard software package which claims to support the maintenance function. During this research project only a few packages were reviewed in depth. This prohibits definite conclusions on the quality of the vendors and their packages. But none of the packages examined provided in depth functional information. The documentation mainly consisted of a reference of the screens where every field constraint is
Conclusions

explained technically, a keyboard layout and function reference and at best a small and
simplistic tutorial. The result is that serious users are forced to find out in a trial and error
fashion how developers perceived how their package should be used.
The evidence suggests, if vendors take their potential clients seriously, the user
documentation needs substantial improvement on how a user is supposed to work with a
standard package. Further research could reveal whether this problem is generally true and in
which way user documentation can be improved.

The validity of functional requirements
In chapter 5 we discussed the problem of measuring the validity of functions. The lack of an
operational measurement method may restrict the use of the framework in practice. To assess
the value of alternative functions all aimed at satisfying a certain output requirement, a
comparison of the quality of the output with respect to the required output must be carried
out. In practice this first check for effectiveness of a function may involve a wide spectrum
of relevant properties, which all somehow relate to a function's perceived value or
effectiveness. E.g. accuracy of the produced output and input requirements. Simplicity can be
a relevant property which is difficult to measure quantitatively and even more difficult to
combine in an assessment. In order to relate a function's effectiveness to its eventual
efficiency, costs and manhours are most likely the only common denominators. Further basic
research in this area is recommended.

9.1.2 The practicality of the framework

In the previous paragraph we discussed the theoretical implications of the prototype
framework. However, in our industrial engineering design methodology field testing is
needed to assess the practicality of the framework. In this section we shall discuss the
practical aspects as a criterion for the evaluation of the prototype framework.

The functional analysis and design effort
Because our framework for design has not been used in practice yet, very little can be
concluded at this time with respect to the efforts required to carry out a functional analysis
and design process. On the structural analysis and design level, in theory, large parts of the
On the determination of functional requirements in a maintenance environment

decomposition can be structured, as has been demonstrated for maintenance planning and control (chapter 6 and 7). In practice structured decision schemes can be used to determine what functions are needed to support mainly the personnel system analysis and design process. The actual effort required is limited to the checking of rather global situational conditions, provided that the information required to use the prestructured decision schemes presented in chapter 7 is readily available. Further research, also in other functional areas (designing maintenance concepts and spare parts management), is necessary to design more complete prestructured structural analysis and design schemes.

Because of the limited potential of prestructuring and of the increased detail of functions on the detailed functional analysis and design level, the major effort has to be spent by the actual group members whenever a (re)design activity has been triggered. The primary effort is dedicated to the formal investigation of the different procedures and practices and to the choice of the best alternative from a functional point of view, provided that the option can be realised efficiently in the situation at hand (see paragraph 8.1). Here, the pseudo-modular structure of a functional analysis and design and the group knowledge and responsibility can help to spread these efforts. From a socio-technical point of view, within a task, or an operational group, a substantial portion of their selfregulative abilities is allocated to the design of their own procedures and practices. Then, once the procedures have been established, it is only a small step to represent them in the conventions presented in chapter 5.

The allocation of all work, including the functional analysis and design, is essentially a group responsibility. Therefore, a group can decide which group members are best qualified and how formal, i.e. how detailed, the functional specifications should be made to be of maximum use for the group.

As a prerequisite to a successful operation of groups, the personnel system designer should realise that more complex structural functional analysis and design functions require group members with a substantial level of knowledge of the particular area at hand, and that a substantial portion of their workload should be dedicated functional analysis and design.

Organisation-wide application of the functional analysis and design framework

One of the starting points of a functional analysis and design process for the maintenance function is the general framework of organisational design and its specialist sub-design processes. From an organisation wide perspective all organisational functions have their own
design stages. These organisational functions are not operating independently, but are up to some extent interrelated. The justification criterion triggers a functional analysis and design process whenever an evaluation function is triggered, or, if an external information requirement needs to be satisfied. In turn, the external information requirement is a result of a functional analysis and design process of another organisational function. In fact, the justification criterion enforces a functional analysis and design design approach of an organisation wide scope. Depending on how various organisational functions are interrelated, certain changes in information requirements may have a snowball effect by triggering various functional analysis and design processes, purely due to the backward chaining in which justification takes effect. Some former functions still can be justifiable, whilst others must be redesigned to fulfil the new information requirement, and so on. Minor design changes in one organisational function may appear small, but the resulting external information requirements may cause substantial redesign efforts for other organisational functions. E.g. in an organisation the personnel management function requires a detailed registration of the absenteeism of all personnel to compute the wages periodically. Such a formal registration of absenteeism does make sense for large maintenance groups where formal capacity planning is required to enable some optimisation with respect to the workload. In small groups the handling of capacity in a strict formal manner may be considered unnecessary. But nevertheless, the functional requirement to compute wages from the personnel administration function has be included to the maintenance group responsibility.

As a consequence, there is a limited benefit of a introduction of our design framework for the maintenance function only. To prevent the loss of important inter-organisational function links and to make a true sociotechnical approach possible, an organisation must decide to apply the framework for all organisational functions.

To prevent redesign of functions individually out of the situational context, a cost-benefit analysis is needed. Especially, if organisational functions are allocated to different groups at least some co-ordination of the redesign efforts which are required to assess the potential benefits of redesign proposals for the organisation as a whole is required.

This PhD project concentrated on the design framework for the maintenance function only. Further research in the application of co-ordination and the "cost-benefit" assessment of functional analysis and design processes on an organisation-wide level is essential to assess
the potential of the approach in real situations passing the borders of the maintenance function.

9.2 Mono-disciplinary insights

9.2.1 Maintenance management

Elementary maintenance planning and control situations
For the development of the functional analysis and design framework with respect to the maintenance management function, available maintenance management theory is essential. However, this aspect has until now, despite the large quantity of publications been under-developed in maintenance theory (see chapter 6). A categorisation of elementary maintenance management planning and control situations was not available. A first attempt has been made to develop such a categorisation specifically for this framework. In essence, this categorisation is based on the analogy to elementary production planning and control situations. However, currently the planning and control functions are identified only on an structural functional analysis and design level. Further research which could assist detailed functional designers in the operationalisation of elementary planning and control situations is recommended.

The maintenance evaluation function
Another result of the development of the framework is the interpretation of the evaluation function. An evaluation function monitors the performance of a (lower level) function; if necessary a correction is applied to keep performance on the desired level. Traditionally, evaluation functions are designed independently from the lower level functions which they are supposed to monitor. This approach inherently introduces the danger of accidental omission of functions, especially in cases where evaluation functions are designed long after operational lower level functions have been designed and realised. The decomposition principle implies the creation of an evaluation function whenever a new decomposition level is created. In other words, the evaluation function is created during functional analysis and design.
Conclusions

9.2.2 Sociotechnology

Functional units are operated by a single, possibly segmented, category of planning and control. Further decomposition of functions within a functional unit provide the necessary evaluation functions which create complete control feedback loops within a functional unit. In this manner, the decomposition principle makes the concept of allocating complete functional units to groups usable as a potential cohesion factor to establish task groups. The idea of this concept is that closed feedback loops within groups can be handled more easily, up to some extent informally, within a group instead of between groups, where more formalisation of co-ordination is almost a prerequisite. Traditionally, if planning and control is used as a cohesion factor, relevant personnel have to be interviewed in order to produce a list of -sometimes uncorrelated- maintenance activities, usually stated in various levels of detail (see case description of Van Amelsvoort [3]). Using an uncorrelated list of activities makes the creation of groups difficult, because it may be necessary to experiment for a longer period of time with different task group settings before a satisfactory grouping is established.

The key benefit of establishing task groups by using functional units as a grouping criterion is that no uncertain list of (low level) activities is required. As soon as the final group structure has been formed, the set of functional units the group is held responsible for can be decomposed further without risking an increase in inter-group co-ordination. The uncertainty in the formation of groups can be reduced further in case the volume of the capacity that is required to carry out the physical activities can be estimated. In the case description (chapter 7) we demonstrated how the maintenance concept in combination with the maintenance history can be used to estimate the maintenance workload for different functional unit configurations.

In other words, it can be expected that valid group structures can be established faster by using the results of an structural functional analysis and design process. More case studies, in which the framework will be fully used (not only for demonstration purposes), are needed to assess the potential for faster development.
9.2.3 Computer science

Information analysis
The purpose of the framework, i.e. to provide a user driven way to determine functional requirements in an individual maintenance situation, could be interpreted as an information analysis approach. The key difference with the traditional information analysis approaches is that it is mostly driven by users, because users are best qualified as experts in the function concerned and because they are responsible for the performance of the functions they design. In chapter 4 we already discussed extensively the expected advantages of a user driven approach, which became a design specification for our framework. It is even expected that users can perform an evaluation of packages for the maintenance function themselves, provided that the packages documentation is function oriented, i.e. in terms of procedures and practices instead of being a button and screen assignment reference. Another perceived benefit is that in-house development of maintenance function related information systems can be speeded up considerably, with lower cost, because of the reduced need of user-developer communication and of the reduced risk of misunderstandings. If the framework can live up to its virtues in practice, then information analysis can not be regarded as the primary occupation of the information analyst profession, but it will be a structural part of the users functional responsibilities. Long term experimentation with the framework in practice is recommended.

The future of standard packages aimed at the maintenance function
In organisations operating in open market conditions, the ability to offer products for which there is sufficient demand is clearly the primary objective. To stay ahead of competition it is vital to make these products as efficient as possible. It goes without saying that all organisational functions, including the maintenance function, are directly or indirectly aimed towards that objective. Of course, the way organisational functions are designed and realised is a major contributing factor in staying ahead of the competition. Therefore, standardised functions which are made available by standard packages, which are available to the competition as well, cannot provide a truly competitive advantage. In addition, the uniqueness of an organisation and its group structures necessitate an organisation specific approach in the first place. Competition is a strong motivational factor in stimulating the improvement of current practices. At best, imitating a competitor only narrows the gap with
Conclusions

the market leader. Successful organisations are not only unique because of the potentially large options which are available in all design stages, but also it is their interest to do things differently, i.e. better than the competition.

The organisational design approach presented in chapter 3 demonstrates how functional analysis and design relates to the personnel and the information system analysis and design (=ISAD). If applied, the approach will lead to relatively unique requirements for the ISAD process. In fact, because the ISAD process is situated at the end of all design processes, and because each design process adds situation specific elements, its design specifications (the detailed functional analysis and design specifications and efficiency requirements) also will be unique.

Software packages aim at providing "ready-made" a program code for this essentially unique design process. The strategy followed by some suppliers of maintenance packages is to maximise the number of functions in their packages to create flexibility by allowing the user to make a choice out of a diversity of itemised alternatives. From a user point of view, this flexibility may be constrained and confusing due to the number of possible options offered. In general, the need to cope with different requirements from clients in a dynamic environment puts suppliers of maintenance function specific packages in a difficult position, since each individual adaptation of their packages increases costs and, implicitly, leads to a new -company specific- version, which potentially jeopardises the suppliers interest in uniformity.

Clearly, suppliers must come up with a solution if they want to stay in the business. As the packages market for maintenance matures there will be little potential for traditional "fixed" function packages. Since detailed functional user requirements increase and cannot be denied, suppliers will have to shift their efforts to supply more customer specific services efficiently. Developers of information systems can follow different ways in satisfying functional requirements such as:

a) Modular design.
b) Open design.
c) Efficient development.

ad a) Modularity in packages design means that encoded modules can be replaced by alternative encoded modules without adaptation of the modules which have been
already installed. Modular packages design does not necessarily imply that a function corresponds exactly to a packages module. From a theoretical point of view it is unclear yet how modularisation can be achieved. E.g. the database and the user interface can be regarded as different modular entities requiring a different type of development expertise, from a developers point of view. On the other hand, from user perspective individual functions will be regarded as a separate modular entities.

ad b) The next step in modularisation would be the design of "open" software packages. Open packages provide the ability to extend effectiveness of a package by adding enhancements from different vendors and, consequently, increasing the number of options for potential customers on the one hand, whilst on the other hand increasing the potential for vendors of packages who specialise in those functional areas in which they are most experienced. To facilitate open connections at least standardisation on a technical level (hardware and software interfaces) must be established. Suppliers of "toolbox" functions, such as hazard plotting techniques, will probably benefit most from open packages, because these narrowly defined non-dynamic functions are widely applicable irrespective of the situation or circumstance. However, the success of this approach will depend largely on the possibility to standardise organisational functions. In maintenance, it is perceived that with the exception of rather generalistic "toolbox" functions, such as algorithmic functions assisting in failure analysis, the potential for standardisation is limited.

ad c) The ultimate step in satisfying the clients functional requirements is to encode the exact individual functional requirements without compromises quickly. This requires the involvement of experienced developers and the use of higher level programming languages and programming environments if this strategy is to lead to success. Traditionally, firms offering in-house development projects have already chosen this path.

With respect to software packages for the maintenance function the strategies described above are already in use. Especially more recent packages provide means to realise options a) and b). Given these approaches suppliers can follow, it is questionable whether the term "standard software package" is still appropriate.
References


On the determination of functional requirements in a maintenance environment


References

[23] "COMGE-questionnaire on behalf of an investigation of computer programs" (in dutch), COMGE, july 1981, 17p.


[25] Danish Hospital Institute, "System for registration and analysis of technical services", report no. 79.04, 168p (Dutch version)


[27] DIN V 66285, Vornorm (preliminary norm), "Application software testing principles" (in german), Normenausschuß Informationsverarbeitung (NI) im DIN Deutsches Institut für Normung e. V., 10p


[29] Er, M.C., "Classic tools of systems analysis - why they have failed", Data processing vol 28 no 10, Butterworth & Co, december 1986


[31] "Evaluating off-the shelf programs", Datamation, december 1980


On the determination of functional requirements in a maintenance environment


References


[45] Grothus, H., "Instandhaltekosten budgetieren" (Eng: budgetting maintenance costs), Grothus Verlag, Dorsten, 1973, 80p


[50] International Iron and Steel Institute, "Maintenance for the 1990's", Committee on Technology, Brussels, June 1989


[54] Küpper, W., "Planung der Instandhaltung" (Eng: Maintenance planning), Betriebswirtschaftlicher Verlag Dr Th. Gabler, Wiesbaden, 1974, 444p

[55] Leeuw, A.C.J. de, "Bedrijfskundige methodologie" (Eng: Industrial engineering methodology), Van Gorcum, Assen/Maastricht, 1990, 180p

On the determination of functional requirements in a maintenance environment


References


[74] Stahlknecht, P. and Nordhauß, R., "Case study in the methodology of hardware and software selection in small and medium sized firms" (in german), Oldenbourg Verlag, München, 1981, 142p (in german)


[80] Zomeren, L. van, "Voorraadreductie door centralisatie van voorraadbeheersing en voorraadlokatie" (Eng: Reduction of stock levels by centralisation of spare parts management and locations), Msc-thesis, Faculty of Industrial Engineering and Management Science, Eindhoven University of Technology, March 1994.

[81] Zwaan, A.H. van der, "Organisatie onderzoek" (Eng: Organisations as a subject for research), Van Gorcum, Assen/Maastricht, 1990, 165p
On the determination of functional requirements in a maintenance environment

[82] Zwaan, A.H. van der & Engelen, J.M.L. van, "Bedrijfskundige methodologie 1" (Eng: Industrial engineering methodology; part 1), Bedrijfskunde special "Methodologie en theorie", jaargang 66, 1994, no 1

APPENDIX I

Workload distribution per main technical system used at the 's Hertogenbosch main sorting centre.
<table>
<thead>
<tr>
<th>FUNC. NAME</th>
<th>TRIGGERING</th>
<th>INPUT DATA</th>
<th>OUTPUT INFORMATION</th>
<th>TRANSFORMATION</th>
<th>ACCEPT (Y/N)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>spec. package</td>
<td>spec. package</td>
<td>spec. package</td>
<td>spec. package</td>
<td></td>
</tr>
<tr>
<td>determin. of failure consequ.</td>
<td>whenever a new failure has been discovered</td>
<td>as soon as failure input data is complete</td>
<td>idem</td>
<td>total consequ. of a new failure</td>
<td>idem (based on historical costs)</td>
</tr>
<tr>
<td>direct consequ. for maint. (in hfi)</td>
<td>idem</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>external consequ. (in hfi)</td>
<td>** unknown</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># failures per period</td>
<td>idem</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX III

The EUT maintenance model is a generalised descriptive model of the internal maintenance function. Primarily, it serves as a tool to distinguish knowledge elements in maintenance and to identify into which maintenance function they fit. In addition, it assists in assessing potential improvements in actual maintenance situations.

The EUT maintenance model is a representation of 14 functional areas (main functions and relationships) within a maintenance function:

- The technical systems to be maintained
- The internal capacity
- The external capacity offered in the market by a diversity of contractors
- The external capacity offered specifically by the OEM's
- The maintenance planning and control
- The inventory control of non-repairable maintenance parts (consumables)
- The maintenance planning and control of rotables
- The evaluation of results
- The terotechnological feedback
- The methodology of design of a technical system
- The specification of requirements for a technical systems
- The design of a technical system
- The manufacture of a technical system
- The design of the maintenance concept for a technical system

It should be noted that the EUT-maintenance model is based on a multi-disciplinary, industrial engineering point of view. The diversity in technological disciplines required to carry out maintenance operations, methodologies for technical design and manufacturing of technical systems have been recognised, but not covered in the model. The disposal of technical systems, although of great importance in some situations, has not been taken up in the model, because it doesn't belong to the core of maintenance theory but to investment theory.

The EUT-maintenance model has been described extensively by Geraerds [37].
On the determination of functional requirements in a maintenance environment
Summary

Due to the decline of prices of computer hardware in combination with the increase in performance the number of potential application areas has increased as well. Nowadays the computer is regarded as a valuable and sometimes even as an indispensable tool to manage organisational functions, provided that suitable software is available. To obtain suitable software, basically, two strategies can be followed, which are actually two extremes on a more or less continuous scale of options. The traditional strategy is to develop software in-house. In this way, theoretically, all functional requirements can be met. The other option is to turn to so called standard software packages offered by software houses and consulting firms. To assess the suitability of a software package an evaluation process has to be carried out.

In this thesis the way in which to evaluate standard software packages offered for the internal maintenance function of organisations is dealt with. The preliminary research in this area resulted in the conclusion that most evaluation techniques are flawed. However, the most serious drawback in current evaluation practices is caused by the lack of attention paid to the process of how to achieve complete, valid, justified evaluation criteria. Essentially, an evaluation aims at the assessment of the suitability of software for an individual situation. The formal procedures, which are preferably carried out by computer, are the basis for evaluation criteria. To determine valid procedures for the maintenance function, a series of design decisions are to be made originating from different scientific disciplines, such as a wide variety of technical disciplines, maintenance management, sociotechnology and computer science. In fact, to arrive at valid procedures for evaluation purposes the entire maintenance function has to be designed taking into account these scientific disciplines. Decisions with respect to technical design are outside the scope of this thesis.

In this research project a framework for design of operational maintenance functions has been developed. The cornerstone of the framework is the principle of decomposition, originating from systems theory. In the process of decomposition the maintenance function is split up into smaller more detailed subfunctions interconnected via data streams. Each decomposition step requires a decomposition criterion, which sets the constraints for decomposition. The available level of knowledge in together with situational factors determine these decomposition criteria.
This stage of decomposition is stopped as soon as personnel settings become part of the decomposition criterion. Next, the personnel system has to be designed by using sociotechnological design principle to form semiautonomous task groups. The maintenance functions are assigned to these task groups. Then, the task groups themselves are responsible to decompose the maintenance functions they responsible for until, in their view, the functions are unambiguously defined.

An evaluation is carried out by comparing functions resulting from the decomposition process with the functionality of a software package at hand. For each mismatch it is first decided whether the functions provided by the software package are satisfying the specified requirements. If this is not the case the possibility to modify the package at hand is assessed. The evaluation process then can result in a list of required modifications, which have to be assessed based on efficiency criteria. In the end, provided that potentially effective packages exist, the one with the lest costly modifications is preferred.

In order to illustrate the framework for design, parts of the framework have been applied at a hub maintenance unit of the Dutch PTT Post organisation.
Samenvatting

Mede als gevolg van de dalende prijzen van computerapparatuur in combinatie met het stijgende prestatieniveau is het aantal potentiële toepassingsgebieden van dit hulpmiddel sterk toegenomen. Tegenwoordig wordt de computer als een volwaardig en vaak ook een onmisbaar hulpmiddel gezien in de bedrijfsvoering, mits voor de beoogde toepassing geschikte programmatuur beschikbaar is. Om geschikte programmatuur te verkrijgen kan men als uitersten op een vrijwel continue schaal kiezen tussen het in eigen beheer specifiek voor het beoogde doel ontwikkelen van programmatuur, of men onderzoeken of een z.g.n. standaardsoftwarepakket dat vrij in de markt beschikbaar is voor het beoogde doel geschikt is.

In dit proefschrift wordt het vraagstuk onderzocht op welke wijze men programmatuur beoordeeld op zijn bruikbaarheid t.b.v. onderhoudsfunctie in een organisatie als toepassingsgebied. Na vooronderzoek is gebleken dat de meeste evaluatietechnieken een aantal tekortkomingen hebben. Echter, het voornaamste probleem vindt zijn oorsprong in de criteria die voor een evaluatie gebruikt worden. De geschiktheid van programmatuur in een individuele situatie wordt in essentie afgemeten aan het aantal en de aard van geformaliseerde functies die door de programmatuur uitgevoerd kunnen worden.

Evaluatiecriteria werden aldus gebaseerd op de gewenste werkwijze in een individuele situatie. De gewenste werkwijze in een onderhoudsituatie, wederom, is een functie van de ontwerpbeslissingen t.a.v. de technische systemen die onderhouden dienen te worden, de procedures die gehanteerd dienen te worden, het personeel dat verantwoordelijk is voor de onderhoudsfunctie en de informatie-infrastructuur. In dit onderzoek ligt het zwaartepunt bij het ontwikkelen van een methodiek van het ontwerpen van een operationele onderhoudsfunctie, opdat hieruit valide, complete en correcte evaluatiecriteria af te leiden zijn.

De ontwikkelde methodiek is gebaseerd op een reeks ontwerpstappen uit het materiegebied van de onderhoudsbeheersing, de sociotechniek en de bestuurlijke informatica. Via het principe van functionele decompositie wordt de onderhoudsfunctie opgesplitst in kleinere meer gedetailleerde subfuncties die via datastromen met elkaar verbonden zijn. De decompositiecriteria worden per decompositions slag bepaald en zijn afhankelijk van het kennisniveau in het onderhoud en situationele factoren. De functionele decompositie wordt
On the determination of functional requirements in a maintenance environment

onderbroken op het moment dat personele aspecten bij een decompositiecriterium betrokken moeten worden. Deze zogenaamde structuurfuncties dienen als uitgangspunt voor het sociotechnisch (her)ontwerp, waarbij de functies aan groepen worden toegewezen. Vervolgens wordt per groep de functionele decompositie voortgezet, totdat naar het oordeel van de groepsleden voldoende formalisatie is bereikt bij alle groepsleden een eenduidige opvatting bestaat over te volgen gewenste werkwijze. De langs deze weg bepaalde functies kunnen bij een evaluatie vergeleken worden met de door een programma geleverde functies. Bij iedere afwijking wordt allereerst bepaald of de door de programmatuur geleverde functie effectiever is, zo niet kan overwogen worden of via aanpassingen in de programmatuur de gewenste effectiviteit wel bereikt kan worden. Aldus ontstaat een lijst met noodzakelijke functionele aanpassingen die op grond van efficiency overwegingen al dan niet acceptabel zijn. Ter illustratie van de ontwerpmethode zijn delen van de ontwerpstappen toegepast bij de PTT Bedrijfsservice.
Curriculum vitae

Harry Martin (born in 1961) started studying Chemical Engineering at Twente University of Technology. In 1980 switched to the Eindhoven University of Technology to study Industrial Engineering and Management Science. He received his M.Sc. in 1986 for his research in information systems for maintenance management. Until March 1987 he participated in a joined project of the EUT and the Philips company with the objective to introduce the principles of designing maintenance concepts at the Philips television tube factory in Aachen (Germany). He then joined the "Directoraat Material Koninklijke Landmacht" (the main military supply organisation of the Dutch Army) where was active in the in-house development of several information systems. After his military service he rejoined the faculty of Industrial Engineering and Management Science as an assistant professor. He professional interest is aimed at the development, application and evaluation of information systems for maintenance. He is Honorary General Secretary and Treasurer of IFRIM.
On the determination of functional requirements in a maintenance environment

van

Harry H. Martin
I
Eén standaardpakket dat voor alle onderhoudssituaties geschikt is, is onmogelijk.
(Dit proefschrift)

II
Het is niet mogelijk een standaardsoftwarepakket voor de onderhoudsfunctie te beoordelen indien de beoordeling niet wordt gebaseerd op criteria ontleend aan wel gedefinieerde onderhoudssituaties.
(Dit proefschrift)

III
Ofschoon het wenselijk is om de beslissing tot aanschaf van standaardsoftware pakketten te baseren op een vergelijking van kosten en baten, is een directe vergelijking alleen bij uitzondering mogelijk.

IV
Het enige voordeel dat standaardsoftwarepakketten voor de onderhoudsfunctie hebben ligt in het feit dat men niet zelf of in eigen regie behoeft te coderen.

V
Het alom gehanteerde verkoopargument dat standaardsoftware voor complexe organisatiefuncties goedkoper dan maatwerk is, heeft uitsluitend betrekking op de verkoopprijs van het initiële pakket.

VI
De introductie van informatiesystemen t.b.v. de onderhoudsfunctie is geen op zich zelfstaande éénmalige activiteit.
VII
Een meer geschikte term voor de in de bestuurlijke informatica populaire term "informatie analysemethode" is "data formalisatiemethode".

VIII
Het in de informatica gehanteerde begrip "onderhoud van informatiesystemen" is misleidend, aangezien er geen sprake is van het opheffen van storingen die optreden in de programmatuur als functie van het gebruik.

IX
Het streven naar vergaande formalisatie in organisaties vindt zijn oorzaak voornamelijk in het onderschatten van het "human recovery" vermogen.

X
De effectiviteit van operations research zou aanmerkelijk worden verhoogd door allereerst research van operations.

XI
Getuige het opkomende nationalisme tijdens grote internationale sportevenementen is het twijfelachtig of het gezegde "sport verbroedert" nog wel algemeen geldig is en of in vele gevallen niet eerder sprake is van "sport verloedert".

XII
Gezien de wijze waarop de overheid budgetten bepaalt, doen bestuurders van overheidsinstanties er goed aan te overwegen om mogelijkheden tot efficiency verbetering niet bekend te maken en niet direct te realiseren, maar ze in reserve te houden voor een volgende arbitraire budgetverlaging.