Requirements Analysis: Problems and the STARTS Approach

John McDermid, University of York

1. Introduction

In recognition of the importance of good software engineering technology to the prosperity of UK industry the DTI funded the STARTS programme, which was then run and managed by the National Computing Centre, or NCC. The primary deliverables from the STARTS programme have been two documents providing comprehensive guidance on software engineering. The STARTS Guide primarily gives assessments of methods and tools aimed at solving particular aspects of the software engineering problem. The STARTS Handbook [STARTS1989] gives a rather more general, management and procurement oriented, view of the problems of software development. However there are a number of areas where the handbook goes into technical detail. One of these is on general guidelines for requirements specification as it is recognised that invalid or inappropriate requirements are often one of the major sources of technical, managerial and procurement difficulties on a software development project.

The aim of the handbook is to provide guidance which is appropriate, independent of the method used for requirements analysis, for eliciting requirements for a large scale system. In particular the handbook gives guidelines for producing and validating specifications, and a set of checklists for assessing the completeness of the coverage of a requirements specification. The aim of this paper is to give an overview of the problems of requirements specification, and to outline the guidance given by the STARTS Handbook in terms of the checklists it presents for requirements. In this way it is hoped to clarify what are the fundamental difficulties in producing requirements, and to summarise what is currently accepted by industry experts as being the fundamental contents of a requirements specification.

2. What is a Requirement Specification?

There is often considerable confusion over the distinction between a design, a set of requirements, and a specification. A common distinction is to say that requirements say **what** the system should do, and design says **how** the system should do it. In practice this distinction is rather artificial, and most requirements contain a mixture of "what" and "how" information. Whilst this may partly be a reflection of errors in analysis, it often also reflects the fact that the customers or users for the system can only specify

something they genuinely require by constraining the way the system is designed.

Perhaps the best definition of a set of requirements is due to Parnas:

> A requirements specification is everything the developer needs to know to produce a system which is acceptable to the customer and users, and nothing more.

This effectively says that a requirement specification contains everything that the users regard as essential, but it contains minimal design information.

3. The Problems of Requirements Specification

There are many difficulties in producing requirements specifications. Most of these stem from the fact that this is the first technical activity in the software development process, and that the production of requirements usually involves the interaction of people with different backgrounds. For example it is quite common for requirements to be written by computer specialists who have no knowledge of the application domain, and for the potential system users to be experts in their application but not to understand computers.

A primary activity in requirements specification is the elicitation of requirements, i.e. extracting information about the requirements from the customer or users. There are quite a large number of problems of elicitation, including those identified below:

- users' incomplete understanding of needs
- conflicting views of different users
- users poor understanding of computer capabilities and limitations
- analysts poor knowledge of problem domain
- it is easy to omit "obvious" information
- user and analyst speak different languages
- requirements evolve over time
- the boundary of the system is ill-defined
- unnecessary design information may be given (psychological trait)
- requirements often vague and untestable, e.g. "user friendly", "robust".

Having elicited a set of requirements it is also necessary to try to validate them, i.e. to gain confidence that the requirements as written down do state clearly and unambiguously what the customer and user wishes. Thus the problems of validation are essentially the inverse of those of elicitation, except that one must also be concerned about completeness of the requirements.

4. The STARTS Requirements Check List

The check list proposed by STARTS falls into two parts. The first level essentially reflects what might be thought of as a cardinal points specification, i.e. a specification which sets out the most critical aspects of a system, but doesn't go into technical details about functionality or performance. This is often a very important part of requirements, as it will identify the fundamental objectives of the system. In a military example the objectives might be set out in terms of some operational scenario, and in a more commercial context they may reflect a position in the marketplace or a financial return to the company. It is fairly rare for these objectives to be identified but it is extremely important that they should be - in many cases this represents the "real requirement" that the system has to satisfy.

The level two specification gives the technical detail against which the system can be developed and acceptance tested. It is always possible to represent the intended functionality of computer systems in two essentially complementary ways. One way involves centering the requirements on data, and then showing how the data will be transformed. The other involves ensuring the functions as the central aspect of the system, and showing how the functions operate on the data in the system. STARTS has adopted the function centred approach.

The STARTS check list identifies a number of facets or attributes of functions which need to be specified. These include the following key points:

- normal operation - this reflects the standard processing carried out by the function and indicates the inputs and outputs used by the function;
- abnormal operation - this deals with potential failure modes, error reports associated with failures and recovery techniques;
- dependability - this covers properties associated with safety, security, reliability, etc.;
- assumptions - this indicates any aspects of the requirements, application or other facets of the system that can be assumed not to change in designing and implementing the function;
- quality - this covers general quality attributes to do with usability, maintainability, etc.;
- performance - this covers important aspects of space and time behaviour including response to stimuli, throughput, period of operation for periodic functions, etc.;
- interface protocols - this explains the ways in which the function interacts with the user, and potentially with other hardware systems and subsystems;
- expected changes - this details ways in which the operation of the function is expected to have to change over the life time of the system.

The above check list covers most of the key aspects of the system that need to be specified.

5. Specification of Critical Systems

The intention in defining the STARTS check list is that it should cover the attributes that are required for critical systems, as well as for those used in less stringent applications. Nonetheless there would be some differences between requirements produced for critical applications, and those used in other circumstances. However the primary difference would be that a number of additional techniques, e.g. failure modes effect analysis, would be used in arriving at the requirements. Also we would expect that the requirements would be driven further into the design domain by the need to specify certain aspects of safety critical behaviour, e.g. the ability to operate correctly even in the presence of faults. Thus we would expect a requirement specification for a safety critical system to say more about the system design and architecture than we would a requirement for a system to be used in a less stringent application.

6. Available Methods

There are a number of methods available for dealing with requirements analysis, although there are far less methods oriented towards requirements than there are to design. None of the available methods deals adequately with all the different facets of a requirement specification. However there are techniques from a number of different requirements analysis methods which are widely useful and applicable. The presentation will conclude by summarising the strengths of some of the better know requirements techniques.

7. References

STARTS1989. STARTS, *STARTS Purchasers' Handbook, Second Edition*, DTI (1989).