ESA PSS-05-11 Issue 1
July 1993

Guide to software quality assurance

Prepared by:
ESA Board for Software
Standardisation and Control
(BSSC)

Approved by:
The Inspector General, ESA

european space agency / agence spatiale européenne
8-10, rue Mario-Nikis, 75738 PARIS CEDEX, France

DOCUMENT STATUS SHEET

DOCUMENT STATUS SHEET			
1. DOCUMENT TITLE: **ESA PSS-05-11 Guide to Software Quality Assurance**			
2. ISSUE	3. REVISION	4. DATE	5. REASON FOR CHANGE
1	0	1993	First issue

Approved, 1st July 1993
Board for Software Standardisation and Control
C. Mazza, Chairman

Approved, July 13th 1993
Telematics Supervisory Board

Approved by:
The Inspector General, ESA

Published by ESA Publications Division,
ESTEC, Noordwijk, The Netherlands.
Printed in the Netherlands.
ESA price code: E1
ISSN 0379 - 4059

TABLE OF CONTENTS

DOCUMENT STATUS SHEET ii

CHAPTER 1 INTRODUCTION 1
1.1 PURPOSE 1
1.2 OVERVIEW 1

CHAPTER 2 SOFTWARE QUALITY ASSURANCE 3
2.1 INTRODUCTION 3
2.2 USER REQUIREMENTS DEFINITION PHASE 6
2.2.1 Checking technical activities 7
2.2.2 Checking the management plans 8
2.2.2.1 Software project management plan 8
2.2.2.2 Software configuration management plan 10
2.2.2.3 Software verification and validation plan 11
2.3 SOFTWARE REQUIREMENTS DEFINITION PHASE 12
2.3.1 Checking technical activities 12
2.3.2 Checking the management plans 14
2.3.2.1 Software project management plan 14
2.3.2.2 Software configuration management plan 15
2.3.2.3 Software verification and validation plan 15
2.4 ARCHITECTURAL DESIGN PHASE 15
2.4.1 Checking technical activities 16
2.4.2 Checking the management plans 17
2.4.2.1 Software project management plan 18
2.4.2.2 Software configuration management plan 18
2.4.2.3 Software verification and validation plan 21
2.5 DETAILED DESIGN AND PRODUCTION PHASE 21
2.5.1 Checking technical activities 21
2.5.1.1 Detailed design 21
2.5.1.2 Coding 22
2.5.1.3 Reuse 22
2.5.1.4 Testing 23
2.5.1.5 Formal review 25
2.5.2 Checking the management plans 26
2.5.2.1 Software project management plan 27
2.5.2.2 Software configuration management plan 27
2.5.2.3 Software verification and validation plan 28
2.6 TRANSFER PHASE 28
2.7 OPERATIONS AND MAINTENANCE PHASE 30

CHAPTER 3 THE SOFTWARE QUALITY ASSURANCE PLAN 31
3.1 INTRODUCTION 31
3.2 STYLE 31
3.3 RESPONSIBILITY 31
3.4 MEDIUM 32
3.5 CONTENT 32
3.5.1 SQAP/1 Purpose 35
3.5.2 SQAP/2 Reference Documents 35
3.5.3 SQAP/3 Management 35
3.5.3.1 SQAP/3.1 Organisation 35
3.5.3.2 SQAP/3.2 Tasks 36

3.5.3.3 SQAP/3.3 Responsibilities 36
3.5.4 SQAP/4 Documentation 36
3.5.5 SQAP/5 Standards, practices, conventions and metrics 36
3.5.5.1 SQAP/5.1 Documentation standards 36
3.5.5.2 SQAP/5.2 Design standards 36
3.5.5.3 SQAP/5.3 Coding standards 36
3.5.5.4 SQAP/5.4 Commentary standards 37
3.5.5.5 SQAP/5.5 Testing standards and practices 37
3.5.5.6 SQAP/5.6 Selected software quality assurance metrics 37
3.5.5.7 SQAP/5.7 Statement of how compliance is to be monitored.. 37
3.5.6 SQAP/6 Review and audits 37
3.5.7 SQAP/7 Test 37
3.5.8 SQAP/8 Problem reporting and corrective action 37
3.5.9 SQAP/9 Tools, techniques and methods 38
3.5.10 SQAP/10 Code (and document) control 38
3.5.11 SQAP/11 Media control 38
3.5.12 SQAP/12 Supplier control 38
3.5.13 SQAP/13 Records collection, maintenance and retention 39
3.5.14 SQAP/14 Training 39
3.5.15 SQAP/15 Risk Management 39
3.5.16 SQAP/16 Outline of the rest of the project 39
3.5.17 SQAP/APPENDIX A Glossary 39
3.6 EVOLUTION 39
3.6.1 UR phase 39
3.6.2 SR phase 40
3.6.3 AD phase 40
3.6.4 DD phase 40

APPENDIX A GLOSSARY A-1
APPENDIX B REFERENCES B-1
APPENDIX C MANDATORY PRACTICES C-1
APPENDIX D INDEX D-1

PREFACE

This document is one of a series of guides to software engineering produced by the Board for Software Standardisation and Control (BSSC), of the European Space Agency. The document tree is described in ESA PSS-05-01, Guide to the Software Engineering Standards.

The guides contain advisory material for software developers conforming to ESA's Software Engineering Standards, ESA PSS-05-0. They have been compiled from discussions with software engineers, research of the software engineering literature, and experience gained from the application of the Software Engineering Standards in projects.

The following past and present BSSC members have contributed to the production of the software engineering guides: Carlo Mazza (chairman), Bryan Melton, Daniel de Pablo, Adriaan Scheffer, Richard Stevens and Michael Jones.

The BSSC wishes to thank Jon Fairclough for his assistance in the development of the Standards and Guides, and to all those software engineers in ESA and Industry who have made contributions.

Requests for clarifications, change proposals or any other comment concerning this guide should be addressed to:

BSSC/ESOC Secretariat
Attention Mr C Mazza
ESOC
Robert Bosch Strasse 5
D-64293 Darmstadt
Germany

BSSC/ESTEC Secretariat
Attention of Mr A Scheffer
ESTEC
Postbus 299
NL-2200 AG Noordwijk
The Netherlands

This page is intentionally left blank.

CHAPTER 1 INTRODUCTION

1.1 PURPOSE

ESA PSS-05-0 describes the software engineering standards that apply to all deliverable software implemented for the European Space Agency (ESA), either in house or by industry [Ref 1].

ESA PSS-05-0 requires that all software projects assure that products and procedures conform to standards and plans. This is called 'Software Quality Assurance' (SQA). Projects must define their software quality assurance activities in a Software Quality Assurance Plan (SQAP)[1].

This guide defines and explains what software quality assurance is, provides guidelines on how to do it, and defines in detail what a Software Quality Assurance Plan should contain.

Everyone who is concerned with software quality should read this guide, i.e. software project managers, software engineers and software quality assurance personnel.

1.2 OVERVIEW

Chapter 2 contains a general discussion of the principles of Software Quality Assurance, expanding upon ESA PSS-05-0. Chapter 3 describes how to write the SQAP, in particular how to fill out the document template.

All the mandatory practices in ESA PSS-05-0 that apply to software quality assurance are repeated in this document. The identifier of the practice in parentheses marks a repetition. Practices that apply to other chapters of ESA PSS-05-0 are referenced in the same way. This document contains no new mandatory practices.

[1] Where ESA PSS-01 series documents are applicable, ESA PSS-01-21 'Software Product Assurance Requirements for ESA Space Systems' also applies [Ref 9].

This page is intentionally left blank.

CHAPTER 2
SOFTWARE QUALITY ASSURANCE

2.1 INTRODUCTION

ESA PSS-05-0 defines Software Quality Assurance (SQA) as a 'planned and systematic pattern of all actions necessary to provide adequate confidence that the item or product conforms to established technical requirements'. SQA does this by checking that:

- plans are defined according to standards;
- procedures are performed according to plans;
- products are implemented according to standards.

A procedure defines how an activity will be conducted. Procedures are defined in plans, such as a Software Configuration Management Plan. A product is a deliverable to a customer. Software products are code, user manuals and technical documents, such as an Architectural Design Document. Examples of product standards are design and coding standards, and the standard document templates in ESA PSS-05-0.

SQA is not the only checking activity in a project. Whereas SQA checks procedures against plans and output products against standards, Software Verification and Validation (SVV) checks output products against input products. Figure 2.1A illustrates the difference.

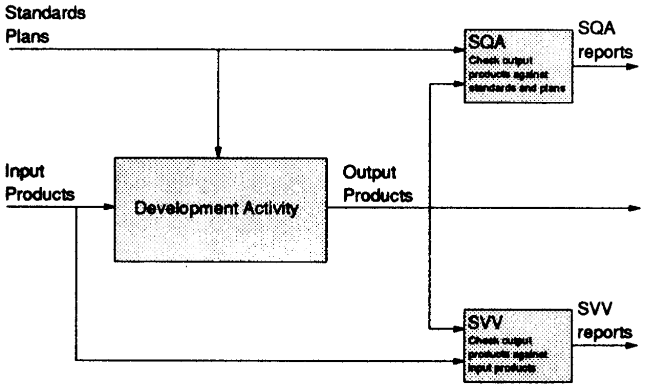
Standards
Plans
SQA
Check output products against standards and plans
SQA reports
Input Products
Development Activity
Output Products
SVV
Check output products against input products
SVV reports

Figure 2.1A: Differences between SQA and SVV

Complete checking is never possible, and the amount required depends upon the type of project. All projects must do some checks. Projects developing safety-critical software should be checked more rigorously.

SQA must plan what checks to do early in the project. The most important selection criterion for software quality assurance planning is risk. Common risk areas in software development are novelty, complexity, staff capability, staff experience, manual procedures and organisational maturity.

SQA staff should concentrate on those items that have a strong influence on product quality. They should check as early as possible that the:

- project is properly organised, with an appropriate life cycle;
- development team members have defined tasks and responsibilities;
- documentation plans are implemented;
- documentation contains what it should contain;
- documentation and coding standards are followed;
- standards, practices and conventions are adhered to;
- metric data is collected and used to improve products and processes;
- reviews and audits take place and are properly conducted;
- tests are specified and rigorously carried out;
- problems are recorded and tracked;
- projects use appropriate tools, techniques and methods;
- software is stored in controlled libraries;
- software is stored safely and securely;
- software from external suppliers meets applicable standards;
- proper records are kept of all activities;
- staff are properly trained;
- risks to the project are minimised.

Project management is responsible for the organisation of SQA activities, the definition of SQA roles and the allocation of staff to those roles.

Within a project, different groups have their own characteristic requirements for SQA. Project management needs to know that the software is built according to the plans and procedures it has defined. Development personnel need to know that their work is of good quality and meets the standards. Users should get visibility of SQA activities, so that they will be assured that the product will be fit for its purpose.

The effective management of a project depends upon the control loop shown in Figure 2.1B.

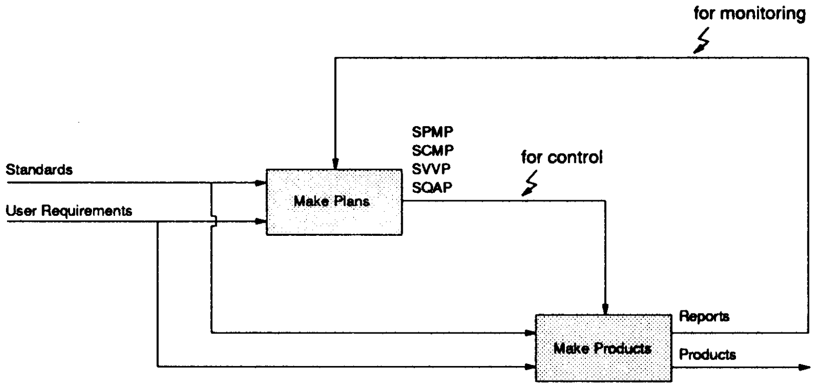

Figure 2.1B: Management control loop

SQA staff make a critical contribution to both control and monitoring activities in the management of a project by:

- writing the SQAP;
- reviewing the SPMP, SCMP and SVVP;
- advising project management on how to apply the standards to planning;
- advising development staff on how to apply the standards to production tasks;
- reporting to the development organisation's management about the implementation of plans and adherence to standards.

The project manager supervises the work of the development staff, and normally carries out the SQA role in small projects (two man years or less). In large projects (twenty man-years or more), specialist personnel should be dedicated to the SQA role because of the amount of work required. It is normal for the SQA staff to report to both the project manager and the corporate SQA manager.

On some projects, SQA functions may be performed in parallel by a separate team, independent of the development organisation, which reports directly to the user or customer. Such teams should have their own SQAP.

In a multi-contractor project, the SQA staff of the prime contractor oversee the activities of the subcontractors by participating in reviews and conducting audits.

The rest of this chapter discusses the software quality assurance activities required in each life cycle phase. As verifying conformance to ESA PSS-05-0 is the primary SQA task, each section discusses all the mandatory practices for the phase from the SQA point of view. Each practice is marked in parentheses and indexed for ease of reference. Management practices that are of special importance in the phase are also included. Each section is therefore an annotated SQA checklist for a life cycle phase. The guidelines should be used to prepare the SQAP for each phase.

In each phase, SQA staff should prepare a report of their activities in time for the formal review (SR/R, AD/R, DD/R). The report should cover all the activities described in the SQAP and should contain recommendations about the readiness to proceed. SQA may also prepare interim reports during the phase to bring concerns to the attention of management.

2.2 USER REQUIREMENTS DEFINITION PHASE

Software quality assurance staff should be involved in a software development project at the earliest opportunity. They should check the user requirements and plans for conformance to standards. These documents have a strong influence on the whole development, and it is extremely cost-effective for any problems in them to be identified and corrected as early as possible.

2.2.1 Checking technical activities

The user requirements document is the users' input to the project. The users are responsible for it (UR01), and SQA should not permit a software development project to start until it has been produced (UR10, UR11).

SQA staff should check the URD for conformance to standards by reading the whole document. They should look for a general description of the software (UR12). This must contain a step-by-step account of what the user wants the software to do (UR14). The URD must also contain specific requirements with identifiers, need attributes, priority attributes and references to sources (UR02, UR03, UR04 and UR05). The URD should contain all the known user requirements (UR13). To confirm completeness, SQA should look for evidence of requirements capture activities such as surveys and interviews.

SQA should check that users have stated in the URD all the constraints they want to impose on the software, such as portability, availability and usability (UR15). The URD must also describe all the external interfaces, or reference them in Interface Control Documents (UR16). External interfaces constrain the design.

The user requirements must be verifiable (UR06). The acceptance test plan defines the scope and approach towards validation, and SQA staff should check that the plan will allow the user requirements to be validated. This plan is documented in the Software Verification and Validation Plan, Acceptance Test section (SVVP/AT).

An important responsibility of SQA staff is to check that the user has described the consequences of losses of availability or breaches of security (UR07). This is needed if the developers are to fully appreciate the criticality of each function.

At the end of the phase the URD is reviewed at the formal user requirements review (UR08). SQA staff should take part in the review, and check that the proper procedures are observed. The review may decide that some requirements should not be made applicable. Such requirements must be clearly flagged in the URD (UR09). They should not be deleted, as they indicate to developers where future growth potential may be needed.

2.2.2 Checking the management plans

Four plans must be produced by the developers by the end of the user requirements review to define how the development will be managed in the SR phase. These are the:

- Software Project Management Plan (SPMP/SR) (SPM02);
- Software Configuration Management Plan (SCMP/SR) (SCM42);
- Software Verification and Validation Plan (SVVP/SR) (SVV09);
- Software Quality Assurance Plan (SQAP/SR) (SQA03).

SQA staff must produce the SQAP/SR. They also need to review the other plans to check that they conform to standards. The SQAP/SR also outlines SQA activities for the whole of the project.

The same structure is used for each plan in every phase of the project. The importance of different parts of the structure varies with the phase. The rest of this section picks out the parts of each plan that are important for the SQA activities in the SR phase.

2.2.2.1 Software project management plan

Every project must produce a software project management plan (SPM01). The plan should declare the objectives of the project. These may take the form of statements of deliverables and delivery dates, or of statements of required functionality, performance and quality. The objectives should be prioritised. SQA staff should examine the statement of objectives and check that all plans are consistent with them.

The plan should describe all the SR phase activities and contain a precise estimate of the SR phase cost (SPM04). The plan should outline activities in the remaining phases (SPM03) and estimate the total cost of the project.

The plan should define the life cycle approach to be used in the project, such as waterfall, incremental delivery or evolutionary. The choice of life cycle approach is crucial to the success of the project. SQA should check that the life cycle approach is appropriate and has been correctly tailored to the project.

A 'process model', based upon the selected life cycle approach should be described in the plan. The process model defines all the major activities. For each activity it should define:

- entry criteria;
- inputs;
- tasks;
- outputs;
- exit criteria.

SQA staff should check the model for consistency. Common problems are documents with no destination, impossible entry criteria and the absence of exit criteria (e.g. successful review).

SQA should check that the plan is sufficient to minimise the risks on the project. For example prototyping is an important risk reduction activity, especially when the project is developing novel software.

SQA should check that the project management plan is realistic. The plan should explicitly declare the:

- assumptions made in planning (e.g. feasibility);
- dependencies upon external events (e.g. supplier delivery);
- constraints on the plan (e.g. availability of staff).

All plans make assumptions, have dependencies, and are subject to constraints. SQA should check that the assumptions, dependencies and constraints have been addressed in the risk analysis.

Quantitative measures (metrics) are important for evaluating project performance [Ref 5]. SQA should check that any metrics proposed in plans are appropriate.

2.2.2.2 Software configuration management plan

SQA need to check that the development of the software is properly controlled. Each project defines the control procedures in the SCMP, before the production starts (SCM41). SQA should verify that they are properly defined and carried out. Examples of poor software configuration management planning are:

- no SCMP;
- an incomplete SCMP;
- no-one with overall responsibility for SCM;
- no identification conventions;
- no change control procedures;
- inefficient change control procedures;
- no approval procedures;
- no storage procedures;
- no mechanisms for tracking changes.

Documents are the primary output of the SR phase. The software configuration management plan for the phase must describe the documentation configuration management procedures. These should be flexible enough to be usable throughout the project. Key aspects that must be addressed are:

- document identification;
- document storage;
- document change control;
- document status accounting.

All relevant project documentation must be uniquely identified (SCM06). Document identifiers should include the company name, the project name, the document type, the document number and/or name, and the document version number (SCM07, SCM08, SCM09, SCM10). The identification scheme must be extensible (SCM11). Every page of the document should be marked with the identifier. SQA staff should check that all project documents have been included in the configuration identification scheme. One way to do this is to examine the process model and decide what kind of documents are input to and output from each activity in the model.

Multiple versions of documents will be generated during a project. A copy of the current issue of every project document must be stored in the master library. Another copy must be archived. SQA staff check that master and archive libraries are complete by performing physical audits.

ESA PSS-05-0 defines the Review Item Discrepancy (RID) procedure for document change control (SCM29). This procedure must be used for reporting problems in documents that undergo formal review. SQA should confirm that all problems discovered during formal review are reported on RID forms and processed by the review.

Document status accounting includes the management of the RID status information (one of CLOSE/UPDATE/ACTION/REJECT). Hundreds of RIDs can be generated on small projects. Large projects, with many reviewers, can produce thousands. Tool support for analysing and reporting RID status is therefore usually needed. SQA staff should ensure that the RID status accounting system is efficient, accurate and gives good visibility of status.

The software configuration management plan should address the identification, storage, control and status accounting of CASE tool outputs, such as files containing design information (SCM43). Plans for the configuration management of prototype software should also be made. Key project decisions may turn upon the results of prototyping, and SQA staff should ensure that the prototypes can be trusted. Poor configuration management of prototypes can cause spurious results.

2.2.2.3 Software verification and validation plan

By the end of the UR phase, the software verification and validation plan needs to include:

- a plan for verifying the outputs of the SR phase;
- an acceptance test plan (SVV11).

Verification techniques used in the SR phase include reviewing, formal proof and tracing. Plans for reviews should include intermediate and formal reviews. Intermediate reviews are used to verify products, and later to clear up problems before formal review. The primary formal review is the Software Requirements Review (SR/R). SQA should examine the software verification and validation plan and check that the review procedures are well-defined, the right people are involved, and that enough reviews are held. Documents that have not been reviewed are just as likely to contain problems as code that has not been tested.

SQA staff should examine the software verification and validation plan and check that the verification approach is sufficiently rigorous to assure the correctness of the software requirements document. Formal methods for specification and verification of software are often necessary when safety and security issues are involved.

The software verification and validation plan needs to define how user requirements will be traced to software requirements. SQA staff often have to check the traceability matrix later in the project, and it is important that traceability procedures are easy to use.

The acceptance test plan must be produced as soon as the URD is available (SVV11). Although the level of detail required depends upon the project, all plans should define the approach to validating the user requirements. SQA staff should check that the plan is sufficiently detailed to allow estimation of the effort required for acceptance testing. Overlooking the need for expensive items of test equipment, such as a simulator, is all too common.

2.3 SOFTWARE REQUIREMENTS DEFINITION PHASE

Software quality assurance staff's primary responsibility in the SR phase is to check that the planned activities are carried out (SR01).

2.3.1 Checking technical activities

All projects should adopt a recognised method for software requirements analysis that is appropriate for the project, and then apply it consistently. The best sign of recognition of a method is publication in a book or journal. All methods should produce a 'logical model' that expresses the logic of the system without using implementation terminology, such as 'record', 'file' or 'event flag', to describe how the software will work (SR02, SR15, SR16, SR17).

Modelling is an essential technical activity in the SR phase and SQA should check that modelling is being done. SQA should check at the beginning of the phase that:

- an analysis method will be used;
- staff have used the analysis method before, or will receive training;
- the analysis method can be supported with CASE tools.

SQA should check that methods, tools and techniques are being properly applied. An important reason for using a tool is that it enforces the correct use of the method. Even though using a tool can slow the creative stages, it pays off in the long run by minimising the amount of work that needs to be redone because rules were broken. CASE tools are not mandatory, but they are strongly recommended.

Techniques such as:

- Failure Modes, Effects and Criticality Analysis (FMECA);
- Common Mode Failure Analysis (CMFA);
- Fault Tree Analysis (FTA);
- Hazard Analysis;

may be employed by developers to verify that software will meet requirements for reliability, maintainability and safety [Ref 8]. The role of SQA is to check that appropriate analysis methods have been properly applied. SQA may introduce requirements for quality, reliability, maintainability and safety based upon the results of the analysis.

The software requirements should specify metrics for measuring quality, reliability (e.g. MTBF), and maintainability (e.g. MTTR). Additional quality-related metrics may be defined by the project. Values of complexity metrics may be defined in the quality requirements to limit design complexity for example.

The Software Requirements Document (SRD) is the primary product of the SR phase (SR10). SQA should read the whole document and check that it has the structure and contents required by the standards (SR18). The general description section should contain the logical model, and SQA should check that it conforms to the rules of the method. If no CASE tool has been used, SQA should check the consistency of the model. An unbalanced data flow can give rise to an interface problem at a later stage, for example. During the intermediate review process, SQA should help analysts make the model understandable to non-experts.

Each requirement in the specific requirements section needs to have an identifier, for traceability (SR04), a flag marking it as essential or desirable (SR05), a flag marking priority (SR06), and references to the URD (SR07). The requirements should be structured according to the logical model.

The SRD needs to be complete, consistent and cover all the requirements in the URD (SR11, SR12, and SR14). SQA should verify completeness by checking the traceability matrix in the SRD (SR13).

The software requirements must be verifiable (SR08). Developers check verifiability by examining each specific requirement and deciding whether it can be verified by testing. If a requirement cannot be tested, other methods, such as inspection or formal proof, must be found. The system test plan defines the scope and approach towards verification of the SRD. This plan is documented in the Software Verification and Validation Plan, System Test section (SVVP/ST). SQA should check that the SVVP/ST defines, in outline, how the software requirements will be verified.

At the end of the phase the SRD is reviewed at the formal software requirements review (SR09). SQA staff should take part in the review, and check that the proper procedures are observed.

2.3.2 Checking the management plans

During the SR phase, four plans must be produced by the developers to define how the development will be managed in the AD phase. These are the:

- Software Project Management Plan (SPMP/AD) (SPM05);
- Software Configuration Management Plan (SCMP/AD) (SCM44);
- Software Verification and Validation Plan (SVVP/AD) (SVV12);
- Software Quality Assurance Plan (SQAP/AD) (SQA06).

SQA staff must produce the SQAP/AD. They also need to review the other plans to check that they conform to standards. The rest of this section picks out the parts of each plan that are important for the AD phase.

2.3.2.1 Software project management plan

In most respects, the SPMP/AD will resemble the SPMP/SR. SQA staff should check that the plan analyses the risks to the project and devotes adequate resources to them. Experimental prototyping may be required for the demonstration of feasibility, for example.

The plan should describe all the AD phase activities and contain a precise estimate of the AD phase cost (SPM07). The most important new component of the plan is the refined cost estimate for the whole project (SPM06). Using the logical model, managers should be able to produce a cost estimate accurate to 30%. SQA staff should check that the cost estimate has been obtained methodically and includes all the necessary development activities. Possible methods include using historical cost data from similar projects, and Albrecht's function point analysis method [Ref 6].

Projects with few technical risks should require little or no experimental prototyping. Accurate cost estimates should therefore be possible. SQA staff should check that there is no discrepancy between the amount of prototyping planned and accuracy of cost estimates.

2.3.2.2 Software configuration management plan

Software configuration management procedures in the AD phase are normally similar to the those used in the SR phase. New procedures might be necessary for the CASE tools used for design, however.

2.3.2.3 Software verification and validation plan

The software verification and validation plan needs to be extended during the SR phase to include:

- a plan for verifying the outputs of the AD phase;
- a system test plan (SVV14).

Verification techniques used in the AD phase are similar to the SR phase. Intermediate reviews are used to agree the design layer by layer, and to clear up problems in the ADD before formal review. The primary formal review is the Architectural Design Review (AD/R).

The system test plan must be produced as soon as the SRD is available (SVV14). Although the level of detail required depends upon the project, all plans should define the approach to verifying the software requirements. Implementation of the verification requirements in the SRD should be considered in the plan.

2.4 ARCHITECTURAL DESIGN PHASE

Software quality assurance staff's primary responsibility in the AD phase is to check that the planned activities are carried out (AD01).

2.4.1 Checking technical activities

A recognised method for software design shall be adopted and applied consistently in the AD phase (AD02). This method must support the construction of 'physical model' that defines how the software works (AD03). The method used to decompose the software into its components must permit a top-down approach (AD04). Just as in the SR phase, SQA should check at the beginning of the phase that:

- an appropriate design method will be used;
- the developers have used the design method before, or will receive training;
- the design method can be supported with CASE tools;
- the lowest level of the architectural design has been agreed.

The architectural design should be reviewed layer-by-layer as it is developed. SQA should attend these intermediate reviews to help, for example:

- prevent the design becoming too complex to test;
- check that feasibility of each major component has been proven;
- ensure that the design will be reliable, maintainable and safe.

SQA should check that design quality has been optimised using the guidelines in ESA PSS-05-04, Guide to the Software Architectural Design Phase [Ref 11].

The primary output of the AD phase is the Architectural Design Document (ADD) (AD19). Although several designs may have been considered in the AD phase, only one design should be presented (AD05). Records of the investigation of alternatives should have been retained for inclusion in the Project History Document.

SQA should check that the ADD defines the functions, inputs and outputs of all the major components of the software (AD06, AD07, AD08, AD17). The ADD must also contain definitions of the data structures that interface components (AD09, AD10, AD11, AD12, AD13, AD18), or reference them in Interface Control Documents (ICDs). The ADD must define the control flow between the components (AD14).

The ADD must define the computer resources needed (AD15). Trade-offs should have been performed during the AD phase to define them. Prototyping may have been required. SQA staff should look for evidence that the resource estimates have been obtained methodically.

The ADD gives everyone on the project visibility of the system as a whole. SQA staff have a special responsibility to ensure that the designers make the document easy to understand. They should review the whole ADD and familiarise themselves with the design. This will improve their effectiveness during the DD phase.

The ADD needs to be complete (AD20) and should include a cross-reference matrix tracing software requirements to components (AD21). SQA should check this table, and that the ADD contains the standard contents (AD24), to confirm completeness. SQA staff should confirm that the ADD has been checked for inconsistencies (AD22), such as mismatched interfaces. The checks need to be very thorough when CASE tools have not been used. The ADD needs to be detailed enough for subgroups of developers to be able to work independently in the DD phase (AD23). The functions of major components and their interfaces need to be well defined for this to be possible.

At the end of the phase the ADD is reviewed at the formal architectural design review (AD16). SQA staff should take part in the review, and check that the proper procedures are observed.

2.4.2 Checking the management plans

During the AD phase, four plans must be produced by the developers to define how the development will be managed in the DD phase. These are the:

- Software Project Management Plan (SPMP/DD)(SPM08);
- Software Configuration Management Plan (SCMP/DD)(SCM46);
- Software Verification and Validation Plan (SVVP/DD) (SVV15);
- Software Quality Assurance Plan (SQAP/DD) (SQA08).

SQA staff must produce the SQAP/DD. They also need to review the other plans to check that they conform to standards. The rest of this section picks out the parts of each plan that are important for the DD phase.

2.4.2.1 Software project management plan

The software project management plan for the DD phase is larger and more complex than SR and AD phase plans because:

- most of the development effort is expended in the DD phase;
- software production work packages must not consume more than one man-month of effort (SPM12);
- parallel working is usually required.

The work-breakdown structure is based upon the component breakdown (SPM10). The level of detail required should result in a cost estimate that is accurate to 10% (SPM09).

SQA should check that the plan contains a network showing the relation of coding, integration and testing activities (SPM11). This network should have been optimised to make the best use of resources such as staff and computing equipment. SQA should check that the plan:

- defines a critical path showing the time required for completion of the phase;
- includes essential activities such as reviews;
- makes realistic estimates of the effort required for each activity (e.g. it is usual to spend at least half the time in the DD phase in testing);
- schedules SQA activities such as audits.

Projects should define criteria, ideally based upon software metrics, to help decide such issues as the readiness of:

- a software component for integration;
- the software for system testing;
- the software for delivery.

SQA should check that the management plan describes the criteria for making these decisions.

2.4.2.2 Software configuration management plan

The primary output of the DD phase is the code, and the software configuration management plan needs to be extended to cover:

- code identification;
- code storage;
- code change control;
- code status accounting.

An efficient configuration management system is vital to the success of the DD phase, and this system must be described in detail in the SCMP/DD (SCM02).

SQA staff should review the SCMP/DD to confirm that all software items, documentation, source code, object code, executable code, data files and test software, are covered by the plan (SCM01). The parts of the SCMP related to document handling can be carried over from earlier phases.

All code must be uniquely identified (SCM06). Identifiers must include the module name, type and version number (SCM07, SCM08, SCM09, SCM10, and SCM11). SQA staff should check that the names of configuration items reflect their purpose.

The SCMP should define the standard header for all source code modules (SCM15, SCM16, SCM17 and SCM18). Documentation and storage media should be clearly labelled (SCM19, SCM20, SCM21 and SCM22). Clear labelling is a sign of a well-run project. Poor labelling is readily apparent in a physical audit. SQA conduct physical audits to verify that the tangible software items such as disks, tapes and files are present and in their correct location.

The core of the software configuration management plan is the software library system (SCM23, SCM24, SCM25). The library procedures should be vetted by SQA to check that access to libraries is controlled (SCM26). SQA should check that software cannot be lost through simultaneous update.

The SCMP must contain a backup plan (SCM27, SCM28). All versions of software should be retained. SQA should check backup logs and stores in physical audits.

Good change control procedures are essential. SQA should examine these procedures and estimate the total time needed to process a change through the configuration management system. This time should be small compared with the time required to do the necessary technical work. If the change process is too time-consuming then there is a high probability of it breaking down at periods of stress, such as when a delivery is approaching. When this happens, configuration management problems will add to technical problems, increasing the pressure on developers. SQA staff should check that the configuration management system can handle the expected volume of change, and if necessary help make it more efficient. Special fast-track procedures may be required for urgent changes, for example.

Configuration status accounting is needed for the effective control of a project. It also gives customers and users visibility of problem tracking and corrective action. SQA should examine the configuration status accounting procedures and confirm that the evolution of baselines is tracked (SCM32, SCM33) and records are kept of the status of RIDs, SPRs, SCRs and SMRs (SCM34). SQA may define software quality metrics that use data in the configuration status accounts. Tools for monitoring quality attributes (e.g. fault rate, repair time) may use the accounts data.

SQA should monitor the execution of the procedures, described in the SCMP, and examine trends in problem occurrence. To make meaningful comparisons, SQA should check that problems are classified. A possible classification is:

- user error;
- user documentation error;
- coding error;
- design error;
- requirements specification error.

Problems may also be categorised by subsystem, or even software component. For each problem category, projects should record:

- number of problems reported;
- number of problems open;
- number of problems closed
- problem reporting rate;
- problem closeout time.

SQA should use these statistics to evaluate product quality and readiness for transfer.

2.4.2.3 Software verification and validation plan

The software verification and validation plan needs to be extended during the AD phase to include:

- a plan for verifying the outputs of the DD phase;
- an integration test plan (SVV17).

The techniques used to verify the detailed design are similar to those used in the AD phase. As before, intermediate reviews are used to agree the design layer by layer. The last review is the code inspection. This is done after modules have been successfully compiled but before unit testing. Peer review of code before unit testing has been shown to be very cost-effective. SQA staff should ensure that this vital step is not missed. Inspection metrics should include:

- lines of code per module;
- cyclomatic complexity per module;
- number of errors per module.

SQA should check that the values of these metrics are within the limits defined in the design and coding standards.

The primary formal review is the Detailed Design Review (DD/R). This is held at the end of the phase to decide whether the software is ready for transfer.

The integration test plan should be produced as soon as the ADD is available. The plan should define the integration approach, and should aim to put the major infrastructure functions in place as early as possible. Thereafter the integration plan should maximise the use of integrated software and minimise the use of test software that substitutes for components that are integrated later.

2.5 DETAILED DESIGN AND PRODUCTION PHASE

Software quality assurance staff's primary responsibility in the DD phase is to check that the planned activities are carried out (DD01).

2.5.1 Checking technical activities

2.5.1.1 Detailed design

ESA PSS-05-0 calls for the detailed design and production of software to be based upon the principles of top-down decomposition and structured programming (DD02, DD03). Technical staff should describe how they will implement these design practices in part 1 of the DDD. SQA should examine the DDD and check that the practices are implemented when they review the software.

SQA should check that the software is documented as it is produced (DD04). Documentation is often deferred when staff are under pressure, and this is always a mistake. SQA can ensure that documentation and production are concurrent by reviewing software as soon as it is produced. If SQA delay their review, developers may defer writing documentation until just before SQA are ready.

The detailed design should be reviewed layer-by-layer. These reviews should include technical managers and the designers concerned. When the design of a major component is finished, a critical design review of the relevant DDD sections must be held to decide upon its readiness for coding (DD10). SQA should attend some of these review meetings, especially when they relate to the implementation of quality, reliability, maintainability and safety requirements. SQA should also check that part 2 of the DDD is a logical extension of the structure defined in the ADD, and that there is a section for every software component (DD14).

2.5.1.2 Coding

SQA should inspect the code. This activity may be part of an inspection process used by development staff, or may be a standalone quality assurance activity. They should verify that the coding standards have been observed. The quality of code should be evaluated with the aid of standard metrics such as module length, complexity and comment rate. Static analysis tools should be used to support metric data collection and evaluation.

2.5.1.3 Reuse

The 'reuse' of software from project to project is increasingly common. SQA should check that the reused software was developed according to acceptable standards.

Software may be reliable in one operational environment but not in another. SQA should treat with extreme caution claims that the quality of reusable software is proven through successful operational use. SQA should check that the development and maintenance standards and records of the software make it fit for reuse.

Reused software is frequently purchased off-the- shelf. Such software is usually called 'commercial software'. SQA should check that the quality certification of the commercial software supplier meets the standards of the project.

2.5.1.4 Testing

Software that has not been tested is very unlikely to work. SQA have a vital role to play in reassuring management and the users that testing has been done properly. To do this, SQA should check that testing activities:

- are appropriate for the degree of criticality of the software (SVV05);
- comply with verification and acceptance testing requirements (stated in the SRD) (SVV06);
- are properly documented in the SVVP (SVV17, SVV18, SVV19, SVV20, SVV21, SVV22).

SQA carry out these checks by reviewing test specifications, observing tests being carried out, participating in selected tests, and reviewing the results. As part of this reviewing activity, SQA may request additions or modification to the test specifications. Normally SQA will not be able to observe all the tests. The SQAP should identify the tests they intend to observe or participate in.

SQA should monitor the progress of the project through the results of tests. SQA should define a metrics programme that includes measures such as:

- number of failures;
- number of failures for each test;
- number of failures per test cycle.

Progress can be measured by a declining number of failures per test cycle (i.e. edit, compile, link, test).

The unit test plan should be written early in the DD phase (SVV18) and SQA should review it. They should check that its level of detail is consistent with the software quality, reliability and safety requirements. The first unit tests should be white-box tests because they give assurance that the software is performing its job in the way it was intended. When full coverage has been achieved (see below), black-box tests should be applied to verify functionality. Black-box tests should also be used to check for the occurrence of likely errors (e.g. invalid input data).

SQA should check that the unit test plan defines the coverage requirements. ESA PSS-05-0's basic requirement is for full statement coverage (DD06). This is a minimum requirement. For most projects, full branch coverage should be achieved during unit testing [Ref 7]. This is because coverage verification requires the tester to trace the path of execution. Debuggers, dynamic testing tools and test harnesses are usually needed to do this, and are most conveniently applied during unit testing. SQA should examine the coverage records. Dynamic testing tools should be used to produce them.

The software verification and validation plan is expanded in the DD phase to include:

- unit, integration and system test designs (SVV19);
- unit, integration and system test cases (SVV20);
- unit, integration and system test procedures (SVV21).

Each test design may have several test cases. A test procedure should be defined for each test case. SQA should review all these extensions to the SVVP test sections.

An important SQA check is to confirm that enough test cases have been specified. When full branch coverage has been specified, SQA should check that the:

- number of test cases for each module is greater than or equal to the cyclomatic complexity of the module [Ref 7];
- paths traversed in the test cases actually result in every branch being traversed.

In applying these rules, a module is assumed to contain a single software component such as a FORTRAN subroutine or a PASCAL procedure. Path coverage should be done for a representative set of test cases. Complete checking is most efficiently done during the testing process with a dynamic testing tool.

There should be an integration test design for every interface (DD07). The test cases should vary the input data so that nominal and limit situations are explored. Integration tests should also exercise the control flows defined in the ADD (DD08).

The system tests must verify compliance with system objectives (DD09). There should be a system test design for each software requirement. System test cases should check for likely ways in which the system may fail to comply with the software requirement.

Test procedures should be written so that they can be executed by someone other than the software author. SQA should review the test procedures and confirm that they are self-explanatory.

The outcome of testing activities must be recorded (SVV22). Reviewing test results is an important SQA responsibility. SQA should check that the causes of test failures are diagnosed and corrective action taken. The problem reporting and corrective action mechanism should be invoked whenever a fault is detected in an item of software whose control authority is not the software author. Normally this means that SPRs are used to record problems discovered during integration testing and system testing. However SPRs also need to be used to record problems when unit tests are done by an independent SVV team.

SQA staff should ensure that failed tests are repeated after repairs have been made. SPRs cannot be closed until the tests that originated them have been passed.

Whereas the integration test plan defines the sequence for building the system from the major components, the software configuration management plan defines the procedures to be used when components are promoted to the master libraries for integration testing (DD05). SQA should check that control authority rights are formally transferred when each component is promoted. The usual means of formal notification is signing-off unit test results. SQA should also check that software authors cannot modify components that are stored in master libraries.

2.5.1.5 Formal review

Every project must hold a DD/R before delivering software to check readiness for transfer (DD11). SQA should participate in the review process and make recommendations based upon:

- test results;
- audit results;
- analysis of outstanding problems.

SQA should conduct a physical audit of the software by checking the configuration item list before transfer. The list should contain every deliverable software item specified in the project plan. The DDD, code and SUM must be included, as these are mandatory outputs of the phase (DD12, DD13 and DD17).

SQA should conduct a functional audit of the software by checking the software requirements versus software components traceability matrix in the DDD (DD16). This checks that the DDD is complete, and accounts for all software requirements (DD15). A second activity is to check that the SVVP/ST contains tests for every software requirement, and that these tests have been run. Functional audit activities should start as early as possible, as soon as the inputs, such as the DDD and test specifications, are available.

SQA's most important role at the DD/R is to analyse the trends in problem occurrence and repair, and advise management about readiness. SQA should categorise failures into major and minor. Major problems put provisional acceptance at risk. Using the records of SPRs in the configuration status accounts, SQA should estimate for each failure category:

- Mean Time Between Failures (MTBF);
- Mean Time To Repair (MTTR);
- number of outstanding problems;
- time required to repair the outstanding problems.

Using this data, management should be able to decide when the software will be ready for transfer. SQA should not approve the transfer of software with major problems outstanding.

2.5.2 Checking the management plans

During the DD phase, three plans must be produced by the developers to define how the development will be managed in the TR phase. These are the:

- Software Project Management Plan (SPMP/TR) (SPM13);
- Software Configuration Management Plan (SCMP/TR) (SCM48);
- Software Quality Assurance Plan (SQAP/TR) (SQA10).

In addition, the Software Verification and Validation Plan must be extended to define the acceptance tests in detail.

SQA staff must produce the SQAP/TR. They also need to review the other plans to check that they conform to standards. The rest of this section picks out the parts of each plan that are important for the TR phase.

2.5.2.1 Software project management plan

Customer confidence in software is greatly increased when the transfer phase is trouble-free. This is most likely when the installation and acceptance testing activities are carefully planned.

The transfer phase plan can be simple for a standalone product developed in the same environment as that used for operations. The software is installed and acceptance tests are run. SQA should check that both these activities have been rehearsed in the DD phase. These rehearsals should allow accurate estimation of TR phase effort.

When the software is to be embedded in a larger system or operated in a different environment, the transfer of software can be much more complicated. The acceptance tests check that the software integrates correctly with the target environment. Problems will always occur when software meets its target environment for the first time. Changes will be necessary and the software project management plan should allow for them.

SQA should check the SPMP/TR plan for realistic estimates for the testing and repair work. They should also check that key development staff will be retained during the phase so that new problems can be diagnosed and corrected quickly. The SPMP/TR should say who is to be involved during the phase. Users, development staff and SQA should attend acceptance tests.

2.5.2.2 Software configuration management plan

The SCMP/TR must define the software configuration management procedures for deliverables in the operational environment (SCM49). SQA should check that the SCMP/TR:

- identifies the deliverable items;
- defines procedures for the storage and backup of deliverables;
- defines the change control procedures;
- defines the problem reporting procedures.

Users may have to apply the procedures defined in the SCMP/TR. This section of the SCMP should be simple and easy to follow, and integrate well with the SUM and STD. SQA should confirm this.

The change control section of the SCMP/TR should include a definition of the terms of reference and procedures of the Software Review Board (SRB). SQA staff should be members of the SRB.

2.5.2.3 Software verification and validation plan

The SVVP is expanded in the DD phase to include test designs, test cases, test procedures and test results for unit tests, integration tests and systems tests, as discussed in Section 2.5.1.4. This section deals with the additions to the SVVP necessary for the TR phase.

During the DD phase, the Acceptance Test section of the Software Verification and Validation Plan (SVVP/AT) is extended to contain:

- acceptance test designs (SVV19);
- acceptance test cases (SVV20);
- acceptance test procedures (SVV21).

SQA should confirm that there is a test design for each user requirement. A test design versus user requirement cross-reference matrix may be inserted in the SVVP/AT to demonstrate compliance.

Test cases should be flagged for use in provisional or final acceptance (TR05). Some properties, such as reliability, may only be demonstrable after a period of operations.

Every project should rehearse the acceptance tests before the DD/R to confirm readiness for transfer. SQA should ensure that the rehearsals (often called ‘dry runs’) are done.

2.6 TRANSFER PHASE

Software quality assurance staff’s primary responsibility in the TR phase is to check that the planned activities are carried out (TR03).

The first TR phase activity is to build the software. This should be done using the components directly modifiable by the maintenance team (TR04). The build procedures must have been defined in the Software Transfer Document (STD). After building, the software is installed using the procedures also defined in the STD. SQA should monitor the installation and build process.

The acceptance tests necessary for provisional acceptance are then run (TR05). SQA should check that the users are present to witness the acceptance tests, and that each test is signed off by developers and users (TR01).

A Software Review Board (SRB) meeting is held after the acceptance tests to review the software's performance and decide whether the software can be provisionally accepted (TR02). For the purposes of acceptance, the 'software' is the outputs of the SR, AD, DD and TR phases. Together with the URD, these outputs constitute the software system (TR07). SQA should attend this meeting. Three outcomes are possible:

- rejection of the software;
- unconditional provisional acceptance of the software;
- conditional provisional acceptance of the software.

The third outcome is the most common. Acceptance is made conditional upon the completion of actions defined by the SRB meeting. These 'close-out' actions usually include the completion of modifications found to be necessary during the acceptance tests.

SQA should check that the statement of provisional acceptance (TR06) is signed by the:

- initiator;
- project manager;
- project SQA manager.

The STD is a mandatory output of the TR phase and SQA should inspect it (TR08). The first STD inspection should be done at the beginning of the phase, before delivery. At this stage the STD contains sections listing the deliverables (i.e. configuration item list), installation procedures and build procedures. The second inspection should be done at the end of the TR phase when the sections describing the acceptance test results, software problem reports, software change requests and software modification results are added (TR10). The last step in the TR phase is the formal hand-over of the STD (TR09).

2.7 OPERATIONS AND MAINTENANCE PHASE

Good software can be ruined by poor maintenance. SQA should monitor software quality throughout the OM phase to check that it is not degraded. SQA should check that the:

- software configuration is properly managed (OM05);
- documentation and code are kept up-to-date (OM06);
- MTBF increases;
- MTTR decreases.

MTBF and MTTR should be regularly estimated from the data in the configuration status accounts.

The SRB authorises all modifications to the software (OM08). SQA should participate in SRB meetings and advise it on issues related to quality, reliability, maintainability and safety.

Development plans should cover the period up to final acceptance (OM01). All projects must have a final acceptance milestone (OM03). Before it arrives, SQA should check that:

- all acceptance tests have been successfully completed (OM02);
- the Project History Document is ready for delivery (OM10);

SQA should check that the statement of final acceptance (OM09) is signed by the:

- initiator;
- project manager;
- project SQA manager.

After final acceptance, an organisation must be defined to take over maintenance (OM04). SQA should check that the maintenance organisation is properly resourced (OM07). Estimates of the resource requirements should be based upon:

- MTBF;
- MTTR;
- size of the system;
- number of subsystems;
- type of system;
- usage pattern.

CHAPTER 3
THE SOFTWARE QUALITY ASSURANCE PLAN

3.1 INTRODUCTION

Plans for software quality assurance activities must be documented in the Software Quality Assurance Plan (SQAP) (SQA02). The first issue of the SQAP must be prepared by the end of the UR review. This issue must outline the SQA activities for the whole project and define in detail SR phase SQA activities (SQA04 and SQA05). Sections of the SQAP must be produced for the AD, DD and TR phases (SQA06, SQA08, SQA10), to cover in detail all the SQA activities that will take place in those phases (SQA07, SQA09 and SQA11).

The table of contents for each section is derived from the IEEE Standard for Software Quality Assurance Plans (ANSI/IEEE Std 730-1989) [Ref 3], which adds the proviso that the table of contents 'should not be construed to prohibit additional content in the SQAP'. The size and content of the SQAP should reflect the complexity of the project. Additional guidelines on the completion of an SQAP can be found in IEEE Guide for Software Quality Assurance Plans (ANSI/IEEE Std 983-1989) [Ref 4].

3.2 STYLE

The SQAP should be plain and concise. The document should be clear, consistent and modifiable.

The author of the SQAP should assume familiarity with the purpose of the software, and not repeat information that is explained in other documents.

3.3 RESPONSIBILITY

A SQAP must be produced by each contractor developing software (SQA01). Review of the SQAPs produced by each contractor is part of the supplier control activity.

The SQAP should be produced by the SQA staff. It should be reviewed by those to whom the SQA personnel report.

3.4 MEDIUM

It is usually assumed that the SQAP is a paper document. There is no reason why the SQAP should not be distributed electronically to participants with the necessary equipment.

3.5 CONTENT

The SQAP is divided into four sections, one for each development phase. These sections are called:

- Software Quality Assurance Plan for the SR phase (SQAP/SR);
- Software Quality Assurance Plan for the AD phase (SQAP/AD);
- Software Quality Assurance Plan for the DD phase (SQAP/DD);
- Software Quality Assurance Plan for the TR phase (SQAP/TR).

ESA PSS-05-0 recommends the following table of contents for each section of the SQAP:

Service Information:

a - Abstract
b - Table of Contents
c - Document Status Sheet
d - Document Change records made since last issue

1 Purpose

2 Reference Documents

3 Management

4 Documentation

5 Standards, practices, conventions and metrics

5.1 Documentation standards
5.2 Design standards
5.3 Coding standards
5.4 Commentary standards
5.5 Testing standards and practices
5.6 Selected software quality assurance metrics
5.7 Statement of how compliance is to be monitored

6 Review and audits[2]

[2] Subsections 6.1 'Purpose' and 6.2 'Minimum Requirements' in ESA PSS-05-0 should not be used. They will be removed from the next issue of ESA PSS-05-0.

7 Test

8 Problem reporting and corrective action

9 Tools, techniques and methods

10 Code control

11 Media control

12 Supplier control

13 Records collection, maintenance and retention

14 Training

15 Risk Management

16 Outline of the rest of the project

Appendix A Glossary

Material unsuitable for the above contents list should be inserted in additional appendices. If there is no material for a section then the phrase 'Not Applicable' should be inserted and the section numbering preserved.

Sections 3 to 15 of the SQAP should describe how the technical activities and management plans will be checked. Table 3.5 traces each section of SQAP to the complementary documents or document sections that describe what is to be checked. The SQAP should identify (i.e. reference) the material in each of the other documents, but not repeat it. Sections 3 to 15 of the SQAP should also describe how SQA activities will be reported.

SQAP section	Document/Document section
Management	SPMP SCMP/Management SVVP/SR-AD-DD-TR/Reporting
Documentation	SPMP/Software documentation
Standards, practices and conventions	SPMP/Methods, tools and techniques SVVP/SR-AD-DD-TR/Administrative procedures DDD/Project standards, conventions and procedures
Reviews and audits	SVVP/SR-AD-DD-TR/Activities
Test	SVVP/AT-ST-IT-UT
Problem reporting and corrective action	SCMP/Change control SVVP/SR-AD-DD-TR/Administrative procedures
Tools, techniques and methods	SPMP/Methods, tools and techniques SCMP/Tools, techniques and methods SVVP/SR-AD-DD-TR/Overview ADD/Design method
Code control	SCMP/Code control
Media control	SCMP/Media control
Records collection, maintenance and retention	SCMP/Configuration status accounting SCMP/Records collection and retention
Supplier control	SCMP/Supplier control
Training	SPMP
Risk management	SPMP/Risk Management

Table 3.5 Software quality assurance plan sections traced to other project documents

3.5.1 SQAP/1 Purpose

This section should briefly:

- describe the purpose of the SQAP;
- specify the intended readership of the SQAP;
- list the software products to be developed;
- describe the intended use of the software;
- identify the phase of the life cycle to which the plan applies.

3.5.2 SQAP/2 Reference Documents

This section should provide a complete list of all the applicable and reference documents, identified by title, author and date. Each document should be marked as applicable or reference. If appropriate, report number, journal name and publishing organisation should be included.

3.5.3 SQAP/3 Management

This section should describe the organisation of quality assurance, and the associated responsibilities. The SQAP should define the roles to be carried out, but not allocate people to roles, or define the effort and schedule. ANSI/IEEE Std 730-1989, 'Standard for Software Quality Assurance Plans' [Ref 3] recommends that the following structure be used for this section.

3.5.3.1 SQAP/3.1 Organisation

This section should:

- identify the organisational roles that control and monitor software quality (e.g. project manager, team leaders, software engineers, software librarian, software verification and validation team leader, software quality assurance engineer);
- describe the relationships between the organisational roles;
- describe the interface with the user organisation.

Relationships between the organisational roles may be shown by means of an organigram. This section may reference the SPMP, SCMP and SVVP.

This section should describe how the implementation of the organisational plan will be verified.

3.5.3.2 SQAP/3.2 Tasks

This section should define the SQA tasks (selected from SQAP sections 3.4 to 3.13) that are to be carried out in the phase of the life cycle to which this SQAP applies. This section should define the sequencing of the selected tasks. Additional tasks may be included.

3.5.3.3 SQAP/3.3 Responsibilities

This section should identify the SQA tasks for which each organisational role is responsible. A cross-reference matrix may be used to show that each SQA task has been allocated.

3.5.4 SQAP/4 Documentation

This section should identify the documents to be produced in the phase. The SPMP normally contains (or references) a documentation plan listing all the documents to be produced in the phase.

This section should state how the documents will be checked for conformance to ESA PSS-05-0 and the project documentation plan.

3.5.5 SQAP/5 Standards, practices, conventions and metrics

The following subsections should identify the standards, practices, conventions and metrics used to specify software quality, and explain how SQA will check that the required quality will be achieved.

3.5.5.1 SQAP/5.1 Documentation standards

This section should identify the standards, practices, and conventions that will be used to produce the documents of the phase. Documentation standards are normally defined in the documentation plan (see section 4 of the SQAP).

3.5.5.2 SQAP/5.2 Design standards

This section should identify the standards, practices and conventions that will be used in the phase to design the software. Design standards are normally defined or referenced in the ADD and DDD.

3.5.5.3 SQAP/5.3 Coding standards

This section should identify the standards, practices, and conventions that will be used in the phase to write code. Coding standards are normally defined or referenced in the DDD.

3.5.5.4 SQAP/5.4 Commentary standards

This section should identify the standards, practices and conventions that will be used in the phase to comment code. Commentary standards are normally included in the coding standards.

3.5.5.5 SQAP/5.5 Testing standards and practices

This section should identify the standards, practices and conventions that will be used in the phase to test the software. Testing standards are normally defined in the SRD and SVVP.

3.5.5.6 SQAP/5.6 Selected software quality assurance metrics

This section should identify the metrics that will be used in the phase to measure the quality of the software. Metrics are normally defined in the project standards and plans.

3.5.5.7 SQAP/5.7 Statement of how compliance is to be monitored

This section should describe how SQA will monitor compliance to the standards, practices, conventions and metrics.

3.5.6 SQAP/6 Review and audits

This section should identify the technical reviews, inspections, walkthroughs and audits that will be held during the phase, and the purpose of each. It should describe how adherence to the review and audit procedures (defined in the SVVP) will be monitored, and the role of SQA personnel in the review and audit process.

3.5.7 SQAP/7 Test

This section should describe how the testing activities described in the SVVP will be monitored and how compliance with verification and acceptance-testing requirements in the SRD will be checked.

3.5.8 SQAP/8 Problem reporting and corrective action

This section should identify (by referencing the SCMP) the problem reporting and corrective action procedures (e.g. RID and SPR handling procedures).

This section should describe how adherence to the problem reporting procedures described in the SCMP will be monitored.

This section may describe the metrics that will be applied to problem reporting process to estimate the software quality.

3.5.9 SQAP/9 Tools, techniques and methods

This section should identify the tools, techniques and methods used to develop the software.

This section should describe how the use of the tools, techniques and methods will be monitored.

Additional tools, techniques and methods for supporting SQA tasks may be described here (or in the section on the task).

3.5.10 SQAP/10 Code (and document) control

This section should identify the procedures used to maintain, store, secure and document software. These procedures should be defined in the SCMP.

This section should describe how adherence to the procedures will be monitored.

3.5.11 SQAP/11 Media control

This section should identify the procedures used to maintain, store, secure and document controlled versions of the physical media on which the identified software resides. These procedures should be defined in the SCMP.

This section should describe how adherence to the procedures will be monitored.

3.5.12 SQAP/12 Supplier control

A supplier is any external organisation that develops or provides software to the project (e.g. subcontractor developing software for the project, or a company providing off-the-shelf commercial software).

This section should identify the standards that will be applied by suppliers.

This section should describe how adherence of the suppliers to the applicable standards will be monitored.

This section should identify the procedures that will be applied to goods supplied, such as commercial software and hardware.

This section should describe how adherence to the incoming inspections (i.e. goods-in procedures) will be monitored.

3.5.13 SQAP/13 Records collection, maintenance and retention

This section should identify the procedures kept by the project for recording activities such as meetings, reviews, walkthroughs, audits and correspondence. It should describe where the records are kept, and for how long.

This section should describe how adherence to the record-keeping procedures will be monitored.

3.5.14 SQAP/14 Training

This section should identify any training programmes defined for the project staff and explain how SQA will check that they have been implemented.

3.5.15 SQAP/15 Risk Management

This section should identify the risk management procedures used in the project (which should be described in the SPMP).

This section should describe how adherence to the risk management procedures will be monitored.

3.5.16 SQAP/16 Outline of the rest of the project

This section should be included in the SQAP/SR to provide an overview of SQA activities in the AD, DD and TR phases.

3.5.17 SQAP/APPENDIX A Glossary

This section should provide the definitions of all terms, acronyms, and abbreviations used in the plan, or refer to other documents where the definitions can be found.

3.6 EVOLUTION

3.6.1 UR phase

By the end of the UR review, the SR phase section of the SQAP must be produced (SQAP/SR) (SQA03). The SQAP/SR must describe, in detail, the quality assurance activities to be carried out in the SR phase (SQA04). The SQAP/SR must outline the quality assurance plan for the rest of the project (SQA05).

3.6.2 SR phase

During the SR phase, the AD phase section of the SQAP must be produced (SQAP/AD) (SQA06). The SQAP/AD must cover in detail all the quality assurance activities to be carried out in the AD phase (SQA07).

In the SR phase, the SRD should be analysed to extract any constraints that relate to software quality assurance (e.g. standards and documentation requirements).

3.6.3 AD phase

During the AD phase, the DD phase section of the SQAP must be produced (SQAP/DD) (SQA08). The SQAP/DD must cover in detail all the quality assurance activities to be carried out in the DD phase (SQA09).

3.6.4 DD phase

During the DD phase, the TR phase section of the SQAP must be produced (SQAP/TR) (SQA10). The SQAP/TR must cover in detail all the quality assurance activities to be carried out from the start of the TR phase until final acceptance in the OM phase (SQA11).

APPENDIX A
GLOSSARY

A.1 LIST OF ACRONYMS

AD	Architectural Design
AD/R	Architectural Design Review
ADD	Architectural Design Document
ANSI	American National Standards Institute
AT	Acceptance Test
BSSC	Board for Software Standardisation and Control
CASE	Computer-Aided Software Engineering
CMFA	Common Mode Failure Analysis
DCR	Document Change Record
DD	Detailed Design and production
DD/R	Detailed Design and production Review
DDD	Detailed Design and production Document
DSS	Document Status Sheet
ESA	European Space Agency
FMECA	Failure Modes, Effects and Criticality Analysis
FTA	Fault Tree Analysis
IEEE	Institute of Electrical and Electronics Engineers
IT	Integration Test
MTBF	Mean Time Between Failures
MTTR	Mean Time To Repair
PA	Product Assurance
PSS	Procedures, Specifications and Standards
QA	Quality Assurance
RID	Review Item Discrepancy
SCM	Software Configuration Management
SCMP	Software Configuration Management Plan
SCR	Software Change Request
SMR	Software Modification Report
SPM	Software Project Management
SPMP	Software Project Management Plan
SPR	Software Problem Report
SQA	Software Quality Assurance
SQAP	Software Quality Assurance Plan
SR	Software Requirements
SR/R	Software Requirements Review
SRD	Software Requirements Document
ST	System Test
STD	Software Transfer Document
SVVP	Software Verification and Validation Plan
UR	User Requirements

UR/R	User Requirements Review
URD	User Requirements Document
UT	Unit Tests

APPENDIX B
REFERENCES

1. ESA Software Engineering Standards, ESA PSS-05-0 Issue 2 February 1991.
2. IEEE Standard Glossary of Software Engineering Terminology, ANSI/IEEE Std 610.12-1990.
3. IEEE Standard for Software Quality Assurance Plans, ANSI/IEEE Std 730-1989.
4. IEEE Guide for Software Quality Assurance Planning, ANSI/IEEE Std 983-1986.
5. Application of Metrics in Industry Handbook, AMI Project, Centre for Systems and Software Engineering, South Bank University, London, UK, 1992.
6. Software function, source lines of code, and development effort prediction: a software science validation, A.J.Albrecht and J.E.Gaffney, IEEE Transactions on Software Engineering, vol SE-9, No 6, November 1983.
7. Structured Testing; A Software testing Methodology Using the Cyclomatic Complexity Metric, T.J. McCabe, NBS Special Publication 500-99, 1992.
8. Software Engineering, Design, Reliability and Management, M.L.Shooman, McGraw-Hill, 1983.
9. Software Product Assurance Requirements for ESA Space Systems, ESA PSS-01-21, Issue 2, April 1991.
10. Guide to the Software Engineering Standards, ESA PSS-05-01, Issue 1, October 1991.
11. Guide to the Software Architectural Design Phase, ESA PSS-05-04, Issue 1, January 1992.

Copies of IEEE standards may be purchased (by credit card) from:

IEEE Service Center,
445 Hoes Lane, P.O. Box 1331,
Piscataway, NJ 08855-1331, USA
Tel: 908-981-0060

This page is intentionally left blank.

APPENDIX C
MANDATORY PRACTICES

This appendix is repeated from ESA PSS-05-0, appendix D.12

SQA01 An SQAP shall be produced by each contractor developing software.

SQA02 All software quality assurance activities shall be documented in the Software Quality Assurance Plan (SQAP).

SQA03 By the end of the UR review, the SR phase section of the SQAP shall be produced (SQAP/SR).

SQA04 The SQAP/SR shall describe, in detail, the quality assurance activities to be carried out in the SR phase.

SQA05 The SQAP/SR shall outline the quality assurance plan for the rest of the project.

SQA06 During the SR phase, the AD phase section of the SQAP shall be produced (SQAP/AD).

SQA07 The SQAP/AD shall cover in detail all the quality assurance activities to be carried out in the AD phase.

SQA08 During the AD phase, the DD phase section of the SQAP shall be produced (SQAP/DD).

SQA09 The SQAP/DD shall cover in detail all the quality assurance activities to be carried out in the DD phase.

SQA10 During the DD phase, the TR phase section of the SQAP shall be produced (SQAP/TR).

SQA11 The SQAP/TR shall cover in detail all the quality assurance activities to be carried out from the start the TR phase until final acceptance in the OM phase.

This page is intentionally left blank.

APPENDIX D
INDEX

This page is intentionally left blank.

AD01, 15
AD02, 16
AD03, 16
AD04, 16
AD05, 16
AD06, 16
AD07, 16
AD08, 16
AD09, 16
AD10, 16
AD11, 16
AD12, 16
AD13, 16
AD14, 16
AD15, 16
AD16, 17
AD17, 16
AD18, 16
AD19, 16
AD20, 17
AD21, 17
AD22, 17
AD23, 17
AD24, 17
audits, 37

CFMA, 13
code control, 38
coding, 22
conventions, 36
corrective action, 37

DD01, 21
DD02, 22
DD03, 22
DD04, 22
DD05, 25
DD06, 24
DD07, 24
DD08, 24
DD09, 25
DD10, 22
DD11, 25
DD12, 26
DD13, 26
DD14, 22
DD15, 26
DD16, 26
DD17, 26
detailed design, 21
document control, 38
documentation, 36

FMECA, 13
FTA, 13

hazard analysis, 13

IEEE
  Std 730, 31
  Std 983, 31

management, 35
media control, 38
methods, 38
metric, 4, 9, 13, 18, 20 - 23, 36

OM01, 30
OM02, 30
OM03, 30
OM04, 30
OM05, 30
OM06, 30
OM07, 30
OM08, 30
OM09, 30
OM10, 30

practices, 36
problem reporting, 37
process model, 9

records collection, 39
records maintenance, 39
records retention, 39
reuse, 22
reviews, 37
risk management, 39

SCM01, 19
SCM02, 19
SCM06, 10, 19
SCM07, 10, 19
SCM08, 10, 19
SCM09, 10, 19
SCM10, 10, 19
SCM11, 10, 19
SCM15, 19
SCM16, 19
SCM17, 19
SCM18, 19
SCM19, 19
SCM20, 19
SCM21, 19
SCM22, 19
SCM23, 19
SCM24, 19
SCM25, 19
SCM26, 19
SCM27, 19
SCM28, 19
SCM29, 11
SCM32, 20
SCM33, 20
SCM34, 20
SCM41, 10
SCM42, 8
SCM43, 11
SCM44, 14
SCM46, 17
SCM48, 26
SCM49, 27
SCMP/AD, 14
SCMP/DD, 17
SCMP/SR, 8
SCMP/TR, 26

Software Quality Assurance
  see SQA
SPM01, 8
SPM02, 8
SPM03, 8
SPM04, 8
SPM05, 14
SPM06, 14
SPM07, 14
SPM08, 17
SPM09, 18
SPM10, 18
SPM11, 18
SPM12, 18
SPM13, 26
SPMP/AD, 14
SPMP/DD, 17
SPMP/SR, 8
SPMP/TR, 26
SQA, 3
SQA01, 31
SQA02, 31
SQA03, 8, 39
SQA04, 31, 40
SQA05, 31, 40
SQA06, 14, 31, 40
SQA07, 31, 40
SQA08, 17, 31, 40
SQA09, 31, 40
SQA10, 26, 31, 40
SQA11, 31, 40
SQAP
  AD phase, 40
  DD phase, 40
  SR phase, 39
  TR phase, 40
SQAP/AD, 14
SQAP/DD, 17
SQAP/SR, 8
SQAP/TR, 26
SR01, 12
SR02, 12
SR04, 13
SR05, 13
SR06, 13
SR07, 13
SR08, 14
SR09, 14
SR10, 13
SR11, 14
SR12, 14
SR13, 14
SR14, 14
SR15, 12
SR16, 12
SR17, 12
SR18, 13
standards, 36
supplier control, 38
SVV05, 23
SVV06, 23
SVV09, 8
SVV11, 11, 12
SVV12, 14
SVV14, 15
SVV15, 17
SVV17, 21, 23
SVV18, 23
SVV19, 23, 24, 28
SVV20, 23, 24, 28
SVV21, 23, 24, 28
SVV22, 23, 25
SVVP/AD, 14
SVVP/AT, 7
SVVP/DD, 17
SVVP/SR, 8
SVVP/ST, 14

techniques, 38
testing, 23, 37
tools, 38
TR01, 29
TR02, 29
TR03, 28
TR04, 28
TR05, 28, 29
TR06, 29
TR07, 29
TR08, 29
TR09, 29
TR10, 29
training, 39

UR01, 7
UR02, 7
UR03, 7
UR04, 7
UR05, 7
UR06, 7
UR07, 7
UR08, 7
UR09, 7
UR10, 7
UR11, 7
UR12, 7
UR13, 7
UR14, 7
UR15, 7
UR16, 7