# SEP projects (2IP35)

## Table of Contents

General Information ................................................................. 2  
Project Management ............................................................... 2  
Documentation ................................................................. 2  
Way of working .................................................................. 3  
Progress control ............................................................. 3  
Verification and Validation ............................................. 4  
Delivery of documents .................................................... 4  
Assessment ................................................................. 5  
URD (User Requirements Document) table of contents .......... 6  
SRD (Software Requirements Document) table of contents .... 7  
ADD (Architectural Design Document) table of contents ........ 8  
DDD (Detailed Design Document) table of contents .......... 9  
SUM (Software User Manual) table of contents ................. 10  
STD (Software Transfer Document) table of contents ........ 12  
The Software Project Management Plan ......................... 13  
The Software Configuration Management Plan ................. 15  
SVVP (Software Verification and Validation Plan) general table of contents ........................................ 16  
UT (Unit Test), IT (Integration Test), ST (System Test) and AT (Acceptance Test ) table of contents ......................... 17  
SQAP (Software Quality Assurance Plan) table of contents .... 19  
The Software Project Progress Report table of contents ........ 21  
Peer evaluation form .............................................................. 22  
Requirements review checklist ........................................ 23  
Design review checklist ...................................................... 24  
User manual guidelines ......................................................... 26
General Information

The fictitious company SEP consisting of two staff members ("senior management") has defined software engineering projects to be executed by groups of students. Each project has an external customer who needs a software program that has to be developed.

Project Management

A project consists of a group of approximately 8 students. They work almost full time for the project. Every group has its own project room. Each group has a project manager and a quality manager (students that follows 2IP45). Furthermore, each group has an advisor (staff member) to advise in technical and other questions and to review the technical documents.

Documentation

The documentation delivered by the projects is according to the ESA standard. This means that the following documents are produced in the course of a project:

- Project documents:
  a. SPMP (software project management plan)
  b. SVVP (software verification and validation plan).
     This includes also the UT (unit test), IT (integration test) and AT (acceptance test) plans.
     In our case, the ST (system test) can be omitted.
  c. SCMP (software configuration management plan)
  d. SQAP (software quality assurance plan)
- Product documents:
  e. URD (user requirements document).
  f. SRD (software requirements document).
  g. ADD (architectural design document).
  h. DDD (detailed design document).
  i. SUM (software use manual).
  j. STD (software transfer document).
Note that the ESA prescribes a number of chapters for each document. These should be taken as a guideline for structuring the document. Non applicable sections should be omitted. For example, the DDD often can be generated to a large degree. Examples of documents from previous years are available.

Way of working
A project starts with the gathering of the requirements of the customer. These have to be documented in the URD, whereas the SRD presents a specification of a solution. Once requirements are clear, a rough architecture can be sketched, finally leading to the architecture document. Parallel to the requirements work, plans are written and configuration management and a test environment are set up.

The coding phase is executed in an agile way, we will use Scrum for this. Five sprints of one week each will be carried out, each preceded by a planning session (1 hour, the customer being present) and followed by a demo (30 minutes, for management and customer) and retrospective (15 minutes). The project manager or quality manager is the Scrum master. We follow the way of working as prescribed in "Scrum and XP from the Trenches" (the first ten chapters are important).

Agile also implies that a continuous integration and test set-up should be used: each new implemented feature (story) should be accompanied by a test case. These test cases are added to the automatic regression test that is run whenever changes are checked in in the main line development tree.

The project has successfully ended when a build and install of the software has been performed at the customer site (in his environment) and the acceptance test has been passed.

Progress control
Each group should have a regular internal progress meeting every week (e.g. on Monday). Topics of these meetings are the work done last week, deviations from the plan(ning), and the work to be done in the coming week. Once the coding has started, the internal progress meeting is replaced by the planning session and retrospective.

Progress meetings between project / quality manager and senior management will be held also every week. For each such progress meeting, a progress report has to be prepared.

All project members should keep track of the hours they spend on each work package. These project metrics should be sent weekly to senior management. This amounts to two lists:

<table>
<thead>
<tr>
<th>Activity</th>
<th>Hours spent this week</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non project related</td>
<td>...</td>
</tr>
<tr>
<td>General project related</td>
<td>...</td>
</tr>
<tr>
<td>(progress meetings, SPMP writing, etc)</td>
<td>...</td>
</tr>
<tr>
<td>Documentation, specification, design</td>
<td>...</td>
</tr>
<tr>
<td>(all activities leading to such results)</td>
<td>...</td>
</tr>
<tr>
<td>Source code</td>
<td>...</td>
</tr>
<tr>
<td>Testing, verification, consolidation</td>
<td>...</td>
</tr>
<tr>
<td>Rework</td>
<td>...</td>
</tr>
<tr>
<td>Total</td>
<td>40 * group size</td>
</tr>
<tr>
<td></td>
<td>(includes project manager)</td>
</tr>
</tbody>
</table>

and

<table>
<thead>
<tr>
<th>Work package</th>
<th>Hours spent this week</th>
<th>Hours to go till end of project (new estimation)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPMP</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>SVVP</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
The last column ("hours to go till end of project") should contain the hours that are left according to the reported status of the project: an up-to-date reliable estimate, not just the original number of planned hours minus the hours spent.

The code work package mentioned in the table will be broken down in separate user stories in the planning sessions, whereas each of the others can be considered as one story (white index card as in “Scrum and XP from the Trenches”).

Peer evaluations, where the team members judge the work of the other group members, will take place halfway and at the end of the project. Use the peer evaluation form at the end of this document, an excel form is also available.

Verification and Validation

The URD, SRD, ADD, SUM and AT have to be reviewed:
- URD, SRD, SUM and AT with customer and advisor,
- ADD with only the advisor.

Before this formal "external review", the project group takes care that an internal review has been held. The advisor decides whether a document is acceptable for the final ("external") review.

There are checklists for reviews at the end of this document.

Review results and metrics should be sent to senior management.

The customer or advisor sign for the acceptance of the reviewed product documents:
- The customer signs for the URD, SRD, SUM, AT.
- The advisor signs for the acceptance of the ADD.

Delivery of documents

All technical product documents should be delivered to the customer, that is:
- URD, SRD, ADD, DDD, source code, SUM, STD and UT, IT, AT.

Senior management should receive all documents at the end of the project:
- The product documents (like the customer).
- The project documents: SPMP, SCMP, SVVP, SQAP.
- Review reports.
- Progress reports.
- Meeting notes.
Assessment

The result of the project is mainly judged by the quality of the product documents\(^1\) delivered (especially the ones that have to be approved by the advisor) and the quality of the software produced.

The mark of a project will be calculated as follows.

(1) The documents and code delivered by each group are marked using the SEP marking form (http://wwwis.win.tue.nl/2IP35/marking/SEP_marking.xlsx). A copy of this form will be filled in by at least two staff members (advisors and/or coordinators). The results will form the main (average) mark for the group.

(2) The individual contribution by each student will be judged based on the peer reviews (using http://wwwis.win.tue.nl/2IP35/marking/SEP_peer_review_template.xlsx), that are held twice during the project. These reviews are filled by (a) the students of the group, (b) the project manager and quality manager, and (c) the advisor.

The mark for each student will be equal to the result from (1) with a correction [-1,+1] from (2).

\(^1\) URD: current situation, environment, users (roles, rights), use cases / scenarios, requirements (SMART); SRD: class diagram, (if necessary) state charts, sequence diagrams for scenarios (use cases), user interface, requirements; ADD: components, interfaces (internal/external), design choices; UT, IT, AT tests should have been documented.
# URD (User Requirements Document) table of contents

The contents of the ESA description of a URD, which is based on the IEEE table of contents.

<table>
<thead>
<tr>
<th>a. Abstract</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>b. Table of Contents</td>
<td></td>
</tr>
<tr>
<td>c. Document Status Sheet</td>
<td>Status sheet for configuration control (previous versions and their main contributions).</td>
</tr>
<tr>
<td>d. Document Change Records since previous issue</td>
<td>A list of document changes.</td>
</tr>
</tbody>
</table>

## 1. Introduction

1.1 Purpose | The purpose of this particular URD and its intended readership. |
1.2 Scope | Scope of the software. Identifies the product by name, explains what the software will do. |
1.3 List of definitions | The definitions of all used terms, acronyms and abbreviations. |
1.4 List of references | All applicable documents. |
1.5 Overview | Short description of the rest of the URD and how it is organized. |

## 2. General description

2.1 Product perspective | The relation to other systems. |
2.2 General capabilities | The main capabilities. |
2.3 General constraints | Reasons why constraints exist: background information and justification. |
2.4 User characteristics | The characteristics of the different user roles. |
2.5 Environment description | A description of the operational environment. |
2.6 Assumptions and dependencies | The assumptions upon which the specific requirements (in the next section) are based. |

## 3. Specific requirements

3.1 Capability requirements | A list of all capability requirements. |
3.2 Constraint requirements | A list of all constraint requirements (interfaces, portability, adaptability, availability, security, safety, standards, resources, time scales, …). |
SRD (Software Requirements Document) table of contents

The contents of the ESA description of an SRD, which based on the IEEE table of contents.

<table>
<thead>
<tr>
<th>a.</th>
<th>Abstract</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>b.</td>
<td>Table of Contents</td>
<td></td>
</tr>
<tr>
<td>c.</td>
<td>Document Status Sheet</td>
<td>Status sheet for configuration control (previous versions and their main contributions).</td>
</tr>
<tr>
<td>d.</td>
<td>Document Change Records since previous issue</td>
<td>A list of document changes.</td>
</tr>
</tbody>
</table>

1. Introduction

1.1 Purpose

The purpose of this particular SRD and its intended readership.

1.2 Scope

Scope of the software. Identifies the product by name, explains what the software will do.

1.3 List of definitions

The definitions of all used terms, acronyms and abbreviations.

1.4 List of references

All applicable documents.

1.5 Overview

Short description of the rest of the SRD and how it is organized.

2. General description

2.1 Relation to current projects

The context of this project in relation to other current projects.

2.2 Relation to predecessor and successor projects

The context of this project in relation to past and future projects.

2.3 Function and purpose

A general overview of the function and purpose of the product.

2.4 Environment

Hardware and operating system of target system and development system. Who will use the system (user roles in URD).

2.5 Relation to other systems

Is the product an independent system, part of a larger system, replacing another system? The essential characteristics of these other systems.

2.6 General constraints

Reasons why constraints exist: background information and justification (analogous to URD).

2.7 Model description

A description of the logical model.

3. Specific requirements

3.1 Functional requirements

A list of all functional requirements (what should the system do).

3.2 Non-functional requirements

A list of all non-functional requirements (performance, interface, operational, resource, verification/testing, portability, maintainability, reliability, security, safety, documentation, other, ...), linked to functional requirements. Each category of non-functional requirements has its own subsection.

4. Requirements traceability matrix

A table showing how each user requirement of the URD is linked to software requirements in the SRD.
ADD (Architectural Design Document) table of contents

The contents of the ESA description of an ADD, which is based on the IEEE table of contents.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>Abstract</td>
</tr>
<tr>
<td>b.</td>
<td>Table of Contents</td>
</tr>
<tr>
<td>c.</td>
<td>Document Status Sheet</td>
</tr>
<tr>
<td>d.</td>
<td>Document Change Records since previous issue</td>
</tr>
</tbody>
</table>

1. Introduction

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Purpose</td>
</tr>
<tr>
<td>1.2</td>
<td>Scope</td>
</tr>
<tr>
<td>1.3</td>
<td>List of definitions</td>
</tr>
<tr>
<td>1.4</td>
<td>List of references</td>
</tr>
<tr>
<td>1.5</td>
<td>Overview</td>
</tr>
</tbody>
</table>

2. System overview

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>External interface definition</td>
</tr>
</tbody>
</table>

3. System context (for each external interface ...)

4. System design

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Design method</td>
</tr>
<tr>
<td>4.2</td>
<td>Decomposition description</td>
</tr>
</tbody>
</table>

5. Component descriptions (for each component ...)

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5.n</td>
<td>Component identifier</td>
</tr>
<tr>
<td>5.n.1</td>
<td>Type</td>
</tr>
<tr>
<td>5.n.2</td>
<td>Purpose</td>
</tr>
<tr>
<td>5.n.3</td>
<td>Function</td>
</tr>
<tr>
<td>5.n.4</td>
<td>Subordinates</td>
</tr>
<tr>
<td>5.n.5</td>
<td>Dependencies</td>
</tr>
<tr>
<td>5.n.6</td>
<td>Interfaces</td>
</tr>
<tr>
<td>5.n.7</td>
<td>Resources</td>
</tr>
<tr>
<td>5.n.8</td>
<td>References</td>
</tr>
<tr>
<td>5.n.9</td>
<td>Processing</td>
</tr>
<tr>
<td>5.n.10</td>
<td>Data</td>
</tr>
</tbody>
</table>

6. Feasibility and resource estimates

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>A summary of computer resources needed to build, operate and maintain the software.</td>
</tr>
</tbody>
</table>

7. Requirements traceability matrix

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1</td>
<td>A table showing how each software requirement of the SRD is linked to components in the ADD.</td>
</tr>
</tbody>
</table>
## DDD (Detailed Design Document) table of contents

The contents of the ESA description of a DDD, which is based on the IEEE table of contents.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>Abstract</td>
</tr>
<tr>
<td>b.</td>
<td>Table of Contents</td>
</tr>
<tr>
<td>c.</td>
<td>Document Status Sheet</td>
</tr>
<tr>
<td>d.</td>
<td>Document Change Records since previous issue</td>
</tr>
</tbody>
</table>

### 1. Introduction

| 1.1 | Purpose | The purpose of this particular DDD and its intended readership. |
| 1.2 | Scope | Scope of the software. Identifies the product by name, explains what the software will do. |
| 1.3 | List of definitions | The definitions of all used terms, acronyms and abbreviations. |
| 1.4 | List of references | All applicable documents. |
| 1.5 | Overview | Short description of the rest of the DDD and how it is organized. |

### 2. Standards and conventions

| 2.1 | Design standards | Name and reference of design method used. |
| 2.2 | Documentation standards | Format, style and tools for design and code documentation. |
| 2.3 | Naming conventions | All file naming conventions. |
| 2.4 | Coding standards | Coding standards for each programming language used. |
| 2.5 | Software development tools | The tools used for software construction. |

### 3. Component descriptions (for each component ...)

| 3.n | Component identifier | A unique identifier for the module. |
| 3.n.1 | Type | Task, procedure, package, program, file, ... |
| 3.n.2 | Purpose | Software requirements implemented. |
| 3.n.3 | Function | What the module does. |
| 3.n.4 | Subordinates | Child modules (modules called, files composed of, classes used). |
| 3.n.5 | Dependencies | Components to be executed before/after, excluded operations during execution. |
| 3.n.6 | Interfaces | Data and control flow in and out. |
| 3.n.7 | Resources | Needed to perform the function. |
| 3.n.8 | References | To other documents. |
| 3.n.9 | Processing | Internal control and data flow. |
| 3.n.10 | Data | Internal data. |

### 4. Build procedure

How to build the system from the source code (`make file`). [This section is not present in the ESA standard.]

### A. Source code listings

Listings of the source code or a list showing where the source code can be found.

### B. Requirements traceability matrix

A table showing how each software requirement of the SRD is linked to components in the DDD.
SUM (Software User Manual) table of contents

The contents of the ESA description of a SUM, which is based on the IEEE table of contents.

| a. | Abstract |
| b. | Table of Contents |
| c. | Document Status Sheet | Status sheet for configuration control (previous versions and their main contributions). |
| d. | Document Change Records since previous issue | A list of document changes. |

1. Introduction
1.1 Intended readership | User categories (end user, operator), level of experience assumed, which sections are most relevant. |
1.2 Applicability | Software releases the SUM applies to. |
1.3 Purpose | Purpose of the SUM and of the software. |
1.4 How to use this document | What each section contains and the relationships between sections. |
1.5 Related documents | All applicable documents. |
1.6 Conventions | All symbols, stylistic conventions and command syntax conventions used. |
1.7 Problem reporting | A summary how to report software problems. |

2. Overview
A summary of the process to be supported, the fundamental principles of the process, what the software does to support the process, what the user needs to supply to the software.

3. Tutorial (for each session ...)
3.n.1 Functional description | What the tutorial session is supposed to achieve. |
3.n.2 Cautions and warnings | A list of precautions that may need to be taken. |
3.n.3 Procedures | How to prepare for and start the task. |
| | A step-by-step description of what the user must do and the response of the system. |
| | What final results to expect. |
3.n.4 Likely errors | An informal description (not a list of errors) of likely errors and possible causes. |

4. Reference (for each operation ...)
4.n.1 Functional description | What the operation (command, menu item, button, ...) achieves. |
4.n.2 Cautions and warnings | A list of cautions and warnings that apply to the operation. |
4.n.3 Formal description | A formal description of what the operation does and how it is used: required parameters, optional parameters, defaults, syntax and semantics. |
4.n.4 Examples | One or more examples of the use of the operation. |
4.n.5 Possible errors | A list of all possible errors for this operation and their causes. |
4.n.6 Related operations | References to, for example, operations to complete a task or logically related operations. |

A. Error messages and recovery procedures | A list of all error messages; for each error: diagnosis and recovery procedure. |

B. Glossary | An explanation of terms used. |
| C. Index | A list of topics and their pages. |
STD (Software Transfer Document) table of contents

The contents of the ESA description of an STD, which is based on the IEEE table of contents.

<table>
<thead>
<tr>
<th>a.</th>
<th>Abstract</th>
</tr>
</thead>
<tbody>
<tr>
<td>b.</td>
<td>Table of Contents</td>
</tr>
<tr>
<td>c.</td>
<td>Document Status Sheet</td>
</tr>
</tbody>
</table>

1. Introduction
   1.1 Purpose | The purpose of this particular STD and its intended readership. |
   1.2 Scope | Scope of the software. Identifies the product by name, explains what the software will do. |
   1.3 List of definitions | The definitions of all used terms, acronyms and abbreviations. |
   1.4 List of references | All applicable documents. |

2. Build procedure | When, where, in which environment the software was built. Version of the software which was built. Problems during build, time needed to build. |

3. Installation procedure | When, where, in which environment the software was installed. Version of the software installed. Problems during installation, time needed to install. |

4. Configuration item list | All configuration items (object under version control) transferred. |

5. Acceptance test report summary | Summary of test reports in the AT, user requirements not met, number of acceptance tests passed/failed. |

6. Software Problem Reports | A list of software problem reports raised during the transfer phase. |

7. Software Change Requests | A list of software change requests raised during the transfer phase. |

8. Software Modification Reports | A list of software modification reports completed during the transfer phase. |
The Software Project Management Plan

In principle, the SR, AD, DD and TR sections of the Software Project Management Plan (SPMP) are separate documents (all having the same structure), describing the project management activities for a specific phase.

However, we write only one SPMP for the whole project.

| a. Abstract |
| b. Table of Contents |
| c. Document Status Sheet | Status sheet for configuration control (previous versions and their main contributions). |
| d. Document Change Records since previous issue | A list of document changes. |

1. Introduction
1.1 Project overview | Summary of objectives, deliverables, schedule, budget. |
1.2 Project deliverables | All ESA documents (mandatory) and other items (prototypes, tools). |
1.3 Evolution of the SPMP | How the SPMP itself will be updated. |
1.4 List of definitions | The definitions of all used terms, acronyms and abbreviations. |
1.5 List of references | All applicable documents. |

2. Project organization
2.1 Process model | The activities of the phase and their inputs and outputs. |
2.2 Organizational structure | An organigram of the internal management structure of the project (roles, communication structure and hierarchy). |
2.3 Boundaries and interfaces | Relationships with external groups (senior management, end users, subcontractors, suppliers). |
2.4 Project responsibilities | For each role (2.2): purpose and responsibilities. |

3. Managerial process
3.1 Objectives and priorities | Management objectives (short). |
3.2 Assumptions, dependencies and constraints | Budget limitations, schedule constraints, availability of external systems. |
3.3 Risk management | Identified risks and planned actions to manage them. |
3.4 Monitoring and controlling mechanisms | Formats for all documents used to monitor and control the project (progress reports, reviews). Frequency of progress meetings with initiators and management. Frequency of progress reports. |
3.5 Staffing plan | Names and roles (utilization). |

4. Technical process
4.1 Methods, tools and techniques | Which are used to produce the deliverables. |
4.2 Software documentation | The review and approval requirements for each document. Also for non-deliverable (internal) documents. May also contain format and layout requirements. |
4.3 Project support functions | Reference to SCMP, SVVP, SQAP. |

5. Work packages, schedule, budget
5.1 Work packages | All work packages including description and manager. |
5.2 Dependencies | The ordering of the work packages (activity network). |
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3</td>
<td>Resource requirements</td>
</tr>
<tr>
<td>5.4</td>
<td>Budget and resource allocation</td>
</tr>
<tr>
<td>5.5</td>
<td>Schedule</td>
</tr>
</tbody>
</table>
The Software Configuration Management Plan

In principle, the SR, AD, DD and TR sections of the Software Configuration Management Plan (SCMP) are separate documents (all having the same structure), describing the configuration management activities for a specific phase.

However, we write only one SCMP for the whole project.

<table>
<thead>
<tr>
<th>a. Abstract</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>b. Table of Contents</td>
<td></td>
</tr>
<tr>
<td>c. Document Status Sheet</td>
<td>Status sheet for configuration control (previous versions and their main contributions).</td>
</tr>
<tr>
<td>d. Document Change Records since previous issue</td>
<td>A list of document changes.</td>
</tr>
</tbody>
</table>

1. Introduction

1.1 Purpose The purpose of this particular section of the SCMP and its intended readership.
1.2 Scope Summary of configuration items and configuration management activities.
1.3 List of definitions The definitions of all used terms, acronyms and abbreviations.
1.4 List of references All applicable documents.

2. Management

2.1 Organization Roles involved in configuration management and their relationships: usually a reference to the SPMP.
2.2 Responsibilities For each role (2.1): responsibilities in identification (3), storage (4.1, 4.2), change (4.3), status accounting (5), review, audit, approval.
For the users: responsibilities in review, audit and approval.
2.3 Interface management Procedures (responsibilities, contact persons) for managing external hardware and software interfaces.
2.4 SCMP implementation Planning for implementing the SCMP (establishment of the review board, release of products). Can reference the SPMP.
2.5 Applicable procedures A reference to all applicable (general or project specific) policies, directives and procedures.

3. Configuration identification

3.1 Naming conventions For configuration items (name, type, version).
3.2 Baselines Identifier, contents (documents, tools, test software, SPR's, etc), reviews, acceptance criteria for each baseline.

4. Configuration control

4.1 Library control Procedures to handle the software libraries.
4.2 Media control How to handle (label and store) the storage media.
4.3 Change control Definition of the change control process: authorities, change procedures, review board definition.
Also: control of supporting external software.

5. Status accounting How the configuration status is recorded and reported.

6. Tools, techniques and methods For identification, storage (like sccs, rcs, cvs), change control and status accounting of configuration items.

7. Supplier control Configuration management requirements on external organizations (suppliers and subcontractors).

8. Records collection and retention Which records are retained, how (for example on paper) and how long (for example only for the last three baselines).
SVVP (Software Verification and Validation Plan) general table of contents

In principle, the SR, AD and DD sections of the Software Verification and Validation Plan (SVVP) are separate documents (all having the same structure), describing how the output of a specific phase (SR, AD or DD) is verified against the requirements of the previous phase.

However, we write only one SVVP for the whole project.

Note: To align them with the other ESA documents, the ESA sections 1, 2 and 3 have been renumbered to 1.1, 1.4 and 1.3. Consequently, the other sections (4..) have also been renumbered (to 2..).

| a. Abstract                                                                 |          |
| b. Table of Contents                                                        |          |
| c. Document Status Sheet                                                    | Status sheet for configuration control (previous versions and their main contributions). |
| d. Document Change Records since previous issue                             | A list of document changes. |

1. Introduction

1.1 Purpose                                                                 | The purpose of this particular section of the SVVP and its intended readership. |
1.2 Scope                                                                  | Summary of products to be verified and specifications to be verified against. |
1.3 List of definitions                                                    | The definitions of all used terms, acronyms and abbreviations. |
1.4 List of references                                                     | All applicable documents. |

2. Verification overview

2.1 Organization                                                           | Roles, reporting, relationships to project management. |
2.2 Schedule                                                               | Schedule for reviews and tracing activities. |
2.3 Resources                                                              | Summary of staff, computer facilities, tools needed. |
2.4 Project responsibilities                                                | For each role (2.1): responsibilities. |
2.5 Tools, techniques and methods                                          | Used for reviews and tracing. |

3. Administrative procedures

3.1 Anomaly reporting and resolution                                         | How to handle anomalies: probably a reference to the procedure in the SCMP. |
3.2 Task iteration policy                                                   | Criteria for deciding whether to repeat a task after a change. |
3.3 Deviation policy                                                        | Procedures for deviating from the plan. |
3.4 Control procedures                                                      | Configuration management procedures for review and tracing. |
3.5 Standards                                                               | For tracing and reviews. |

4. Verification activities

4.1 Reviews                                                                | Inspection, walkthrough and technical review procedures. |
4.2 Formal proofs                                                          | Which methods are used (if any) for formal proof. |
4.3 Tracing                                                                | How to trace inputs to outputs. |

5. Verification reporting                                                  | Which reports will be produced (summary report, walkthrough reports, ...). |
UT (Unit Test), IT (Integration Test), ST (System Test) and AT (Acceptance Test) table of contents

These are separate documents, all having the same structure.

In our case the ST is equal to the AT, thus it does not have to be written.

Below is a simplified version of an SVVP/UT, SVVP/IT, SVVP/ST or SVVP/AT section. This is a short version of the ESA description.

<table>
<thead>
<tr>
<th>a. Abstract</th>
<th>b. Table of Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>c. Document Status Sheet</td>
<td>Status sheet for configuration control (previous versions and their main contributions).</td>
</tr>
<tr>
<td>d. Document Change Records since previous issue</td>
<td>A list of document changes.</td>
</tr>
</tbody>
</table>

1. Introduction

1.1 Purpose The purpose of this particular section of the SVVP and its intended readership.

1.2 Overview

1.3 List of definitions The definitions of all used terms, acronyms and abbreviations.

1.4 List of references All applicable documents.

2. Test plan

2.1 Test items The items to be tested.

2.2 Features to be tested

2.3 Test deliverables Items that must be delivered before testing starts and when testing ends.

2.4 Testing tasks To prepare and carry out the tests.

2.5 Environmental needs Properties required of the test environment.

2.6 Test case pass/fail criteria

3. Test case specifications (for each test case ...)

3.n.1 Test case identifier A unique identifier.

3.n.2 Test items The items to be tested.

3.n.3 Input specifications Input for this test case.

3.n.4 Output specifications Output required from this test case.

3.n.5 Environmental needs The test environment.

4. Test procedures (for each test procedure ...)

4.n.1 Test procedure identifier A unique identifier.

4.n.2 Purpose The purpose of this test procedure and the test cases this procedure executes.

4.n.3 Procedure steps How to log, set up, start, proceed, measure, shut down, restart, stop the test.

5. Test reports (for each execution of a test procedure ...)

5.n.1 Test report identifier A unique identifier.

5.n.2 Description The items tested.

5.n.3 Activity and event entries Identification of the test procedure. When was the test performed, by who and who witnessed it. For each test case in the procedure: did the software...
| pass or fail, what are the problems. |
SQAP (Software Quality Assurance Plan) table of contents

The SR, AD, DD and TR sections of the Software Quality Assurance Plan (SQAP) are separate documents (each having the same structure), describing the quality assurance activities for a specific phase.

However, we write only one SQAP for the whole project.

Note: To align them with the other ESA documents, the ESA sections 1, 2 and A have been renumbered to 1.1, 1.4 and 1.3. Consequently, the other sections (3..) have also been renumbered (to 2..).

<table>
<thead>
<tr>
<th>a. Abstract</th>
<th>b. Table of Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>c. Document Status Sheet</td>
<td>Status sheet for configuration control (previous versions and their main contributions).</td>
</tr>
<tr>
<td>d. Document Change Records since previous issue</td>
<td>A list of document changes.</td>
</tr>
</tbody>
</table>

1. Introduction

1.1 Purpose

The purpose of this particular section of the SQAP and its intended readership.

1.2 Scope

A list of software products to be developed and their intended use.

1.3 List of definitions

The definitions of all used terms, acronyms and abbreviations.

1.4 List of references

All applicable documents.

2. Management

2.1 Organization

Roles involved in quality assurance and their relationships; usually a reference to the SPMP. How the implementation of this plan is verified.

2.2 Tasks

Quality assurance tasks that will be carried out in this phase.

2.3 Responsibilities

For each role (2.1) in each task (2.2).

3. Documentation

Documents to be produced in this phase: usually a reference to the software documentation in the SPMP. How the documents are checked for conformance to standards and plan.

4. Standards, practices, conventions and metrics

4.1 Documentation standards

Usually defined in the software documentation in the SPMP.

4.2 Design standards

Usually defined in the ADD and DDD.

4.3 Coding standards

Usually defined in the DDD.

4.4 Comment standards

Usually defined in the DDD.

4.5 Testing standards

Usually defined in the SVVP.

4.6 Metrics

Used to monitor quality.

4.7 Compliance monitoring

How adherence to standards and metrics (4.1-4.5) is checked.

5. Review

Usually defined in the SVVP. How adherence to the procedures is checked.

6. Test

How testing activities defined in the SVVP are monitored. How compliance with verification and acceptance testing requirements in the SRD is checked.

7. Problem reporting and corrective action

Usually defined in the change control in the SCMP. How adherence to these problem reporting procedures is checked.
| **8. Tools, techniques and methods** | Defined in SPMP, SCMP, SVVP, and the design method of the ADD. Additional quality assurance tools. How their use is monitored. |
| **9. Code control** | Usually defined in the library control in the SCMP. How adherence to these procedures is checked. |
| **10. Media control** | Usually defined in the media control in the SCMP. How adherence to these procedures is checked. |
| **11. Supplier control** | Standards to be applied by external suppliers. Procedures to inspect incoming goods. How adherence to these standards and procedures is checked. |
| **12. Records collection, maintenance and retention** | How, where, for how long, records of meetings, reviews and correspondence are kept. How adherence to these procedures is checked. |
| **13. Training** | Training programs for staff and how to check that they are implemented. |
| **14. Risk management** | Usually defined in the SPMP. How risk management procedures are checked. |
| **15. Outline of the rest of the project** | In SR section: overview of quality assurance activities in AD, DD and TR phases. |
Software Project Progress Report table of contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.</td>
<td>Abstract</td>
</tr>
<tr>
<td>b.</td>
<td>Table of Contents</td>
</tr>
<tr>
<td>1.</td>
<td>Introduction</td>
</tr>
<tr>
<td>1.1</td>
<td>Purpose</td>
</tr>
<tr>
<td>1.2</td>
<td>Summary</td>
</tr>
<tr>
<td>1.3</td>
<td>List of definitions</td>
</tr>
<tr>
<td>1.4</td>
<td>List of references</td>
</tr>
<tr>
<td>2.</td>
<td>Technical status</td>
</tr>
<tr>
<td>2.1</td>
<td>Workpackage technical status</td>
</tr>
<tr>
<td>2.2</td>
<td>Configuration status</td>
</tr>
<tr>
<td>2.3</td>
<td>Forecast for the next reporting period</td>
</tr>
<tr>
<td>3.</td>
<td>Resource status</td>
</tr>
<tr>
<td>3.1</td>
<td>Staff utilization</td>
</tr>
<tr>
<td>3.2</td>
<td>Workpackage resource status</td>
</tr>
<tr>
<td>3.3</td>
<td>Resource summary</td>
</tr>
<tr>
<td>4.</td>
<td>Schedule status</td>
</tr>
<tr>
<td>4.1</td>
<td>Milestone trends</td>
</tr>
<tr>
<td>4.2</td>
<td>Schedule summary</td>
</tr>
<tr>
<td>5.</td>
<td>Problems</td>
</tr>
<tr>
<td>6.</td>
<td>Financial status report</td>
</tr>
<tr>
<td>6.1</td>
<td>Costs for the reporting period</td>
</tr>
<tr>
<td>6.2</td>
<td>Cost to completion</td>
</tr>
<tr>
<td>6.3</td>
<td>Limit of liability</td>
</tr>
<tr>
<td>6.4</td>
<td>Payments</td>
</tr>
</tbody>
</table>
# Peer evaluation form

**Peer Evaluation Form**

Pre: Discuss the notion of commitment (inzet) in the group to obtain a common understanding.

<table>
<thead>
<tr>
<th>Project</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student</td>
<td>Period</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Student</th>
<th>Judgment</th>
<th>Advice</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Further Comments**

**Legend**

<table>
<thead>
<tr>
<th>Project</th>
<th>Name of project</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Number of project (1 - 8 or A - G)</td>
</tr>
<tr>
<td>Student</td>
<td>Name of student doing the evaluation</td>
</tr>
<tr>
<td>Period</td>
<td>Evaluation period: e.g. January-March, or April-June</td>
</tr>
</tbody>
</table>

Evaluate all student members in the group, including yourself and the project manager.

<table>
<thead>
<tr>
<th>Student</th>
<th>Name of student being evaluated</th>
</tr>
</thead>
</table>
| Judgment| Your impression of the student's commitment (inzet) encoded as follows:

++ Indispensable (voortrekker)
+
Valueable (nuttig)
0 Neutral
-
Mediocre (matig)
-- Lacking (maakt zich er vanaf)

| Advice | Explain how the student might improve a low judgment |
Requirements review checklist

Requirements Review Checklist

1. Does the (software) product have a succinct name, and a clearly described purpose?
2. Are the characteristics of users and of typical usage mentioned? (No user categories missing.)
3. Are all external interfaces of the software explicitly mentioned? (No interfaces missing.)
4. Does each specific requirement have a unique identifier?
5. Is each requirement atomic and simply formulated? (Typically a single sentence. Composite requirements must be split.)
6. Are requirements organized into coherent groups? (If necessary, hierarchical; not more than about ten per group.)
7. Is each requirement prioritized? (Is the meaning of the priority levels clear?)
8. Are all unstable requirements marked as such? (TBC=’To Be Confirmed’, TBD=’To Be Defined’)
9. Is each requirement verifiable (in a provisional acceptance test)? (Measurable: where possible, quantify; capacity, performance, accuracy)
10. Are the requirements consistent? (Non-conflicting.)
11. Are the requirements sufficiently precise and unambiguous? (Which interfaces are involved, who has the initiative, who supplies what data, no passive voice.)
12. Are the requirements complete? Can everything not explicitly constrained indeed be viewed as developer freedom? Is a product that satisfies every requirement indeed acceptable? (No requirements missing.)
13. Are the requirements understandable to those who will need to work with them later?
14. Are the requirements realizable within budget?
15. Do the requirements express actual customer needs (in the language of the problem domain), rather than solutions (in developer jargon)?
## Design Review Checklist

**1. Are the following attributes well-defined for each design entity?**
- **Identification** (unique name)
- **Type** (describing what kind of design entity it is)
- **Purpose** (describing why it was introduced, in terms of the requirements)
- **Function** (summarizing what the component does)
- **Dependencies** (possibly 'none'; describing the requires or uses relationship)
- **Interface** (provided by the design entity)
- **Processing** (including autonomous activities)
- **Data** (information 'hidden' inside)

**2. Is the relationship to the requirements clearly motivated? Is it clear why the proposed architecture realizes the requirements?**

**3. Is the software architecture as simple as possible (but no simpler)?**
- No more than 7 loosely-coupled coherent high-level components.
- Lower-level components possibly clustered into high-level components (hierarchy).
- Using standard(ized) components.
- Is deviation from intuitively obvious solution motivated?

**4. Is the architecture complete?**
- Are all requirements covered?
- Trace some critical requirements through the architecture (e.g. via use cases).

**5. Are the component descriptions sufficiently precise?**
- Do they allow independent construction?
- Are interfaces and external functionality of the high-level components described in sufficient detail?
- Interface details:
  - Routine kind, name, parameters and their types, return type, pre- and postcondition, usage protocol w.r.t. other routines.
  - File name, format, permissions.
  - Socket number and protocol.
  - Shared variables, synchronization primitives (locks).
- Have features of the target programming language been used where appropriate?
- Have implementation details been avoided? (No details of internal classes.)

**6. Are the relationships between the components explicitly documented?**
- Preferably use a diagram

**7. Is the proposed solution realizable?**
- Can the components be implemented or bought, and then integrated together.
- Possibly introduce a second layer of decomposition to get a better grip on realizability.

**8. Are all relevant architectural views documented?**
- **Logical** (Structural) view (class diagram per component expresses functionality).
- **Process** view (how control threads are set up, interact, evolve, and die).
- **Physical** view (deployment diagram relates components to equipment).
- **Development** view (how code is organized in files).

**9. Are cross-cutting issues clearly and generally resolved?**
- Exception handling.
- Initialization and reset.
- Memory management.
- Security.
- Internationalization.
• Built-in help.
• Built-in test facilities.

10. Is all **formalized material** and **diagrammatic material** accompanied by sufficient explanatory text in natural language?

11. Are **design decisions** documented explicitly and motivated?
• Restrictions on developer freedom w.r.t. the requirements.

12. Has an evaluation of the software architecture been documented?
• Have alternative architectures been considered?
• Have non-functional requirements also been considered?
• **Negative indicators:**
  o High **complexity**: a component has a complex interface or functionality.
  o Low **cohesion**: a component contains unrelated functionality.
  o High **coupling**: two or more components have many (mutual) connections.
  o High **fan-in**: a component is needed by many other components.
  o High **fan-out**: a component depends on many other components.

13. Is the **flexibility** of the architecture demonstrated?
• How can it cope with likely changes in the requirements?
• Have the most relevant change scenarios been documented?
• Do these change cases match the short and long term vision of the customer?
User manual guidelines

Some basic rules for clear writing:

- Keep sentences short (less than 20 words).
- Avoid long words.
- Keep paragraphs short (less than 10 sentences).
- Do not use passive terms (‘has been’).
- Use correct grammar.
- Make each paragraph express one point.
- Tell what you are going to say, say it, tell what you have said.

A good SUM is consistent:

- Do not use different terms for the same thing.
- Do not use the same term for different things.