Document Type	Specification
Document Number	ESS-0233615
Date	Feb 7, 2018
Revision	1 (8)
State	Review
Confidentiality Level	Internal
Page	1 (23)

PSS0 Verification and Validation Procedure

	Name	Role/Title
Owner	Yong Kian Sin	Electrical Controls Engineer, ICS Division, PS Group
Reviewer	Stuart Birch Hector Novella	Senior Engineer Personnel Safety Systems, ICS Division, PS Group Deputy Project Manager, ICS Division
Approver	Annika Nordt	Group Leader for Protection Systems Group, ICS Division

TABLE OF CONTENT **PAGE**

1. EXECUTIVE SUMMARY

PSS0 aims to mitigate electrical hazards for personnel arising from operating the ISrc and LEBT test stand.

This document will address the verification and validation for hardware and software of PSS0 system.

This document is a generic procedure document and describes the main verification and validation activities. A generic template for the verification reports is defined in this document.

2. ABBREVIATIONS

E/E/PE	Electrical/Electronic/Programmable Electronic safety related systems
ESS	European Spallation Source
CCB	Change Control Board
CCR	Configuration Correlation Record
CTL	Change Request List
DCR	Design Change Request
DES	Designer
HMI	Human Machine Interface
HW Config	Hardware Configuration
ISrc	Ion Source
ISTQB	International Software Testing Qualification Board
LEBT	Low Energy Beam Transport
PLC	Programmable Logic Controller
VAL	Validator
VER	Verifier

3. INTRODUCTION

3.1. Objectives

This document describes the generic Verification and Validation Procedure of the PSS0 such as test activities in different stages of the project lifecycle, different test levels and types, acceptance driven testing approach. The proposed test philosophy is mainly based on ISTQB.

The detailed test case specifications will be defined during the software development activities. This means that test cases will be defined per feature to be developed and according to Safety Integrity Level (SIL) defined, and the development of features will start by focusing on the core functionality necessary for PSS0. The features should be integrated incrementally, so that as far as possible have a working system as earlier as possible. This allows the PSS0 to make technical adjustments early if necessary and provides possibility to implement changes if integrity level modified, in order to reduce the knock-on effect of unclear requirements on other features during the development phase.

The verification and validation procedure is divided in two parts. The first part describes the verification activities and the second part the validation activities.

The verification plan shall address the following:

- the selection of verification strategies and techniques.
- the selection and utilisation of the test equipment;
- the selection and documentation of verification activities;
- the evaluation of verification results gained;
- the roles and responsibilities of those involved in the test process;
- the degree of test coverage required to be achieved.

The validation plan shall identify the steps necessary to demonstrate the adequacy of the development documents and shall analyse and test the integrated system to ensure compliance with ESS Process for Validation [2] and Safety Requirements Specification [1] with particular emphasis on the functional aspects.

3.2. ISTQB

The International Software Testing Qualifications Board (ISTQB) is a software testing qualification and certification organisation that operates internationally. Founded in Edinburgh in November 2002, ISTQB is a non-profit association legally registered in

Belgium. ISTQB® Certified Tester is a standardized qualification for software testers and the certification is offered by the ISTQB (International Software Testing Qualifications Board). The qualifications are based on a syllabus [], and there is a hierarchy of qualifications and guidelines for accreditation and examination.

3.3. Lifecycle

3.3.1. Requirements and Specification

The first stages of the IEC 61508 safety life cycle define the concept and scope of the system, assess the potential system hazards and estimate the risks they pose. This is followed by safety requirements specification and the allocation of these safety requirements to different sub-systems. The specification of the requirements for software safety shall be derived from the specified safety requirements of the safety-related system and any requirements of safety planning. The requirements for software safety shall be sufficiently detailed to allow design and implementation. The requirements must be clear, precise, verifiable, testable, maintainable, and feasible. The requirements must also be appropriate for the safety integrity level and traceable back to the specification of the safety requirements of the safety-related system. All modes of operation for the safety-related system must be listed. The requirements must detail any relevant constraints between the hardware and the software.

3.3.2. Design and Realisation

Design methods shall be chosen that support abstraction, modularity, and other good software engineering practices. The design method shall allow clear and unambiguous expression of functionality, data flow, sequencing, and time-dependent data, timing constraints, concurrency, data structures, design assumptions, and their dependencies. During design, the overall complexity of the design, its testability, and the ability to make safe modifications shall be considered. The entire design is considered safety-related even if non safety functions are included unless sufficient independence between safety and non-safety can be demonstrated.

The architectural design defines the major components and subsystems of the software. The architectural design description must include:

1. interconnections of these components;
2. the “techniques and measures” necessary during the software safety life cycle phases to satisfy requirements for software safety at the required safety integrity level including software design strategies for fault tolerance and/or fault avoidance (redundancy/diversity)

3. the software safety integrity level of the subsystem/component;
4. all software/hardware interactions and their significance;
5. the design features for maintaining the safety integrity of all data;
6. software architecture integration tests to ensure that the software architecture satisfies the requirements for software. It is assumed and permitted that iteration occurs between the design and the requirements phases. Any resulting changes in requirements must be documented and approved according to Configuration Management [5]. Detailed design and coding shall follow the software safety life cycle.

3.3.3. Integration and Testing

Tests of the integration between the hardware and software are created during the design and development phases and specify the following:

1 test environment, tools, and configuration;
2 test criteria;
3 procedures for corrective action on failure of test. The integration testing results shall state each test and the pass/fail results.

3.3.4. Verification

The verification process (with refer to ESS Process for Verification [6]) tests and evaluates the results of the safety life cycle phases to insure they are correct and consistent with the input information to those phases. Verification of the steps used in the safety life cycle must be performed according to the plan and must be done concurrently with design and development. The verification plan must indicate the activities performed and the items to be verified (documents, reviews, etc.). A verification report must include an explanation of all activities and results.

3.3.5. Validation

Validation (with refer to ESS Process for Validation [2]) is done as an overall check to insure that the design meets the safety requirements and must include the appropriate documentation. The validation may be done as part of overall system validation. Testing must be the primary method of validation with analysis used only to supplement.

If discrepancies occur, a change request [5] must be created and an analysis must be done to determine if the validation may continue.

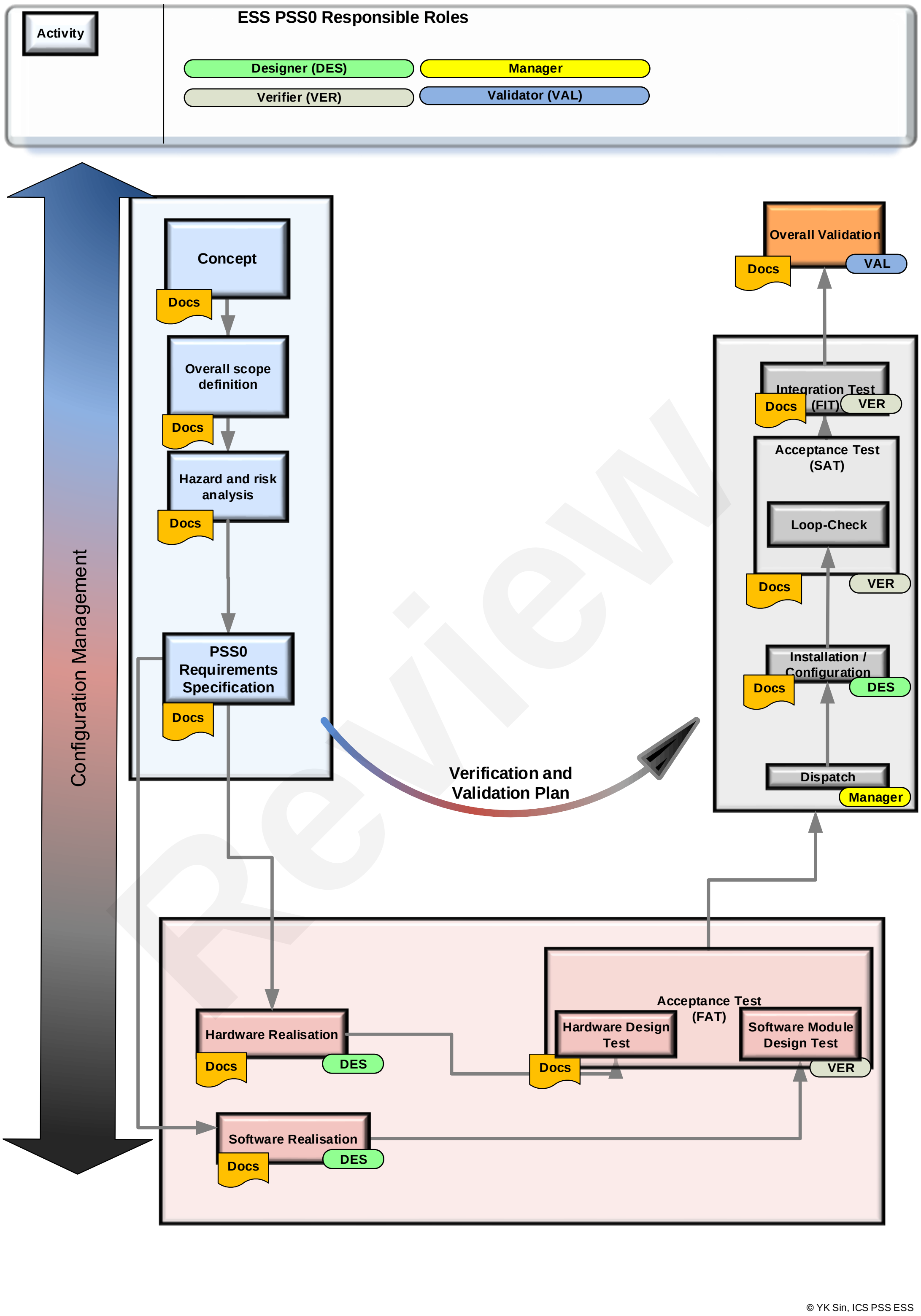

Figure 1: PSS0 Development Lifecycle

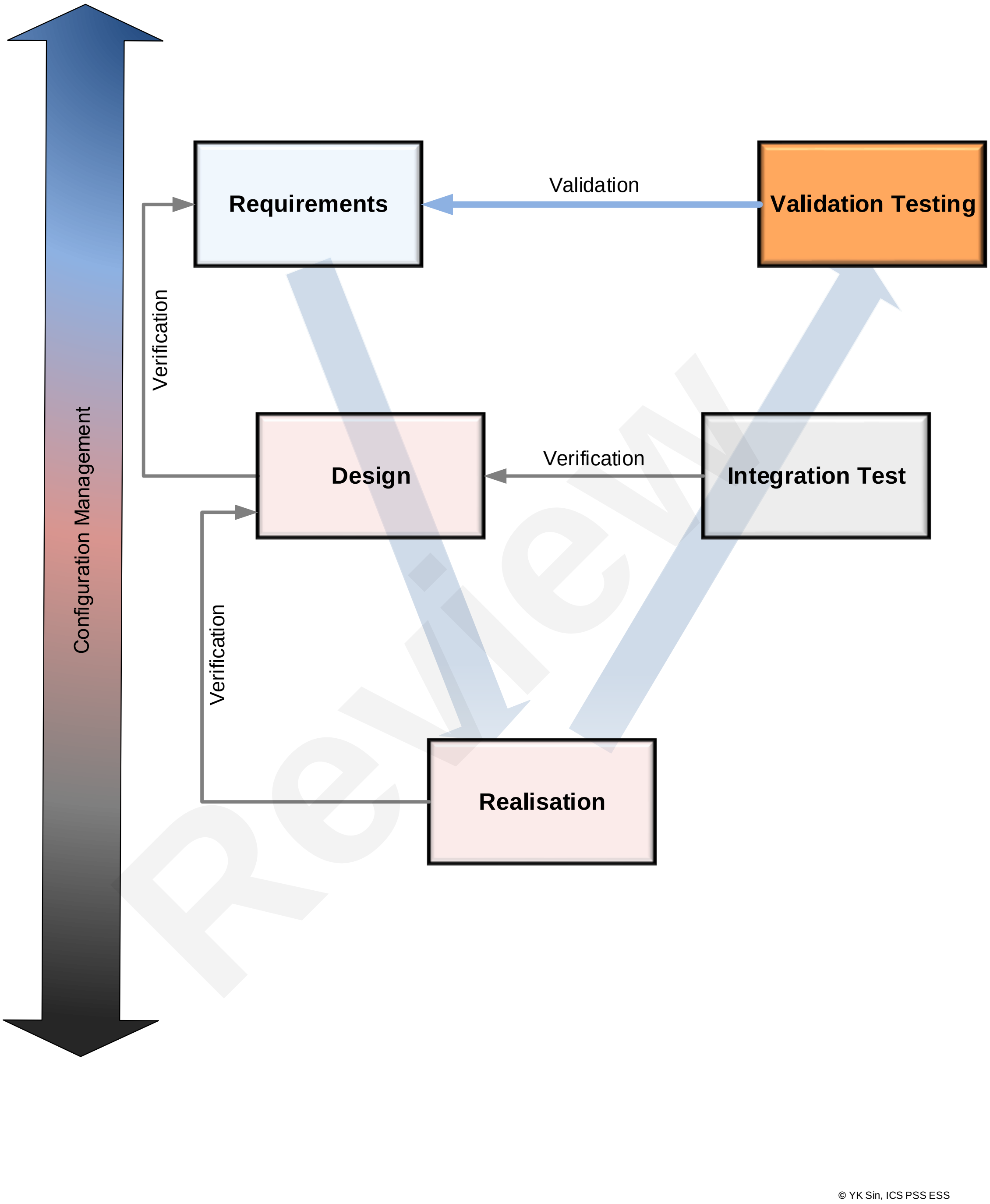

Figure 2: PSS0 V Lifecycle with Verification and Validation

4. MANAGEMENT APPROACH

This chapter talks about the schedule, the roles and the responsibilities required in the Verification and Validation Procedure. Tools, techniques and methodology are also part of this chapter.

4.1. Roles and Responsibilities

The roles are defined for PSS0.

The parties responsible for the various clauses are as follows:

Requirements Test Specification	Validator (VAL)
Verification and Testing	Verifier (VER)
Software/Hardware Integration	Designer (DES)
Validation	Validator (VAL)

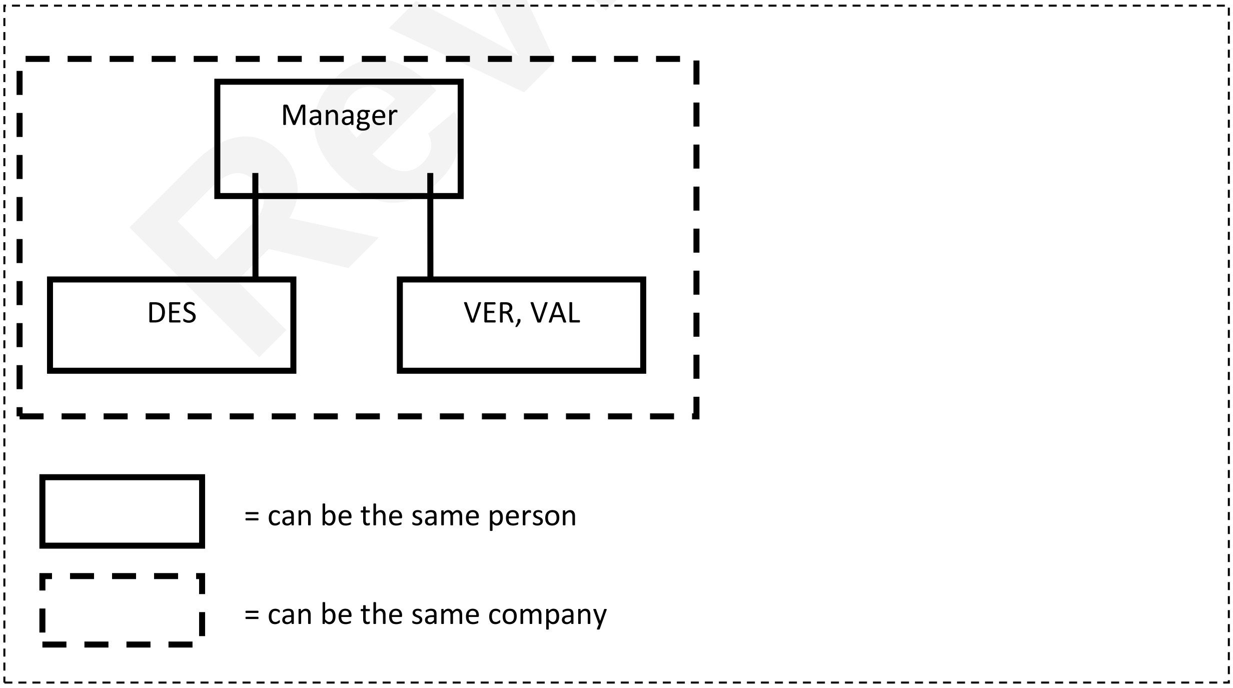

Figure 3: Independence of different roles

4.2. Tools, Techniques, and Methodology [4]

There is a list of techniques and measures for the verification and testing activities.

For Test Case Execution the following two points are highly recommended:

- Static Analysis
- Dynamic Analysis and Testing

The following technique is chosen from the set of static analysis:

- Technical design review

The following technique is chosen from the set of Dynamic Analysis and Testing:

- Test Case Execution from Boundary Value Analysis

For the Software/Hardware Integration there is also one list with techniques and measures. Functional and Black-box Testing is highly recommended in this list.

For the Functional and Black-box Testing are use the highly recommended techniques:

- Boundary Value Analysis
- Equivalence Classes and Input Partition Testing

For Software Validation activities

- Functional and Black-box Testing

are highly recommended.

4.3. Level of test

These are the test levels as defined by ISTQB [4]:

Component testing - (Also known as unit, module or program testing) searches for defects in, and verifies the function of, software modules programs, objects, classes, etc., that are separately testable. It may be done in isolation from the rest of the system, depending of the context of the development life cycle and the system. Stubs, drivers and simulators may be used.

Integration testing - Tests interfaces between components, interactions with different parts of a system, such as the operating system, file systems and hardware, and interfaces between systems. Systematic integration strategies may be based on system architecture (such as top-down and bottom-up), functional tasks, transaction processing sequences or some other aspect of the system or components. In order to ease fault isolation and detect defects early, integration should normally be incremental

System testing - Concerned with the behaviour of a whole system/product. In system testing, the test environment should correspond to the final target or production environment as much as possible in order to minimize the risk of environment-specific failures not being found in testing. System testing may include tests based on risks and/or on requirements specifications, business processes, use cases, or other high level text descriptions or models of system behaviour, interactions with the operating system, and system resources.

Acceptance testing - Often the responsibility of the customers or users of a system; other stakeholders may be involved as well. The goal in acceptance testing is to establish confidence in the system, parts of the system or specific non-functional characteristics of the system. Finding defects is not the main focus in acceptance testing. Acceptance testing may assess the system's readiness for deployment and use, although it is not necessarily the final level of testing.

5. VERIFICATION

After each verification activity a verification report shall be produced stating either that the verification object has passed the verification or the reasons for the failures.

The verification reports shall address the following:

- items which do not conform to the
  - Safety Requirements Specification [1],
  - System Architecture Specification,
  - Software Module Design Specification,
  - Hardware Design Specification,
- modules, data, structures and algorithms poorly adapted to the problem;
- detected errors or deficiencies;
- the identity and configuration of the items verified

Testing which is not fully documented and is performed by the designer prior to verification shall not be regarded as part of the verification.

5.1. Types of verification

For each lifecycle phase a corresponding verification activity exists.

Any modification or change to the system shall be subject to an impact study which shall identify all modules impacted and the necessary re-verification activities. For change management please refer to PSS Configuration Management Plan [5].

5.1.1. Requirements Specification Verification

Responsible	VER
Input Data	Safety Requirements Specification [1]
Output Data	Requirements Specification Verification Report
Technique	Technical Design Review
Verification address	▪ Adequacy of the Requirements Specification ▪ Internal consistency
Approval	Manager

The results shall be recorded in Requirements Specification Verification Report.

5.1.2. Software and Hardware Architecture Verification

Responsible	VER
Input Data	Software Architecture Hardware Architecture System Architecture Requirements Specification
Output Data	Software Architecture Verification Report Hardware Architecture Verification Report
Technique	Walkthrough/Design Review
Verification address	▪ Adequacy of the Software Architecture ▪ Adequacy of the Hardware Architecture ▪ Consistency ○ internal ○ with PSS0 Requirements Specification ▪ Completeness
Approval	Manager

The finished Software and Hardware Architecture Verification shall be reviewed against the System Architecture and the PSS Requirement Specification.

The Software and Hardware Architecture verification shall be compared by the following parameters

- Interfaces of each hardware unit with subsystem software interfaces
- Interconnection to other hardware units with communication requirements

5.1.3. Hardware Design Test (Component testing)

Responsible	VER
Input Data	PSS0 Requirement Specification Hardware Design Hardware Documentation HW Config TIA Portal
Output Data	Hardware Design Test Report
Approval	DES, Manager

Produced the Hardware Configuration (HW Config) from TIA Portal according to electrical drawing.

Each sensor and actuator need to be tested and trigger to ensure items install without defect.

5.1.4. Software Module Design Test (Component testing)

Responsible	VER
Input Data	PSS0 Requirement Specification Software Module Design Software Documentation
Output Data	Software Module Design Test Report
Approval	DES, Manager

During the Software Module Design Test, the implementations of the single modules of the subsystems are checked for correct functionality. A simulation of the activation and deactivation according to requirements will be checked.

5.1.5. Acceptance Test (FAT) (System testing)

Responsible	VER
Input Data	PSS0 Requirement Specification System Architecture Hardware Documentation Software Documentation
Output Data	System Integration Test Report
Approval	DES, Manager

The main activity is the verification of failure scenarios. The relevant failure scenarios are part of the requirements specification.

Place:

- Pre-FAT: in the laboratory at Lund, Sweden and on site.
- FAT: on site (Gothenburg)
-

Methods:

- ○ Test case execution from Boundary Value Analysis
- ○ Equivalence classes and input partition testing

Equipment:

- with the final hardware,
- without the original sensor and actuators,
- with the hardware configuration
- with the original connections
- with test board (if any)
-

Tools:

- Engineering station (Notebook, Computer) with TIA Portal
- Electrical tester or oscilloscope

Steps:

- create connections between the hardware modules (control panels)
- create connections to the external power supply
- integration of the software (HW Config)
- testing of the communication, incl. failure injection te

5.1.6. Loop – Check

Responsible	VER
Input Data	PSS0 Requirement Specification Hardware Documentation Software Documentation Loop – Check Concept
Output Data	Loop – Check Report
Approval	Manager

During Loop – Check the installed system is checked for the right wiring.

Points to verify during Loop check are:

- Interconnections from one unit to the assigned unit
- Correct evaluation at the I/O card (Current vs. Voltage)
- Correct protocol for network connections
- Correct polarity

5.1.7. Site Acceptance Test (SAT) (System testing)

Responsible	VER
Input Data	PSS0 Requirement Specification Hardware Documentation Software Documentation Loop – Check Concept System Integration Test Report

Output Data	Site Integration Test Report
Approval	Manager

Purpose of SAT is to confirm the installation of the equipment with complete software integrated of Input / Output.

5.1.8. Final Integration Test (FIT) (Acceptance testing)

Responsible	VER
Input Data	PSS0 Requirement Specification Hardware Documentation Software Documentation
Output Data	Final Integration Test Report
Approval	DES, Manager

Final Integration Test is similar to FAT with full systems ready. The concept for the FAT can be used for Final Integration Test but the focus of this test is to re-verify the parts, changed during commissioning and the interfaces to systems that are not part of PSS0.

Failure scenarios have to be evaluated for noncertified components like e.g. relays. For dangerous failure scenarios, a measure for detecting this failure has to be specified during commissioning if changes are applied. These measures must be verified for correctness during Final Integration Test.

Place: on site (Lund)

Methods:

- Functional and Black box Testing
    - Test case execution from Boundary Value Analysis
    - Equivalence classes and input partition testing

Equipment:

- with the final hardware (cubicles),
- with the original sensor and actuators,
- with the updated software version
- with the original connections between the cubicles according to electrical drawing

Tools:

- Engineering station (Notebook, Computer) with TIA Portal .
- Electrical tester or oscilloscope

Steps:

- Testing of the function, incl. failure injection test with focus on changes after the FAT.

6. VALIDATION

The objective is to analyse and test the integrated system to ensure compliance with the ESS Process for Validation [2] and Safety Requirements Specification [1].

The Validator shall check that the verification process is complete.

The Validation Plan shall identify the steps necessary to demonstrate the consistency of:

- Safety Requirements Specification [1],
- System Architecture Specification,
- Software and Hardware Architecture Specification,
- Software Module Design Specification,
- Hardware Design Specification,
- Corresponding test concepts if any.

Validation activities:

- Checking of completeness and capability of the verification reports.
- Checking the state of the configuration list.
- Checking the state of the Change Management
- The system will be tested against the Requirements Test Specification
    - under normal operation,
    - under failure conditions.

A Validation Report shall be produced as follows:

- documentation in chronological form of the validation activities;
- the version of the specification for the overall safety requirements being used;
- tools and equipment used, along with calibration data;
- the results of the validation activities;
- configuration identification of the item under test, the procedures applied and the test environment;
- Discrepancies between expected and actual results.

6.1. Requirement Validation

Responsible	VAL
Input Data	PSS0 Requirement Specification PSS0 Verification and Validation Plan Validation Concept
Output Data	Requirement Validation Report
Approval	Independent Safety Assessor depending on SIL

During the Validation the whole system is validated against the PSS0 Requirement Specification.

The functional tests, required for validation, are carried out during Final Integration Test

Aim of the validation phase is to make sure that

- Every Requirement from PSS0 Requirements Specification has been tested adequately
- all necessary tests have been done
- all necessary tests are documented adequately

During the Validation all documentation of each stage of the development are checked for actual status and for consistency to the relevant design document. The validation ensures that the successive stage starts with the right documents and thus leads to an effective processing of the development.

7. DOCUMENTATION AND REPORTING

After each verification activity a verification report shall be produced stating either the document and the system has passed the verification or the reasons for the failures.

The verification reports shall address the following:

- items which do not conform to the
  - Safety Requirements Specification [1],
  - System Architecture Specification ,
  - Software Architecture Specification,
  - Hardware Architecture Specification'
  - Software Module Design Specification,
  - Hardware Station Design Specification ,
- modules, data, structures and algorithms poorly adapted to the problem;
- detected errors or deficiencies;

After the validation activity a validation report shall be produced stating either that the system has passed the validation or the reasons for the failures.

The Validation Report shall document the identity and configuration of all of the following items:

- hardware and software used;
- equipment used;
- equipment's calibration;
- simulation models used;
- discrepancies found;
- corrective actions performed.

All requirements mentioned in the list before which have a functional behaviour, are not listed in the validation report, but in the Final Integration Test Report.

Test cases and their results shall be documented for subsequent analysis and independent assessment as required by the safety integrity level.

8. REFERENCES

[1] ESS-0238059, Safety Requirements Specification

[2] ESS-0015098, ESS Process for Validation

[3] IEC61508: 2010 Functional Safety of electrical/electronic/programmable electronic safety-related systems

[4] ISTQB, https://www.istqb.org/downloads/glossary.html

[5] ESS-0058389, PSS Configuration Management Plan

[6] ESS-0015098, ESS Process for Verification

DOCUMENT REVISION HISTORY

Revision	Reason for and description of change	Author	Date
1	First issue	Yong Kian Sin	2018-01-29