System Integration Test

**Change History**

| **Version** | **Date** | **Change Description / Reason** | **Author** |
| --- | --- | --- | --- |
| 1.0 | 2017-12-08 | Initial version based on Hella Process (State Avenue) | Hae-Min, Woo |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

[1. Purpose Description 5](#_Toc500323881)

[1.1 Purpose 5](#_Toc500323882)

[1.2 Main Description 5](#_Toc500323883)

[1.3 Result of Process 5](#_Toc500323884)

[2. Major Roles Acting in this Process 5](#_Toc500323885)

[2.1 Refinement of the major roles 6](#_Toc500323886)

[3. Process 6](#_Toc500323887)

[3.1 Process Input – Output Definition 6](#_Toc500323888)

[3.2 Introducing Aspects 7](#_Toc500323889)

[3.3 Phase Model 7](#_Toc500323890)

[3.4 Process Flow Chart 9](#_Toc500323891)

[3.5 Test Planning Phase 10](#_Toc500323892)

[3.5.1 Task 1: Determine integration strategy 11](#_Toc500323893)

[3.5.2 Task 2: Temporal planning integration steps 11](#_Toc500323894)

[3.5.3 Task 3: Specifying integration environment 11](#_Toc500323895)

[3.5.4 Task 4: Determine framework conditions 12](#_Toc500323896)

[3.5.5 Task 5: Determine content of test level 12](#_Toc500323897)

[3.5.6 Task 6: Develop test strategy 13](#_Toc500323898)

[3.5.7 Task 7: Organize test project 19](#_Toc500323899)

[3.5.8 Task 8: Review and release test plan 20](#_Toc500323900)

[3.6 Test Control Phase 20](#_Toc500323901)

[3.6.1 Task 9: Manage test team and project 21](#_Toc500323902)

[3.6.2 Task 10: Monitor and communicate test progress 21](#_Toc500323903)

[3.7 Test Infrastructure Phase 22](#_Toc500323904)

[3.7.1 Task 11: Specify infrastructure 22](#_Toc500323905)

[3.7.2 Task 12: Realize infrastructure 22](#_Toc500323906)

[3.7.3 Task 13: Putting infrastructure into operation 22](#_Toc500323907)

[3.8 Test Analysis and Design Phase 23](#_Toc500323908)

[3.8.1 Task 14: Analyze test basis 23](#_Toc500323909)

[3.8.2 Task 15: Specify abstract test cases 23](#_Toc500323910)

[3.8.3 Task 16: Specify concrete test cases 24](#_Toc500323911)

[3.8.4 Task 17: Ensure traceability and consistency 24](#_Toc500323912)

[3.8.5 Task 18: Review and release test specification 25](#_Toc500323913)

[3.9 Test Implementation and Execution Phase 25](#_Toc500323914)

[3.9.1 Task 19: Implementing integration environment 25](#_Toc500323915)

[3.9.2 Task 20: Integration of modules or system parts 25](#_Toc500323916)

[3.9.3 Task 21: Logging integration results 26](#_Toc500323917)

[3.9.4 Task 22: Analyze integration results 26](#_Toc500323918)

[3.9.5 Task 23: Implement test automation 26](#_Toc500323919)

[3.9.6 Task 24: Set up test execution 27](#_Toc500323920)

[3.9.7 Task 25: Execute tests 27](#_Toc500323921)

[3.9.8 Decision 1: Completed test run? 27](#_Toc500323922)

[3.10 Test Evaluation and Report Phase 28](#_Toc500323923)

[3.10.1 Task 26: Analyze test results 28](#_Toc500323924)

[3.10.2 Task 27: Review test results 28](#_Toc500323925)

[3.10.3 Task 28: Assess and communicate maturity of test object 29](#_Toc500323926)

[3.11 Test Closure Phase 29](#_Toc500323927)

[3.11.1 Task 29: Analyze test experience 29](#_Toc500323928)

[3.11.2 Task 30: Archiving test environment 29](#_Toc500323929)

[4. Changes, References, Appendix, Terms 30](#_Toc500323930)

[4.1 References 30](#_Toc500323931)

[4.2 Template 30](#_Toc500323932)

# Purpose Description

## Purpose

The purpose of the system integration test process is to integrate the system elements to produce an integrated system that will satisfy the System Architectural Design and the customers’ expectations expressed in the system requirements.

## Main Description

The system integration test is the next test level after the software test level with the goal to ensure the system architecture design and the customers’ expectations. Focus of the system integration test level is

* System requirements specification
* Interaction and communication between the system elements (e.g. integrated software, hardware items, mechanical items)
* Functionality of the entire system
* Non-functional requirements
* Hardware parts

The following problems can be found:

* Functional and non-functional failures
* Problems between software and hardware
* Conceptual failure

## Result of Process

As a result of successful implementation of this process:

1. A system integration and system integration test strategy is developed for system elements consistent with the system architectural design according to the priorities and categorization of the system requirements;
2. A test specification system integration test is developed to verify compliance with the system architectural design, including the interfaces between system elements;
3. An integrated system is integrated as defined by the integration strategy;
4. The integrated system is verified using the test cases;
5. Test results of system integration testing are recorded;
6. Consistency and bilateral traceability are established between System Architectural Design and test specification system integration test including test cases; and
7. A regression strategy is developed and applied for re-testing the system elements when changes are made;

NOTE 1: The test specification for system integration includes the test design specification, test procedure specification and test case specifications.

NOTE 2: The test results for system integration include the test logs, test incident reports and test summary reports.

# Major Roles Acting in this Process

The following table gives an overview of the major roles involved with a short explanation. The definition of the roles can be found in the supplement A of MHE-PE-01.

|  |  |
| --- | --- |
| **Role** | **Contribution and Responsibilities** |
| Test Manager | Plans, coordinates, and controls all test activities across the test levels and creates the Master Test Plan (8316) and the SW Module Test Plans (8317). In large projects, this role should be executed by a team of “Domain” Test Managers, with one member acting as overall responsible. If there is no explicit Test Manager in a project (or in a domain in case of a test manager team), then the tasks of this role must be taken over by another member of the team and the corresponding project manager is responsible for decisions. |
| System Integration Tester | Analyzes the test basis, creates the test specification, the test scripts, and carries out the test including the corresponding documentation. |
| System Integrator | Only for Software and System Integration Test. Specifies integration strategy, determines the requirements for and maintains the integration environment, and integrates software items into larger assemblies. |
| Project Manager | In this context relevant for all software development aspects like integration, System module classification, System schedule, etc. |
| Safety Manager | In case of safety related system responsible to support the test team, review / decide work products concerning their safety related content. |

Due to the generic character of this process description, the assignments of the generic roles to the test level specific roles are defined at cf. 5.2 Mapping of roles. Additionally, it is defined there, who may take over the role of the Test Manager at each respective test level. Should the Test Manager role be assumed by another role in personal union, then this must be documented in the (SW or HW) Project Manual (8314 or 8622).

## Refinement of the major roles

The following role refinement is made locally in this process description to improve the mapping into this process description of the specific work focuses and skills of the “System Integration Tester” role (MHE-PE-01, Supplement A):

|  |  |
| --- | --- |
| **Role** | **Contribution and Responsibilities** |
| System Integration Test Designer | Tester with very good experiences with test design techniques and extensive knowledge of the test object responsible for designing the test cases (creates the test specification). |
| System Integration Tester | Carries out the tests as defined in the test specification and draws up the reports including measurements. |
| System Integration Test Automation Engineer | Developer with know-how to automate test activities; implementation of modular, maintainable test automation solutions, Tester dealing with the provision, version assignment, archiving, and administration of the necessary test environment. |

# Process

## Process Input – Output Definition

The following process input is required:

|  |  |
| --- | --- |
| **Process input** | **From supplier** |
| Master Test Plan (especially chapter 4 “Master test strategy”, if implemented) | Test Manager |
| Safety Plan (in case of safety related system) | Safety Manager |
| Customer Requirements Specification  System Requirements Specification | OEM  System Analyst |
| System Architecture Specification | System Architect |
| Software Design (Definition of all software modules and interfaces) | Software Architect |
| Hardware Design (Definition of all hardware modules and interfaces) | Hardware Developer |
| Attributes for economic risk classification on system integration level | HW / SW Project Manager  System Architect  HW / SW Analyst |
| Release Plan | SW-Project Manager |
| Integrated system elements (test object) | System Integrator |

The following process output is produced:

|  |  |
| --- | --- |
| **Process output** | **To customer** |
| System Integration (Test) Plan | SW + HW Project Manager  Safety Manager |
| Review Checklist System Integration (Test) Plan | System Integration Test  Designer |
| System Integration Test Specification | System Integration Tester |
| Review Checklist System Integration Test Specification | System Integration Test  Designer |
| System Integration Test Report | Project Manager  Safety Manager |
| Test Completion Report | Project Manager |

## Introducing Aspects

None

## Phase Model

Following the fundamental test process the process description distinguishes five phases: test planning and controlling, test analysis and design, test implementation and execution, test evaluation and report and test closure.



Figure 1 : Phase model of the test process

## Process Flow Chart





## Test Planning Phase

The objective of the first phase Test planning is to plan the test activities for a system integration test level and document the planning decisions like how, by whom, what with and when what test activities are carried out in the System Integration Test Plan.

### Task 1: Determine integration strategy

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | System Architecture Specification  System Release Plan | | | | | |
| Output | | System Integration Plan | | | | | |
| D: | PM | E: | System Integrator | S: | Test Manager,  System Architect,  System Developer,  Safety Manager | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to define an integration strategy concerning order and proceeding for the integration of system parts (e.g. hardware, mechanics, and software) into larger assemblies, producing integrated items consistent with the design and consistent with the release strategy. The integration strategy has to fulfill both the integration and test aspects. The integration strategy must contain an integration exit criteria.

Necessary inputs of this activity are the design description of the test object and the requirements on the test object, as well as the release and System Release Plan (8305).

For safety related systems, the Safety Plan as well as the Guidelines to apply test methods conform to ISO26262 must be considered to determine the reasonable integration strategy.

The integration strategy should be developed according to priorities and categorization (based on the economic risk classification, safety classification as well as release relation) of the requirements.

The details of the integration strategy are documented in the software Integration Plan, chapter 1.3: Integration strategy and chapter 1.4: Definition of the Integration Exit Criteria.

The planning of the system integration shall describe the steps for integrating

* The developed hardware and the developed software
* The individual elements incorporated in the system in accordance with the system design

### Task 2: Temporal planning integration steps

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | Build List | | | | | |
| D: | PM | E: | System Integrator | S: | Test Manager | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to define a time schedule for the integration.

Inputs of this activity are the release plan and the integration strategy described in System Integration Plan.

The time planning of the integration needs to be documented in the System Integration Plan, chapter 1.6: Scheduling of the Integration Steps. If MS Project is used, then a link to the appropriate MS Project file must exist in the System Integration Plan.

### Task 3: Specifying integration environment

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | System Integration Plan | | | | | |
| D: |  | E: | System Integrator | S: | System Test Automation Engineer | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to specify the required software integration environment (e.g. build environment) with the necessary degree of detail.

The details for the integration environment are documented in the System Integration Plan, chapter 1.5: Integration Environment.

### Task 4: Determine framework conditions

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Master Test Plan  Safety Plan | | | | | |
| Output | | System Integration Test Plan | | | | | |
| D: |  | E: | Test Manager | S: | PM,  Safety Manager | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to define the framework conditions and influencing factors for the test project from the stakeholder point of view. This includes project-specific particulars like release milestones and objectives for releases as well as test object project-specific particulars. It also includes the scope of test content in demarcation with the tests executed by the customer.

For safety related systems, the Safety Plan as well as the Guidelines to apply test methods conform to ISO26262 must be considered to determine the specific framework conditions.

The result of this task must be documented in the System Integration Test Plan, chapter 2: Introduction.

### Task 5: Determine content of test level

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Input documents  System Integration Plan | | | | | |
| Output | | System Integration Test Plan | | | | | |
| D: | Test Manager | E: | System Integration Test Designer | S: | PM, Architect, Analyst or Developer,  Safety Manager | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to determine the test content of the test of this test (level) from the test object point of view. This includes analyzing the project-specific particulars, to determine the reference documents as well as the test object, and to define the scope of testing, in terms of “Features to be tested / Features not to be tested.

The test planning shall determine the content of the work products to be tested. This is done in this process task by

* Defining the test object of this test level (The test object for the first phase, integration of the hardware and software, is the integrated hardware and software, the test object of the second phase, integration of the elements that comprise an item to form a complete system, is the integrated system related to our demarcation of the system)
* Analyzing the project- and test object-specific particularities
* Determining and referencing the specification documents

The following information shall be available and referenced in detail:

* Safety goals, functional safety concept, and technical safety concept shall be determined and referenced as project-specific particularities and shall be used as input for the test strategy
* Vehicle architecture can be determined and referenced as project-specific particularities and can be used as input for the test strategy
* Master Test Plan shall be determined and referenced as project-specific particularities and shall be used as input for the test strategy and in the temporal planning
* Hardware-software interface specification and system design specification shall be determined and referenced as specification documents and shall be used as input for the creation of test cases
* Safety Plan shall be determined and referenced as project-specific particularities and shall be used as input for the test strategy

The following test goals are addressed using the test methods defined in the Safety Verification Plan:

* The correct implementation of functional safety and technical safety requirements
* The correct functional performance, accuracy and timing of safety mechanisms
* The consistent and correct implementation of interfaces
* The effectiveness of a safety mechanism's diagnostic or failure coverage
* The level of robustness

For safety related systems the specific content of test must be determined

The result of this task must be documented in the System Integration Test Plan, chapter 3: Test objects.

### Task 6: Develop test strategy

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Master Test Plan  Safety Plan | | | | | |
| Output | | SW Integration Test Plan | | | | | |
| D: |  | E: | Test Manager | S: | SW Integration Test  Designer,  Safety Manager | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The objective of the SW Integration test strategy is to find errors as early as possible, with as little cost as possible, and to test the test object with an appropriate breadth and depth which achieves the desired quality and compliance to the requirements over the complete range of allowed application parameter combinations (for non integration test level) or which verify that only allowed application parameter combinations can influence functionality and that wrong parameter or parameter combinations are failsafe (for integration test level). The test strategy has to be consistent with the release strategy.

In case of an integration test level the test strategy has to identify test steps according to the order of integration defined in the integration strategy.

Based on a suitable granularity of the shares of the test object and on already defined attributes (“Extent of economic damage” caused by implemented defects, the “Implementation probability of defects”, and the “Frequency of use”, e.g. documented using 8325), the Test Manager has to determine the Economic risk classes of these shares before determining the test strategy.

The test strategy includes the definition of

* Suitable methods and techniques to be used and their coverage rates, based on the results of a risk assessment.
* Criteria for test exit, test interruption and resumption.
* A strategy concerning test automation
* A re-test and regression test /verification strategy

For safety related systems, the Guidelines to apply test methods conform to ISO26262 must be considered to determine the reasonable test strategy. The tests of the safety related functions are to prioritize appropriately high in the case of conflicts regarding a complete execution of the planned tests.

The test strategy should be developed according to priorities and categorization (based on the economic risk classification, safety classification as well as release relation) of the requirements.

The verification specification (spanned over the different verification level) shall select and specify the methods to be used for the verification, and shall include:

1. Review or analysis checklists; or
2. Simulation scenarios; or
3. Test cases, test data and test objects.

Although the test process is responsible for the planning and execution of the 3rd topic, the development of the test strategy considers the planned verification activities of the development phase (1st and 2nd topic) while determining the necessary breadth and depth of testing to ensure an appropriate overall verification depth.

The following information shall be used for the determination of the integration and test strategy:

* System design specification
* Functional safety concept
* Technical safety concept
* Item integration and testing plan

The integration and test strategy shall provide evidence that the test goals are covered sufficiently.

The integration and test strategy shall ensure that open issues from hardware-software verifications are addressed.

For ASIL C and D: The hardware-software interface requirements shall be tested with appropriate coverage, with consideration to the ASIL or a rationale shall be given that no issues with respect to the hardware software interface remain.

If the system uses configurations or calibration data the verification at the system (or vehicle level) shall provide evidence of compliance with safety requirements for each configuration at implementation level or for every configuration that is intended for serial production. (If a complete verification of each configuration at the system (or vehicle level) is not feasible, then a reasonable subset might be selected.)

Each functional and technical safety requirement shall be verified (if applicable by test) at least once in the complete integration test level.

(A common practice is to verify a safety requirement at the next higher level of integration to which it has been specified, safety anomalies identified during integration testing are reported)

The following methods (Table 1, Table 7 - Table 11) shall to be applied at the system integration test and at the system test level. The assessment of the completeness of the application of these methods regarding the attainment of the test aim must be carried out comprehensive over these two test levels. (See Remark)

**General methods**

To enable the appropriate specification of test cases for the system test as well as for the system integration test, test cases shall be derived using an appropriate combination of methods, as listed in Table 1, and by considering the integration level.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Analysis of requirements | ++ | ++ | ++ | ++ |
| 1b | Analysis of external and internal interfaces | + | ++ | ++ | ++ |
| 1c | Generation and analysis of equivalence classes for hardware-software  integration | + | + | ++ | ++ |
| 1d | Analysis of boundary values | + | + | ++ | ++ |
| 1e | Error guessing (or Explorative testing) a | + | + | ++ | ++ |
| 1f | Analysis of functional dependencies | + | + | ++ | ++ |
| 1g | Analysis of common limit conditions, sequences, and sources of dependent failures | + | + | ++ | ++ |
| 1h | Analysis of environmental conditions and operational use cases | + | ++ | ++ | ++ |
| 1i | Analysis of field experience | + | ++ | ++ | ++ |
| a An error guessing test uses expert knowledge and data collected through lessons learned to anticipate errors in the test object. Then a set of tests along with adequate test facilities is designed to check for these errors. Error guessing is an effective method given a tester who has previous experience with similar test objects.  MHE remark: Alternative to Error guessing the method Explorative testing can be applied. | | | | | |

Table 1 Methods for deriving test cases for integration testing

**Hardware-software integration and testing related methods**

The correct implementation of the technical safety requirements at the hardware-software level shall be demonstrated using feasible test methods given in Table 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Requirements-based test a | ++ | ++ | ++ | ++ |
| 1b | Fault injection test b | + | ++ | ++ | ++ |
| 1c | Back-to-back test c | + | + | ++ | ++ |
| a A requirements-based test denotes a test against functional and non-functional requirements.  b A fault injection test uses special means to introduce faults into the test object during runtime. This can be done within the software via a special test interface or specially prepared hardware. The method is often used to improve the test coverage of the safety requirements, because during normal operation safety mechanisms are not invoked.  c A back-to-back test compares the responses of the test object with the responses of a simulation model to the same stimuli, to detect differences between the behaviour of the model and its implementation. | | | | | |

Table 2: The correct implementation of technical safety requirements at the hardware-software level

The correct functional performance, accuracy and timing of the safety mechanisms at the hardware-software level shall be demonstrated using feasible test methods given in Table 3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Back-to-back test a | + | + | ++ | ++ |
| 1b | Performance test b | + | ++ | ++ | ++ |
| a A back-to-back test compares the responses of the test object with the responses of a simulation model to the same stimuli, to detect differences between the behavior of the model and its implementation.  b A performance test can verify the performance (e.g. task scheduling, timing, power output) in the context of the whole test object, and can verify the ability of the intended control software to run with the hardware. | | | | | |

Table 3: The correct functional performance, accuracy and timing of safety mechanisms at the hardware software level

The consistent and correct implementation of the external and internal interfaces at the hardware-software level shall be demonstrated using feasible test methods given in Table 4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Test of external interfaces a | + | ++ | ++ | ++ |
| 1b | Test of internal interfaces a | + | ++ | ++ | ++ |
| 1c | Interface consistency check a | + | ++ | ++ | ++ |
| a Interface tests of the test object include tests of analogue and digital inputs and outputs, boundary tests and equivalence-class tests to completely test the specified interfaces, compatibility, timings and other specified ratings for the test object. Internal interfaces of an ECU can be tested by static tests for the compatibility of software and hardware as well as dynamic tests of Serial Peripheral Interface- (SPI) or Integrated Circuit- (I2C) communications or any other interface between elements of an ECU. | | | | | |

Table 4: The consistent and correct implementation of external and internal interfaces at the hardware software level

The effectiveness of the hardware fault detection mechanisms' diagnostic coverage at the hardware-software level, with respect to the fault models, shall be demonstrated using feasible test methods given in Table 5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Fault injection test a | + | + | ++ | ++ |
| 1b | Error guessing (or Explorative testing) b | + | + | ++ | ++ |
| a A fault injection test uses special means to introduce faults into the test object during runtime. This can be done within the software via a special test interface or specially prepared hardware. The method is often used to improve the test coverage of the safety requirements, because during normal operation safety mechanisms are not invoked.  b An error guessing test uses expert knowledge and data collected through lessons learned to anticipate errors in the test object. Then a set of tests along with adequate test facilities is designed to check for these errors. Error guessing is an effective method given a tester who has previous experience with similar test objects.  MHE remark: Alternative to Error guessing the method Explorative testing can be applied. | | | | | |

Table 5: The effectiveness of a safety mechanism's diagnostic coverage at the hardware-software level

The level of robustness of the elements at the hardware-software level shall be demonstrated using feasible test methods given in Table 6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Resource usage test a | + | + | + | ++ |
| 1b | Stress test b | + | + | + | ++ |
| a A resources usage test can be done statically, (e.g. by checking for code sizes or analyzing the code regarding interrupt usage, in order to verify that worst-case scenarios do not run out of resources), or dynamically by runtime monitoring.  b A stress test verifies the test object for correct operation under high operational loads or high demands from the environment. Therefore, tests under high loads on the test object, or with exceptional interface loads, or values (bus loads, electrical shocks etc.), as well as tests with extreme temperatures, humidity or mechanical shocks, can be applied. | | | | | |

Table 6: The level of robustness at the hardware-software level

**System integration and testing related methods**

The correct implementation of the functional and technical requirements at the system level shall be demonstrated using feasible test methods given in Table 7

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Requirement-based test a | ++ | ++ | ++ | ++ |
| 1b | Fault injection test b | + | + | ++ | ++ |
| 1c | Back-to-back test c | o | + | + | ++ |
| a A requirements-based test denotes a test against functional and non-functional requirements.  b A fault injection test uses special means to introduce faults into the test object during runtime. This can be done within the software via a special test interface or specially prepared hardware. The method is often used to improve the test coverage of the safety requirements, because during normal operation safety mechanisms are not invoked.  c A back-to-back test compares the responses of the test object with the responses of a simulation model to the same stimuli, to detect differences between the behaviour of the model and its implementation. | | | | | |

Table 7: The correct implementation of functional safety and technical safety requirements at the system level

The correct functional performance, accuracy and timing of the safety mechanisms at the system level shall be demonstrated using feasible test methods given in Table 8

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Back-to-back test a | o | + | + | ++ |
| 1b | Performance test b | o | + | + | ++ |
| a A back-to-back test compares the responses of the test object with the responses of a simulation model to the same stimuli, to detect differences between the behaviour of the model and its implementation.  b A performance test can verify the performance (e.g. task scheduling, timing, power output) in the context of the whole test object, and can verify the ability of the intended control software to run with the hardware. | | | | | |

Table 8: The correct functional performance, accuracy and timing of safety mechanisms at the system level

The consistent and correct implementation of the external and internal interfaces at the system level shall be demonstrated using feasible test methods given in Table 9

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Test of external interfaces a | + | ++ | ++ | ++ |
| 1b | Test of internal interfaces a | + | ++ | ++ | ++ |
| 1c | Interface consistency check a | o | + | ++ | ++ |
| 1d | Test of interaction/communication b | ++ | ++ | ++ | ++ |
| a An interface test of the system includes tests of analogue and digital inputs and outputs, boundary tests, and equivalence-class tests, to completely test the specified interfaces, compatibility, timings, and other specified characteristics of the system. Internal interfaces of the system can be tested by static tests, (e.g. match of plug connectors) as well as by dynamic tests concerning bus communications or any other interface between system elements.  b A communication and interaction test includes tests of the communication between the system elements, as well as between the system under test and other vehicle systems during runtime, against the functional and non-functional requirements. | | | | | |

Table 9: The consistent and correct implementation of external and internal interfaces at the sysetm level

The effectiveness of the safety mechanisms' failure coverage at the system level shall be demonstrated using feasible test methods given in Table 10

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Fault injection test a | + | + | ++ | ++ |
| 1b | Error guessing (or Explorative testing) b | + | + | ++ | ++ |
| 1c | Test derived from field experience c | o | + | ++ | ++ |
| a A fault injection test uses special means to introduce faults into the system. This can be done within the system via a special test interface, specially prepared elements, or communication devices. The method is often used to improve the test coverage of the safety requirements, because during normal operation safety measures are not invoked.  b An error guessing test uses expert knowledge and data collected through lessons learned and field experience to anticipate errors in the system. Then a set of tests along with adequate test facilities is designed to check for these errors. Error guessing is an effective method given a tester who has previous experience with similar systems.  MHE remark: Alternative to Error guessing the method Explorative testing can be applied.  C ?? | | | | | |

Table 10: The effectiveness of a safety mechanism's failure coverage at the system level

The level of robustness at the system level shall be demonstrated using feasible test methods given in Table

11.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Resource usage test a | o | + | ++ | ++ |
| 1b | Stress test b | o | + | ++ | ++ |
| 1c | Test for interference resistance and robustness under certain environmental conditions c | ++ | ++ | ++ | ++ |
| a At the system level resource usage testing is usually performed in dynamic environments (e.g. lab cars or prototypes). Issues to test include power consumption and bus load.  b A stress test verifies the correct operation of the system under high operational loads or high demands from the environment. Therefore, tests under high loads on the system, or with extreme user inputs or requests from other systems, as well as tests with extreme temperatures, humidity or mechanical shocks, can be applied.  c A test for interference resistance and robustness, under certain environmental conditions, is a special case of stress testing. This includes EMC and ESD tests (e.g. see [2], [3]) | | | | | |

Table 11: The level of robustness at the system level

The verification planning shall define the pass and fail criteria for testing. This is defined in this process task by

* Defining the test exit criteria

The verification planning shall define the actions to be taken if anomalies are detected. This is defined in this process task by

* Defining particularities of the incident handling

The verification planning shall define the actions to be taken after changes of the system. This is done in this process task by defining the re- and regression test strategy (A regression strategy specifies how verification is repeated after changes have been made to the item or element. Verification can be repeated fully or partially and can include other items or elements that might affect the results of the verification.)

### Task 7: Organize test project

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | System Integration Test Plan | | | | | |
| D: |  | E: | Test Manager | S: | PM, System Integration Test Designer,  System Integration Tester,  System Integration Test Automation Engineer,  Safety Manager | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to define and to set up the internal project structure for the test activities. This includes:

* To define and establish test organization, including role assignments, trainings needs, team meetings and team communication
* To estimate the test effort and setup the temporal planning (incl. the resource assignment to work packages)
* To define approach to ensure bilateral traceability between the requirements and the corresponding test cases.
* Determining project specific procedures as “How to’s” like:  
  - assignment of requirements to test level  
  - to define particularities of the incident handling
* To determine the document planning like test deliverables,
* To define the necessary test environment and the demands on this environment
* To define the necessary tools
* To define how to manage, to supervise, and to report the test progress. This includes determining whether additional people, in addition to those already defined by default, must be informed about the test status. Furthermore metrics should be applied as control measure.

For safety related systems, the Safety Plan, the Guidelines to apply test methods conform to ISO26262, and specific time schedules must be considered to determine the organization of the test project.

Additionally specific monitoring and communication needs must be determined.

The test planning shall determine the test environment and the tools used for testing, if applicable. This is done in this process task by

* Defining the demands on test environment

The results are documented in the System Integration Test Plan, chapter 5: Test organization and chapter 6: Test environment / infrastructure.

### Task 8: Review and release test plan

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | System Integration Test Plan | | | | | |
| D: | Test Manager,  Safety Manager | E: | Test Manager | S: | PM,  Quality Planner,  SW Integration Test  Designer , System Integration Tester,  Safety Manager | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim is to review and to release the System Integration Test Plan.

The result is documented in the Review Checklist System Integration Test Plan.

## Test Control Phase

As shown in Figure 1: Phase model of the test process the test control activities are running in parallel to the other phases of the test process. Control comprises the continuous monitoring of the test activities in comparison with the planning, the reporting of deviations or abortion, and the definition of counter measures in the case of deviations from the planned targets.

### Task 9: Manage test team and project

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | Schedule  Meeting Minutes  Review Master  PTC-IM Tasks | | | | | |
| D: |  | E: | Test Manager | S: | System Integration Tester,  System Integration Test Designer,  System Integration Test Automation Engineer | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to manage the test team and the test project in general and to determine counter measures in the case of deviations.

The results are documented in dependence of the managed aspects or the counter measures.

### Task 10: Monitor and communicate test progress

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | Test Metrics | | | | | |
| D: |  | E: | Test Manager | S: | System Integration Tester,  System Integration Test Designer, System Integration Test Automation Engineer | I: | PM,  Safety Manager |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

In this task, the test progress is assessed and test exit criteria are analyzed with regard to the actual test results. In contrast to “Task 26: Analyze test results”, the analysis of the test exit criteria is here characterized more by the Test Manager’s point of view like timing aspects and test budget.

The input of this task is the project schedule; test exit criteria defined in the System Test Plan and the results logged in the System Integration Test Report.

The states of the test progress are documented in the Project Cockpit Chart (8386).

Permanent changes to the test exit criteria must be documented in the System Integration Test Plan; temporary changes in the Integration Test Report.

The structural coverage metrics at software integration test level according to the recommended methods and the definition how to map these methods to software shares from the System Integration (Test) Plan, must be gathered and reported in this task.

If the achieved structural coverage is considered insufficient, either additional test cases shall be specified or a rationale shall be provided.

Example Analysis of structural coverage can reveal shortcomings in the requirement-based test cases, inadequacies in the requirements, dead code, deactivated code or unintended functionality.

## Test Infrastructure Phase

In every project, requirements for the test infrastructure arise which cannot be covered with the standard infrastructure. The activities which must be executed within the test project are taken into account in this phase. This includes small customizations of the existing solutions done by the test team themselves as well as the initiation of parallel projects (like e.g. development of a HiL system) which must be initiated and tracked by the project.

### Task 11: Specify infrastructure

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | System Integration Test Plan | | | | | |
| Output | |  | | | | | |
| D: |  | E: | System Integration Test Automation Engineer | S: | System Integration Test Designer,  Safety Manager | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

During the test planning phase, “Task 7: Organize test project” the requirements on the test environment were defined in the SW Integration Test Plan. The aim of this task is to specify the required test environment with the necessary degree of detail.

The test environment is documented in the System Integration Test Specification or in a separate document, depending on the content of the necessary infrastructure.

For safety related systems, specific demands on the test infrastructure must be determined.

The test planning shall determine the demands on the test environment, if applicable. The activities of Task 7: Organize test project are continued by specifying the environment in this and in the following tasks.

### Task 12: Realize infrastructure

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | |  | | | | | |
| D: |  | E: | System Integration Test Automation Engineer | S: |  | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

Aim of this task is to implement the required infrastructure by project - if the implementation will be done by the project team - or to initiate and control the realization outside of the project.

### Task 13: Putting infrastructure into operation

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | |  | | | | | |
| D: |  | E: | System Integration Tester | S: | System Integration Test Automation Engineer | I: | Test Manager |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to put the test infrastructure into operation in accordance with the specification, as defined in Task 11: Specify infrastructure.

The result is communicated to the Test Manager.

## Test Analysis and Design Phase

In the test analysis and design phase, the test basis is examined with regard to testability first and then the test cases are specified. The test cases are documented in the System Integration Test Specification.

### Task 14: Analyze test basis

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Input documents | | | | | |
| Output | | System Integration Test Specification | | | | | |
| D: |  | E: | System Integration Test Designer | S: | Architect, Analyst or  Developer | I: | PM,  Test Manager |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to analyze the test basis to see whether test cases can be derived from the documents.

If the test basis is not sufficient for the construction of test cases and for test execution, suitable measures must be initiated by the test manager.

The inputs to this task are the necessary documents defined in the test planning phase, “Task 5: Determine content of test (level)”. These reference documents compose the test basis.

The results shall be assigned with the documents, describing the test basis and summarized in the System Test Specification, chapter 3: Examination of the System Test Basis.

### Task 15: Specify abstract test cases

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Input documents | | | | | |
| Output | | System Integration Test Specification | | | | | |
| D: |  | E: | System Integration Test Designer | S: |  | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to specify the abstract test cases and to define their priorities. Additionally the used test design techniques according to the regulation of the test strategy must be documented.

The inputs to this task are the necessary documents defined in the test planning phase, “Task 5: Determine content of test (level)”.

The test case descriptions and the assignment of priorities of the test cases are documented in the System Integration Test Specification, chapter 4-6: Refinement of the Test Approach, Test Environment, and Specification of the Test Cases.

The specification of each test case shall include the following content:

* A unique identification,
* The reference to the version of the associated work product to be verified,
* The preconditions and configurations, (If a complete test of the possible configurations of a work product (e.g. variants of a system) is not feasible, a reasonable subset is selected (e.g. minimum or maximum functionality configurations of a system)).
* The environmental conditions, if appropriate, (Environmental conditions relate to the physical properties (e.g. temperature) of the surroundings in which the test is conducted or is simulated as part of the test.)
* The input data, their time sequence and their values, and
* The expected behavior which includes output data, acceptable ranges of output values, time behavior and tolerance behavior (When specifying the expected behavior, it might be necessary to specify the initial output data in order to detect changes. To avoid the redundant specification and storage of preconditions, configurations and environmental conditions used for various test cases, the use of an unambiguous reference to such data is recommended.)

Test cases shall be grouped according to the test methods to be applied. For each test method, in addition to the test cases, the following shall be specified:

* The test environment,
* The logical and temporal dependencies, and
* The resources.

Together with the test cases or (at least) the test case groups it must be documented

* Which methods for system integration testing are actually used according to the recommended methods
* Which methods for deriving test cases are actually used according to the recommended methods and the definition how to map these methods to system shares defined in the System Integration (Test) Plan

### Task 16: Specify concrete test cases

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Input documents | | | | | |
| Output | | System Integration Test Specification | | | | | |
| D: |  | E: | System Integration Test Designer | S: |  | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to detail the abstract test cases to concrete test cases and to define the order in which the test cases are processed (according to the order of integration defined in the integration strategy in case of integration tests).

The inputs to this task are the necessary documents defined in the test planning phase, “Task 5: Determine content of test (level)” and the already defined abstract test cases.

The detailed concrete test case descriptions are documented in the System Integration Test Specification, chapter 6: Specification of the Test Cases.

### Task 17: Ensure traceability and consistency

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Input documents | | | | | |
| Output | | System Integration Test Specification | | | | | |
| D: |  | E: | System Integration Test Designer | S: |  | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to ensure consistency and bilateral traceability of test specifications to the corresponding requirements, specifications and design, relevant for this test level. Consistency has to be ensured by establishing and maintaining the bilateral traceability.

### Task 18: Review and release test specification

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | System Integration Test Specification | | | | | |
| Output | | Review Checklist System Integration Test Specification | | | | | |
| D: | Test Manager,  Safety Manager | E: | System Integration Test Designer | S: | Architect, Analyst or  Developer,  Safety Manager | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim is to review and to release the System Integration Test Specification.

The result is documented in the Review Checklist System Integration Test Specification.

## Test Implementation and Execution Phase

The test cases as defined in the SW Integration Test Specification are executed and the test results are documented. The Tester is responsible for the implementation of all necessary preconditions before executing the test cases. The trial run must be carried out in the test environment and recorded in the SW Integration Test Report.

As mentioned above the integration aspects of the Software Integration as well as the System Integration Test Process are incorporated in the process flow charts.

### Task 19: Implementing integration environment

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Build List | | | | | |
| Output | | Integration Log | | | | | |
| D: |  | E: | System Integrator | S: | System Developer,  System Integration Test Automation Engineer | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to implement and to maintain the required integration environment.

The integration environment used for the integration must be documented in an Integration Log. The format is informal, the content defined by the project.

### Task 20: Integration of modules or system parts

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Build List | | | | | |
| Output | | Integrated System | | | | | |
| D: |  | E: | System Integrator | S: | System Developer,  System Integration Tester | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to integrate software modules into larger assemblies, producing an integrated System consistent with the software design and to test the interaction between the software or system items according to the integration strategy.

### Task 21: Logging integration results

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | Integration Log | | | | | |
| D: |  | E: | System Integrator | S: |  | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to document the results of each integration step.

The results of each integration step must be documented within the System Integration Log.

### Task 22: Analyze integration results

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | System Integration Log | | | | | |
| Output | | System Integration Test Report  Deviation | | | | | |
| D: |  | E: | System Integrator | S: | System Developer,  System Integration Tester | I: | Test Manager,  PM |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to analyze the integration result, conform to the integration and test strategy, and assess the maturity of the integrated System for the next test level.

The summary of the integration must be documented within the System Integration Test Report.

### Task 23: Implement test automation

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | System Integration Test Specification | | | | | |
| Output | | Test Scripts | | | | | |
| D: |  | E: | System Integration Tester | S: |  | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to implement the test scripts for the corresponding test cases which shall be executed automated, fitting to the test system defined for the project.

The resulting test scripts must be stored in the tool specific format.

### Task 24: Set up test execution

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | |  | | | | | |
| D: |  | E: | System Integration Test Designer | S: |  | I: | PM,  Test Manager |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to intake the test object, to check whether the input conditions are met, and to set up the test object and test environment for the start of testing. If the test object is not sufficient for test execution, suitable measures must be initiated by the test manager.

No documentation is required.

The SW and HW Test Report shall be used to determine the verification progress during the development phase before this test level and assess the maturity of the test object.

### Task 25: Execute tests

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | System Integration Test Object  System Integration Test Specification  Test Scripts | | | | | |
| Output | | System Integration Test Log  Deviations | | | | | |
| D: |  | E: | System Integration Tester | S: |  | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to execute the test cases as specified in the System Integration Test Specification according to the test strategy and to obtain the test results. This includes the execution of the re- and regression tests, following the re- and regression test strategy.

The necessary input of this task is the test object and the System Integration Test Specification including the prioritized test cases or the test scripts in case of automated tests.

The test execution is documented in log files. Deviations from the planned test sequence and abortion of tests are documented in the System Integration Test Report. Deviations of the test results from the desired behavior are documented and propagated by PTC error tickets.

The testing shall be executed as planned in accordance with test planning documented in System Integration (Test) Plan and specified in accordance with the test specification documented in the System Integration (Test) Specification.

The test planning shall determine the test environment and the tools used for testing, if applicable. During the test execution the test environment incl. the tools actually used must be documented, particularly deviations compared with the original planning.

### Decision 1: Completed test run?

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | |  | | | | | |
| D: | System Integration Tester | E: |  | S: |  | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

If this test level can’t be completed before the test execution of the next level an early passing on procedure can be applied. In this case the test process for this level must split in two parallel paths.

* On the one side the test execution of the remaining test cases shall be continued.
* On the other side the current test progress shall be analyzed as described in the next tasks which end with the “Task 28: Assess and communicate maturity of test object”. In this case the report will be done as a System Integration Interims Test Status Report in the standard test report template.

## Test Evaluation and Report Phase

In this phase of the test process, the results of the current test period are summarized and assessed in the System Integration Test Report. In the case of an early passing on of the test object, the maturity of the test object is preliminary assessed.

### Task 26: Analyze test results

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | System Integration Test Logs  Deviations  Metrics | | | | | |
| Output | | System Integration Test Report | | | | | |
| D: |  | E: | System Integration Tester | S: | System Developer | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

In this task, the test results and the fulfillment of the test exit criteria are analyzed and summarized.

The inputs of this task are the test logs, the derivations, and the metrics.

The result is documented in the System Integration Test Report.

The evaluation of the test results shall contain the following information:

* The unique identification of the tested work product,
* The reference to the test plan and test specification,
* The configuration of the test environment and test tools used, and the calibration data used during the evaluation, if applicable,
* The level of compliance of the test results with the expected results,
* An unambiguous statement of whether the test passed or failed; if the test failed the statement shall include the rationale for failure and suggestions for changes in the tested work product, (The test is evaluated according to the criteria for completion and termination of the verification and to the expected test results.)
* The reasons for any test steps not executed.

### Task 27: Review test results

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | System Integration Test Report | | | | | |
| Output | | Review Checklist System Integration Test Report | | | | | |
| D: | Test Manager | E: | System Integration Tester | S: |  | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim is to hand over of the System Integration Test Report to the test manager in form of a review.

The result is documented in the Review Checklist System Integration Test Report.

### Task 28: Assess and communicate maturity of test object

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | System Integration Test Report | | | | | |
| D: | Test Manager,  Safety Manager | E: | Test Manager | S: | Quality Planner | I: | PM |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to provide an assessment of the maturity of the test object and to communicate the test results to all relevant parties as defined in Task 7: Organize test project.

Not performed tests regarding safety related functions have to be documented in the System Integration Test Report and have to be communicated to the customer.

The result is added in the System Integration Test Report.

## Test Closure Phase

In the test closure phase at project closure, the experiences are consolidated and for each test level a Test Completion Report is drawn up. Another important issue is the archiving of the test environment for reproduction and possible reuse.

### Task 29: Analyze test experience

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | System Integration Test Completion Report | | | | | |
| D: |  | E: | Test Manager | S: | PM,  Quality Planner,  Test Team,  Representatives of  System Development | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to analyze the experiences gained in the System Integration test level.

The result is the System Integration Test Completion Report.

### Task 30: Archiving test environment

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | |  | | | | | |
| D: |  | E: | System Integration Tester,  System Integration Test Automation Engineer | S: |  | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

The aim of this task is to archive the test environment, test specification, test cases and other aids for subsequent tests (e.g. reuse).

The relevant test basis, documents, and test environment are stored in the PTC Source Integrity.

# Changes, References, Appendix, Terms

## References

|  |  |  |
| --- | --- | --- |
| **Category** | **Document Name** | **Document Number** |
| Process | Test of Products Containing Software | AD-PE1-1-03 |

## Template

|  |  |  |
| --- | --- | --- |
| **Category** | **Document Name** | **Document Number** |
| Template | System Integration Plan | 8517 |
| Template | System Integration Test Plan | 8517 |
| Template | System Integration Test Specification | 8518 |
| Template | System Integration Test Report | 8519 |
| Template | Review-Checklist System Integration Test Plan | 8393 |
| Template | Review-Checklist System Integration Test Specification | 8588 |
| Template | Review-Checklist System Integration Test Report | 8832 |
| Template | System Integration Test Completion Report | 8391 |