SW 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](#_Toc496026499)

[1.1 Purpose 5](#_Toc496026500)

[1.2 Main Description 5](#_Toc496026501)

[1.3 Result of Process 5](#_Toc496026502)

[2. Major Roles Acting in this Process 5](#_Toc496026503)

[2.1 Refinement of the major roles 6](#_Toc496026504)

[3. Process 6](#_Toc496026505)

[3.1 Process Input – Output Definition 6](#_Toc496026506)

[3.2 Introducing Aspects 7](#_Toc496026507)

[3.3 Phase Model 7](#_Toc496026508)

[3.4 Process Flow Chart 9](#_Toc496026509)

[3.5 Test Planning Phase 10](#_Toc496026510)

[3.5.1 Task 1: Determine integration strategy 11](#_Toc496026511)

[3.5.2 Task 2: Temporal planning integration steps 11](#_Toc496026512)

[3.5.3 Task 3: Specifying integration environment 12](#_Toc496026513)

[3.5.4 Task 4: Determine framework conditions 12](#_Toc496026514)

[3.5.5 Task 5: Determine content of test level 12](#_Toc496026515)

[3.5.6 Task 6: Develop test strategy 13](#_Toc496026516)

[3.5.7 Task 7: Organize test project 16](#_Toc496026517)

[3.5.8 Task 8: Review and release test plan 17](#_Toc496026518)

[3.6 Test Control Phase 18](#_Toc496026519)

[3.6.1 Task 9: Manage test team and project 18](#_Toc496026520)

[3.6.2 Task 10: Monitor and communicate test progress 18](#_Toc496026521)

[3.7 Test Infrastructure Phase 19](#_Toc496026522)

[3.7.1 Task 11: Specify infrastructure 19](#_Toc496026523)

[3.7.2 Task 12: Realize infrastructure 19](#_Toc496026524)

[3.7.3 Task 13: Putting infrastructure into operation 20](#_Toc496026525)

[3.8 Test Analysis and Design Phase 20](#_Toc496026526)

[3.8.1 Task 14: Analyze test basis 20](#_Toc496026527)

[3.8.2 Task 15: Specify abstract test cases 21](#_Toc496026528)

[3.8.3 Task 16: Specify concrete test cases 22](#_Toc496026529)

[3.8.4 Task 17: Ensure traceability and consistency 22](#_Toc496026530)

[3.8.5 Task 18: Review and release test specification 22](#_Toc496026531)

[3.9 Test Implementation and Execution Phase 23](#_Toc496026532)

[3.9.1 Task 19: Implementing integration environment (specific for integration) 23](#_Toc496026533)

[3.9.2 Task 20: Integration of modules or system parts 23](#_Toc496026534)

[3.9.3 Task 21: Logging integration results 23](#_Toc496026535)

[3.9.4 Task 22: Analyze integration results 24](#_Toc496026536)

[3.9.5 Task 23: Implement test automation 24](#_Toc496026537)

[3.9.6 Task 24: Set up test execution 24](#_Toc496026538)

[3.9.7 Task 25: Execute tests 24](#_Toc496026539)

[3.9.8 Decision 1: Completed test run? 25](#_Toc496026540)

[3.10 Test Evaluation and Report Phase 25](#_Toc496026541)

[3.10.1 Task 26: Analyze test results 26](#_Toc496026542)

[3.10.2 Task 27: Review test results 26](#_Toc496026543)

[3.10.3 Task 28: Assess and communicate maturity of test object 26](#_Toc496026544)

[3.11 Test Closure Phase 27](#_Toc496026545)

[3.11.1 Task 29: Analyze test experience 27](#_Toc496026546)

[3.11.2 Task 30: Archiving test environment 27](#_Toc496026547)

[4. Changes, References, Appendix, Terms 27](#_Toc496026548)

[4.1 References 27](#_Toc496026549)

[4.2 Template 28](#_Toc496026550)

# Purpose Description

## Purpose

The purpose of the software integration test process is to integrate the software modules into larger assemblies, producing integrated software consistent with the software design and to test the interaction between the software items.

## Main Description

The software integration test is the next test level after the software module test level with the goal to stabilize the software system in order to perform the software test in an efficient and effective way. Focus of the integration test level is

* Interaction and communication between the software modules (interfaces)
* Functionality, which is realized over several software modules
* Stack usage
* Memory usage
* Runtime aspects: processing time, response times and latencies
* Error handling
* Stability and robustness

The following problems can be found:

* Failures within the interaction and communication of the integrated software modules
* Problems with the resource management
* Runtime failures

## Result of Process

As a result of successful implementation of this process:

* A software integration and integration test strategy is developed for software items consistent with the software design according to the priorities and categorization of the software requirements;
* A test specification software integration is developed that ensures compliance with the software architectural design, software detailed design, allocated to the items;
* Software units and software items are integrated as defined by the integration strategy;
* Integrated software items are verified using the test cases;
* Test results of the software integration test are recorded;
* Consistency and bilateral traceability are established between software architectural design and detailed design and test specification software integration test including test cases; and
* A regression strategy is developed and applied for re-integrating and re-verifying software items when a change in software items (including associated requirements, design and code) occurs.

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

NOTE 2: The test results for software 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. |
| SW Integration Tester | Analyzes the test basis, creates the test specification, the test scripts, and carries out the test including the corresponding documentation. |
| SW Project Manager | In this context relevant for all software development aspects like integration, SW module classification, SW 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 “SW Integration Tester” role (MHE-PE-01,

Supplement A):

|  |  |
| --- | --- |
| **Role** | **Contribution and Responsibilities** |
| SW 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). |
| SW Integration Tester | Carries out the tests as defined in the test specification and draws up the reports including measurements. |
| SW 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 |
| Software Requirements Specification | SW Analyst |
| Software Architecture (e.g. Innovator SD model) | SW Architect |
| Software design (Definition of all software modules and interfaces) | SW Architect,  SW Developer |
| Attributes for economic risk classification on software integration level | SW Architect |
| Release Plan | SW Project Manager |
| Integrated modules/software items (test object) | SW Developer |

The following process output is produced:

|  |  |
| --- | --- |
| **Process output** | **To customer** |
| SW Integration (Test) Plan | SW Project Manager  Safety Manager |
| Review Checklist SW Integration (Test) Plan | SW Integration Test  Designer |
| SW Integration Test Specification | SW Integration Tester |
| Review Checklist SW Integration (Test) Specification | SW Integration Test  Designer |
| SW Integration Test Report | SW Project Manager  Safety Manager |
| Test Completion Report | SW 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 specific test level and document the planning decisions like how, by whom, what with and when what test activities are carried out in the SW Integration Test Plan.

### Task 1: Determine integration strategy

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | SW Architecture Specification  SW Release Plan | | | | | |
| Output | | SW Integration Plan | | | | | |
| D: | SW PM | E: | SW Integrator | S: | Test Manager,  SW Architect,  SW 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 software modules 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 SW 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 software integration shall describe the steps for integrating the individual software hierarchically into software components until the embedded software is fully integrated, and shall consider:

* The functional dependencies that are relevant for software integration; and
* The dependencies between the software integration and the hardware-software integration. (For model-based development, the software integration can be replaced with integration at the model level and subsequent automatic code generation from the integrated model.)

### Task 2: Temporal planning integration steps

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | Build List | | | | | |
| D: | SW PM | E: | SW 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 SW Integration Plan.

The time planning of the integration needs to be documented in the SW 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 SW Integration Plan.

### Task 3: Specifying integration environment

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | SW Integration Plan | | | | | |
| D: |  | E: | SW Integrator | S: | SW Integrator 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 SW Integration Plan, chapter 1.5: Integration Environment.

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

* Specifying the test environment

### Task 4: Determine framework conditions

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Master Test Plan  Safety Plan | | | | | |
| Output | | SW Integration Test Plan | | | | | |
| D: |  | E: | Test Manager | S: | SW 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 SW Integration Test Plan, chapter 2: Introduction.

### Task 5: Determine content of test level

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Input documents  SW Integration Plan | | | | | |
| Output | | SW Integration Test Plan | | | | | |
| D: | Test Manager | E: | SW Integration Test Designer | S: | SW 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
* Analyzing the project- and test object-specific particularities
* Determining and referencing the specification documents

The available qualified software components shall be determined and referenced as project- and test object specific particularities and shall be considered during the determination of the test strategy.

The following information shall be available and referenced in detail:

* Hardware-software interface specification and software architectural 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 software integration test objects are the software components. (For model-based development, the test objects can be the models associated with the software components.)

The software integration testing shall be applied to demonstrate that both the software components and the embedded software achieve:

* Compliance with the software architectural design
* Compliance with the specification of the hardware-software interface
* The specified functionality \*
* Robustness \* (e.g. the absence of inaccessible software, the effectiveness of error detection and error handling mechanisms.)
* Sufficient resources to support the functionality \*

These topics are defined as scope of testing of the software integration test level within this task and considered in the following process steps.

Remark: \* The marked topics can be tested partly in this software integration test level and partly in the software test level, which has the test of the complete embedded software in focus. The test strategies must ensure, that these topics are tested in an appropriate extend.

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

The result of this task must be documented in the SW 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 test strategy:

* The content of the Safety Plan determined for the software integration and integration testing shall be take over and integrated in this test strategy. This content shall contain
  + Methods for software integration testing
  + Methods for deriving test cases for software integration testing
  + Structural coverage metrics at the software architectural level

The following methods shall be applied at the software integration test and partly at the software 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)

The test methods listed in Table 1 shall be applied to demonstrate that both the software components and the embedded software achieve the scope of test content defined:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Requirements-based test a | ++ | ++ | ++ | ++ |
| 1b | Interface test | ++ | ++ | ++ | ++ |
| 1c | Fault injection test b | + | + | + | ++ |
| 1d | Resource usage test c | + | + | + | ++ |
| 1e | Back-to-back comparison test between model and code, if applicable d | + | + | ++ | ++ |
| a The software requirements at the module level are the basis for this requirements-based test.  b This includes injection of arbitrary faults (e.g. by corrupting values of variables, by introducing code mutations, or by corrupting values of CPU registers)  c Some aspects of the resource usage test can only be evaluated properly when the software module tests are executed on the target hardware or if the emulator for the target processor supports resource usage tests.  d This method requires a model that can simulate the functionality of the software modules. Here, the model and code are stimulated in the same way and results compared with each other. | | | | | |

Table 1: Methods for software integration testing

The selection of the method for software integration testing or the criteria to define the selection of these methods for specific types of software interfaces must be defined and justified in the test strategy. Furthermore the omission of methods also must be justified and documented.

To enable the specification of appropriate test cases for the software integration test methods selected in accordance with Table 1, test cases shall be derived using the methods listed in Table 2.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Analysis of requirements | ++ | ++ | ++ | ++ |
| 1b | Generation and analysis of equivalence classes a | + | ++ | ++ | ++ |
| 1c | Analysis of boundary values b | + | ++ | ++ | ++ |
| 1d | Error guessing (or Explorative testing) c | + | + | + | + |
| a Equivalence classes can be identified based on the division of inputs and outputs, such that a representative test value can be selected for each class.  b This method applies to interfaces, values approaching and crossing the boundaries and out of range values.  c Error guessing tests can be based on data collected through a “lessons learned” process and expert judgment. MHE remark: Alternative to Error guessing the method Explorative testing can be applied. | | | | | |

Table 2: Methods for deriving test cases for software integration testing

The selection of the methods for deriving test cases for software integration testing or the criteria to define the selection of these methods for specific types of software interfaces must be defined and justified in the test strategy. Furthermore the omission of methods also must be justified and documented.

To evaluate the completeness of tests and to obtain confidence that there is no unintended functionality the coverage of requirements at the software architectural level by test cases shall be determined. If necessary, additional test cases shall be specified or a rationale shall be provided.

To evaluate the completeness of test cases and to obtain confidence that there is no unintended functionality the structural coverage shall be measured in accordance with the metrics listed in Table 3.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Methods | | ASIL | | | |
| A | B | C | D |
| 1a | Function coverage a | + | + | ++ | ++ |
| 1b | Call coverage b | + | + | ++ | ++ |
| a Method 1a refers to the percentage of executed software functions. This evidence can be achieved by an appropriate software integration strategy.  b Method 1b refers to the percentage of executed software function calls. | | | | | |

Table 3: Structural coverage metrics at the software architectural level

Note 1 The structural coverage can be determined using appropriate software tools.

Note2 In the case of model-based development, software architecture testing can be performed at the model level using analogous structural coverage metrics for models.

It shall be verified that the embedded software that is to be included as part of a production release contains all the specified functions, and only contains other unspecified functions if these functions do not impair the compliance with the software safety requirements.

(In this context unspecified functions include code used for debugging or instrumentation. If deactivation of these unspecified functions can be ensured, this is an acceptable means of compliance with this requirement, otherwise the removal of such code is a change).

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.)

The developed test strategy and regression test strategy are documented in the SW Integration Test Plan, chapter 4: Test strategy.

### Task 7: Organize test project

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | SW Integration Test Plan | | | | | |
| D: |  | E: | Test Manager | S: | SW PM, SW Integration Test Designer,  SW Integration Tester,  SW 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 test environment for software integration testing shall correspond as closely as possible to the target environment.

The gathering and reporting procedure concerning the structural coverage metrics at software integration test level according to the recommended methods and the definition of how to map these methods to software shares, must be defined in this task.

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

### Task 8: Review and release test plan

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

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

The result is documented in the Review Checklist SW 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: | SW Integration Tester,  SW Integration Test  Designer,  SW 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: | SW Integration Tester,  SW Integration Test  Designer, SW Integration Test Automation Engineer | I: | SW 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 SW Integration Test Plan and the results logged in the SW 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 SW 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 SW 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 | | SW Integration Test Plan | | | | | |
| Output | |  | | | | | |
| D: |  | E: | SW Integration Test Automation Engineer | S: | SW 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 SW 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: | SW 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: | SW Integration Tester | S: | SW 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 SW Integration Test Specification (for the corresponding templates see 5.1 References – Forms).

### Task 14: Analyze test basis

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Input documents | | | | | |
| Output | | SW Integration Test Specification | | | | | |
| D: |  | E: | SW Integration Test Designer | S: | Architect, Analyst or  Developer | I: | SW 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 SW Module Test Specification, chapter 3: Examination of the System Test Basis.

The following information shall be used for the creation of test cases:

* Hardware-software interface specification and software architectural design specification determined and referenced as specification documents

If the software integration testing is not carried out in the target environment, the differences in the source and object code, and the differences between the test environment and the target environment, shall be analysed in order to specify additional tests in the target environment during the subsequent test phases.

Note 1 Differences between the test environment and the target environment can arise in the source code or object code, for example, due to different bit widths of data words and address words of the processors.

Note 2 Depending on the scope of the tests and hierarchical level of integration, the appropriate test environment for the execution of the software module is used. Such test environments can be the target processor for final integration, or a processor emulator or a development system for the previous integration steps.

Note 3 Software integration testing can be executed in different environments, for example:

* Model-in-the-loop tests;
* Software-in-the-loop tests;
* Processor-in-the-loop tests; and
* Hardware-in-the-loop tests.

### Task 15: Specify abstract test cases

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Input documents | | | | | |
| Output | | SW Integration Test Specification | | | | | |
| D: |  | E: | SW 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 SW 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 software 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 software shares defined in the SW Integration (Test) Plan,

### Task 16: Specify concrete test cases

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Input documents | | | | | |
| Output | | SW Integration Test Specification | | | | | |
| D: |  | E: | SW 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 SW Integration Test Specification, chapter 6: Specification of the Test Cases.

### Task 17: Ensure traceability and consistency

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Input documents | | | | | |
| Output | | SW Integration Test Specification | | | | | |
| D: |  | E: | SW 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 | | SW Integration Test Specification | | | | | |
| Output | | Review Checklist SW Integration Test Specification | | | | | |
| D: | Test Manager,  Safety Manager | E: | SW 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 SW Integration Test Specification.

The result is documented in the Review Checklist SW 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 (for the corresponding templates see 5.1 References – Forms).

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 (specific for integration)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Build List | | | | | |
| Output | | Integration Log | | | | | |
| D: |  | E: | SW Integrator | S: | SW Developer,  SW 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. (Examples for such Log-Files are given in Confluence)

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | Build List | | | | | |
| Output | | Integrated SW | | | | | |
| D: |  | E: | SW Integrator | S: | SW Developer,  SW 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 software 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: | SW 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 SW Integration Log.

### Task 22: Analyze integration results

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | SW Integration Log | | | | | |
| Output | | SW Integration Test Report  Deviation | | | | | |
| D: |  | E: | SW Integrator | S: | SW Developer,  SW Integration Tester | I: | Test Manager,  SW 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 SW for the next test level.

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

### Task 23: Implement test automation

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | SW Integration Test Specification | | | | | |
| Output | | Test Scripts | | | | | |
| D: |  | E: | SW 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: | SW Integration Test Designer | S: |  | I: | SW 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 Software Verification 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 | | SW Integration Test Object  SW Integration Test Specification  Test Scripts | | | | | |
| Output | | SW Integration Test Log  Deviations | | | | | |
| D: |  | E: | SW 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 SW 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 SW 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 SW 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 SW Integration

(Test) Plan and specified in accordance with the test specification documented in the SW 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: | SW 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 an SW 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 SW 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 | | SW Integration Test Logs  Deviations  Metrics | | | | | |
| Output | | SW Integration Test Report | | | | | |
| D: |  | E: | SW Integration Tester | S: | SW 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 SW 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 | | SW Integration Test Report | | | | | |
| Output | | Review Checklist SW Integration Test Report | | | | | |
| D: | Test Manager | E: | SW Integration Tester | S: |  | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

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

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

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | | SW Integration Test Report | | | | | |
| D: | Test Manager,  Safety Manager | E: | Test Manager | S: | SW Quality Planner | I: | SW 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 SW Integration Test Report and have to be communicated to the customer.

The result is added in the SW 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 | | SW Integration Test Completion Report | | | | | |
| D: |  | E: | Test Manager | S: | SW PM,  SW Quality Planner,  Test Team,  Representatives of  SW Development | I: |  |
| D = Decision | | E = Execution | | S = Support | | I = Information | |

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

The result is the SW Integration Test Completion Report.

### Task 30: Archiving test environment

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Input | |  | | | | | |
| Output | |  | | | | | |
| D: |  | E: | SW Integration Tester,  SW 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 | SW Integration Plan | 8831 |
| Template | SW Integration Test Plan | 8517 |
| Template | SW Integration Test Specification | 8518 |
| Template | SW Integration Test Report | 8519 |
| Template | Review-Checklist SW Integration Test Plan | 8589 |
| Template | Review-Checklist SW Integration Test Specification | 8354 |
| Template | Review-Checklist SW Integration Test Report | 8832 |
| Template | SW Integration Test Completion Report | 8391 |