**TESTING PROCESSES**

**UNIT-II**

* A process is a series of activities performed to fulfill a purpose and produce a concrete output based on a given input.
* Process models are defined to assist organizations in process improvement—that is, in making their work more structured and efficient.
* Testing can also be regarded as a process.
* Like all processes the test process can be viewed at different levels of detail
* Test development is what is usually understood as the real test work. This is sometimes divided into two subprocesses, namely:
* Test analysis and design;
* Test implementation and execution.

**Processes in General**

1. **The Concept of a Process**

A process is a series of activities performed to fulfill a specific purpose. Based on an input to the process and following the activities—also called the procedure—a concrete output is produced.

Processes can be described and hence monitored and improved. A process description must always include:

* A definition of the input
* A list of activities—the procedure
* A description of the output

For a more comprehensive and more useful process description the following information could also be included:

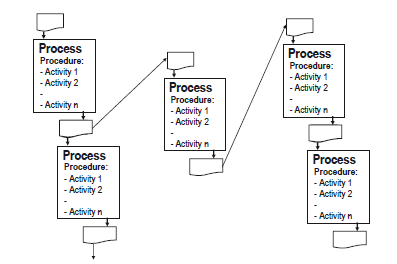
* Entry criteria—What must be in place before we can start?
* Purpose—A description of what must be achieved ?
* Role—Who is going to perform the activities?
* Methods, techniques, tools—How exactly are we going to perform the activities?
* Measurements—What metrics are we going to collect for the process?
* Templates—What should the output look like?
* Verification points—Are we on the right track?
* Exit criteria—What do we need to fulfill before we can say that we have finished?

1. **Monitoring Processes**

* It is the responsibility of management in charge of a specific area to know how the pertaining processes are performed.
* For testing processes it is of course important for the test leader to know how the testing is performed and progressing.

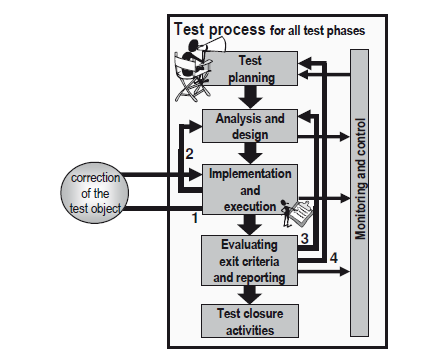
1. **Processes Depend on Each Other**

* The input to a process must be the output from one or more proceeding process(es)—except perhaps for the very first.
* The output from a process must be the input to one or more other processes—even the final product, which is the input to the maintenance process.
* The dependencies between processes can be depicted in a process model, where it is shown how outputs from processes serve as inputs to other processes.



1. **The Overall Generic Test Process**

* Testing is a process.
* The purpose of the test process is to provide information to assure the quality of the product, decisions, and the processes for a testing assignment.
* The inputs on which this process is based are:
* Test strategy
* Project plan
* Master test plan
* Information about how the testing is progressing
* The activities are:
* Test planning and control
* Test development
* Test analysis and design
* Test implementation and execution
* Evaluating exit criteria and reporting
* Test closure activities
* The output consists of:
  + - Level test plan
    - Test specification in the form of test conditions, test design, test cases, and test procedures and/or test scripts
    - Test environment design and specification and actual test environment including test data
    - Test logs
    - Progress reports
    - Test summary report
    - Test experience report
* The generic test process is applicable for each of the dynamic test levels. So the process should be used in testing such as:
* Component testing
* Integration testing
* System testing
* Acceptance testing
* There are many more dependencies. Some of them are described in the following sections.
  + *The test activities need not be performed in strict sequential order.*
  + *The generic test process is iterative*—*not a simple straightforward process.*

**

* + The resulting iterations may be:

1. The defect is in the test object.
2. The defect is in the test procedure
3. More test cases must be specified to increase coverage, and these must then be executed.
4. The exit criteria are relaxed or strengthened in the test plan
5. **Other Testing Processes**

* A test process may be created from scratch or it may be created as a tailoring of a standard process.
* The various process improvement models that exist provide frameworks for processes. Some cover all the process areas in a development organization;
* *Test Process Improvement Model* (TPI), defines a list of 20 key areas.
* The 20 key areas are grouped into four so-called cornerstones as follows:
* Life cycle—Test strategy, life cycle model, moment of involvement
* Techniques—Estimating and planning, test specification techniques, static test techniques, metrics
* Infrastructure—Test tools, test environment, office environment
* Organization—Commitment and motivation, test functions and training, scope of methodology, communication, reporting, defect management, testware management, test process management, evaluating, low-level testing.
* Another process model is the Critical Testing Processes (CTP).
* This model also defines a number of process areas. In this model the process areas are
* grouped into four classes:
  + Plan—Establish context, analyze risks, estimate, plan
  + Prepare—Grow and train team, create testware, test environment, and test processes
  + Perform—Receive test object(s), execute and log tests
  + Perfect—Report bugs, report test results, manage changes

**Test Planning and Control**

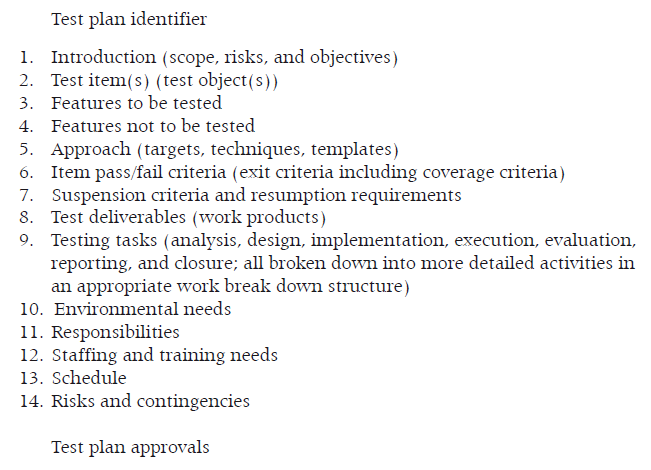
* The purpose of the test planning process is to verify the mission of the testing, to define the objectives of the testing, and to make the necessary decisions to transform the test strategy into an operational plan for the performance of the actual testing task at hand.
* The purpose of the control part is to ensure that the planned activities are on track by monitoring what is going on and take corrective actions as appropriate.
* The inputs on which this process is based are:
  + Test strategy
  + Master test plan
  + Information about how the testing is progressing
* The activities are:
  + Verify the mission and define the objectives of the testing
  + Decide and document how the general test strategy and the project test plan apply to the specific test level: what, how, where, who
  + Make decisions and initiate corrective actions as appropriate as the testing progresses
* The output consists of:
  + Level test plan

1. **Input to Test Planning and Control**

* The planning of a test level is based on the relevant test strategy, the project plan for the project to which the test assignment belongs, and the master test plan.
* The level test plan outlines how the strategy is being implemented in the specific test level in the specific project at hand.
* The test level plan must be consistent with the master test plan.
* *The decisions are not made for the purpose of writing the plan, but for the purpose of getting agreement and commitment* of all the stakeholders in the test to be performed.
* The planning and control of the test are continuous activities.

1. **Documentation of Test Planning and Control**

* The tangible output of this process is the level test plan for the testing level to which the process is applied. The structure of the level test plan should be tailored to the organization.
* A template could be based on the IEEE 829 standard.

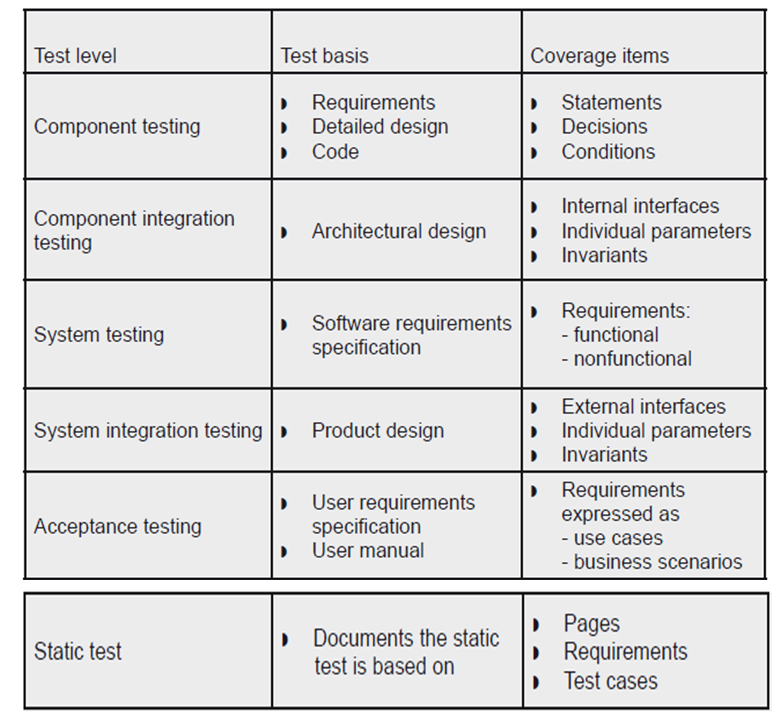


1. **Activities in Test Planning**

* *Test planning should start as early* as possible.
* The initial detailed planning for each of the test levels can start as soon as the documentation on which the testing is based has reached a suitable draft level.
* Early planning has a number of advantages. It provides, for example, time to do a proper planning job, adequate time to include the stakeholders, early visibility of potential problems, and means of influencing the development plan.
* The test planning activities must first of all aim at setting the scene for the testing assignment for the actors in accordance with the framework.

**Defining Test Object and Test Basis**

* The test planning must identify the test basis and define what it is we are going to test in relation to this.
* This includes determination of the coverage to achieve for the appropriate coverage item(s).
* The expected coverage must be documented in the level test plan as (part of) the completion criteria.
* The coverage items depend on the test basis.

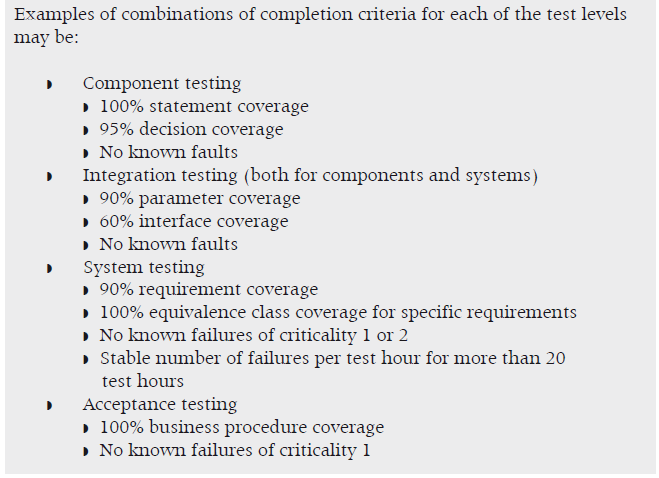
****

**Defining the Approach**

* The test approach must be based on the strategy for the test at hand.
* The approach must at least cover:
  + The test methods and test techniques to use
  + The structure of the test specification to be produced and used
  + The tools to be used
  + The interface with configuration management
  + Measurements to collect
  + Important constraints, such as availability or “fixed” deadline.
* For the testing we are planning for.
* First of all, the test object determines the *method:*
  + If the test object is something that can be read or looked at, the method is static test—the specific choice of static test type(s) depends on the criticality of the object.
  + If the test object is executable software, the method is dynamic test.
* The interface with *configuration management* covers:
  + How to identify and store the configuration items we produce in the test process
  + How to get the configuration items we need (for example, design specifications, source code, and requirements specifications)
  + How to handle traceability
  + How to register and handle incidents

**Defining the Completion Criteria**

* The completion criteria are what we use to determine if we can stop the testing or if we have to go on to reach the objective of the testing.
* The completion criteria are derived from the strategy and should be based on a risk analysis; the higher the risk, the stricter the completion criteria; the lower the risk the less demanding and specific the completion criteria.
* The most appropriate completion criteria vary from test level to test level.
* Completion criteria for the test may be specified as follows:
  + Specified coverage has been achieved
  + Specified number of failures found per test effort has been achieved
  + No known serious faults
  + The benefits of the system are bigger than known problems
  + (The time has run out)

****

**Defining Work Products and Their Relationships**

* The number of deliverables, their characteristics, and estimates of their sizes must be defined, not least because this is used as input for the detailed estimation and scheduling of all the test activities, but also because the precision of what is going to be delivered sets stakeholders’ expectations.
* Typical deliveries or work products from a test level are:
* Level test plan(s)
* Test specification(s)
* Test environment(s)
* Test logs and journals
* Test reports
* The structure shown and explained here is based on the structure suggested in IEEE 829. A full test specification may consists of:
  + A *test design* consisting of a number of test groups (or designs) with test conditions and high-level test cases derived from the basis documentation. The designs will typically reflect the structure of the test basis documentation. The relationships between the elements in the basis documentation and the high-level test cases may well be quite complicated, often including even many-to-many relationships.
  + A number of *low-level test cases* extracted from the high-level test cases and being made explicit with precise input and output specifications .
  + A number of test procedures each encompassing a number of test cases to be executed in sequence. The relationships between highlevel test cases and test procedures may also be complicated and include many-to-many relationships.
* This structure is applicable to test specifications at all test levels, for example, for:
  + Component testing
  + Integration testing
  + System testing
  + Acceptance testing

**Scoping the Test Effort**

* The definition of exhaustive testing is: test case design technique in which the test case suites comprise all combinations of input values and preconditions for component variables.
* No matter how much we as testers would like to do the ultimate good job, *exhaustive testing is a utopian goal.*
* The parameters are:
  + *Time:* The available calendar time
  + *Resources:* The available people and tools
  + *Quality:* The quality of the testing
* These parameters form what we call the quality triangle
* The basic principle of the quality triangle is: *It is not possible to change one of the parameters and leave the other two unchanged*—*and still be in balance!*

***Work Breakdown Structure***

* One of the things on which the test planning is based, is a list of all the tasks to be performed. This list should be in the form of a work breakdown structure of the test process at hand.
* If we use the test process defined here the overall tasks are planning, monitoring, control, analysis, design, implementation, execution, evaluation, reporting, and closure, all broken down into more detailed activities in an appropriate work breakdown structure.

***Defining Test Roles***

* A (software test) project is like a play in which all roles must be filled in order for the play to be performed. Some roles are big, some are small, but they are all important for the whole.
* The roles to handle the testing tasks may be:
  + Test leader (manager or responsible)
  + Test analyst/designer
  + Test executer
  + Reviewer/inspector
  + Domain expert
  + Test environment responsible
  + (Test)tool responsible
* Test teams are formed by all these roles. We need different teams depending on which test phase we are working in, but the principles are the same:
  + *All relevant roles must be present and filled in the team*
  + A role can be filled by one person or more people, depending on the size of the testing assignment at hand
  + One person can fill one role or more roles, again depending on the size (but keep in mind that less than 25% time for a role = 0% in real life)

***Producing the Schedule***

* In scheduling the tasks, the staffing and the estimates are brought together and transformed into a schedule.
* Risk analysis may be used to prioritize the testing for the scheduling: the higher the risk, the more time for testing and the earlier the scheduled start of the testing task.
* The result of this is a schedule that shows precisely who should do what at which point in time and for how long.
* A framework for the resources and the schedule must be obtained from the overall project plan, and the result of the test scheduling must be reconciled with the project plan.
* Estimations for all the tasks are input to the scheduling.
* Once the tasks are estimated they can be fitted into the project time line

1. **Activities in Test Control**

* As the testing progresses the control part of test management is about staying in control and making necessary corrections to the plan when it no longer reflects the reality.
* Measurements are collected in the test monitoring activities for all the detailed activities in the test processes, and these measurements are analyzed to understand and follow the actual progress of the planned test activities and the resulting coverage.
* Decisions must be made if things are deviating significantly from the plan, and corrective actions may be necessary.
* The testing often gets pressed for time, since it is the last activity before the product is released.
* When development is delayed it is tempting to shorten the test to be able to keep the release date.
* But if our testing time is cut, we have to change at least one other parameter in the quality triangle; anything else is impossible.
* It is important to point this out to management.

1. **Metrics for Test Planning and Control**

* Metrics to be defined for the monitoring and control of the test planning and control activities themselves may include:
  + Number of tasks commenced over time
  + Task completion percentage over time
  + Number of tasks completed over time
  + Time spent on each task over time
* This will of course have to be compared to the estimates and schedule of the test planning and control activities.

**Test Analysis and Design**

* The purpose of the test analysis and design activities is to produce test designs with test conditions and tests cases and the necessary test environment based on the test basis and the test goals and approach outlined in the test plan.
* The inputs on which this process is based are:
  + Level test plan
  + Basis documentation
* The activities are:
  + Analysis of basis documentation
  + Design of high-level test cases and test environment
* The output consists of:
  + Test design
  + Test environment design and specification

1. **Input to Test Analysis and Design**

* The input from the level test plan that we need for this process is:
  + Test objectives
  + Scheduling and staffing for the activities
  + Definition of test object(s)
  + Approach—especially test case design techniques to use and structure and contents of the test specification
  + Completion criteria, not least required coverage
  + Deliverables

1. **Documentation of Test Analysis and Design**

* The result of the test analysis and design should be documented in the test specification. This document or series of documents encompasses
  + The test designs—also called test groups
  + The test cases—many test cases per test design
  + Test procedures—often many-to-many relationship with test cases
* The overall structure of the test specification is defined in the level test plan.

1. **Activities in Test Analysis and Design**

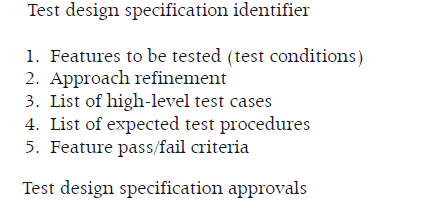
* The idea in structured testing is that the test is specified before the execution.
* The test specification activity can already start when the basis documentation is under preparation.
* The test specification aims at designing tests that provide the largest possible coverage to meet the coverage demands in the test plan.

**Defining Test Designs**

In test design the testing task is broken into a number of test design or test groups. This makes the test development easier to cope with, especially for the higher test levels.

Test groups may also be known as test topics or test areas.

A *test design* or test group specification should have the following contents according to IEEE 829:

****

* The groups and the procedures must be uniquely identified. The number of test groups we can define depends on the test level and the nature, size, and architecture of the test object:
  + In component testing we usually have one test group per component
  + For integration testing there are usually a few groups per interface
  + For system and acceptance testing we typically have many test groups

**Identification of Test Conditions**

* The features to be tested mentioned in the test design can be expressed as test conditions or test requirements.
* A test condition is a verifiable item or element.
* The nature of a test condition depends on the nature of the test basis documentation. It may for example be a function, a transaction, a feature, a requirement, or a structural element like an interface parameter or a statement in the code.
* The test conditions are based on or identical to our coverage items.
* They are the items we are covering when we test the test object.
* The completion criteria often include the percentage of the coverage items we must cover, called the coverage.
* We select the test conditions to get the highest coverage. Prioritization criteria identified in the risk analysis and test planning.
* The documentation of a test condition must at least include:
  + Unique identification
  + Description
  + Reference to test basis documentation, if not taken from there directly

**Creation of Test Cases**

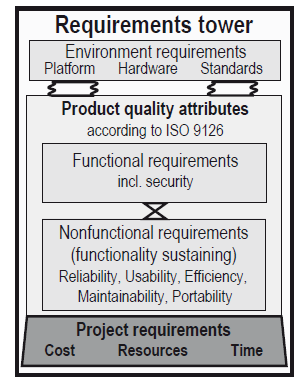
* Based on the test conditions, we can now produce our first high-level test cases and subsequently low-level test cases.
* A high-level test case is a test case without specific values for input data and expected results, but with logical operators or other means of defining what to test in general terms.
* The test cases we design should strike the best possible balance between being:
  + Effective: Have a reasonable probability of detecting errors
  + Exemplary: Be practical and have a low redundancy
  + Economic: Have a reasonable development cost and return on investment
  + Evolvable: Be flexible, structured, and maintainable
* The test case design techniques make it possible to create test cases that satisfy these demands.
* The test techniques help us identify the input values for the test cases.
* *The techniques cannot supply the expected result.*
* The documentation of a *test case* at this stage must at least include:
  + Unique identification
  + Description
  + References to test condition(s) on which the test case is based and to test design(s) to which the test case belongs
* The analysis of the basis documentation will also reveal requirements concerning
* the test environment, not least the required test data From the high-level test cases we go on to define the low-level test cases.
* It is not always possible to execute all the test cases we have identified.
* A low-level test case is a test case with specific values defined for both input and expected result.
* The documentation of a *low-level test case* must at least include:
  + Unique identification
  + Execution preconditions
  + Inputs: data and actions
  + Expected results including post conditions
  + Reference(s) to test conditions and/or directly to basis documentation
* The expected result must be determined from the basis documentation where the expectations for the coverage items are described. *The expected result must never, ever be derived from the code!*

1. **Requirements**

* All product development starts with the requirements.
* The higher level testing is done directly against requirements.
* The lower level testing is done against design that is based on the requirements.
* All testing is hence based on the requirements.

**Requirement Levels**

* Requirements should exist at different levels, for example:
  + Business requirements
  + User requirements
  + System requirements

****

**Requirement Types**

* The requirement specification at each level must cover all types of requirements.
* The most obvious requirements type is functional. No functionality entails no system.
* Some requirements expressing how the functionality should behave and present itself. These requirements are usually known as **nonfunctional requirements**. We could also call them functionality- supporting requirements

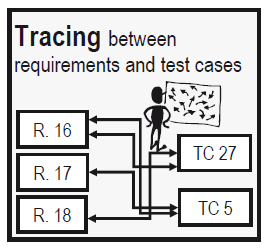
**Requirement Styles**

Requirements can be expressed in many ways. Typical styles are:

* Statements
* Tasks
* Models
* Tables
* To make statement requirements more precise and testable we can use metrics and include information such as the scale to use, the way to measure, the target, and maybe acceptable limits. This is especially important for nonfunctional requirements!
* A *task* is a series of actions to achieve a goal.
* A *model* is a small representation of an existing or planned object.

1. **Traceability**

* References are an important part of the information to be documented in the test specification. A few words are needed about these.
* There are two sets of references:
  + References between test specification elements
  + References from test specification elements to basis documentation

****

* Traces should be two way
  + Forward
  + Backward

1. **Metrics for Analysis and Design**

* Metrics to be defined for the monitoring and control of the test analysis and design activities may include:
  + Number of specified test conditions and high-level requirements over time
  + Coverage achieved in the specification (for example, for code structures, requirements, risks, business processes), over time
  + Number of defects found during analysis and design
  + Other tasks commenced and completed over time, for example, in connection with test environment specifications
  + Time spent on each task over time
* This will, of course, have to be compared to the estimates and schedule of the test analysis and design activities.

**Test Implementation and Execution**

The purpose of the test implementation is to organize the test cases in procedures and/or scripts and to perform the physical test in the correct environment.

The inputs on which this process is based are:

* Level test plan
* Test conditions and test design
* Other relevant documents
* The test object

The activities are:

* Organizing test procedures
* Design and verify the test environment
* Execute the tests
* Record the testing
* Check the test results

The output consists of:

* Test specification
* Test environment
* Test logs
* Incident reports
* Tested test object

1. **Input to Test Implementation and Execution**

* The input from the level test plan that we need for this process is:
  + Scheduling and staffing for the activities
  + Definition of the test object(s)
  + Specification of test environment
  + Entry criteria for the test execution
  + Exit criteria, including coverage criteria

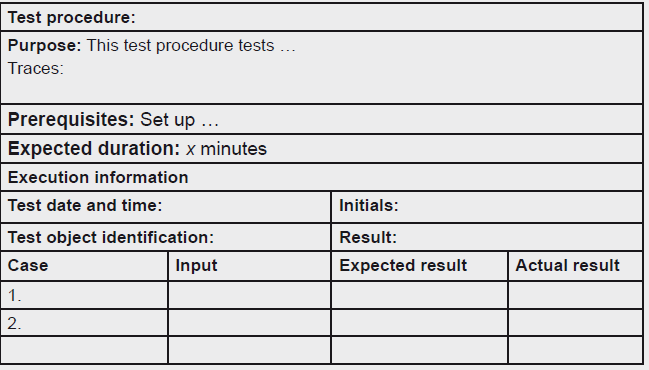
1. **Documentation of Test Implementation and Execution**

* The test specification is finished in this process where the test procedures are laid out.
* During this work the requirements concerning the test environment are finalized.
* The test environment must be established before the test execution may start. In some cases the test environment is explicitly documented.
* The test execution is documented in test logs. When failures occur these should be documented in incident reports.

1. **Activities in Test Implementation and Execution**
2. **Organizing Test Procedures:** The low-level test cases should now be organized and assembled in test procedures and/or test scripts

The documentation of a *test procedure* must at least include:

* Unique identification
* Description
* References to high-level test cases and/or to test conditions and/or directly to basis documentation to be covered by the procedure
* An explicit description of the preconditions to be fulfilled before the actual test execution can start
* Included low-level test cases

****

***Quality Assurance of the Test Specification****:*

* Before the test specification is used in the test execution it should be reviewed.
* Apart from the obvious benefits of having the test specification reviewed, it also has some psychological benefits.
* Usually we as testers review and test the work products of the analysts and developers, and we deliver feedback in the form of verbal or written review reporting and incident reports.

**2. Test Environment Specification and Testing**

* The test environment is a necessary prerequisite for the test execution—without a proper environment the test is either not executable at all or the results will be open to doubt.
* The environment is first outlined in the test plan based on the strategy.
* The test plan also describes by whom and when the test environment is to be created and maintained. Some additional requirements for the environment may be specified in the test specification in the form of prerequisites for the test procedures, and especially for test data
* The descriptions of the test environment must cover:
  + Hardware—to run on and/or to interface with
  + Software—on the test platform and other applications
  + Peripherals (printers including correct paper, fax, CD reader/burner)
  + Network—provider agreements, access, hardware, and software
  + Tools and utilities
  + Data—actual test data, anonymization, security, and rollback facilities
  + Other aspects—security, load patterns, timing, and availability
  + Physical environment (room, furniture, conditions)
  + Communication (phones, Internet, paper forms, paper, word processor)
  + Sundry (paper, pencils, coffee, candy, fruit, water)

1. **Checking Execution Entry Criteria**

* Even though we are eager to start the test execution we should not be tempted to make a false start.
* We need to make sure that the execution entry criteria are fulfilled.
* If the test object has not passed the entry criteria defined for it, do not start the test execution. You will waste your time, and you risk teaching the developers or your fellow testers that they don’t need to take the entry criteria seriously
* the testing process and the configuration management process, including:
  1. The ability to get the correct version of the test object, the test specification, and/or the ability to get the correct versions of any other necessary material
  2. The ability to be able to report the failures and other incidents found during the testing
  3. The ability to follow the progress of the failures and plan any necessary confirmation testing and regression testing
  4. The ability to register approval of successful removal of failures

1. **Test Execution**

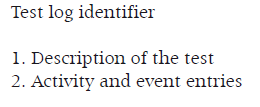
* The execution of the tests is what everybody has been waiting for: the moment of truth!
* In structured testing, in principle all the testers have to do during test execution is to follow the test specification and register all incidents on the way.
* If the execution is done by a tool, this is exactly what will happen.
* We have taken great care in writing the test procedures, and it is important to follow them. There are several reasons for this:
  1. we need to be able to trust that the specified testing has actually been executed.
  2. We need to be able to collect actual time spent and compare it with the estimates to improve our estimation techniques.
  3. We need to be able to compare the progress with the plan.
  4. We need to be able to repeat the tests exactly as they were executed before for the sake of confirmation testing and regression testing.
  5. It should be possible to make a complete audit of the test.

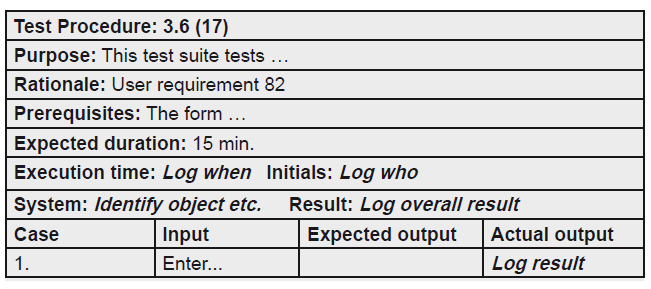
1. **Identifying Failures**

* For each test case we execute the actual result should be logged and compared to the expected result, defined as part of the test case.
* This can be done in various ways depending on the formality of the test. For fairly informal testing a tick mark, √, is sufficient to indicate when the actual result matched the expected result.
* For more formal testing, for example, for safety-critical software, the authorities require that the actual result is recorded explicitly.
* This could be in the form of screen dumps, included reports, or simply writing the actual result in the log.
* This type of logging may also serve as part of the proof that the test has actually been executed

1. **Test Execution Logging**

* As we execute, manually or by the use of tools, we must log what is going on.
* We must record the precise identification of what we are testing and the test environment and test procedures we use.
* We must also log the result of the checking, as discussed above. Last but not least we must log any significant event that has an effect on the testing.
* The IEEE 829 standard suggests the following contents of a test log:

****

****

1. **Confirmation Testing and Regression Testing**

* During testing we get failures. In most cases the underlying defects are corrected and the corrected test object is handed over to the testers for confirmation.
* This is the situation where we iterate in the test process and go back to the test execution process.
* We go back to perform confirmation testing and regression testing.
* Confirmation testing and regression testing are important activities in test execution.
* They can appear in all the test levels from component testing to (one hopes rarely) acceptance testing and even during maintenance of a product in operation.
* These two types of change-related testing have one thing in common: they are executed after defect correction. Apart from that, they have very different goals.
* *Confirmation Testing*
  + - Confirmation testing is the first to be performed after defect correction. It is done to ensure that the defect has indeed been successfully removed.
* *Regression Testing*
* Regression testing may—and should—then be performed.
* Regression testing is repetition of tests that have already been performed without problems to ensure that defects have not been introduced or uncovered as a result of the change.
* In other words it is to ensure the object under test has not regressed.

**Metrics for Implementation and Execution**

test implementation and execution activities may include:

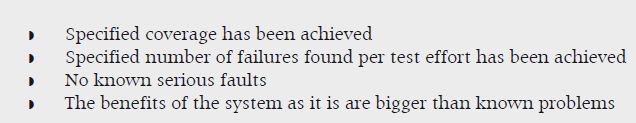
* Number of created test environments over time
* Number of created test data over time
* Number of created test procedures over time
* Number of initiated test procedures over time
* Number of passed test procedures over time
* Number of failed test procedures over time
* Number of passed confirmation tests over time
* Number of test procedures run for regression testing over time
* Time spent on the various tasks

**Evaluating Exit Criteria and Reporting**

* Test execution, recording, control, retesting, and regression testing must be continued until we believe that the exit criteria have been achieved. All the way we need to follow what is going on.
* The inputs on which this process is based are:
  + Test plan
  + Measurements from the test development and execution processes
* The activities are:
  + Comparing actual measurements with estimates and planned values
  + Reporting test results
* The output consists of:
* Presentation of test progress
* Test report
  + - 1. **Input to Test Progress and Completion Reporting**
* The input from the level test plan that we need for this process is:
  + Scheduling and staffing for the activities
  + Exit criteria
    - 1. **Documentation of Test Progress and Completion Reporting**
* The documentation of the progress must be presented in various ways according to who is receiving it.
* The audience may be the customer, higher management, project management and participants, and testers.
  + 1. **Activities in Test Progress and Completion Reporting**

The activities related to the test progress and completion reporting are

* 1. **Checking for Completion**
* A check against the test exit criteria is mandatory before we can say that the testing is completed at any level.
* To warrant a stop it is important to ensure that the product has the required quality.
* The exit criteria are tightly connected to the coverage items for the test, the test case design techniques used, and the risk of the product.
* The exit criteria therefore vary from test level to test level.



* + 1. **Metrics for Progress and Completion Reporting**

Metrics to be defined for the progress and control activities themselves may include:

* Number of tasks commenced over time
* Task completion percentage over time
* Number of task completed over time.
* Time spent on each task over time

**Test Closure**

* The purpose of the test closure activities is to consolidate experience and place test ware under proper control for future use.
* The inputs on which this process is based are:
* Level test plan
* Test ware, including test environment
* The overall procedure consists of the activities:
* Final check of deliveries and incident reports
* Secure storage/handover of test ware
* Retrospection
* The output generated in this process is:
*  Test experience report
*  Configuration management documentation

**Input to Test Closure**

* The input from the test plan that we need for this process is:
* Scheduling and staffing for the activities
* Planned deliveries

**Documentation of Test Closure**

* The documentation from this process is an experience report or a retrospective report from the retrospective meeting.

**Activities in Test Closure**

**Check Completion Again**

* Before we definitively close the door to the testing assignment we need to make extra sure that we have met the part of the exit criteria.
* This is both in terms of test coverage and deliveries we are to produce.
* If this is not in order or any discrepancies not clearly documented we‘ll have to make sure it is before we proceed.

Delivering and Archiving Test Ware

* The test ware we have produced are valuable assets for the organization and should be handled carefully.
* For the sake of easy and economically sound future testing in connection with defect correction and development of new versions of the product we should keep the assets we have produced.
* *It is a waste of time and money not to keep the test ware we have produced.*

Retrospective Meeting

* The last thing we have to do is to report the experiences we have gained during our testing.
* The measurements we have collected should be analyzed and any other experiences collected and synthesized as well.
* It is important that we as testers finish our testing assignment properly by producing an experience report.
* For the sake of the entire process improvement activity, and hence the entire organization, it is important that higher management is involved and asks for and actively uses the test experience reports. Otherwise, the retrospective meetings might not be held, because people quickly get engrossed in new (test) projects and forget about the previous one.

**Metrics for Test Closure Activities**

* Metrics to be defined for these activities may include number of tasks commenced over time, task completion percentage over time, number of tasks completed over time, and time spent on each task over time as for the other processes.