CHAPTER

3

Test Management

Contents

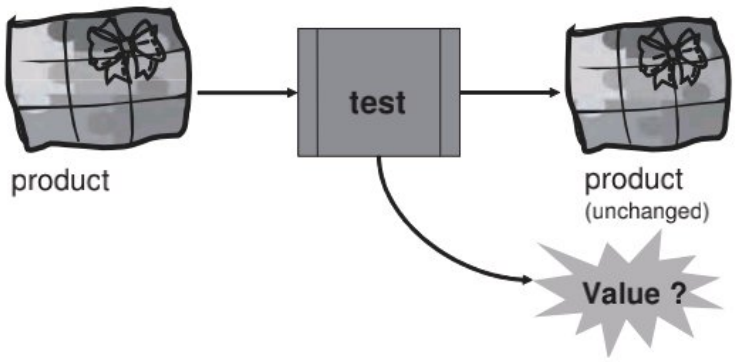
Test management is the art of planning and directing a test assignment to success. It is in many ways like project management, and yet not quite the same.

Test management must be done in close cooperation with project management, sometimes by the same person, sometimes by different people.

The test manager is the link between the test team and the development team and between the test team and higher management. It is therefore essential that the test manager is the ambassador of testing and truly understands how testing contributes to the business goals.

3.1 Business Value of Testing

On the face of it, testing adds no value. The product under testing is—in principle—not changed after the test has been executed.



But we are paid to test, so we must add some value to be in business. And we do!

The business value of testing lies in the savings that the organization can achieve from improvements based on the information the testing provides.

Improvements can be obtained in three places:

- The product under development
- The decisions to be made about the product
- The processes used in both testing and development

It may, however, sometimes be difficult to understand and express what the value is, both to ourselves and to others in the organization.

It is essential that test managers know and understand the value of testing and know how to express it to others to make them understand as well. Test managers must communicate the value to the testers, to other project participants, and to higher management.

Testers are often engrossed in the testing tasks at hand and don't see the big picture they are a part of; higher management is often fairly remote from the project as such and doesn't see the detailed activities.

3.1.1 Purpose of Testing

What testing does and therefore the immediate *purpose of testing is getting information about the product under testing*. We could say (with Paul Gerrard, founder of Aqastra): Testing is the intelligence office of the company.

The places we gather our raw data from are the test logs and the incident reports, if these are used sensibly and updated as the testing and the incident are progressing. From the raw data we can count and calculate a lot of useful quantitative information.

A few examples of such information are:

- Number of passed test cases
- Coverage of the performed test
- Number and types of failures
- Defects corrected over time
- Root causes of the failures

Most of this information is "invisible" or indigestible unless we testers make it available in appropriate formats. There is more about this in Section 3.5. This section also discusses how the information can be used to monitor the progress of the development in general and the testing in particular.

3.1.2 The Testing Business Case

It is not straightforward to establish a business case for testing, since we don't know in advance what savings we are going to enable. We don't know how many defects in the product we are going to unveil.

A well-established way to express the value of testing for the product is based on the cost of quality. This can be expressed as value of product improvement:

Value of product improvement =
(cost of failure not found – cost failure found) – cost of detection
To this we can add
Value of decision improvement =
(cost of wrong decision – cost of right decision) – cost of getting decision basis
Value of process improvement =
(cost using old process – cost using better process) – cost of process improvement

These three aspects add up to form the entire business case for testing. The aim is to get as high a value as possible.

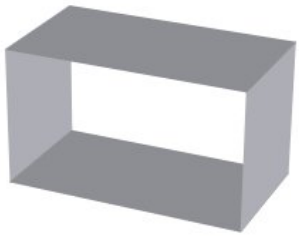
A value may be expressed either quantitatively or qualitatively. Quantitative values can be expressed in actual numbers—euros, pounds, or dollars or numbers of something, for example. Qualitative values cannot be calculated like that, but may be expressed in other terms or "felt."

3.1.2.1 The Value of Product Improvement

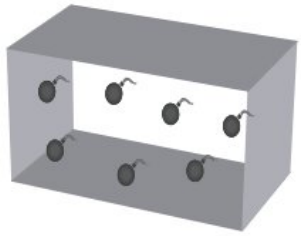
The value of product improvement is the easiest to assess.

One goal of all development is reliability in the products we deliver to the customers. Reliability is the probability that software will not cause the failure of a system for a specified time under specified conditions.

A product's reliability is measured by the probability that faults materialize in the product when it is in use.



No faults
= 100% reliability



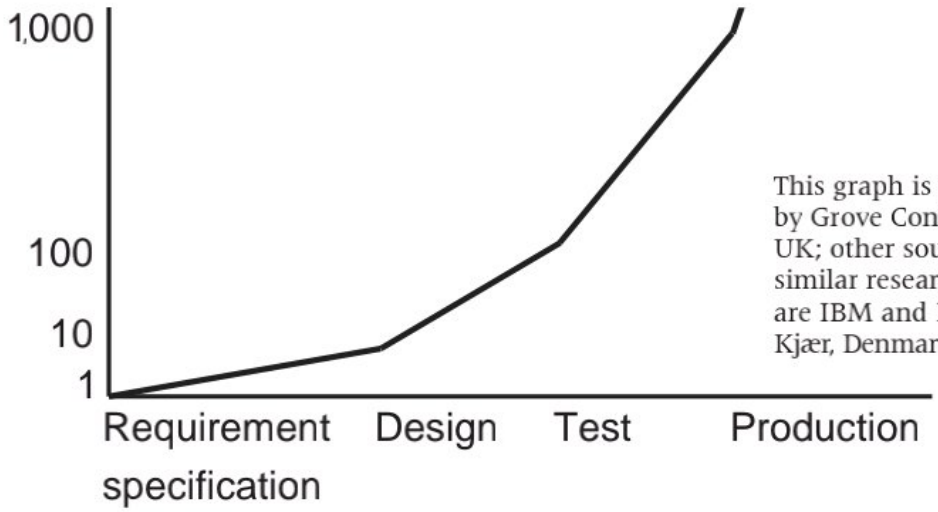
Many faults
= *x*% reliability

The less failures that remain in the product we release, the higher is the reliability of the product and the lower the risk of the product failing and thereby jeopardizing its environment. Project risks range from ignorable to endangering the lives of people or companies. There is more about risk management in Section 3.6.

The earlier we get a defect removed the cheaper it is. Reviews find defects and dynamic testing finds failures, and this enables the correction of the underlying defects.

The cost of the defect correction depends on when the defect is found. Defects found and corrected early are much cheaper to correct than defects found at a later point in time. Research shows that if we set the cost of correcting a defect found in the requirements specification to 1 unit, then it will cost 10 units to make the necessary correction if the defect is first found in the design.

If the defect remains in the product and is not found until encountered as a failure in dynamic test, it costs 100 units to correct it. The failures found during development and testing are called internal failures, and they are relatively cheap.

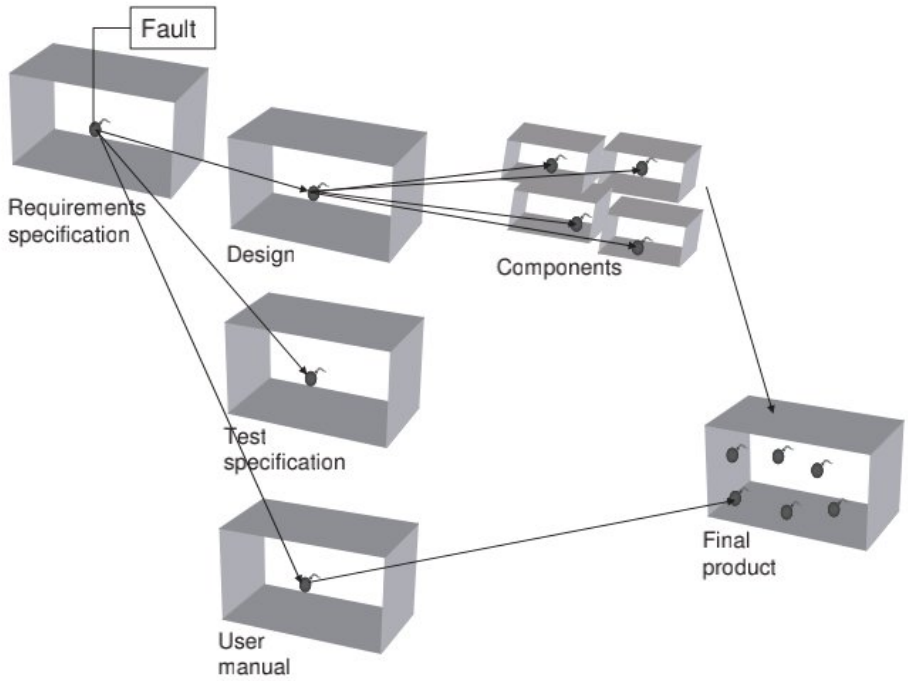


This graph is provided by Grove Consultants, UK; other sources for similar research results are IBM and Brüel & Kjær, Denmark.

If the customer gets a failure in production—an external failure—it may costs more than 1,000 units to make the necessary corrections, including the cost that the customer may incur. The analysts and programmers who can/must correct the defects may even be moved to new assignments, which are then in turn delayed because of (emergency) changes to the previous product.

The basic reason for this raise in cost is that defects in software do not go away if left unattended; they multiply. There are many steps in software

development from requirements specification to manufacturing and for each step a defect can be transformed into many defects.



There is some element of estimation in preparing the business case for product improvement. Many organizations don't know how many defects to expect, how much it costs to find defects, and how much it costs to fix them, or how much it would have cost to fix them later. The more historical data about the testing and defect correction an organization has, the easier it is to establish a realistic business case.

Ex.

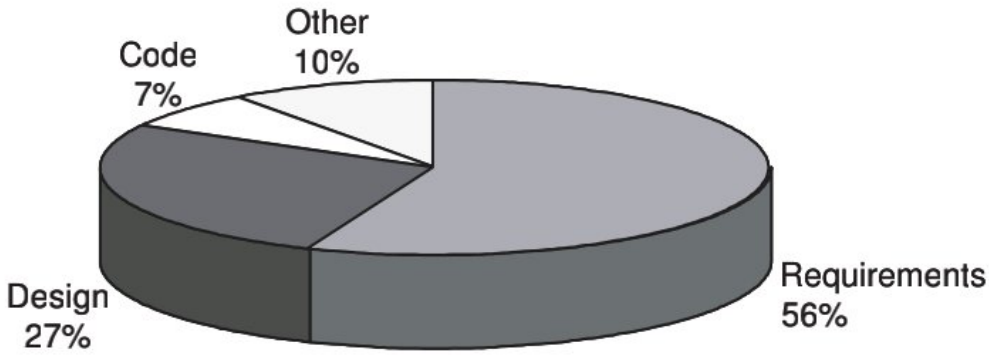
Let's look at a few calculation examples.

If we assume that it costs 4 units to correct a defect in the requirements phase, and 6 units to detect a defect or a failure, we can make calculations like:

Value of finding a defect in system testing rather than in production at the customer's site = (4,000 – 400) – 6 = 3,594 units

Value of finding a defect in requirements specification rather than in system testing = (400 – 4) – 6 = 390 units

Other research show that about 50% of the defects found in the entire life of a product can be traced back to defects introduced during the requirements specification work. This is illustrated in the following figure where the origins of defects are shown.



If we combine these two pieces of research results we have a really strong case for testing, and for starting testing early on in the project!

To get the full value of the test, it should start as early as possible in the course of a development project, preferably on day 1!

3.1.2.2 The Value of Decision Improvement

From the point of view of decision making such as decisions concerning release (or not) of a product the confidence in the product and quality of the decisions are proportional to the quality and the amount of the information provided by testing. As testing progresses, more and more information is gathered and this enhances the basis for the decisions.

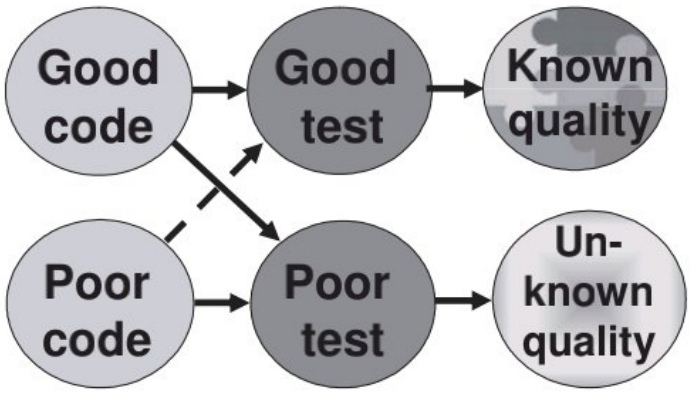
The more knowledge the decision makers have about what parts of the product have been tested to which depth—coverage—and which detected defects have been removed and which are still remaining, the more informed are the decisions made. The value of more informed decisions rather than less informed decisions is qualitative; it is very rarely possible to calculate this quantitatively.

It follows from the concept of test as an information collection activity that it is not possible to test good quality into a product. But the quality of the testing reflects in the quality of the information it provides.

Good testing provides trustworthy information and poor testing leave us in ignorance.

If the starting point is a good product, a good test will provide information to give us confidence that the quality is good.

If the starting point is a poor product, a good test will reveal that the quality is low.



But if the testing is poor we will not know if we have a good or a poor product.

The line from "poor code" to "good test" is dashed, because poor coding and good testing is not often seen together. Our goal as professional test practitioners is to reduce the occurrence of poor testing.

More important decisions may also be based on the information from testing. A test report with documentation of the test and the test results can be used to prove that we fulfilled contractual obligations, if needed. It may even in some (one hopes rare) cases provide a judicial shield for the company in that it provides evidence against suits for negligence or the like. This is of qualitative value to the business.

3.1.2.3 The Value of Process Improvement

From *the process improvement* point of view the information gained from testing is invaluable in the analysis of how well processes fit and serve the organization. The results of such analysis can be used to identify the process that could be the subject for process improvement. The process to improve may be both the testing process and other processes.

As time goes by the information can tell us how a process improvement initiative has worked in the organization.

When the testing process improves, the number of failures sent out to the customers falls, and the organization's reputation for delivering quality products will rise (all else being equal). The value of this is qualitative.

There is more about process improvement in Chapter 8.

3.2 Test Management Documentation

3.2.1 Overview

Proper test management requires that information about the decisions that test management makes is available and comprehensive to all stakeholders. These decisions are normally captured in a number of documents.

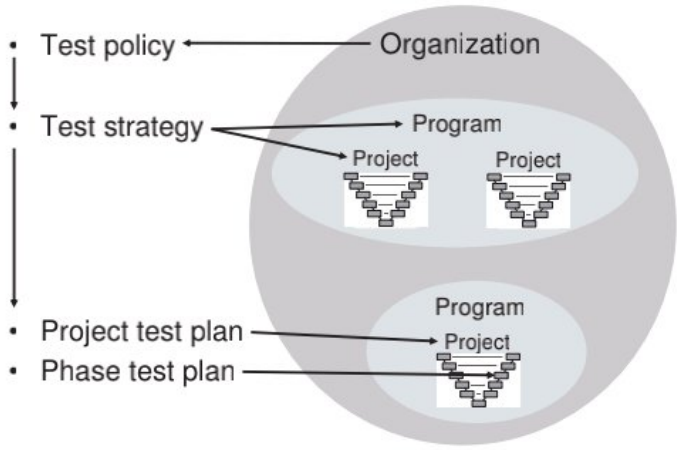
The test management documentation comprises:

- Test policy
- Test strategy
- Project test plan
- Level test plan

The test management documentation belongs to different organizational levels as shown in the next figure.

The *test policy* holds the organization's philosophy toward software testing.

The *test strategy* is based on the policy. It can have the scope of an organizational unit or a program (one or more similar projects). It contains the generic requirements for the test for the defined scope.



A *master test plan* is for a particular project. It makes the strategy operational and defines the test levels to be performed and the testing within those levels.

A *level test plan* is for a particular test level in a particular project. It provides the details for performing testing within a level.

The presentation of this documentation depends on the organization's needs, general standards, size, and maturity. The presentation can vary from oral (not recommended!) over loose notes to formal documents based on organizational templates. It can also vary from all the information being presented together in one document, or even as part of a bigger document, to it being split into a number of individual documents.

The more mature an organization is the more the presentation of the test management documentation is tailored to the organization's needs. The way the information is presented is not important; the information is.

3.2.2 Higher Management Documentation

Higher management, that is management above project managers and test managers, is responsible for the two types of test management documentation discussed in this section. The documentation is used by everybody in the organization involved in testing.

3.2.2.1 Test Policy

The test policy defines the organization's philosophy toward software testing. It is the basis for all the test work in the organization. A policy must be behavior-regulating in the good way—*it is like a lighthouse for all the testing activties*. And like every lighthouse has its own signal, every organization must have its own policy, tailored to its specific business needs.

The test policy must be short and to the point.

It is the responsibility of the top management to formulate the policy. It may, however, be difficult for top management if managers are not familiar

with professional testing, so it is often seen that the IT department (or equivalent) steps in and develops the test policy on behalf of the management.

The test policy must include:

1. Definition of testing
2. The testing process to use
3. Evaluation of testing
4. Quality targets
5. Approach to test process improvement

The test policy applies to all testing. The policy must cover all test targets. This means that there must be a policy for:

- Testing new products
- Change-related testing
- Maintenance testing

Test Policy; Definition of Testing

The definition of testing is a brief statement formulating the overall purpose of the test in the organization.

"Checking that the software solves a business problem"
"Activity to provide information about the quality of the products"
"A tool box for minimization of the product risks"

Test Policy; The Testing Process

The testing process is an overview of the activities to be performed or a reference to a full description of the testing process.

"Development and execution of a test plan in accordance with departmental procedures and user requirements found on the intranet"

Another possibility is a reference to the test process defined in standards or other literature, for example the ISTQB syllabus, the test process on which this book based. This test process was described in detail in Chapter 2.

Test Policy; Evaluation of Testing

The evaluation of testing is the measurement to be made in order for the quality of the testing to be determined.

"The number of failures reported by the field is measured every three months."
"The cost of the fault correction done after release is measured."
"The customer satisfaction is measured once a year by means of a questionnaire sent out to 200 selected customers."

Test Policy; Quality Targets

The quality targets to be achieved should be expressed so that the measurements can be used to see if we reach the goals.

Examples are:

"No more than one high severity fault per 1,000 lines of delivered code to be found in the first six months of operation."

"The overall effectiveness of the test must be over 98% after the first three months in production."

"The customers must not be reporting more than three severity 1 failures during the first year of use."

"The system must not have a breakdown lasting longer than 15 minutes during the first six months in production."

Test Policy; Approach to Test Process Improvement

The organizational approach to test process improvement is the process to be used for learning from experiences. This would often be the same as the organization's general approach to software process improvement, but there might be a specific policy for the testing process improvement.

"A postproject workshop where all the observations during the test process are collected shall be held within the first month after turnover to production."

"Failure reports shall be analyzed to determine any trends in the faults found in system test."

"The root cause shall be found for every severity 1 and 2 fault found during testing, and improvement actions shall be determined."

3.2.2.2 Test Strategy

The Latin word "stratagem" means a plan for deceiving an enemy in war. The enemy here is not the developers, but rather the defects! The strategy is based on the test policy and should of course be compliant with it. The strategy outlines how the risks of defects in the product will be fought. It could be said to express the generic requirements for the test. The strategy comprises the navigation rules for the testing.

The test strategy is high-level, and it should be short. It should also be readily available to all with a stake in the testing within the scope of the strategy. The strategy could be issued in a document, but it would be a good idea to present it in table form on a poster or on the intranet in the organization.

A test strategy must be for a specified scope. The scope may be all the projects in an entire organization, a specific site or department, or a program (a number of similar projects).

The overall test strategy may be chosen among the following possible approaches to the testing:

- Analytical—Using for example a risk analysis as the basis
- Model-based—Using for example statistical models for usage
- Consultative—Using technology guidance or domain experts
- Methodical—Using for example checklists or experience
- Heuristic—Using exploratory techniques
- Standard-compliant—Using given standards or processes
- Regression-averse—Using automation and reuse of scripts

There is nothing wrong with mixing the approaches. They address different aspects of testing and more approaches can support each other.

We could, for example, decide:

The component test shall be structured in compliance with the tool used for component testing.

The component integration testing shall be bottom-up integration based on a design model and in compliance with standard xxx.

The system test shall be risk-based and structured and the initial risk analysis shall be supplemented with exploratory testing.

The decisions about approaches or overall strategies have a great influence on some of the decisions that have to be made for specific topics in the strategy.

Remember that the strategy must be short—the test approach is to be refined and detailed in the test plans.

The test strategy should not be "once-written-never-changed." As the experiences gained from finished testing activities are collected and analyzed, the results in terms of test process improvement initiatives must constantly be considered when the test strategy is formulated.

A test strategy for a defined scope could contain the following information:

Test strategy identifier

1. Introduction
2. Standards to use
3. Risks to be addressed
4. Levels of testing and their relationships
   For each level, as appropriate
   - 4.1 Entry criteria
   - 4.2 Exit criteria
   - 4.3 Degree of independence
   - 4.4 Techniques to use
   - 4.5 Extent of reuse
   - 4.6 Environments
   - 4.7 Automation
   - 4.8 Measurements
   - 4.9 Confirmation and regression testing
5. Incident management
6. Configuration management of testware
7. Test process improvement activities

Approvals

The numbered topics are indented as sections in the strategy. The identifier and the approvals are information about the strategy usually found on the front page.

The strategy identifier is the configuration management identification information for the strategy itself. It could be formed by the:

- Name of the strategy
- Organizational affiliation
- Version
- Status

This should adhere to the organization´s standards for configuration management, if there is one.

Strategy; Introduction

The introduction sets the scene for the strategy. It contains general information of use to the reader.

The introduction is usually the most organization specific chapter of the plan. It should be based on the organization's own standard. It should in any case cover:

- Purpose of the document
- Scope of the strategy
- References to other plans, standards, contracts, and so forth
- Readers' guide

Strategy; Standards to Be Complied With

In this section references to the standard(s) that the test must adhere to are provided.

Standards may be both external to the organization and proprietary standards.

Standards are very useful. Many people with a lot of experience have contributed to standards. Even though no standard is perfect and no standard fits any organization completely standards can facilitate the work by providing ideas and guidelines.

The more standards it is possible to reference the easier the work in the strategy, the planning, and the specification. Information given in standards must not be repeated in specific documents, just referenced.

IEEE 829, Test Documentation.
"Test-Nice"—the company standard for test specifications.

Some appropriate standards are discussed in Chapter 8.

Strategy; Risks

The basis for the strategy can be the product risks to mitigate by the testing. Appropriate project risks may also be taken into consideration.

The strategy must include a list of the relevant risks or a reference to such a list.

Risks in relation to testing are discussed in Section 3.8.

Strategy; Test Levels and Their Relationships

The typical test strategy will include a list and description of the test levels into which the test assignments within the scope should be broken.

The levels can for example be:

- Component testing
- Component integration testing
- System testing
- System integration testing
- Acceptance testing

The levels are described in detail in Chapter 1.

The following strategy topics must be addressed for each of the levels that the strategy includes. It is a good idea to give a rationale for the decisions made for each topic, if it is not obvious to everybody.

Strategy; Level Entry Criteria

This is a description of what needs to be in place before the work in the test level can start.

The strictness of the entry criteria depends on the risk: The higher the risk the stricter the criteria.

An entry criterion for the system test could be that the system requirements specification has passed the first review.

Strategy; Level Exit Criteria

The testing exit or completion criteria are a specification of what needs to be achieved by the test. It is a guideline for when to stop the testing—for when it is "good enough." It is a description of what needs to be in place before the work in the test level can be said to be finished.

Testing completion criteria represent one of the most important items in a comprehensive test strategy, since they have a great influence on the subsequent testing and the quality of a whole system.

Some completion criteria are closely linked to the chosen test case design techniques; some are linked to the progress of the test.

The strategy does however not need to be very specific. The completion criteria will be detailed and made explicit in the test plans. A detailed discussion of completion criteria is found in Section 3.2.3.3.

The strictness of the completion criteria depends on the risk as described above.

Descriptions of the strategy for completion criteria could for example be:

Component test: Decisions coverage must be between 85% and 100% dependent on the criticality of the component. No known faults may be outstanding.

System test: At least 95% functional requirements coverage for priority 1 requirements must be achieved. No known priority 1 failures may be outstanding.

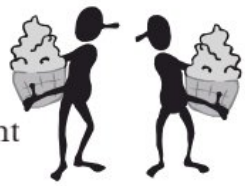
The test report has been approved by the project manager.

Strategy; Degree of Independence

Testing should be as objective as possible. The closer the tester is to the producer of the test object, the more difficult it is to be objective.

The concept of independence in testing has therefore been introduced.

The degree of independence increases with the "distance" between the producer and the tester. These degrees of independence in testing have been defined:



1. The producer tests his or her own product
2. Tests are designed by another nontester team member
3. Tests are designed by a tester who is a member of the development team
4. Tests are designed by independent testers in the same organization
5. Tests are designed by organizationally independent testers (consultants)
6. Tests are design by external testers (third-party testing)

As it can be seen in the list the point is who designs the test cases. In structured testing the execution must follow the specification strictly, so the degree of independence is not affected by who is executing the test. In testing with little or no scripting, like exploratory testing, the independence must be between producer and test executor.

The strategy must determine the necessary degree of independence for the test at hand. The higher the risk the higher the degree of independence.

There is more about independence in testing in Section 10.4.

Strategy; Test Case Design Techniques to Be Used

A list of the test case design techniques to be used for the test level should be provided here. The choice of test case design techniques is very much dependent on the risk—high risk: few, comprehensive techniques to choose from; low risk: looser selection criteria.

Test case design techniques could be equivalence partitioning, boundary value analysis, and branch testing for component testing.

Test case design techniques are described in great detail in Chapter 4.

Strategy; Extent of Reuse

Reuse can be a big money and time saver in an organization.

Effective reuse requires a certain degree of maturity in the organization. Configuration management needs to be working well in order to keep track of items that can be reused.

This section must provide a description of what to reuse under which circumstances.

Work product for reuse could, for example, be:

- Generic test specifications
- Specific test environment(s)
- Test data

Strategy; Environment in Which the Test Will Be Executed

Generic requirements for the test environment must be given here. The specific environment must be described in the test plan, based on what the strategy states.

The requirements for the test environment depend very much on the degree of independence and on the test level at which we are working.

We could for example find:

Component testing: The developer's own PC, but in a specific test area.

System test: A specific test environment established on the test company's own machine and reflecting the production environment as closely as possible.

Strategy; Approach to Test Automation

This is an area where the strategy needs to be rather precise in order for tool investments not to get out of hand.

Technical people—including testers—love tools. Tools are very useful and can ease a lot of tedious work.

Tools also cost a lot both in terms of money over the counter and in terms of time to implement, learn, use, and maintain. Furthermore, no single tool covers all the requirements for tool support in a test organization, and only a few tools are on speaking terms. It can be costly and risky, or indeed impossible to get information across from one tool to another.

It is important that the strategy includes a list of already existing testing tools to be used, and/or guidelines for considerations of implementation of new tools.

Test tools are described in Chapter 9.

Strategy; Measures to Be Captured

In the test policy it has been defined how the test shall be evaluated. It has also been defined what the approach to process improvement is. This governs the measures we have to collect.

Measures are also necessary to be able to monitor and control the progress of the testing. We need to know how the correspondence is between the reality and the plan. We also need to know if and when our completion criteria have been met.

Based on this, this section must contain a definition of all the metrics for testing activities. Descriptions of metrics include scales, ways of capturing the measurements, and the usage of the collected information.

Metrics and measurements are discussed in general in Section 1.3.

Strategy; Approach to Confirmation Testing and Regression Testing

Confirmation testing is done after fault correction to confirm that the fault has indeed been removed.

Fault correction is NOT part of the test process.

Regression testing should be done whenever something has changed in the product. It is done to ensure that the change has had no adverse effect on something that was previously working OK. Regression testing should follow any confirmation test; it should also for example follow an upgrade of the operation system underlying the product. The amount of regression testing to perform after a change is dependent on the risk associated with the change.

This section must outline when and how to perform confirmation testing and regression testing in the test level it is covering.

System testing: Re-execute the test case(s) that identified the fault and rerun at least 1/3 of the rest of the already executed test cases. The choice of test cases to rerun must be explained.

Strategy; Approach to Incident Management

It is hoped that a reference to the configurations management system is sufficient here.

If this is not the case it must be described how incidents are to be reported and who the incident reports should be sent to for further handling.

Close cooperation with the general configuration management function in the organization is strongly recommended on this. There is no need to reinvent procedures that others have already invented.

Strategy; Approach to Configuration Management of Testware

Configuration management of testware is important for the reliability of the test results. The test specification and the test environment including the data must be of the right versions corresponding to the version of product under testing. A good configuration management system will also help prevent extra work in finding or possibly remaking testware that has gone missing—something that happens all too often in testing.

Configuration management is a general support process, and if a configuration management system is in place this is of course the one the testers should use as well, and the one to which the strategy should refer.

If such a system is not in place the approach to local testing configuration management must be described. Configuration management is discussed in Section 1.1.3.

Strategy; Approach to Test Process Improvement

This could be a refinement of the approach described in the policy; see Section 3.1.1.5.

3.2.3 Project Level Test Management Documentation

The two types of test management documentation discussed in this section belong to a particular project. The master test plan should be produced by the person responsible for testing on the project, ideally a test manager. The level test plans should be produced by the stakeholder(s) carrying the appropriate responsibility. This could be anybody from a developer planning a component test to the test manager planning the system or acceptance test.

3.2.3.1 Master Test Plan

The plan outlines the journey.

The master test plan documents the implementation of the overall test strategy for a particular project. This is where the strategy hits reality for the first time. The master test plan must comply with the strategy; any noncompliance must be explained.

The master test plan must be closely connected to the overall project plan, especially concerning the schedule and the budget! The master test plan should be referenced from the project plan or it could be an integrated part of it.

The master test plan has many stakeholders and missions, and it must at least provide the information indicated in the following list to the main stakeholders.

Stakeholder	Information
All	Test object = scope of the test for each level
	Involvement in the testing activities
	Contribution to the testing activities
	Relevant testing deliverables (get/produce)
Management	Business justification and value of testing
	Budget and schedule
	Test quantity and quality
Development	Expectation concerning delivery times
	Entry criteria for deliverables
Test team	Test levels
	Schedule
	Test execution cycles
	Suspension criteria and exit criteria
Customer	Test quantity and quality

All stakeholders in the master test plan must agree to the contents according to their interest and involvement—otherwise the plan is not valid!

As mentioned previously, the way the information in the master test plan is presented is not important; the information is.

The detailed structure and contents of a master test plan are discussed in Section 3.2.3.3.

3.2.3.2 Level Test Plan

A level test plan documents a detailed approach to a specific test level, for example a component test or acceptance test. The level test plan describes the implementation of the master test plan for the specific level in even more precise detail. For instance, it would normally include a sequence of test activities, day-to-day plan of activities, and associated milestones.

The size of a level test plan depends on the level it covers—a component test plan for single components may be just 5–10 lines; system test plans may be several pages.

As for the master test plan it is vital to include all relevant stakeholders in the planning process and to get their sign-off on the plan.

3.2.3.3 Test Plan Template

The structure of the test plans, both the master test plan and any level test plans, should be tailored to the organization's needs.

In order not to start from scratch each time it is, however, a good idea to have a template. A template could be based on the IEEE 829 standard. This standard suggests the following contents of a test plan:

Test plan identifier

1. Introduction (scope, risks, and objectives)
2. Test item(s) or test object(s)
3. Features (quality attributes) to be tested
4. Features (quality attributes) not to be tested
5. Approach (targets, techniques, templates)
6. Item pass/fail criteria (exit criteria including coverage criteria)
7. Suspension criteria and resumption requirements
8. Test deliverables (work products)
9. Testing tasks
10. Environmental needs
11. Responsibilities
12. Staffing and training needs
13. Schedule
14. Risks and contingencies

Test plan approvals

The numbered topics are intended as sections in the plan; the identifier and the approvals for the plan are usually found on the front page.

The test plan identifier is the configuration management identification information for the test plan itself. It could be formed by the

- Name of the plan
- Organizational affiliation
- Version
- Status

This should adhere to the organization's standards for configuration management, if there is one.

Test Plan; Introduction

The introduction sets the scene for the test plan as a whole. It contains general information of use to the reader.

The introduction is usually the most organization-specific chapter of the plan. It should be based on the organization's own standard. It should in any case cover:

- Purpose of the document
- Scope of the plan, possibly including intended readership
- References to other plans, standards, contracts, and so forth
- Definitions
- Abbreviations

- Typographical conventions used
- Readers' guide

It is important to get the references precise and correct. Test planning is influenced by many aspects, including the organization's test policy, the test strategy, the development or maintenance plan, risks, constraints (time, money, resources), and the test basis and its availability and testability. References must be made to all this information—and it must be respected.

If this chapter gets too voluminous you can place some of the information in appendices.

Test Plan; Test Item(s)

Here the test object(s) or item(s) and additional information are identified as precisely and explicitly as possible. The additional information may be the appropriate source specification, for example detailed design or requirements specification, and helpful information such as the design guide, coding rules, checklists, user manual, and relevant test reports.

The test object depends on whether the plan is the master test plan or a level test plan, and in the latter case of the specific test level the plan is for.

Product XZX V2.3, based on XZX System Requirements Specification V.4.2.
The individual component: pre_tbuly V2.3.

Test Plan; Features to Be Tested

Within the scope of the test item(s), an overview of the features to be tested is provided along with references to where the test is specified, or will be specified as the case may be. Features include both functional and nonfunctional quality attributes.

The decision about which features are to be tested and which are not is based on the applicable test strategy, the identified risks, and the mitigation activities for them. The identification of the features to be tested is also closely linked to the specified coverage items.

All functional requirements, specified in System test specification STS-XX.doc must be covered in this test.

All methods in the classes are to be tested in the component testing.

Test Plan; Features Not to Be Tested

To set the expectations of the stakeholders correctly, it is just as important to state what features are not tested as it is to state which are.

With regards to what might be expected to be tested in relation to the test item(s), we must provide a list of the features not to be tested. A reason must be given for each of the features omitted.

Performance testing is not part of this test because it will be carried out by third-party company PTESTIT. They are experts in performance testing.

In this component the function M-bladoo is not tested. It is too costly to simulate the error situation that it handles. A formal inspection has been performed on the code.

Test Plan; Approach

The test approach must be based on the strategy for the test at hand. This section expands the approach and makes it operational.

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.

Test Plan; Item Pass/Fail Criteria

The item pass/fail criteria are the American counterpart to what we Europeans call 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.

Examples of appropriate completion criteria for some test levels are:

- Component testing
  - 100% statement coverage
  - 95% decision coverage
  - No known faults
- Acceptance testing
  - 100% business procedure coverage
  - No known failures of criticality 1

Test Plan; Suspension Criteria and Resumption Requirements

Sometimes it does not make sense to persevere with the test execution. It can be a very good idea to try to identify such situations beforehand. In this section in the plan, the circumstances that may lead to a suspension of the test for a shorter or longer period are described.

More than 20% of the time is spent on reporting banal failures, caused by faults that should have been found in an earlier test phase.

It must also be decided and documented what must be fulfilled for the test to be resumed.

Evidence of required coverage of component testing must be provided.

Finally it should be stated which test activities must be repeated at resumption. Maybe every test case must be re-executed; maybe it is OK to proceed from where we stopped.

Test Plan; Test Deliverables

The deliverables are a listing and a brief description of all the documentation, logs, and reports that are going to be produced in the test process at hand. Everything must be included for the purpose of estimation and the setting of expectations.

Example of test deliverables are:

- Test plans
- Test specifications
- Test environment
- Logs, journals, and test reports
- Release documentation for the test object

Test Plan; Testing Tasks

This section in the plan is the work breakdown structure of the test process at hand. If we use the test process used here, it is analysis, design, implementation, execution, evaluation, reporting, and closure, all broken down into more detailed activities in an appropriate work breakdown structure. When defining the test tasks in detail it is important to remember and mention everything. Even the smallest task, which may seem insignificant, may have a significant influence on the schedule.

The tasks, together with resources and responsibilities, are input items to the test schedule.

Test Plan; Environmental Needs

The test environment is a description of the environment in which the test is to be executed. It is important to be as specific as possible in order to get the right test environment established at the right time (and at the right cost).

Test Plan; Responsibilities

In this section we must describe who is responsible for what. The distribution of testing roles or tasks on organizational units or named people can be shown in a responsibility distribution matrix (RDM). This is a simple two-dimensional matrix or table. On one axis we have organizational units or people, on the other axis we have roles or tasks. In the cross-field we can indicate the type of involvement an organizational unit has for the role.

A completed responsibility distribution matrix might look like this.

	1	2	3	4	5	6
Test leader	R	C	I	I	I	I
Test department	C	R	R	P	P	R
Quality assurance	C	C	R	-	-	I
Sales/marketing	C	C	C	-	-	P
The customer	C	C	C	-	R	P
Method department	I	I	P	R	-	-
Responsible **P**erforming **C**onsulted **I**nformed						

Where:
1. Test management
2. Test analysis and design
3. Test environment
4. Test tools
5. Test data
6. Test execution

Test Plan; Staffing and Training Needs

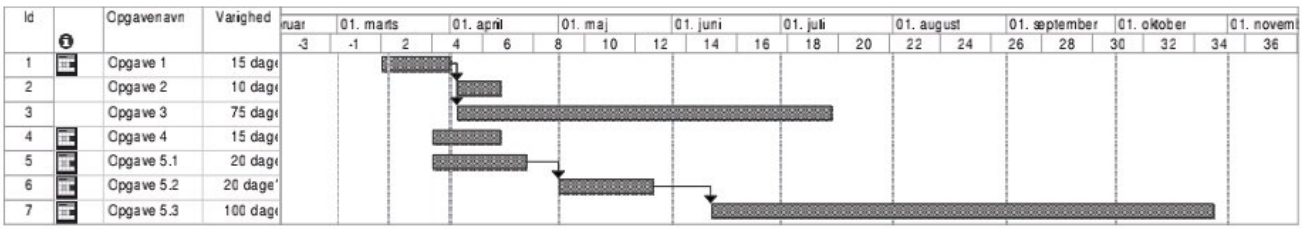
The necessary staff to fulfill the roles and take on the responsibilities must be determined and described here.

Each of the roles requires a number of specific skills. If these skills are not available in the people you have at your disposal, you must describe any training needs here. The training should then be part of the activities to put in the schedule.

Test Plan; Schedule

In the 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.

Scheduling testing is just like any other project scheduling. The result may be presented graphically, typically as Gantt diagrams.



Test Plan; Risks and Contingencies

This is the management of the risks specifically connected to the task of testing itself, not to the object under test.

The risks to consider here are hence:

- Project risks—What can jeopardize the plan
- Process risks—What can jeopardize the best possible performance of the tasks

The risks must be identified, analyzed, mitigated as appropriate, and followed up like any other risk management task.

Risk management is discussed in Section 3.5.

The *approvals* are the sign-off on the plan by the relevant stakeholders.

3.2.3.4 Scheduling Test Planning

Planning is important and planning takes time.

If you fail to plan—you plan to fail!

It is important to plan activities rather than just jump headfirst into action. The work on the planning provides a deeper understanding of the task at hand, and it is much easier to change something you have written down or sketched out on a piece of paper, than something that has already taken place in the real world.

Because planning takes time and because it is important, it should be planned so that it can start as early as possible. Take your planning seriously, so that you don't end up like this poster painter once did:

The benefits of starting test planning early are many:

- There is time to do a proper job of planning.
- There is more time to talk and/or negotiate with stakeholders.
- Potential problems might be spotted in time to warn all the relevant stakeholders.
- It is possible to influence the overall project plan.

When you plan you have to keep in mind that a plan needs to be SMART:

Specific—Make it clear what the scope is
Measurable—Make it possible to determine if the plan still holds at any time
Accepted—Make every stakeholder agree to his or her involvement
Relevant—Make references to additional information; don't copy it
Time-specific—Provide dates

The test plan should be reviewed and approved by all stakeholders to ensure their commitment. A plan is invalid without commitment from the contributors.

Remember that *a plan is just a plan*; it is not unchangeable once written. A plan must be a living document that should constantly be updated to reflect the changes in the real world. Contrary to what many people think it is not a virtue to keep to a plan at any cost—the virtue lies in getting the plan to align with the real world. No matter how hard you try, you are not able to see what is going to happen in the future.

You should always plan *The New Yorker* way: Adjust the detailing of the planning with the visibility at any given moment. When close to an activity provide many details; for activities further away provide fewer details.

As the time for the execution of activities approaches, more details can be provided, and the necessary adjustments done.

All this takes time and it should not be "invisible" work (i.e., work that is not scheduled reported anywhere). The same in fact holds true for the monitoring activities and for the test reporting.

3.2.3.5 Test Report

The purpose of test reporting is to summarize the results and provide evaluations based on these results.

A test report should be issued at the completion of each test level and the end of the entire testing assignment task. The test reports should include analysis of result information to allow management decisions, based on risk, on whether to proceed to the next level of test or to project implementation, or whether more testing is required. Top management may also need information for regularly scheduled project status meetings and at the end of the project in order to adjust policy and strategy.

Test policy
- Definition of testing
- Test process to use
- Test evaluation
- Quality targets
- Test process improvement activities

R

According to IEEE 829 the test report should contain:

Test report identifier

1. Summary
2. Variances
3. Comprehensiveness assessment
4. Summary of results

5. Evaluation
6. Summary of activities

Approvals

The *test report identifier* is the identification information for the report itself. As for all the other documentation, it could be formed by the name of the report, the organizational affiliation, the version, and the status.

If the report is to be placed under configuration management, the identification should adhere to the organization's standards for configuration management, if there is one.

Test Report; Summary

The summary provides an overview of test activities. This section could refer to the test plan. The summary should also include any conclusion. It should be possible to read the summary in isolation and get the main information about the test.

Test Report; Variances

The variances to be reported here are all incidents that have happened for any of the items used as a basis for the test. It must also include a summary of what was done and not done with regard to the original plan.

A variance could for example be the issue of a new version of the requirements specification after the approval of the test specification.

Test Report; Comprehensiveness Assessment

In this section we report whether we made it or not according to the original (or modified) plan. It should describe which of the planned tests were not performed, if any, and why not.

This is where we must describe how we met the original completion criteria. If they where modified, this is where we explain why.

Any statistically valid conclusions that can be drawn from these analyses could be used to predict the quality level achieved by the tested product. They can also be used to compare with the target level established in the test plans.

Test Report; Summary of Results

We must provide an overview of incidents found and incidents solved during the testing.

We can also list findings about which functions are working and which functions are not; or about which risks have been eliminated and which are still outstanding.

The evaluation sums up the expectations versus the actual findings. Any out-of-scope situations should also be documented as should outstanding issues.

We can draw conclusions regarding the quality of software by comparing the planned quality levels with the actual. We can also give recommendations, but it is not the responsibility of the testers to decide whether the test object should be released or not. That is a management decision—project management, product management, or even higher up.

Test Report; Evaluation

In this section we should give an overall evaluation of the test item, preferably based on a risk analysis of possible outstanding risks related to the release of the item.

The evaluation must be based on the result of the test compared to the completion criteria.

Test Report; Summary of Activities

Here we must provide an overview of the resource usage for the testing. This could be in terms of time used and other costs such as investments in tools.

The *approvals* here are the approvals of the *test report,* not of the test object.

3.3 Test Estimation

3.3.1 General Estimation Principles

Estimation is a prediction of how much time it takes to perform an activity. It is an approximate calculation or judgment, not something carved in stone. An estimate is typically based on the professional understanding of experienced practitioners.

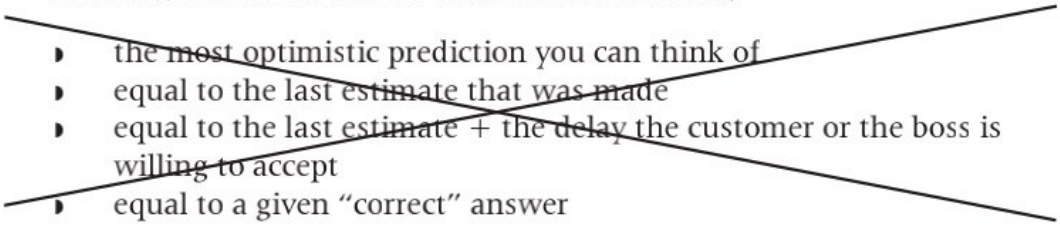
There are many ways in which to express estimations, but the best way is in hours. In that case we don't get problems with holidays, effective working hours, and so forth. You must never express estimates using dates; dates and estimates are incompatible.

Estimation is input to the scheduling. Only in that activity will we transform the estimated hours into dates.

We can also estimate other elements than just time, for example, number of test cases, number of faults to be found, and number of iterations in the test process needed to fulfill the completion criteria. We may also estimate any other costs, such as hardware and tools.

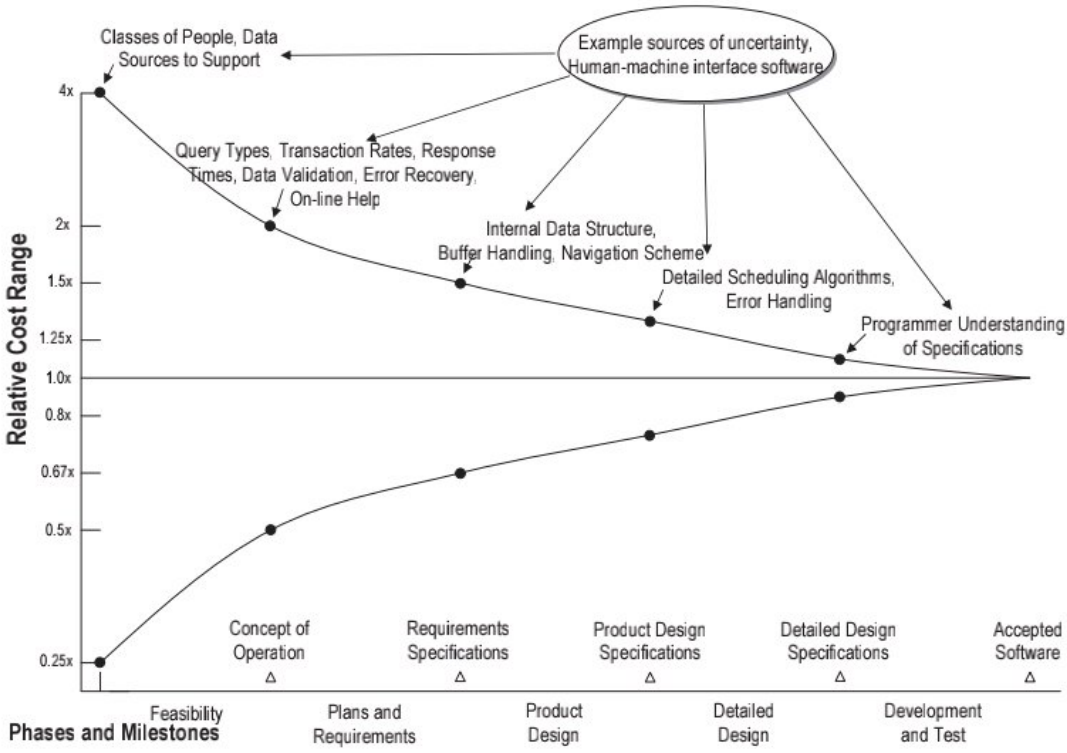
We should always take our estimations seriously. Be honest when you estimate, even though it is often easier to get forgiveness than permission. Keep your original estimates for future reference.

In line with this remember that estimation is not:



- the most optimistic prediction you can think of
- equal to the last estimate that was made
- equal to the last estimate + the delay the customer or the boss is willing to accept
- equal to a given "correct" answer

Estimates are predictions about the future and predictions are by definition uncertain. The closer we come to the actual result, the less is the uncertainty as illustrated here.



You should always calculate with an uncertainty in every estimate and document this uncertainty with the estimate. Furthermore, estimates should always be accompanied by the rationale or justification for the estimation values along with any assumptions and prerequisites.

3.3.2 Test Estimation Principles

Estimating test activities is in many ways like all other estimation in a project. We need to take all tasks, even the smallest and seemingly insignificant, into account.

The time to complete must be estimated for each task defined in the task section, including all the test process activities from test planning to checking for completion.

Even though estimation of testing tasks is in many ways identical to the estimation for any other process, there are also important differences. A test is a success if it detects faults—this is the paradox with which we have to deal.

The test estimation is different from other project estimations, because the number of failures is not known in advance—though it can be estimated as well. The number of necessary iterations before the completion criteria are met is usually not known either. As a rule of thumb, at least three iterations must be reckoned with—one is definitely not enough, unless the completion criterion is a simple execution of all test cases, and independent of the number of outstanding faults and coverage.

Nevertheless, we have to do our best. The estimation must include:

- Time to produce incident registrations
- Possible time to wait for fault analysis
- Possible time to wait for fault correction
- Time for retest and regression test (minimum three iterations!)

The reason why we have to cater for several iterations is that, well: "*Errare humanum est*!"

When we report incidents and the underlying faults are corrected by development or support staff, not all reported faults are actually corrected. Furthermore, fault correction introduces new faults, and fault correction unveils existing faults that we could not see before.

Experience in the testing business shows that *50% of the original number of faults remains after correction*. These are distributed like this:

Remaining faults after correction	20%
Unveiled faults after correction	10%
New faults after correction	20%

So if we report 100 faults, we have 20 + 20 + 10 = 50 faults to report in the next iteration, 10 + 10 + 5 = 25 faults in the third, and 5 + 5 + 2 = 12 in the forth.

These are general experience numbers. It is important that you collect your own metrics!

3.3.3 The Estimation Process

Estimation is a process like anything else we do. You should of course use your organization's standard process for estimation, if there is one. Otherwise, you can adapt an estimation procedure like the generic one described here.

1. Define the purpose of the estimation—Is this estimation the first approach, for a proposal, or for detailed planning?
2. Plan the estimating task—Estimation is not a two-minute task; set sufficient time aside for it.
3. Write down the basis for the estimation—Here the scope and the size of the work are determined, and all factors that may influence the estimates are registered. This includes factors related to the nature of the processes we are working by, the nature of the project we are working in, the people we are working with, and any risks we are facing.
4. Break down the work—This is the work breakdown (i.e., the listing of all the tasks to estimate). Do this as well as possible in relation to the purpose.
5. Estimate—Use more than one technique as appropriate.
6. Compare with reality and reestimate—This is the ongoing monitoring and control of how the work that we have estimated is actually going.

3.3.4 Estimation Techniques

The following estimation techniques are the most used and an expression of the best practice within estimation.

- FIA (finger in the air) or best guess
- Experience-based estimation
- Analogies and experts
- Delphi technique
- Three-point estimation (successive calculation)
- Model-based estimation
- Function points
- Test points
- Percentage distribution

3.3.4.1 Estimation; Best Guess (FIA)

This technique is more or less pure guesswork, but it will always be based on some sort of experience and a number of (unconscious) assumptions. The technique is very widespread, but since it is based on your gut feeling it is bound to be inaccurate. It is often not repeatable, and it is not always trusted.

The uncertainty contingency is probably around 200%–400% for estimates based on best guess. We can do better than that.

3.3.4.2 Estimation; Analogies and Experts

In the analogy techniques you base your estimate on something you have experienced before.

For example: "This looks very much like the system I tested in my previous job. That took us three months, and we were four people. This is slightly smaller and we are five people—so I guess this will take two months to complete."

If you have participated in a testing project that is comparable to the one you are estimating, you might use that as a baseline to do your estimation.

Analogies may also be based on metrics collected from previous tests. We may estimate the number of iterations of the test based on recent records of comparable test efforts. We can calculate the average effort required per test on a previous test effort and multiply by the number of tests estimated for this test effort.

Experts, in the estimation context, know what they are talking about and have relevant knowledge. It is almost always possible to find experts somewhere in the organization.

If experts on this kind of testing are available, then by all means make use of them. They have been there before, so they know what they are talking about.

3.3.4.3 Estimation; Delphi Technique

This is a simple technique that has proved remarkably resilient even in highly complex situations.

You must appoint an estimation group as appropriate. This can be stakeholders and/or experts in the tasks to estimate.

The steps in this estimation process are:

- Each member of the group gives an estimate.
- The group is informed about the average and distribution of the estimates.
- Those giving estimates in the lower quartile and in the upper quartile are asked to tell the rest of the group why their estimates were as they were.
- The group estimates again—this time taking the previous result and the provided arguments for the "extreme" estimates into account.
- This may continue two, three, four, or more times until the variation in the estimates is sufficiently small.

Usually the average of the estimations does not change much, but the variation is rapidly decreased. This gives confidence in the final estimation result.

The Delphi techniques can be used in many ways. The people taking part can be in the same room, but they may also be continents apart and the technique used via e-mail.

The technique can be combined with other techniques. Most often the participants give their initial estimates based on experience and/or they are experts in a specific area. The initial estimates may also be obtained using some of the other estimation techniques to make them even more trustworthy.

3.3.4.4 Estimation; Three-Point Estimation

Three-point estimation is a statistical calculation of the probability of finishing within a given time. The technique is useful for quantifying uncertainty to the estimate. The technique is also called successive calculation because tasks are broken down and the estimates successively calculated until the variance is within acceptable limits.

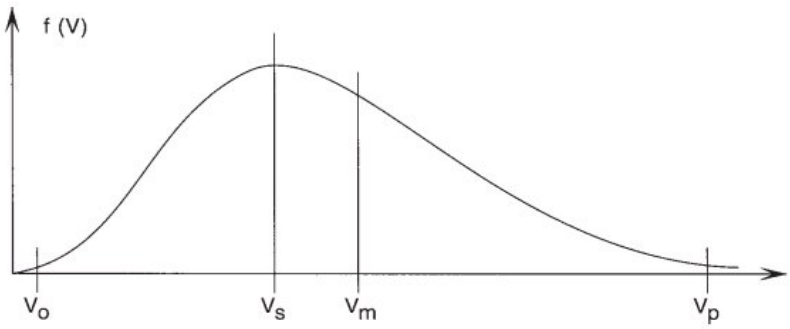
Three point estimation is based on three estimates:

- The most optimistic time (ideal conditions)
- The most likely time (if we do business as usual)
- The most pessimistic time (Murphy is with us all the way)

The three estimates to be used can be provided in a number of ways. One person can be asked to provide all of them, or a group, for example some of the test team members, can take part in the estimation. The estimates can be provided using the Delphi technique or other recognized techniques. High and low values may either be estimated separately (i.e., "what are the best and the worst cases?") or they may be the highest and the lowest of the individual estimates.

If the "best and worst and most likely" values are used, the estimators should be 99% sure that the actual value will fall between the low and the high values.

From these three estimates it is possible to define the distribution function for the time to finish. It could look like the figure shown where V_o = most optimistic; V_s = most likely; V_p = most pessimistic; and V_m = mean.



We can use the approximated formula to derive:
$V_m = (V_o + 3*V_s + V_p) / 5$
$S = (V_p - V_o) / 5$ (the standard deviation)
Based on this distribution we can calculate the time needed for any probability of finishing the task we want, by using the appropriate formula.

For the 5% interval the formulas are:
5%: Vm – 2S 95%: Vm + 2S
Let us say that for a testing task we have reckoned:
Vo = 70 hours Vs = 80 hours Vp = 110 hours
We calculate:
Vm = (Vo + 3*Vs + Vp)/5 = (70 + 3*80 + 110)/5 = 84
S = (Vp – Vo)/5 = (110 – 70)/5 = 8
The upper value in the 95% interval = 84 + 2 * 8 = 100
Therefore if we want to be 95% sure that we'll finish in time our estimate for the given task should be 100 hours.

All tasks or a selection of the most critical tasks can be estimated using this technique.

3.3.4.5 Estimation; Function Points

This technique is a factor estimation technique initially published by Albrecht in 1979. It has been revised several times, and it now maintained by IFPUG —International Function Points User Group. The group has been permanent since 1992. Version 4.0 of the technique was published in 1994.

The estimation is based on a model of the product, for example, a requirements specification and/or a prototype.

Five aspects of the product are counted from the model:

- External inputs
- External outputs
- External enquiries
- Internal logical files
- External interface files

The counts are then multiplied with a weight and the total of the weighted counts is the unadjusted sum. The actual effort in person hours is then calculated with an adjustment factor obtained from previous project data.

It requires some training to be able to count function points correctly. Continuous comparisons of actual time spent with the estimates are essential to get the best possible local adjustment factor.

The disadvantage of using function points is that they require detailed requirements in advance. Many modern systems are specified using use cases, and use cases are incompatible with this technique.

3.3.4.6 Estimation; Test Points

In 1999 Martin Pol et al. published a dedicated test estimation technique called test points as part of the TMAP method.

The technique is based on the function point technique, and it provides a unit of measurement for the size of the high-level test (system and acceptance tests) to be executed.

The technique converts function points into test points based on the impact of specific factors that affect tests, such as:

- Quality requirements
- The system's size and complexity
- The quality of the test basis (the document(s) the test is specified toward)
- The extent to which test tools are used

3.3.4.7 Estimation; Percentage Distribution

Unlike all the other techniques discussed here, this technique is a so-called top-down estimation technique. The fundamental idea is that test efforts can be derived from the development effort.

The estimation using this technique starts from an estimate of the total effort for a project. This estimate may be the result of the usage of appropriate estimation techniques at the project management level.

The next step is to use formulas (usually just percentages) to distribute this total effort over defined tasks, including the testing tasks. The formulas are based on empirical data, and they vary widely from organization to organization.

It is essential that you get your own empirical data and constantly trim it according to experiences gained.

If you do not have any data you could assume that the total testing effort is 25–30% of the total project effort. The testing effort should then be spread out on the test levels with an appropriate amount for each level.

This example is from Capers Jones *Applied software measurements*. It is for in-house development of administrative systems. The left-hand table shows the distribution of the total effort on overall tasks, including all tests as one task only. The right-hand table shows the distribution of the effort on detailed testing tasks (the terminology is that of Capers Jones.)

Activity	%
Requirements	9.5
Design	15.5
Coding	20
Test (all test phases)	27
Project management	13
Quality assurance	0
Configuration management	3
Documentation	9
Installation and training	3

All phases	%
Component testing	16
Independent testing	84
	100
Independent testing	%
Integration testing	24
System testing	52
Acceptance testing	24
	100
System testing	%
Functional system testing	65
Nonfunctional system testing	35
	100

3.3.5 From Estimations to Plan and Back Again

The estimation is done to provide input to the scheduling activity in the project planning.

In the scheduling we bring the estimates for the defined testing tasks together with the people, who are going to be performing the tasks. Based on the start date for the first task and the dependencies between the tasks we can then puzzle the tasks together and calculate the expected finishing date.

Estimations should be in hours. The scheduling provides the dates: dates for when the performance of each of the tasks should begin, and dates for when they are expected to be finished.

When defining the expected finish date for a task we need to take several aspects into account:

- The start and/or finish dates of others tasks that this task depends on to start, if any
- The earliest possible start date for the task
- The general calendar regarding public holidays
- The pure estimate for the time to finish the task
- The efficiency of the employee(s) to perform the task—typically 70–80% for a full time assignment
- The employee(s)'s availability—this should NOT be less than 25%

We should not expect that our estimations are accepted straightaway. Making plans for a development project is a very delicate balance between

resources (including cost), time, and quality of the work to be done. Testing is often on the critical path for a project, and testing estimates are likely to be the subject of negotiations between stakeholders—typically the customer or higher management, the project manager, and the test manager.

The estimating does not stop with the preparation of the first schedule. Once the actual testing has started—from the first planning activities and onwards, we need to frequently monitor how realities correspond to the estimates. Based on the new information gathered through the monitoring, we must re-estimate, when the deviations between estimates and reality get too large to stay in control. Only when all the testing activities are completed can we stop the monitoring and re-estimation.

3.3.6 Get Your Own Measurements

All estimates are based on experience—maybe very informally (FIA), maybe very formally (like function points). The better the basis for the estimation is, the better the estimation gets. Better estimation means more reliable estimations, and that is what we both, management and customers, want.

In order to get better estimates we need to collect actual data. The more empirical data we have, the better will the estimates be. In general we can say that (almost) any data is better than no data.

We do, however, always need to objectively evaluate the empirical data we have—is it collected from tasks that can be compared with the ones we are dealing with now? When we use the empirical data available, we also have an obligation to contribute to and refine the empirical data on an ongoing basis.

Empirical data for estimation is part of the measurements we are collecting. So we need to chip in to establish a set of simple measurements of time, costs, and size in all projects we participate in. This requires procedure(s) for collection of data to be established and maintained, and training of the (test) managers in these procedure(s).

From an organizational point of view it must be checked that all completed projects collect data, and the usage of these data must, of course, also be enforced.

3.4 Test Progress Monitoring and Control

Continuous monitoring of how the test is progressing compared to the plan is absolutely necessary to stay in control. If we don't control the test project, it will control us—and that is not a nice experience.

You need to collect information about facts, compare these with the estimates, and analyze the findings. This is needed to minimize divergence from the test plan. If there is any discrepancy you need to take action to keep in control, and you need to inform the stakeholders.

There are a few rules that you must adhere to when you do the follow-up on the actual activities. Follow-up must be guided by:

- Honesty
- Visibility
- Action

First of all you need to be honest, not only when you estimate, but also when you collect information about reality. In the long run you loose integrity and trust if you "tailor" the numbers, or come up with "political" results of the monitoring.

You also need to make the information visible to all stakeholders. Again you lose trust if you hide the truth, be it a positive truth (we are ahead of schedule) or a negative truth (we are behind schedule). Information about progress and findings must be made readily available to the stakeholders in appropriate forms.

The last thing you need to do to stay in control is to take action whenever needed. *It is your duty as test manager to intervene as soon as deviations appear!*

3.4.1 Collecting Data

The data to collect during testing should be specified in the approach section in the test plan, based on the requirements outlined in the policy and the strategy.

The concept of metrics and measurements is discussed in Section 1.3.

No matter which data we have planned to collect it is not enough to just collect it. It must be presented and analyzed to be of real value.

3.4.2 Presenting the Measurements

Test reports are used to communicate test progress. These reports must be tailored to the different recipients or stakeholders. The immediate stakeholders for test monitoring information are the customer, the project and/or product management (or higher), the test management, and the testers.

The customer and the management above test management need test reports (described below) when the test is completed. The test management needs information on a continuous basis to keep in control. The testers need to be kept informed on progress at a very regular basis—at least daily when the activities are at their peak.

A picture speaks a thousand words. The best way to present progress information for testing is by using graphics. This holds true for all stakeholders, though especially for the testers in the middle of the action. Graphics used in the right way give an immediate overview—or feeling—for the state of the testing.

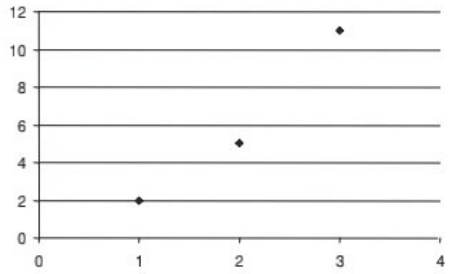
Sources:
Tufle and Huff

The flip side of the coin is that graphics can "lie." You can do it deliberately —which is outside the scope of this book—or you can make it happen accidentally if you are too eager to make your presentation "interesting" and "lively." The truth is usually boring, but adding decoration does not help.

One of the common mistakes is to use too many dimensions. Most of our information is one-dimensional: the number of something. Many graphs, however, present one-dimensional information in a two- or even three-dimensional way.

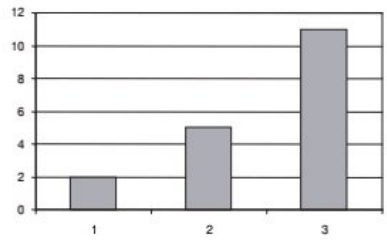
Consider the following information:

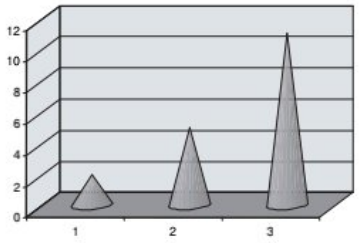
Day 1:	2 faults found
Day 2:	5 faults found
Day 3	11 faults found



The simplest way to present this is as shown to the right. See the trend? Yes, that is perfectly clear! Need anything else? Not really.

But all too often we may see exactly the same information presented like this: or, even worse, like this:





Does that add to the understanding? No.

There is a "metric" called the ink-factor. That is defined as the amount of ink used to convey the message compared to the amount of ink used in the graph. You should keep the ink-factor as low as possible.

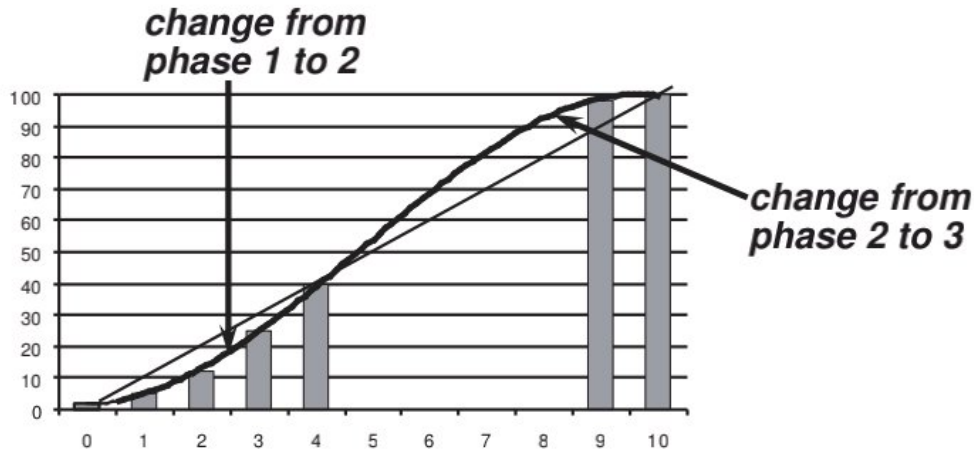
Also *avoid highlighting (read: hiding) the message in decoration, patterns, shading, or color*. A graph that presents the number of failures found each day as the size of the corollas of a line of flowers is perhaps cute, but not professional.

More obvious ways to misinform is by changing the scale across the axis, or by omitting or distorting the context of the information or the way it has been collected.

Whichever way you choose to present the information you have collected, it is your responsibility to ensure that the recipients understand and interpret the data correctly. In the following some of the most common and useful ways of presenting test progress information are described.

3.4.2.1 S-Curves

The most used, most loved, and most useful way of presenting progress information and controlling what's happening is S-curves. They are named so because of the shape of the wanted curve.



Source: Marnie Hutcheson, Unicom Seminar, Oct 95.

S-curves can be used for many different metrics. You need two that are related to each other—one is typically time. The other could, for example, be:

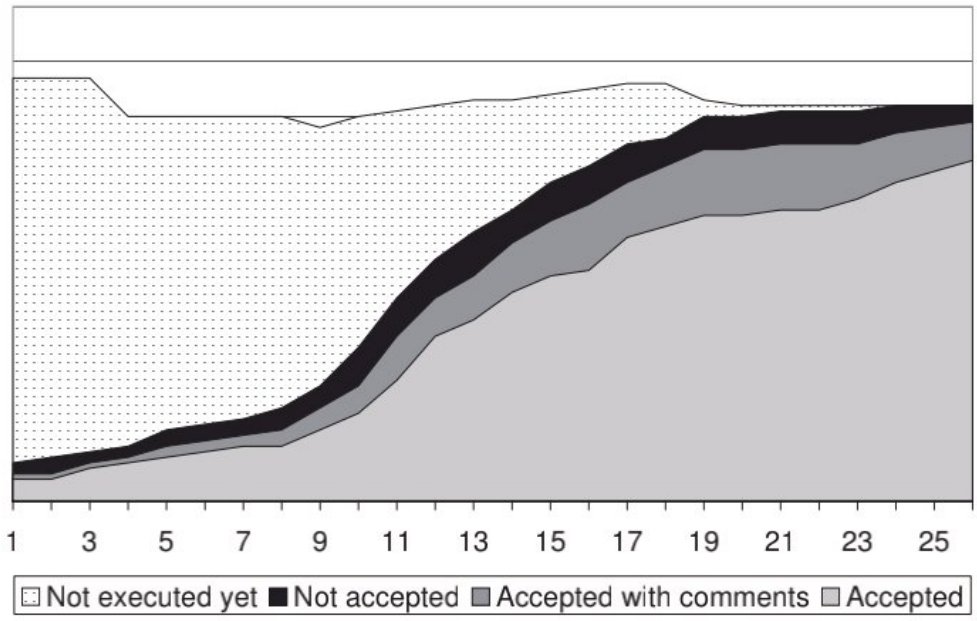
- Test cases (run, attempted, passed, completed)
- Incidents (encountered, fixed, retested)

S-curves can give us early warnings of something being wrong. It can also give us reassurance that (so far) things are progressing as planned.

The principle in S-curves is that our work falls in three phases:

- Phase 1: Slow start—not more than 15–25%
  An initial period of reduced efficiency to allow for testing teams to become familiar with the testing task, for example with the test object, the test environment, and execution and logging practices.
- Phase 2: Productive phase—55–65%
  After the initial period, the second phase is the period of maximum efficiency.
- Phase 3: The difficult part—10–25%
  The final phase reflects the need to be able to ease off the work as the testing window nears completion.

The following figure shows how real data are reported as several S-curves in the same graph.



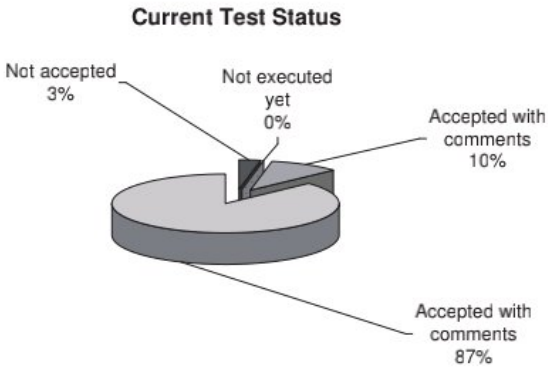
To use an S-curve you need to know what the expected start point (usually 0,0) and the expected end point are. The end point is your estimation of what needs to be achieved by the end time; for example 300 test cases passed after 21 days of testing.

You mark the start point and the end point, and you draw (or get a tool to draw) a third-order polynomial that fits. A straight-line approximation between the two points may do.

As the time goes and you do you work, you plot in your achievements—for example, the sum of test cases passed day by day. Follow the progress to see if it fits the predicted curve. If it does, we are happy!

If the upward turn, marking the start of the second phase, comes too late, we are in trouble. But the good news is that we know it and can take action well before the end date! If the curve is rising too fast, we may also be in trouble, and we must investigate what could be wrong. Maybe our test cases are not giving enough failures? Maybe we have run the smallest first and are pushing work in front of us?

3.4.2.2 Pie Chart



Pie charts are used to give an overview of the proportions of different aspects relative to each other. Pie charts are perhaps the most used graph in the world.

The graph shown here gives a nice impression of the testing going well.

But think about the inkfactor—maybe the third dimension is not needed to present the information. Maybe it is even disturbing the impression: Should the lightest gray volume be almost nine times as big as the medium gray?

3.4.2.3 Check Sheets

Check sheets are a good way to give a quick overview of status. They can be used to show progress compared to plan, for example, for planned test cases, planned test areas, or defined risks.

Check sheets can be presented as colorful graphics or expressed as lists or hierarchies. They are usually easy to produce, and easy to understand. Some organizations make wall-sized check sheets and stick them in the meeting room or the corridor. This way everyone has easy access to the information about the progress of the test.

A few examples of check sheets are shown next.

The first is an extract of the check sheet presented on Systematic's intranet. It is updated every day. Even though the text has been deliberately blurred and the extract is small, it gives an impression of things going well.—no black or light gray fields in the list!

Ex.

Status for project: ksdf				
Area	**Remain**	**% Complete**	**Status**	**Comment**
agfdg	8	56		
gstyk	0	100		
jl,flli	0	100		
dsrahjtdulk	1	80		
ths	2	56		
jdvw	0	100		
yjdtek	0	100		
	0	87		

Legend
- Completed
- In progress
- Blocked (see comment)
- Not started

Ex.

The next is a dashboard suggested by James Bach:

Area	Test effort	Coverage planned	Coverage achieved	Quality	Comments
Start up	High	>80%	27%	😐	ER 52
Discount	Low	>40% <70%	53%	☺	
Pricing	**Blocked**	>40% <70%	14%	☹	**ER 86**

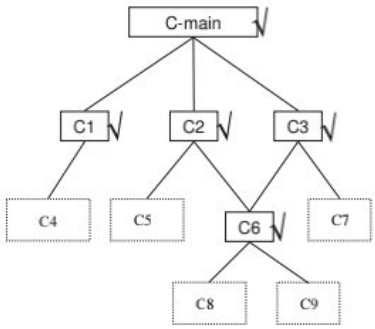
James Bach's recommendations for the presentation are: draw it on the wall, make it huge, and update it every day.

Ex.

The last example here is a tiny extract of a hierarchical check sheet showing the progress of a component test for a system.

The marked components have been successfully component-tested and are ready for integration.

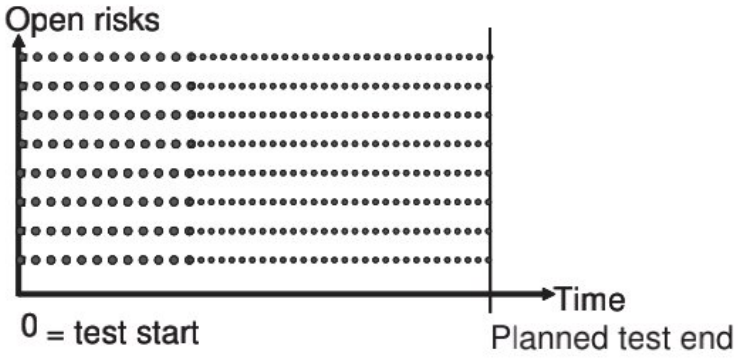
The integration testing has not yet started—no interfaces are marked as having been successfully tested. It is, however, easy to see which interfaces we have to test.



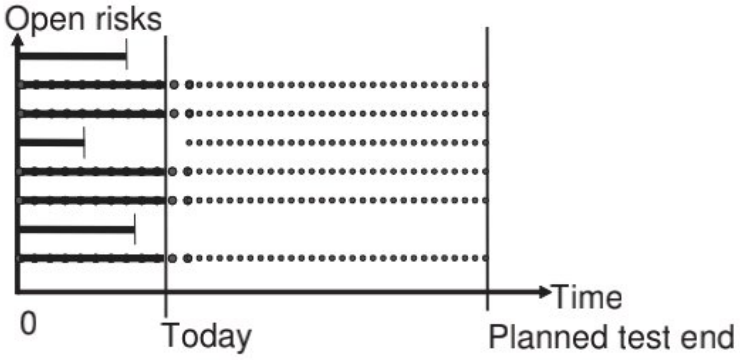
3.4.2.4 Risk-Based Reporting

If our test approach is based on identified and analyzed risks it is appropriate to report on the test progress in terms of these risks.

The purpose of this test is to eliminate the risk, so the reporting must be in terms of eliminated risks. The open risks at the test start could be illustrated like this:



At any point in time we must make it possible to see from the updated progress graph which risks are still open, if any. The risks with the small vertical line across them are eliminated and hence no longer present any threat to the system!



3.4.2.5 Statistical Reporting

The way a process is performed is different from project to project and over time, because processes are performed by people, not machines.

Statistics is the science of patterns in a variable world. We can say that statistics make the invisible visible. This means that statistical methods can be used to help us:

- Understand the past
- Control the present
- Predict the future

Statistics also include handling of "fortuitousness," that is, happenings that are out of the ordinary.

When we have to deal with many happenings assumed to be "alike," we need to find out what "alike" means. To do that we must find out what the norm is, and what variances are allowed to still call things "normal."

Ex.

Norm and variation vary. In our family the norm is that we are friendly and talk to each other in nice, calm tones of voices. I, however, have a short temper, and sometimes raise my voice without anything being really out of the normal. If, on the other hand, I keep quiet, then my mood is not within the norm. My husband is different: If he raises his voice just a little bit, he is sure to be in a very bad mood.

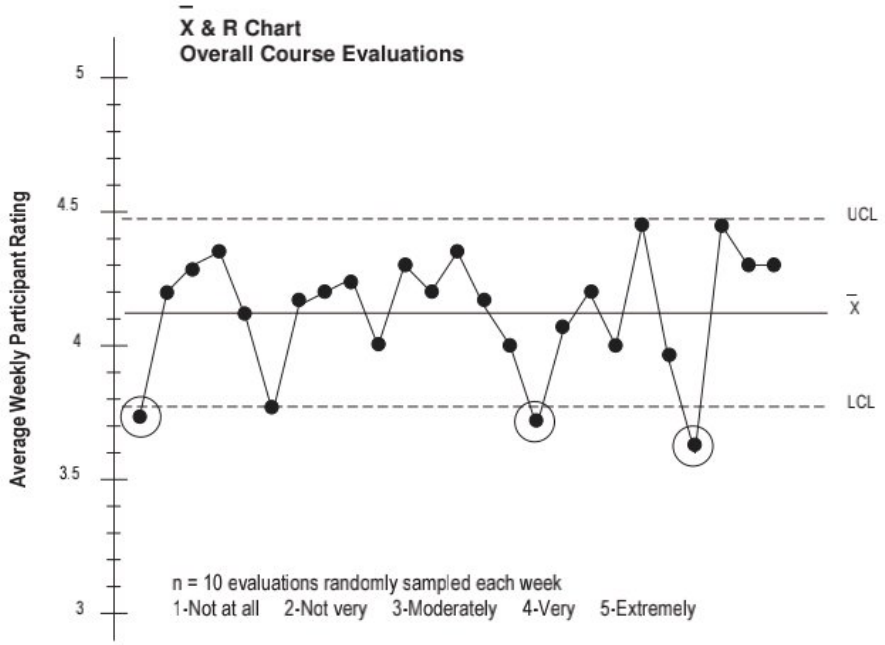
The norm in statistics can be calculated in three much used ways, namely: mean—the arithmetic average; median—the value that splits the group in the

middle; and modus—the most frequent value.

But how far from the "normal" value, can a given value be and still be considered within the norm? This can be seen in a control sheet.

Ex.

An example of a control sheet is shown here. The values are ratings for a course. Every week 10 evaluations are randomly sampled and the average is plotted in the graphs. The graph shows the upper and lower control levels (UCL and LCL) for the series of ratings.



The values here are indicators for the course performance. We can choose other values to be indicator values for our processes, if we want to control how they are performed.

Ex.

An indication value may, for example, be the average time per test case it takes to produce a test specification.

When we examine the control sheet we must be looking for warnings of something being out of the borders of the norm. Such warnings may, for example, be:

- One value outside either CL
- Two out of three values on the same side
- Six values in a row either up or down
- Fourteen values in a row alternately up and down

Many tools can assist in the necessary statistical calculations. The use of control sheets and statistics is rather advanced process control—belonging to maturity level 4—and we will not go into further depth here.

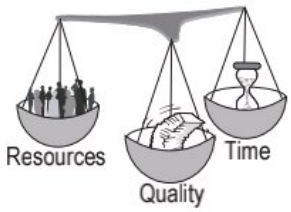
3.4.3 Stay in Control

Sometimes we need to take action to stay in control.

Keeping the triangle of test quality in mind, you have three aspects you can change—and you must change at least two at the time.

The aspects are:

- The resources for the task
- The time for the task
- The quality of the work to be performed



Usually when things are getting out of control it is because we are behind schedule or because our time frame has been squeezed. To compensate for this we must (try to) obtain additional resources and/or change the quality of the testing. The latter can be done by changing the test completion criteria and/or changing the amount or depths of the tests to be performed.

Any change you make must be reflected in the plan. The plan must be updated with the new decisions based on the new information. The new plan must be reviewed and approved, just like the first one.

Remember that *it is not a virtue to comply with the plan at any cost—the virtue lies in the plan complying with reality.*

No matter which way the progress is measured and presented, the test manager must analyze the measurements. If something seems to be going the adverse way, further analysis must be made to determine what may be wrong.

Examples of things not being as they should be are:

- The delivered software is not ready for test
- The easy test cases have been run first
- The test cases are not sufficiently specified
- The test case does not give the right coverage
- General faults have wide effects
- Fault correction is too slow
- Fault correction is not sufficiently effective

Based on this analysis the test manager must identify what can be done to remedy or mitigate the problems.

Possible actions may, for example, be:

- Tighten entry criteria
- Cancel the project
- Improve the test specifications
- Improve the incident reporting
- Perform a new risk analysis and replan
- Do more regression testing

The important message to the test manager is that he or she must intervene as soon as deviations appear! Or in other words:

If you do not control the test, it will control you!

To sum up we can say that as test managers we must set out the destination and plan how to get there; collect data as we go along; analyze data to obtain information; and act on the information and change destination and plan as appropriate.

3.5 Testing and Risk

The golden rule of testing is:

Always test so that whenever you have to stop you have done the best possible test.

Everybody with some understanding of requirements and tests will know that a requirement like this cannot be verified. What does it mean: the best possible test? It is not immediately measurable.

What is the best possible test then? The answer to that is: It depends!

The best possible test depends on the risk associated with having defects left in the product when it is released to the customer!

The best possible test is determined by the risks we are facing and the risks we are willing to run. Obtaining consensus from stakeholders on the most important risks to cover is essential.

3.5.1 Introduction to Risk-Based Testing

We have to live with the fact that it is impossible to test everything. Testing is sample control. There is a risk involved in all sample control: the risk of overlooking defects in the areas we are not testing.

3.5.1.1 Risk Definition

A risk is defined as: "The possibility of realizing an unwanted negative consequence of an incident."

Alternatively, a risk may be defined as: *"A problem that has not materialized yet and possibly never will."*

There are two important points in these definitions:

- A risk entails something negative
- A risk may or may not happen—we don't know

A risk therefore has two aspects:

- Effect (impact—consequence)
- Probability (likelihood—frequency)

It is not a risk (to us), if there is no effect of an event that might happen. We can therefore ignore it even if the probability is high.

There is a probability that there are defects in the new version of our database management system, but that will have no effect on the quality of our product if it happens, because it is not used in our system.

It is not a risk if there is no (or an extremely small) probability that an event will happen, even if the effect would be extremely big, if it did. We can therefore ignore that as well.

There is no (detectable) risk of our department closing down, because we have lots of orders, and are making good money, and both management and employees like their jobs. If we did close down the effect on the project would be pretty bad, if not disastrous.

It is not a risk either if the probability of an event with a negative effect is 100%. In this case we have a real problem on our hands, and we will have to deal with that in our planning.

It is a problem—not a risk—that we will have to do without one of our test experts because she has found another job and is leaving in three months.

The two aspects of risk can be combined in

Risk level = probability x effect

From this it is quite clear that if we have no probability or no effect we have no risk.

The risks that do have a risk exposure greater than zero are the risks we have to deal with.

3.5.1.2 Risk Types

It is quite common to treat all the risks we can think of in connection with a development project in one big bundle. This can be quite overwhelming.

It is therefore a very good idea to take a closer look at the risks and divide them into classes, corresponding to where they may hit, or what they are threatening.

Risks hit in different places, namely;

- The business
- The processes
- The project
- The product

The risks threatening the product are the testers' main concern. This is where we can make a difference.

The business risks are things threatening the entire company or organization from a "staying-in-business" point of view. This is out of the scope of this book, and will not be discussed further.

Process risks are related to the processes and/or the way work is performed. It is also out of the scope of this book, but will be briefly discussed because knowledge about processes is indispensable in a modern development organization.

Process risk threatens the effectiveness and efficiency with which we work on an assignment. Process risks may be originated in:

- Missing process(es)
- The organization's lack of knowledge about the processes
- Inadequate processes
- Inconsistencies between processes
- Unsuitable processes
- Lack of support in the form of templates and techniques

Process risks jeopardize the way the work in the project is being performed. These risks should be the concern of the project manager and those responsible for the processes in the organization.

Process risks may influence business, as well as project and product risks.

Project Risks

Project risks are related to the project and the successful completion of the project.

A project consists of a number of activities and phases from requirements development to the final acceptance test. These activities are supported by activities like quality assurance, configuration management, and project management.

All the activities in a project are estimated, get allocated resources, and are scheduled. As the project progresses the activities are monitored according to the plan.

Risks concerning the project may be originated in:

- People assigned to the project (e.g., their availability, adequate skills and knowledge, and personalities)
- Time
- Money
- Development and test environment, including tools
- External interfaces
- Customer/supplier relationships

Project risks jeopardize the project's progress and successful completion according to the plan.

Examples of project risks are:

- The necessary analysts are not available when the requirements development is expected to start
- Two of the senior designers are not on speaking terms and useful information exchange between them is not done—this causes the design phase to take longer than expected
- The complexity of the user interface has been underestimated
- The testers are not adequately trained in testing techniques, so testing requires more resources than expected
- The integration is more time-consuming than expected
- The access to external data is not possible with the technique chosen in the design

The project risks are the main concern of the project manager and higher management.

The test manager is concerned with the project risks related to the test project as it is specified in the test plan documents.

Product Risks

Product risks are related to the final product. They are the risks of defects remaining in the product when it is delivered.

We want to deliver the required quality and reliability. This cannot be tested into the product at the end of the development, but must be worked into the product through the work products produced during development and in the implementation of the components.

Product risks may be originated in:

- Functional and nonfunctional requirements
- Missing requirements
- Ambiguous requirements
- Misunderstood requirements
- Requirements on which stakeholders do not agree

More than 50% of all defects in products can be traced back to defects in the requirements.

- Design
- Not traceable to requirement
- Incorrect
- Incomplete (too superficial)
- Coding
- Testing
- Not traceable
- Inadequate

Product risks jeopardize customer satisfaction and maybe even the customer's life and livelihood.

Product risks may be related to different requirement types, such as functionality, safety, and security and political and technical factors.

Examples of product risks are:

- A small, but important functionality has been overlooked in the requirements and is therefore not implemented
- A calculation of discounts is wrongly implemented, and the customer may lose a lot of money
- The instrument may reset to default values if it is dropped on the floor
- It is possible to print a report of confidential customer information through a loophole in the reporting facility
- The installation procedure is difficult to follow

These risks are the main concern of the testers, since testing may mitigate the risks. The test strategy for a product should be based on the product risks that have been identified and analyzed.

Project risks and product risks can influence and be the cause of each other. A project risk may cause a product risk, and a product risk may cause a project risk.

If a project risk results in time being cut from component testing, this may cause the product risk of defects remaining in the components that are not tested or not tested sufficiently. This may further cause the project risk that there is not sufficient time to perform a proper system test because too many trivial failures are encountered in the system test.

3.5.1.3 Testing and Risk Management

Testing and management of risks should be tightly interwoven as they support each other. Testing can be based on the results of risk analysis, and test results can give valuable feedback to support continuous risk analysis.

The result of a product risk analysis can be used in test planning to make the test as effective as possible. It can be used to target the testing effort, since the different types of testing are most effective for different risks.

Component testing is most effective for testing where the product risk exposure related to complex calculations is highest.

The risk analysis results can also be used to prioritize and distribute the test effort. The areas with the risk exposure should be planned to be tested first and given the most time and other resources.

Finally the product risk analysis can be used to qualify the testing already done. If test effort is related to risks it should be possible to report on dissolved and remaining risks at any time.

Testing can, as mentioned earlier, dissolve or mitigate the product risks. The probability of sending a product with defects out to the customers is reduced by the testing finding failures and the subsequent correction of the defects.

Testing can also mitigate project risks if an appropriate test strategy is applied, especially if testing is started early.

Even process risks may be reduced by analyzing failure reports and taking appropriate process improvement initiatives.

3.5.2 Risk Management

Risk management consists of the following activities:

- Risk identification
- Risk analysis
- Risk mitigation
- Risk follow-up

In risk identification we are finding out what may happen to threaten the process, the project, or, in this particular context, the customer satisfaction for the product.

The identified risks are evaluated in the risk analysis and ordered relatively to each other. The analysis means that we assign probability and effect to the risks. Based on this we can determine the risk exposure and hence which risk is the worst and how the others relate to that.

One of the points in risk management is to use the results of the analysis to mitigate the risks. Actions can be planned to lower the probability and/or the effect of the risks. In this context of product risks and testing, test activities can be planned to mitigate the risks by lowering the probability of having remaining defects in the product when it is released. The more defects we can remove from the product as a result of the testing, the more the probability falls.

Contingency planning is a part of classic risk management, but this is not relevant for product risks in relation to testing. Testing is concerned with lowering the probability of remaining defects for the defects that remain; support and maintenance must be prepared to provide work-arounds and/or corrections and updates.

Risk identification, analysis, and mitigation must not be a one-time activity. It is necessary to follow-up on the risks as testing progresses. The results of the testing activities provide input to continuous risk management.

Information about the failure frequency over time can be used to assess if the probability of a risk is falling or rising.

There are many stakeholders in a development project, and they come from different places and have different view points, also on risks.

The following table shows examples of stakeholders or stakeholder representatives for a number of aspects of a development project.

Aspect	Stakeholder representatives
The business	Product line manager Business analysts Marketing personnel Finance executives
Future users	Future users Users of existing system Users representing different types of user groups
User representations	Marketing personnel Salespeople Employees from relevant support function
Technical, development	Analysts, designers, programmers, testers People responsible for manufacturing People responsible for operation People responsible for configuration management People responsible for quality assurance
Technical, product	Experts in and people responsible for usability, security reliability, performance, portability

All stakeholders identified for a project should be involved in the risk management activities.

3.5.2.1 Risk Identification

Risk identification is finding out where things can go wrong and what can go wrong, and writing it down to form the basis for risk analysis.

Risks are found in areas where the fulfillment of expectations may be threatened (i.e., where customer satisfaction is jeopardized). *Satisfying expectations is the way to success!*

All stakeholders have expectations towards the product, but we are most concerned with the expectations of the customer. The customer is the one to order the product, to pay for it, and to take advantage of it, the latter possibly through end users.

In the ideal world the customer's expectations are expressed in user or product requirements. These requirements are transformed into design, and the requirements are fulfilled in the code and maybe also in other subsystems, being integrated into the final products.

In less ideal worlds expectations may be derived from other sources.

Risks are not always evident. Even when we work with experienced and knowledgeable stakeholders it can be efficient to use a risk identification technique.

Useful techniques are:

- Lessons learned
- Checklists
- Risk workshops
- Brainstorms
- Expert interviews
- Independent risk assessments

Techniques may be mixed to be even more efficient.

3.5.2.2 Lessons Learned and Checklists

Lessons learned and checklists are closely related. A checklist is a list of generic risks formed and maintained by experience (i.e., lessons learned from previous projects).

Risk checklists are valuable assets in an organization and should always be treated as such.

One or more product risk checklists should be kept in the organization, depending on the diversity of the nature of the projects performed in the organization.

A product risk checklist could be structured as the requirements specification or the design specification is structured, or both.

The checklists used by the pilots before takeoff are long and must be run through very carefully before every takeoff. A pilot was once hurried on by a busy business man and asked to drop the checklists and get going. He carried on with his work as he answered: "These checklists are written in blood!"

3.5.2.3 Risk Workshops

Workshops are an effective way to identify risks. There are no strict rules as to how a workshop should be conducted, but a few guidelines can be given.

As many stakeholders as possible should be involved, though the number of participants should not exceed 10–12 in order to give everybody a chance to talk within reasonable intervals.

Risk workshops can get emotional, and a neutral facilitator—somebody who is not by any account a stakeholder—should be present to guide the discussions. *Encourage discussions and new ideas, but avoid conflicts.*

Make sure that all participants agree that the objective has been reached and that it is clear how work can proceed after the workshop.

3.5.2.4 Brainstorming

A brainstorm (in this context of risk identification) is an informal session with the purpose of identifying possible risks connected with the product when it is released.

The only rule that should apply during a brainstorm *is that no possible risk must be commented on* in any way by the participants. Ideas should be allowed to flow freely, the rationale being that even the most seemingly stupid, silly, or strange thought may be the inspiration for valuable potential risks.

A brainstorm must have a facilitator who may act as a catalyst if ideas do not flow freely. At the end of the session the facilitator must make sure that whatever is being brought forward as possible risks is expressed as risk and documented.

The stakeholders to be involved in this technique may be any type of stakeholders.

3.5.2.5 Expert Interviews

Interviews may be conducted as individual interviews or as group interviews. *An interview is not as easy to conduct* as many people think. It requires specific skills and thorough preparation to get as much information as possible from an interview.

First of all, an interview is not like an ordinary conversation. People in an interview have different roles (i.e., the interviewer and the interviewee(s)), and they may have a number of expectations and prejudices related to these roles. Interviews must be prepared. The interviewer must, for example, make sure that the right people are being interviewed and the right information is gathered. A list of questions or a framework for the course of the interview must be prepared.

Ample notes must be taken and/or the interview can be recorded with the permission of the interviewee(s). *The interviewer must extract a list of possible risks from the interview* and get agreement from all the participants.

3.5.2.6 Independent Risk Assessments

In cases where conflicts are threatening external consultants may be called in to identify risks. External consultants could also be used if time is short or if specific expertise not present with the immediate stakeholders is required.

The external consultants identify risks and usually perform or facilitate the risk analysis.

The consultants may be external to the project organization or third-party consultants entirely external to the developing organization.

3.5.3 Risk Analysis

Risk analysis is the study of the identified risks. One thing is to identify and list the risks; another is to put them into perspective relative to each other. This is what the analysis of the risks helps us do.

The analysis must be performed by all the appropriate stakeholders, because they all have different perspectives and the risk analysis aims at providing a common and agreed perspective. Experts may be called in to contribute if adequate expertise cannot be found among the immediate stakeholders.

Risk analysis can be performed more or less rigorously, but it should always be taken seriously.

3.5.3.1 Risk Template

A risk template or a risk register is a very useful tool in risk management. It can be used to support risk analysis and risk mitigation.

Risk templates can be held in office tools, for example, spreadsheets that support calculations.

A risk template should include:

- Risk identification (e.g., number or title)
- Risk description
- Probability
- Effect
- Exposure
- Test priority
- Mitigation action = test type
- Dependencies and assumptions

3.5.3.2 Perception of Risks

The performance of risk analysis can be more or less objective. In fact *most risk analysis is based on perceptions*; it is usually not possible to determine risk probability and effect totally objectively. There is an element of prediction in risk analysis since we have to do with something that has not happened and maybe never will. There are usually very few trusted measurements applicable to identified risks.

Perceptions are personal and different people have different "pain thresholds." Just look around you: Some people use their holidays to explore new places; others always spend their holidays at the same place. In connection with process improvements we sometimes say that if it blows hard some people build shelters; others build windmills.

"If you do that, I'll never see you again!!!"
"Is that a promise or a threat?"

People in different professions may also have different view points on risks, partly because people choose jobs according to their personalities, partly because job-related experiences influence their perception of different risks.

The following descriptions of job-related risk perceptions are of course gross generalizations, but they can be used as guidelines in understanding different viewpoints on "the same risks." The descriptions encompass:

- Project managers
- Developers
- Testers
- End users

Project managers are often under time pressure; they are used to compromises. They know that even though things may look dark, the world usually keeps standing.

Developers, that is analysts, designers, and programmers, are proud of their work, and they know how it was done. They have really done their best, and they are usually reluctant to accept that there may still be defects left in there.

Testers often have a pessimistic view on work products, product components, and products. We remember previous experiences where we received objects for testing and got far more failures than we expected.

The *end users* are, despite what we might think, usually highly failure-tolerant. They also tend to remember previous experiences, but what they remember is that even though the system failed, they found a way around it or another way of doing their work. End users use our product as a tool in their job, and nothing more. If it does not help them, they'll find another way of using the tool, find another tool, or just live with it.

In risk analysis we must encourage communication and understanding between stakeholders. *Stakeholders need to be able to, if not agree with then at least be aware of and accept others' points of view.* If need be, stakeholders will have to compromise or use composite analysis. This is explained next.

3.5.3.3 Scales for Risk Analysis

The analysis of risks uses metrics for probability and effect. For all work with metrics it is mandatory to use agreed and understood scales. This is therefore also the case in risk analysis.

We can work with two different kinds of scale, namely:

- Qualitative
- Quantitative

In a qualitative scale we work with feelings or assessments.

For effect we could use
bad—worse—worst
For probability this could be expressed as
not likely—likely—very likely

In a quantitative scale, on the other hand, we work with exact measures or numbers.

For effect we could use actual cost in $ or Kr. or €.
For probability this could be expressed as
<= 10%, >10% & <= 50%, >50% & <=80%, > 80%

Whichever way to do it, we must define and agree on scales for both probability and effect before we start the risk analysis, that is before we assign metrics to the probability of the identified risks actually materializing, and metrics for the effects if they do.

3.5.3.4 Effect

The effect is the impact or consequences of a risk if (when?) it occurs. The first thing we have to do is agree on a scale for the effect.

The obvious *quantitative scale* for the effect is the actual cost imparted by a failure occurring out in the field. The actual cost can be measured in any agreed currency ($, €, Kr.). This is an open scale; in theory there is no limit to actual cost.

It can be very difficult to assess what the actual cost in real money might be. On the other hand it can be quite an eye-opener to sit down and consider all the sources of extra cost associated with a failure.

Expenses may for example be considered for:

- Time for the end user to realize that something is wrong
- Time to report the incident to first-line support
- Time for first-line support to understand the report and try to help
- Time for any double or extra work to be performed by the end users
- Loss of production because the system is down or malfunctioning
- Time for escalation to secondhand support
- Time for secondhand support to try to help
- Time to investigate the failure and decide what to do about it
- Time for finding the defect(s)
- Time for corrections to be implemented and tested in all affected objects

- Time for retesting and regression testing
- Time to reinstall the new version
- Time to update what has been done by other means while the system was unavailable or malfunctioning

These are all examples of time spent in connection with a failure. There may also be costs associated with, for example, renting or replacing parts of the system or the entire system.

Furthermore there may be an effect in the form of indirect losses from, for example, people getting hurt, the environment being destroyed, or the company getting an adverse reputation or losing trustworthiness.

Failures have been known to cost lives or to put companies out of the market completely. Fortunately, it is usually not that bad, but still the effects of failures can be significant.

Another way to measure effect is by using a qualitative scale. Such a scale could, for example, be expressed as shown in the following table.

Inspired by Paul Gerrard

Effect	Description	Score
Critical	Goals cannot be achieved	6
High	Goals will be jeopardized	5
Above middle	Goals will be significantly affected	4
Below middle	Goals will be affected	3
Low	Goals will be slightly affected	2
Negligible	Goals will be barely noticeably affected	1

In the table there is a column for a mnemonic for the effect, a column describing the effect more precisely, and a column for the actual score.

Using a numeric score makes it possible to calculate the risk exposure even when a qualitative scale is used for the effect.

Despite the above example it can be useful to define a scale with an even number of scores. This can mitigate the effect of some people having a tendency for choosing the middle value if they are not sure what to score or can't be bothered to think deeper about their opinion. A scale with an even number of scores does not have a middle value and the stakeholders will have to decide if they want to score over the middle or under.

The important point before the analysis of the effects can start is that *the stakeholders agree to and understand the scale.*

When you perform the analysis of the effect of the risks you have identified, you must keep your focus on the effect. You must *NEVER let the probability influence the effect!* It can sometimes be tempting to give the effect an extra little turn upwards if we know (or think) that the probability of the risk materializing is high. This will give a twisted picture of the risk exposure and should be avoided.

A simple effect analysis for the risks pertaining to the four top-level architectural areas defined for a product may look like this, using a scale from 1–6 where 6 is worst.

Risk area	Effect
Setup	2
Conveyor	2
Concentration calculation	6
Compound determination	5

Often it is not enough to have one single score for the effect. Stakeholders see the effect from different perspectives. An end user sees the effect in the light of how a failure will influence his or her daily work. A customer may look at the effect of failures on the overall business goals. A supplier organization may assess the effect in terms of correct efforts for failures or loss of credibility in the market.

These different perspectives can be honored if we use a more complex or composite effect analysis. The score should be the same for all the perspectives, but the descriptions should be tailored to make sense for each of the viewpoints.

A composite effect analysis taking more perspectives into account may look like this:

Risk area	Effect for perspective			Final effect
	User	Customer	Supplier	
Setup	5	3	2	3.3
Conveyor	3	3	5	3.7
Concentration calculation	2	5	2	3
Compound determination	1	5	3	3

Here all perspectives have the same weight, and the final effect is a simple average of the effect contributions.

If the scale is not sufficiently differentiated the individual perspectives may be assigned independent weights, and the final effect can then be calculated as the weighted average:

$$\text{Final effect} = \sum(\text{effect}*\text{weight}) / \sum(\text{weight})$$

The effect analysis taking more perspectives into account and assigning different weights to the perspectives may look as shown next.

Ex.

Risk area	Effect for perspective			Final effect
	User W=2	Customer W=7	Supplier W=1	
Setup	5	3	2	3.3
Conveyor	3	3	5	3.2
Concentration calculation	2	5	2	4.1
Compound determination	1	5	3	3.9

3.5.3.5 Probability

The probability is the likelihood of the materialization of a risk.

Also here we first of all need to agree on a scale. On a *quantitative scale* probability can be measured on a scale from 0 to 1 or a scale from 0% to 100%. For most risks it is, however, almost impossible to determine the probability with such a precision.

A *qualitative scale* for probability is usually much more useful, as long as it doesn't get too loose, like "likely," "very likely," "extremely likely."

A qualitative scale could be expressed as in the following table where there is a column for probability intervals, a column describing the probability, and a column for the actual score. Again using a numeric score makes it possible to calculate the risk exposure even when a qualitative scale is used for the probability.

Ex.

Probability	Description	Score
99–82	Highly likely	6
81–	Likely	5
–50	Above 50–50	4
49–	Below 50–50	3
Low	Unlikely	2
–1	Highly unlikely	1

Inspired by Paul Gerrard

For effect, you must keep your focus on the probability when you perform the analysis of the probability of the risks you have identified. You must *NEVER let the effect influence the probability!* It can sometimes be tempting to give the probability an extra little turn upwards if we know (or think) that the effect of the risk if it materializes is high. This will give an untrue picture of the risk exposure and should be avoided.

The probability of a risk materializing may be a function of many factors, for example:

- Complexity of the product or the code
- Size of the product or the code
- The producer of the work product(s) or component(s)
- Whether it is a new product or code or maintenance
- The previous defect record for the product or area
- The developers' familiarity with tools and processes

Just like it is explained for the effect above the final probability can be calculated as the weighted average of the probabilities pertaining to the different factors.

Final probability = ∑(probability*weight) / ∑(weight)

A composite probability analysis may look like this:

Risk area	Probability for factor			Final prob.
	Size W=1	History W=5	Complexity W=2	
Setup	4	2	1	2
Conveyor	5	3	5	3.8
Concentration calculation	3	1	2	1.5
Compound determination	3	1	5	2.2

The same quantitative scale must be used for all the factors.

3.5.3.6 Risk Level

The risk level is calculated for each of the identified risks as

Risk level = final effect x final probability

It is not a difficult task to perform a risk analysis as explained above. A full analysis including identifying about 30 risks and assessing and calculating the effect, probability, and final level can be done in a couple of hours. It is well worth the effort because it gives everybody involved a much clearer picture of why test is necessary and how the test should be planned.

Spreadsheets can be used for easy calculation of the levels and for maintenance of the risk analysis.

Using the above examples for final effect and final probability, the final risk level may look as shown in the following table.

Risk area	Final effect	Final probability	Final risk level
Setup	3.3	2	**6.6**
Conveyor	3.3	3.8	**12.2**
Concentration calculation	4.1	1.5	**6.2**
Compound determination	3.9	2.2	**8.6**

It sometimes happens that some *stakeholders are unhappy with the final level*. If a stakeholder has high rates for a particular risk and the risk comes out with a relatively low final risk level, this can "seem unfair." In such cases the perspectives and the scales will have to be discussed once more.

The point of the perspectives and the scales is that they should satisfy every stakeholder's viewpoint. If that is not the case they must be adjusted. Most of the time, however, stakeholders recognize that the perspectives and scales are OK and that their viewpoint is fairly overruled by others, different viewpoints.

The distribution of the final risk level over individual risks is used to plan the test activities. It can be used to prioritize the test activities and to distribute the available time and other resources according to the relative risk level. A test plan based on a risk analysis is more trusted than a plan based on "gut feeling."

It is difficult to predict events, and therefore all risk analysis has some built-in uncertainty. *A risk analysis must be repeated at regular intervals as the testing progresses*.

The testing results can be used as input to the continuous risk analysis. If we get more defects than expected in a particular area it means that the probability is higher than we expected, and the area hence has a higher risk level. On the other hand if we get fewer defects than expected the risk level is lower.

3.5.4 Risk Mitigation

We use the results of the risk analysis as the basis for the risk mitigation, the last activity of the sequential risk management activities. "To mitigate" means "to make or become milder, less severe or less painful." That is what we'll try to do.

Faced with the list of risks and their individual risk level we have to go through each of the risks and decide:

- What we are going to do
- How we are going to do it
- When we are going to do it

3.5.4.1 What to Do to Mitigate Risks

In terms of what to do we have the following choices:

- Do nothing
- Share the pain
- Plan contingency action(s)
- Take preventive action

We can choose to do nothing if the benefit of waiting to see how things develop is greater than the cost of doing something.

You would not buy a safe for € 1,000 to protect your jewels if they were only worth € 500 (including the sentimental value). If the jewels were stolen you could buy new ones and still have money left.

Sharing the pain is outside the scope of testing, but it is a possibility for the project management or higher management to negotiate sharing the pain of the effect of a materializing risk with other parties. This other party could be an insurance company or it could be a supplier or even the customer.

Planning contingency action is a natural part of most risk mitigation. The contingency action is what we are going to do to mitigate the effect of a risk once it actually has materialized. For other risk types than product risks and other processes than testing the response to the risk analysis may be production of contingency plans. But it is not something that is applicable in the test planning for mitigating product risks.

Extra courses are planned if it turns out that the system is more difficult to learn than expected.

Testing is one of a number of possible preventive actions. The aim is to mitigate the risks. Testing can be used to mitigate the risk exposure by lowering or eliminating the probability of the risk.

Product risks are associated with presence of defects. The effect is associated with the effect if a defect causes a failure of the product in use. The probability is associated with the probability of undetected defects still being present in the product when it is released.

Testing aims at identifying defects by making the product fail—before it reaches the customer. Defects found in testing can be corrected—before the system reaches the customer. Hence testing and defect correction reduces the risk level by reducing the probability.

3.5.4.2 How to Mitigate Risks by Testing

When we have decided to do something to mitigate a risk, we must find out what to do. The nature of the risk can be used to determine the phases and types of testing to perform to mitigate the risk and the level of formality applied. The decisions must be documented in the applicable test strategy or test plan.

Certain test phases are especially applicable for certain types of risks. We need to look at the risk source and determine the phases and activities that are most likely to unveil defects.

Some examples are given next.

Risk source	Recommended test phases
High risk of defects in algorithms	◗ Review of detailed design ◗ Review of code ◗ Component testing ◗ Functional system test
Risk of problems in the user interface	◗ Usability evaluation of prototype (requirements review) ◗ Usability test (nonfunctional system test)
Risk of performance problems	◗ Performance test (nonfunctional system test)
Risk concerning external interface	◗ Review of design ◗ Review of code ◗ System integration test

The formality of the test can also be determined from the risk exposure. The rule is simple:

The higher the risk level => The higher the formality

The formality can change from level to level and it can change over the product. Some areas can have more formal testing than others, even within the same test level.

At any level, for example, system testing, we can have the different levels for formality as shown in the following table.

System test

Risk level	Recommended test phases
High	◗ Specific test case design techniques to be used ◗ Strict test completion criteria ◗ Strict regression test procedures
Low	◗ Free choice of test case design techniques ◗ Less strict test completion criteria ◗ Less strict regression test procedures

Ex.

3.5.4.3 When to Mitigate Risks by Testing

We can use the results of the risk analysis to prioritize the test activities that we have identified for the risks and to distribute the test time (and possibly other resources).

In the prioritization we are going to determine the order in which to attack the risks. Even if we are not going to perform all the testing activities identified for the risks in strictly sequential order, it is a help in the planning to have them prioritized.

The priority can follow the final risk exposure. This means that the final exposure can be as the sorting criteria. This takes every perspective of the risks into consideration in one attempt.

With the example from above the priority of the risk areas can then be as shown in the following, where 1 is the highest.

Risk area	Final risk level	Priority
Setup	**6.6**	3
Conveyor	**12.2**	1
Concentration calculation	**6.2**	4
Compound determination	**8.6**	2

Ex.

The stakeholders could also choose to let the prioritization be guided by either the final effect or the final probability, or they may even agree to use one particular perspective, for example, the probability related to the complexity, to prioritize from.

In order to calculate the distribution of the time to spend on the testing we need to calculate the sum of the final exposure.

Ex.

Risk area	Final level
Setup	6.6
Conveyor	12.2
Concentration calculation	6.2
Compound determination	8.6
Total	33.6

The next step is to calculate the distribution of the final levels over the risk areas. This could look as shown here, where the percentages have been rounded to the nearest whole number.

Ex.

Risk area	Final level	% distribution
Setup	6.6	20 %
Conveyor	12.2	36 %
Concentration calculation	6.2	10 %
Compound determination	8.6	26 %
Total	33.6	100 %

With a table like this we have a strong planning tool. No matter which resource we have at our disposal we can distribute it on the risk areas and hence ensure that each area is indeed tested, but neither more nor less than it deserves.

If the project manager for example allocates 400 hours for the complete test of our example system, we can distribute this time over the areas as shown here:

Ex.

Risk area	% distribution	Hours for testing
Setup	20 %	80
Conveyor	36 %	144
Concentration calculation	18 %	72
Compound determination	26 %	104
Total	**100**	**400**

The list of prioritized risks with their allocated resources and identified testing phases and activities allows us to produce a substantiated plan and schedule for the test.

The list also allows us to immediately assess the results of a proposed change in resource allocation. If the resource we have distributed is cut, we will have to find out how to make do with what is left.

Usually we are operating with time; having a number of hours allocated for the testing and consequently a number of hours may be cut. If time is cut we must ask management what to do with our distribution of time on the risks. We can't leave our plan and schedule untouched; the cut must have an effect. What we can do is, in principle:

- Reduce testing time proportionally to the cut
- Take risk areas out of the testing completely

The best thing to do is to reduce the time proportionally. This ensures that all areas are still tested, that is, we will get some information relating to all the risks. We can combine the two approaches but we should be very careful if we take areas out completely.

If some testing has already been performed, a renewed risk analysis is necessary before we can act on any cuts. In this case we must distribute the remaining resources over the remaining risks according the new final exposure and prioritize as we did before or by a new, more relevant criterion.

Questions

1. How does testing provide business value?
2. What is the purpose of testing?
3. What is product reliability?
4. On what does the cost of defect correction depend?
5. Where do most of the defects originate from?
6. What does good testing provide?
7. What four types of test management documents are defined?
8. What is the purpose of the test policy?
9. What should the test policy include?
10. What could a quality target for example be?
11. What is the purpose of the test strategy?
12. What are the approaches to the testing?
13. What are the strategy topics to be considered?
14. Why may standards be useful?
15. What should a strategy include in relation to risks?
16. Why are completion criteria important?
17. What is the idea in degree of independence?
18. What may be reused in testing?
19. Why does the strategy need to be specific about tools?
20. Why should we measure during testing?
21. To which support process does the incident management belong?
22. What is a master plan for?