

# ICT DOCUMENT

Information and Communication Technologies

## ICT Test Strategy V 1.0a.

Approval(s)	08/02/2018, Stanislav Danchev, Acting Head of ICT Unit. Approved for using as a reference when planning testing activities.
Document owner	ICT Quality Management officer
Consulted	ICT Security Officer, Enterprise Architect, Heads of ICT sections and group leads, Back Office Tester
Reference to IT Test policy	<a href="#">IT Test policy</a>

## Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>4</b>
1.1	Purpose .....	4
1.2	Scope .....	4
1.3	How to use the document.....	4
<b>2</b>	<b>Testing Principles .....</b>	<b>6</b>
<b>3</b>	<b>Test Levels and Types.....</b>	<b>8</b>
3.1	Test Levels .....	8
3.2	Test Types .....	12
<b>4</b>	<b>Test Approaches .....</b>	<b>13</b>
4.1	Risk-based Testing .....	13
4.2	Requirements-based Testing .....	14
4.3	Business Process-based Testing.....	14
4.4	Session-based Testing .....	14
4.5	Methodical Testing .....	15
<b>5</b>	<b>Test Processes .....</b>	<b>16</b>
<b>6</b>	<b>Test roles and key stakeholders .....</b>	<b>17</b>
6.1	Test roles .....	17
6.2	Testing stakeholders .....	17
<b>7</b>	<b>Test Strategies per IT service activities .....</b>	<b>19</b>
7.1	Introduction .....	19
7.2	Test Strategy per IT service activity .....	20
7.3	Selection Criteria for light or full Test Strategy .....	21
7.4	Test approaches per Test Level .....	21
<b>8</b>	<b>Test Metrics .....</b>	<b>22</b>
<b>9</b>	<b>Test Tools .....</b>	<b>28</b>
<b>10</b>	<b>Documentation .....</b>	<b>29</b>
10.1	Documentation Structure .....	29
10.2	Documents .....	29
	<b>Annex A: Quality Attributes .....</b>	<b>31</b>
	<b>Annex B: Definition of testing terms.....</b>	<b>34</b>

## List of Figures

Figure 1: Product risk resulting in test case priority .....	14
Figure 2: Test Processes.....	16
Figure 3: Quality of the testing done by the Software Development Service Provider .....	23
Figure 4: Average duration of defect solving.....	24
Figure 5: Number of IT and AT defects solved outside the agreed time .....	24
Figure 6: Defects in production in relation to major releases .....	25
Figure 7: Testing as a percentage of the total project .....	27
Figure 8: Effort per test level in percentages .....	27
Figure 9: Documentation structure .....	29

# 1 Introduction

## 1.1 Purpose

According to the International Software Testing Qualifications Board (ISTQB<sup>1</sup>) a Test Strategy provides a high-level description of Test Levels to be performed and testing activities within those levels for an Organization or Programme.

The purpose of this document is to provide the general test strategy for projects and IT Products under responsibility of ICT Unit. The strategy describes how the testing should be applied to manage product maintenance and project risks, the division of testing into levels, and the high-level activities associated with the testing. Why testing is needed is already defined in the IT Test Policy<sup>2</sup>.

The intention, over time, is for this strategy to cover all software-testing activities in ECDC to have harmonized approach, even if the initial scope is limited to the ICT Unit.

## 1.2 Scope

The description of the software testing managed by ICT Unit for new software development and IT Product maintenance activities is in the scope of the Test Strategy. The Test Strategy concerns the internal development activities in ICT Unit, as well as development by an external service provider. In the document Infrastructure or Software Development Service Providers refers to both whether the service is provided by ECDC internal resources or external services.

Out of scope is:

- The testing of the software developed or IT product maintained by other Units of ECDC. At a later stage and in accordance with the adopted IT Target Operating Model, the software development and IT product maintenance should be centralised in ICT, until then the strategy shall be used as a source of reference.
- Static testing: the reviews of the requirements are managed in the Requirement management process and the review of high level design and technical design are managed in the Software development process

## 1.3 How to use the document

In order to get acknowledgment on the Test Strategy it is recommended to study the whole document once.

During the planning of project or maintenance activity, the approach to use this strategy is as follows:

1. From the *Table 6: Test Strategy per IT service (Section 7.2)* select the relevant column(s) with activity applicable to your project or product maintenance e.g. "Development of a new solution or IT Product", minor development, (in the future this might be called "further development"). In case there are two possible strategies (light and full), select one of them according to the criteria defined in *Section 0*

---

<sup>1</sup> ISTQB is International Software Testing Qualifications Board, detailed information on it can be found at [\[LINK\]](#)

<sup>2</sup> [The link](#) to the IT Test Policy

2. *Selection Criteria for light or full Test Strategy* .
3. The chosen columns from *Table 6: Test Strategy per IT service (Section 7.2)* indicates whether the applicability of test levels and regression or smoke tests. Refer to section *10.2 Documents* for the more information on documentation per test processes.
4. Refer to the *Table 7 Test approaches per Test Level (Section 7.4)* for test approaches applicable for each Test Level.

The strategy may be tailored to fit the specific project or product needs, but this should be justified and documented in the Test section of the PMP or Release plan.

## 2 Testing Principles

ECDC adheres to the "7 principles"<sup>3</sup> of testing, as follows:

**Principle 1 – Testing shows presence of defects**

*Testing can show that defects are present, but cannot prove that there are no defects. Testing reduces the probability of undiscovered defects remaining in the developed solution but, even if no defects are found, it is not a proof of correctness.*

**Principle 2 – Exhaustive testing is impossible**

*Testing of all combinations of inputs and preconditions is not feasible, except for trivial cases. Instead of exhaustive testing a risk analysis and priorities should be used to focus testing efforts.*

**Principle 3 – Early testing**

*To find defects early, testing activities shall be started as early as possible in the software development lifecycle and shall be focussed on defined objectives.*

**Principle 4 – Defect clustering**

*Testing effort shall be focused proportionally on the expected and later observed defect density of modules. A small number of modules usually contains most of the defects discovered during pre-release testing, or is responsible for most of the operational failures.*

**Principle 5 – Pesticide paradox**

*If the same tests are repeated over and over again, the same set of test cases will no longer find any defects. To overcome this "pesticide paradox", test cases need to be regularly reviewed and revised and new and different tests need to be written to exercise different parts of the software or system to be able to detect potentially more defects.*

**Principle 6 – Testing is context dependent**

*Testing is done differently in different context. For example, EWRS (business critical application) is tested differently from an application that supports an internal administrative process, e.g. Allegro.*

**Principle 7 – Absence-of-errors fallacy**

*Finding and fixing defects does not bring value if the system built is unusable and does not fulfil the users' needs and expectations.*

The following principles should also be considered:

**Principle 8 – Testers are no Quality Police**

The tester should be a collaborator, not an enforcer. Testers, developers and business owners have one common goal to produce a solution that complies with the IT Product Quality Acceptance criteria and fit for the user's needs.

**Principle 9 - Keep it simple**

Test documentation should be practical. The goal of testing is not to produce lots of documents, but to achieve a good quality IT product, therefore there is no need for a test plan of tens of pages, but a simple one-page test plan with references to the test strategy is enough.

---

<sup>3</sup> As it is recognised by ISTQB, the principles offer general guidelines that are common for all testing.

**Principle 10 - Keep it simple II**

The test execution should be started with the "happy paths" for the highest risks and highest priority requirements. Once the "happy path" is acceptable, then continue with the exceptions.

**Principle 11 - Enough is enough**

One of the most important aspects of test management is finding the balance between 'not enough testing that results in possible production defects that are costly' and 'too much testing that has no extra value and also increases the costs' .

**Principle 12 - Testing should be objective**

As testing results are in the assessment of the solution under test, this assessment should be objective.

Therefore, testing should be is being performed

- conducted against pre-defined test cases and agreed acceptance criteria
- performed by an independent role to ensure that quality is not compromised with project or maintenance priorities.

## 3 Test Levels and Types

### 3.1 Test Levels

A Test Level is a group of test activities that are organised and managed together.

A Test Level is linked to the responsibilities in a project or maintenance.

The Test Levels are:

- Unit Testing<sup>4</sup> , including unit integration (Table 1)
- System Testing (Table 2);
- System Integration Testing, incl. integration with systems in Member States (Table 3):
- Acceptance Testing (Table 4):
  - Validation Feedback (will be executed after or in parallel with the System Testing)
  - User Acceptance
  - Operational Acceptance
  - Security Acceptance
  - Service Desk Acceptance

The Test Levels are presented in detail in the following tables.

*Table 1: Test Level – Unit Testing and Unit Integration Testing*

Unit Testing (incl. Unit Integration Testing)	
Definition	Testing performed to test individual units and the integration between the different units to expose defects in the interfaces and interaction between integrated units.
Test Objective	To ensure that the units are working as defined and are integrated in a correct way and working together as expected
Test Basis	<ul style="list-style-type: none"><li>• Code</li><li>• Detailed design</li><li>• Unit requirements</li><li>• For the unit integration test: Unit tested Units</li></ul>
Test Objects	<ul style="list-style-type: none"><li>• Interfaces</li><li>• Units</li><li>• Programs</li><li>• Data conversion / migration programs</li><li>• Database modules</li></ul>
Test executed by:	Software Development Service Provider

---

<sup>4</sup> Part of development activity and added for information



Unit Testing (incl. Unit Integration Testing)	
Entry Criteria	<ul style="list-style-type: none"> <li>Code is finished (not in the case of test driven development)</li> </ul>
Exit Criteria	<ul style="list-style-type: none"> <li>All unit tests are executed successfully</li> <li>No open defects</li> </ul>
Test Deliverables	N.A.

Table 2: Test Level - System Testing

System Testing	
Definition	Testing an integrated system to verify that it meets specified requirements.
Test Objective	System testing is concerned with the behaviour of the whole system / IT Product. Its objective is to ensure that the system is correctly behaving as defined in the requirements.
Test Basis	<ul style="list-style-type: none"> <li>Solution requirements (functional and non-functional<sup>5</sup>).</li> <li>Stakeholder requirements</li> <li>Risk analysis (if available)</li> </ul>
Test Objects	<ul style="list-style-type: none"> <li>System, user and operation manuals</li> <li>System configuration and configuration data</li> </ul>
Test prepared by	Test Analyst from Software Development Service Provider
Test executed by	Tester from Software Development Service Provider
Entry Criteria	<ul style="list-style-type: none"> <li>Unit testing is finished</li> </ul>
Exit Criteria	<ul style="list-style-type: none"> <li>100% of all Solution requirements according to the agreed scope are verified with no open defects.</li> <li>Exception: if it is justified and agreed to leave select defects open. This should be documented</li> </ul>
Test Deliverables	QA Report (with the system test results)

Table 3: Test Level - Integration Testing (System Integration)

Integration Testing (System Integration)	
Definition	<p>Testing performed to:</p> <ul style="list-style-type: none"> <li>- expose defects in the interfaces and in the interactions between integrated units or systems;</li> <li>- Testing performed to expose defects in the interfaces and interaction</li> </ul>

<sup>5</sup> only a subset of non-functional requirements can be tested during System Testing as the environment of the test does not fully enable e.g. security and performance testing.

Integration Testing (System Integration)	
	<p>between hardware and software components;</p> <ul style="list-style-type: none"> <li>- Testing the integration of systems and packages (ECDC internal or external, e.g. in Member States)</li> <li>- Testing interfaces to external organizations (e.g., Electronic Data Interchange, Internet).</li> </ul>
Test Objective	The objective is to ensure that the system interacts with its environment (different systems, hardware and software, external environment such as the web) as expected.
Test Basis	<ul style="list-style-type: none"> <li>• Solution requirements (functional and non-functional) (including data and user migration)</li> <li>• Stakeholder requirements</li> <li>• High level design</li> <li>• Risk analysis (if available)</li> </ul>
Test Objects	<ul style="list-style-type: none"> <li>• Subsystems</li> <li>• Database implementation</li> <li>• Infrastructure</li> <li>• Interfaces</li> <li>• System configuration and configuration data</li> </ul>
Test prepared by	Test Analyst from IT Infrastructure Service Provider <sup>6</sup>
Test executed by:	Tester from IT Infrastructure Service Provider
Entry Criteria	<ul style="list-style-type: none"> <li>• System test successfully completed</li> <li>• Release package is installed successfully</li> <li>• Smoke test has run successfully</li> </ul>
Exit Criteria	<ul style="list-style-type: none"> <li>• System Integration test has run successfully</li> </ul>
Test Deliverables	QA Report (with integration test results)

Table 4: Test Level - Acceptance Testing

Acceptance Testing	
Definition	Testing with respect to user needs, requirements and business processes, which is conducted to determine whether a system satisfies the acceptance criteria (defined as part of the stakeholders requirements) and to enable the users and other stakeholders to determine whether to accept the solution.

<sup>6</sup> Integration testing activities may be partly outsourced. The Master test plan should cover this.

Test Objective	The goal of acceptance testing is to establish confidence in the solution or specific non-functional characteristics of the solution. The main focus of the acceptance testing is the solution readiness for deployment to the production environment, and detecting defects is in the focus as well.
Test Basis	<ul style="list-style-type: none"> <li>• Stakeholder requirements</li> <li>• Solution requirements (including, but not limited, to data and user migration)</li> <li>• Business processes</li> <li>• Risk analysis report (if available)</li> </ul>
Test Objects	<ul style="list-style-type: none"> <li>• Business processes in fully integrated system</li> <li>• Operational and maintenance processes</li> <li>• User procedures</li> <li>• Forms</li> <li>• Reports</li> <li>• Configuration data</li> </ul>
Test prepared by:	<ul style="list-style-type: none"> <li>- Functional validation - tester from Software Development Service Provider executed during system testing</li> <li>- User Acceptance: BO and requirement provider or delegated role (BA)</li> <li>- Operational Acceptance: system administrators</li> <li>- Security Acceptance – ICT Security Officer</li> <li>- Service desk Acceptance –the Head of the Service desk</li> </ul>
Test executed by:	<ul style="list-style-type: none"> <li>- Functional validation - tester from Software Development Service Provider together with requirement provider (during system test)</li> <li>- User Acceptance - BO, requirement provider, user.</li> <li>- Operational Acceptance – Tester from IT Infrastructure Service Provider</li> <li>- Security Acceptance – Tester from IT Infrastructure Service Provider and if needed Tester with expertise in Security<sup>7</sup></li> <li>- Service desk Acceptance – review by the Head of the Service desk</li> </ul>
Entry Criteria	<ul style="list-style-type: none"> <li>• System Integration testing is finished successfully</li> </ul>
Exit Criteria	<ul style="list-style-type: none"> <li>• Sign off (acceptance by the stakeholders that were involved in the acceptance tests)</li> </ul>
Test Deliverables	QA Report (with the acceptance test results) Official Acceptance per testing stakeholder

<sup>7</sup>Security testing can be divided indifferent tests such as roles & authorisation, fire walls, penetration (Pen) testing etc. For some of these test very specific knowledge is needed

## 3.2 Test Types

A Test Type is a group of test activities aimed at testing a Unit or IT product on a specific Test Objective. A Test Type may take place on one or more Test Levels.

The following Test Types should be performed:

- **Functional testing:**  
Testing based on an analysis of the specification of the functionality of a component or solution (testing “what” the solution does)
- **Non-functional testing<sup>8</sup>:**  
Testing of the attributes of a component or system that do not relate to functionality (testing “how well” the solution works). Non-functional testing includes, but not limited to: Deployment, Performance, Load, Stress, Security, Privacy, Usability, Maintainability, Reliability, Portability, Installability testing etc.
- **Regression Testing:**  
Testing of a previously tested solution following modification to ensure that defects have not been occurred or are uncovered in unchanged areas of the solution. It is performed when the software or its environment is changed.  
The regression test consists of previously defined and used test cases.  
The size of the regression test set is based on the product risk of the solution.
- **Confirmation testing**  
Testing that runs test cases that failed the last time they were run, in order to verify the success of corrective actions.
- **Smoke testing:**  
Testing of a subset of all defined or planned test cases that cover the main functionality of a component or system in order to ensure that the most crucial functions of a solution work (there is no need to test in detail). It is recommended to automate the smoke tests as much as possible.

---

<sup>8</sup> A list of the quality attributes is available in Annex A [\[LINK\]](#)

## 4 Test Approaches

As per test Principle 2, an exhaustive testing is not possible therefore a choice should be made what are the priorities and what to test.

There are different ways for selection, allocation and prioritization of tests that are called Test Approaches.

In this Test Strategy only the approaches to be used in ICT Unit are described:

- Risk-based Testing
- Requirement-based Testing
- Business Processes-based Testing
- Session-based Testing
- Methodical Testing

Risk based testing is the basis for the test strategy but techniques from other approaches will also be used such as requirement, business process and, for exploratory testing, session based testing.

Different test levels may require different approaches, for example, User Acceptance Test may require business process and requirements based testing, System Integration Test requires risk and (non-)functional requirements based testing. The applicable approaches for each test levels are defined in section 7.4.

### 4.1 Risk-based Testing

**Definition:** An approach to testing aiming to reduce the level of product risks and inform stakeholders of their status, starting in the initial stages of a project or maintenance. It involves the identification of product risks and the use of risk levels to guide the test process.

The product risks are taken into account to define what to test, when, and how much, based on risks, thus ensuring that risky IT Product parts are tested more intensively and earlier than parts with a lower risk. The risks guide the planning, test specification and test execution activities. These risks are related to requirements.

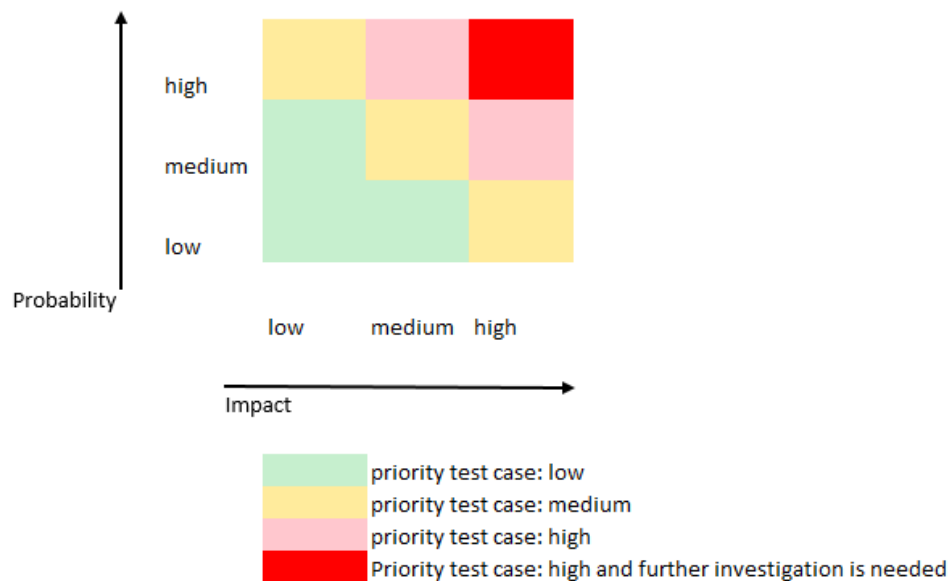
It is also a way to confirm that any potential risk remaining after testing is visible and acceptable to relevant stakeholders.

In the case of time constraints to test all requirements, Risk-based testing involves testing the scenarios with the highest impact and probability of failure.

#### Deciding on priority of test cases.

During requirement elicitation Business Analyst for each requirement determines the impact (low, medium or high) and the probability (low, medium or high) if the requirement is not correctly implemented. The risk value is decided by considering the impact and probability of the risk as shown in the Figure 1. Where each colour represents a level of risk.

Figure 1: Product risk resulting in test case priority



The requirements in the green area will be tested with test cases with a low priority, those in the yellow area result in test cases with a medium priority and all requirements in the pink and red area will result in test cases with a high priority

Requirements red area require extra verification, this is either done by creation of extra test cases or by exploratory testing, in which case one or more test charters are created.

The level of risk is used to sequence and prioritize the testing, thus ensuring early coverage of the most important areas and discovery of the most important defects during test execution.

## 4.2 Requirements-based Testing

**Definition:** An approach to testing in which test cases are designed based on test objectives and test conditions derived from requirements, e.g. tests that exercise specific functions or probe non-functional attributes such as reliability or usability.

If the requirements are prioritised, these priorities can be used to allocate effort and priorities to the test cases.

Prerequisite: The quality of the requirements must be sufficient (complete, testable and not ambiguous).

## 4.3 Business Process-based Testing

**Definition:** An approach to testing in which test cases are designed based on descriptions and/or knowledge of business processes.

By applying this approach, it is possible to simulate the day to day work for the users.

## 4.4 Session-based Testing

**Definition:** An approach to testing in which test activities are planned as uninterrupted sessions of test design and execution, often used in conjunction with exploratory testing.

During the creation of the test documentation, first the decision should be made on which areas to focus during exploratory testing and then the decision should be applied. In practice, for each of the areas a small form is created called a Test charter.

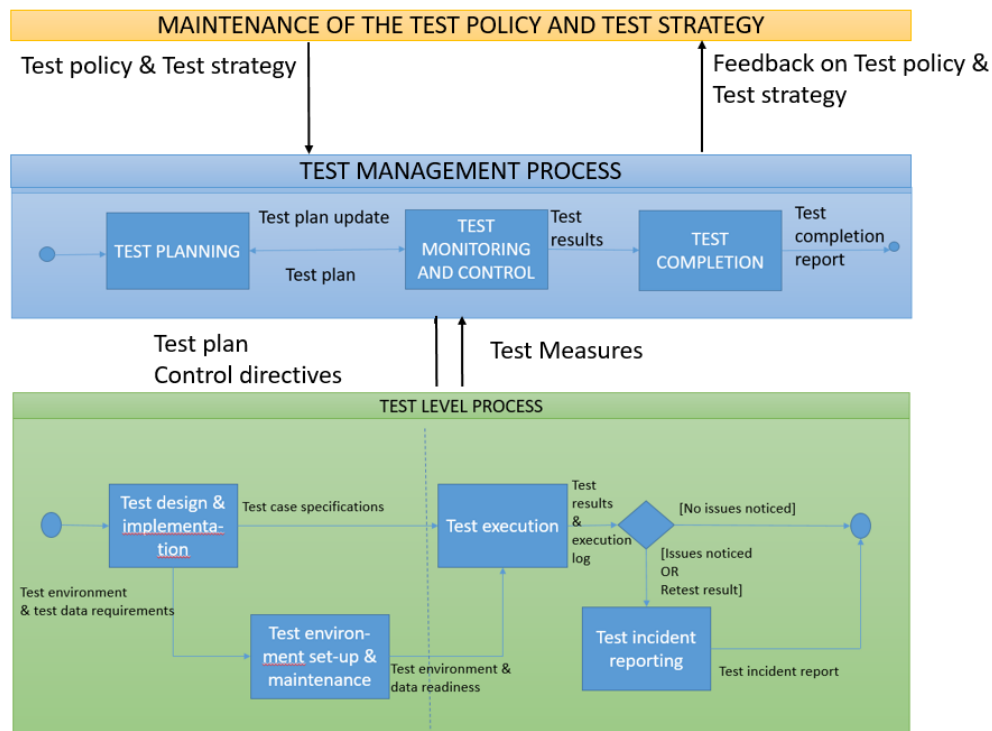
The exploratory testing is split up in test sessions, two hours duration of each of them, and each session should cover a test charter to maintain focus and ensure that at the end of the testing all areas are covered.

## 4.5 Methodical Testing

**Definition:** An approach to testing whereby a pre-determined set of test conditions, such as a quality standard, a checklist or a collection of generalized, logical test conditions which may relate to a particular domain, application or type of testing (e.g. checklist for Security and privacy) is used.

## 5 Test Processes

Figure 2: Test Processes



The test process model consists of three layers:<sup>9</sup>

- **The maintenance of the Test Policy and Test Strategy**  
IT Test Policy and Test Strategy documents will be maintained as part of ICT Process documents. The process is described in ICT Process manual (in development, it will replace the ICT Quality policy)
- **The Test Management process** describes planning, monitoring and control of test activities, as well as the test completion. The Master Test plan is incorporated in the PMP. The process is described in the Internal Procedure on Project Governance and Management.
- **The Test Level process** takes place at each test level, and it can also be executed for a test type, for example performance test.

The elements of test level processes are described<sup>10</sup> in :

- Software Development process: Unit Test, System Test, Functional validation, User Acceptance Test (UAT), and references to the System Integration test;
- Infrastructure Change Management process: System Integration Test and Operational Acceptance Test.

<sup>9</sup> Reference documentation: [ISO/IEC/IEEE International Standard 29119-Software and systems engineering – Software testing](#)

<sup>10</sup> The process documentation will be reviewed and adapted where needed in 2018



## 6 Test roles and key stakeholders

### 6.1 Test roles

The test roles and their responsibilities are detailed in the table below Table 5: Testing Roles. One individual can assume more than one test role. For example, for testing as part of minor maintenance, all test roles can be assigned to one individual, if the individual has corresponding skills and expertise. For larger projects there can be more than one individual with a specific test role, such as Tester with expertise in Security. The roles and assignments are decided and allocated during the planning of the testing activities in the project or maintenance by IT Project/Product Manager.

*Table 5: Testing Roles*

Role	Responsibilities
Test Manager	Testing planning, monitoring, control, reporting, collecting and reporting on metrics
Test Analyst	<ul style="list-style-type: none"> <li>• Definition and creation of needed test data</li> <li>• Setting up and maintenance of the test environments</li> <li>• Definition of test conditions</li> <li>• Review of the requirements</li> <li>• Review of high-level designs</li> </ul>
Tester	<ul style="list-style-type: none"> <li>• Creation of the test cases</li> <li>• Execution of test cases</li> </ul>

Note: these roles apply to all levels of testing and though their responsibilities remain the same, they need different expertise. E.g. a tester for a penetration tests needs very different expertise than a system tester

### 6.2 Testing stakeholders

Testing Stakeholder is anyone who has an interest in the testing activities, the testing work products, or the quality of the final system or deliverable. The stakeholder's interest can be direct or indirect involvement in the testing activities, direct or indirect receipt of testing work products, or direct or indirect effect by the quality of the deliverables produced by the project or maintenance.

Testing Stakeholders may vary, depending on the project, the product, the organization and other factors. They can include the following roles (this list is not comprehensive):

- **Business Owner**  
S/He is overall responsible to validate that the system requirements are delivered as agreed and the solution satisfies user and other stakeholder needs and is fit for business purpose (e.g. through User Acceptance Testing). Business Owner takes decision to 'go-live' based on the test results.
- **Users**  
These stakeholders use the software directly (i.e., they are the end-users), or receive outputs or services produced or supported by the software.
- **Developers, Lead Developers and Development Managers**  
These stakeholders can be both internal or external. They implement the software under test, receive test results, and often should take action based on those results (e.g. fix reported defects).

- **Testing team**  
These stakeholders can be both internal or external. They plan, define, execute and report on the testing.
- **Project Manager, Product Manager**  
These stakeholders are responsible for managing their projects to success or monitoring maintenance activities, which requires balancing quality, schedule, feature and budget priorities. They collaborate with the Test Manager in test planning and control.
- **Solution Architect and Designers**  
These stakeholders design the software, receive test results, and may need to take action on those results.
- **Requirement Providers**  
These stakeholders determine the features and the level of quality inherent in those features that must be present in the software. They validate the requirements and their implementation (e.g. through User Acceptance Testing).
- **Help Desk**  
These stakeholder supports the users and customers. This stakeholder's activity depends on the quality of the solution and the quality of the system documentation.
- **Operations**  
These stakeholders are responsible for the future day to day availability of the solution under test. They have an interest in the quality of the solution.
- **IT Service provider for Maintenance**  
This stakeholder is responsible for maintaining the solution, thus their activities depend on the quality of the source and the documentation to perform their role adequately.
- **ICT Security Officer**  
This stakeholder needs to know that the security and privacy aspects of the solution are well guarded.
- **Legal Services, Legal Department**  
This stakeholder is involved in case the solution falls under legislation to ensure all legal aspects are covered.

## 7 Test Strategies per IT service activities

### 7.1 Introduction

As testing is context dependent, the Test Strategy can be adapted to the situation. In this case the tailoring use of the Test Strategy should be documented in the Test Plan or test section of the project plan or the Release Plan.

The testing process depends on the different IT services as they have their own specific activities.

The IT services<sup>11</sup> with their activities that are within the scope of the Testing Strategy are (reference to column in Table 6):

- **New IT Solution Services:**

- Development of a new solution or IT Product (N-01)
- New Commercial off-the-shelf (COTS) products with N-02) and without customisation (N-03)
- Software as a service (SaaS) products with (N-04) and without customisation (N-05):  
Working with SaaS brings a number of challenges:
  - Data is stored on the vendor's servers, so extra attention must be given to data security and privacy. It is not just the vendor's responsibility. This means that special attention must be given to **security aspects and security testing**;
  - In the SaaS model, new versions, which might result in instability from defects in the newer software, are put in place outside the view of ECDC;
  - Maintenance and patch revision processes are not controlled by ECDC and therefore it is not clear what is done and if new features etc. are in line with what ECDC needs;

- **IT Product Maintenance Services:**

- maintenance carried out as a project<sup>12</sup> (M-01)
- IT Product Maintenance (M-02)
- Maintenance, COTS customisation (M-03)
- Maintenance, SaaS customisation (M-04)
- Maintenance, environment change:

There are two types of changes:

- Changes that have impact on security, data, software, such as a database upgrade, service pack with security impact (M-05). On a case by cases basis together with the ICT Security Officer it will be decided which strategy will be applied: strategy M-05, its customization or that they will be tested in the same way as changes without impact (M-06).
- Changes without any impact, such as service pack without impact on security, data, software (M-06).

---

<sup>11</sup> IT services are listed as per Service Catalogue from IT Target Operating Model, approved document version (SMTWritten22)

<sup>12</sup> The decision if selecting Maintenance as a project or IT Product Maintenance should be made according to [ICT Guideline - Project and Product Maintenance separation](#)

## 7.2 Test Strategy per IT service activity

Table 6: Test Strategy per IT service activity

Strategy id	N-01	N-02	N-03	N-04	N-05
Activity	Development of a new solution or IT Product	New COTS products (Customisation)	New COTS products (No customisation)	Saas (Customisation)	SaaS (No customisation)
Strategy	Full	Full	Light	Full	Light
Prepare Test Plan	Test section in PMP	Test section in PMP	Test section in PMP	Test section in PMP	Test section in PMP
Review requirements	X	X		X	
System test	X	X		X	
Integration test	X	X	X	X (if integration with other systems)	X (if integration with other systems)
Acceptance test					
Functional validation	X	X		X	
User acceptance test	X	X		X	
Security	X			X	x
Operations	X	X	X		
Service desk	X	X		X	
Production					
Smoke test	X	X	X		
Regression test				daily	daily

Strategy id	M-01	M-02	M-03	M-04	M-05
Activity	Maintenance as a project	IT Product Maintenance	Maintenance, COTS customisation	Maintenance, Saas Customisation	Maintenance environmental change (with security or data impact)
Strategy	Full	Light	Light	Light	Light
Prepare Test Plan	Test section in PMP	Test section in release plan	Test section in release plan	Test section in release plan	Test section in release plan
Review requirements	X	X	X	X	X
System test	X	X	X	X	X
Integration test	X	X	X	X (if integration with other systems)	X
Acceptance test					
Functional validation	X	X	X	X	X
User acceptance test	X	X	X	X	X
Security	X	X		X	
Operations	X	X	X		X
Service desk	X	X	X		X
Production					
Smoke test	X	X	X		X
Regression test				daily	

M-06 Maintenance environment changes with no security or data impact, such as service pack without impact on security, data, software, are tested as follows: the service pack is installed in the development environment, and if after a predefined period no change related incidents occur, the change is implemented into the integration test environment for further testing. No specific test cases are necessary.

Any tests executed in Production should have no impacted to, the production data or functionality. This can be ensured either by executing only test cases not affecting data (e.g. generate report, view pages) or by using a specific test data set (incl. users) or by removing removed/inactivating test data after test completion. The chosen strategy shall be documented in corresponding planning document (Release plan, PMP or a separate Test plan).

## 7.3 Selection Criteria for light or full Test Strategy

For the different IT services there will be a light (if applicable) and/or a full Test Strategy.

The full Test Strategy is mandatory for:

- Development of a new solution or IT Product;
- Maintenance release of mission critical systems (Business Impact Priority 1);
- Maintenance release for complex changes;
- Maintenance release with security impact.

Together with the ICT Security Officer the need to do security testing will be decided on a case by case basis

The differences between the full and the light Test Strategy are in selection of test cases:

- the full strategy will be risk based combined with requirements based
- the light strategy is only requirements based

A different test strategy may be chosen (e.g. a trivial change for a mission critical IT Product). In that case the test section of the release plan should contain a short justification for such deviation.

## 7.4 Test approaches per Test Level

The table below indicates test approaches from section 4 applicable for each test level.

*Table 7 Test approaches per Test Level*

Test level / Test approach	Risk-based	Requirement-based	Business process-based	Session-based	Methodical
System Test	full	full + light		full + light	full
Validation Feedback*		full + light			
System Integration Test	full + light	full + light			
User Acceptance Test		full + light	full + light		
Security Acceptance Test	full + light t.b.d**.	full + light t.b.d.			full + light t.b.d.
Operations Acceptance Test	full + light	full + light			
Service Desk Acceptance		full review			

\* Part of Acceptance test level

\*\* To be discussed with the ICT Security Officer on a case by case basis

## 8 Test Metrics

Test Metrics are used to provide insight on the quality and progress of the testing process.

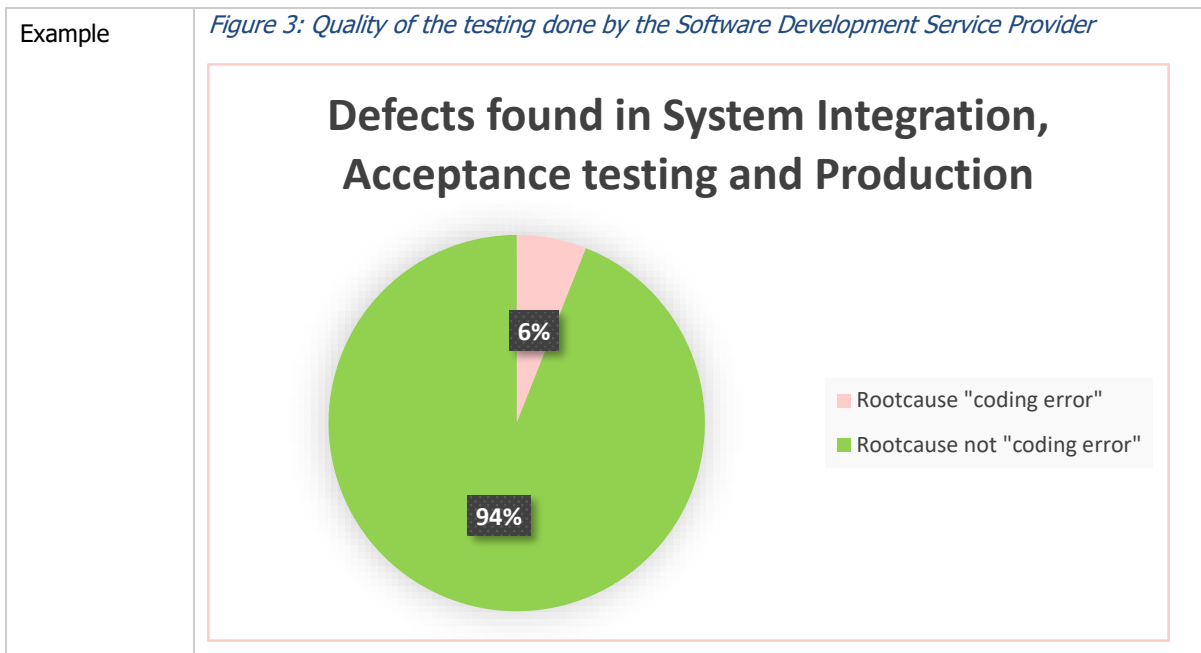
Test Metrics are defined to give an answer to the following questions regarding testing and testing related processes:

- What is the quality of the testing done by the Software Development Service Provider?
- What is the duration of fixing defects from System Integration and Acceptance testing by the Software Development Service Provider?
- What is quality of the performed System Integration Testing and Acceptance Testing?
- What is the progress of the ongoing testing?
- How much effort is spent on testing as a percentage of the total effort on project or maintenance? <sup>13</sup>

The measurements apply to both external and internal service providers. The metrics below are proposal and shall be revised during their implementation and, if necessary, incorporated as part of project or product maintenance measurements.

Information needed	What is the quality of the testing done by the Software Development Service Provider
Explanation	When the tests performed by Software Development Service Provider are executed, defects in the system itself like wrong dialog, wrong calculation etc. should not be found anymore in the later test levels or production.
Goal	To improve the quality of the delivered solution by the Software Development Service Provider
Measurement	All defects in later Test Levels (System Integration and Acceptance Testing) and Production with root-cause "Coding error" will be counted
Frequency	At each deployment to production all defects from System Integration Testing and Acceptance testing are investigated
Gathered by	Test Manager (role)
Representation	Percentage (pie chart)
Limit	<i>To be defined</i> (e.g. 5 % of total defects)
Prerequisites	The root cause field in the defect should be filled in

<sup>13</sup> This measure is covered by KPI-13 of the Overview of Key Performance Indicators in the SLA (concept)



Information needed	What is the duration of fixing defects from System Integration and Acceptance testing by the Software Development Service Provider
Explanation	When the Software Development Service Provider does not fix the defects found during System Integration and Acceptance testing within the agreed time-frame, planning of further testing will be difficult, progress of the test executing might be delayed and it is not feasible to achieve deadlines of the project or maintenance activities.
Goal	To improve timely delivery of fixed defects from System Integration testing and Acceptance testing
Measurement	For all tickets with critical, high and medium severity the time between the status assigned and ready for re-test is measured. Both an average and the number of tickets with a duration longer than agreed will be given.
Frequency	During the testing period on a weekly base
Gathered by	Test Manager (role)
Representation	The values will be in column and line charts
Limit	<i>To be defined</i>
Prerequisites	The status of the defect tickets must be filled in immediately

Example

Figure 4: Average duration of defect solving

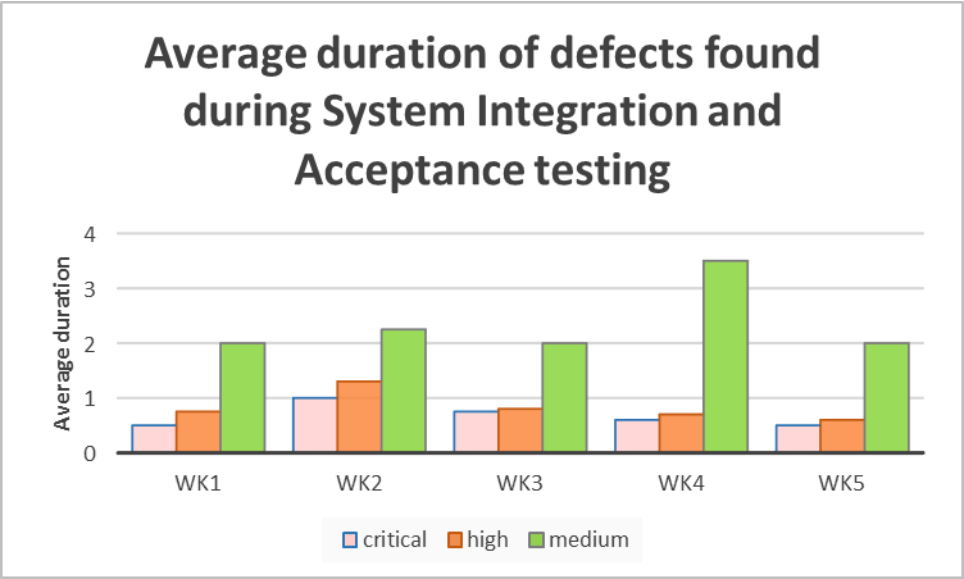
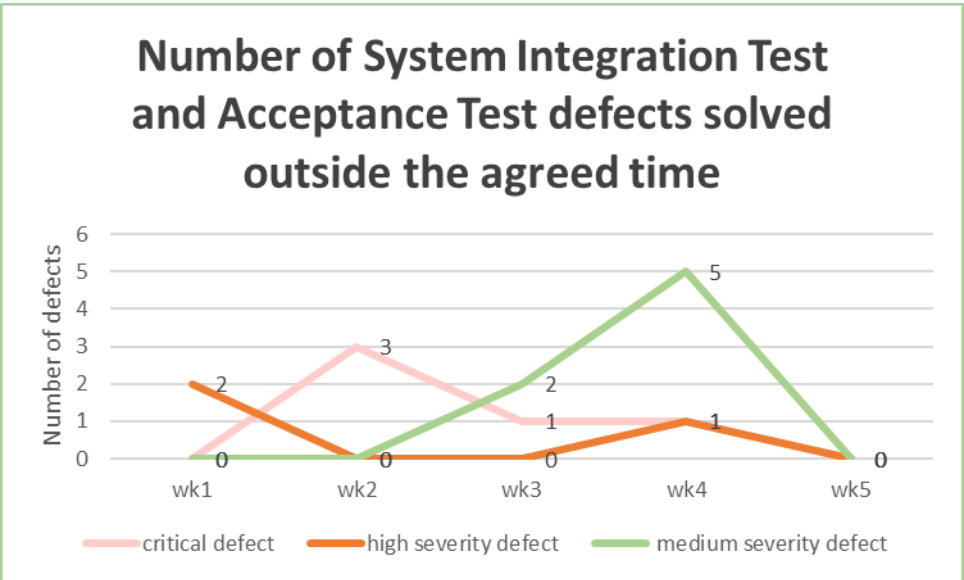


Figure 5: Number of IT and AT defects solved outside the agreed time

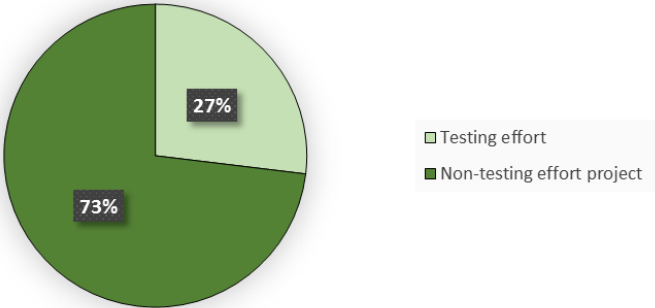
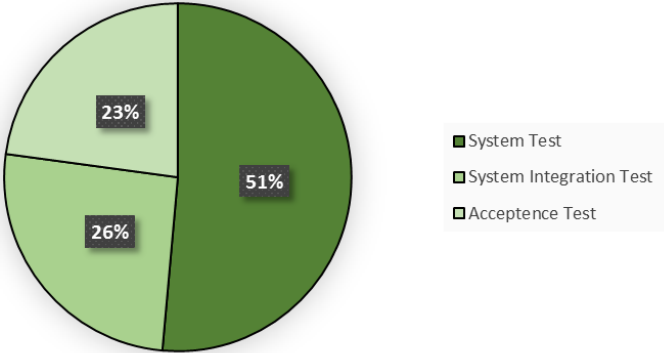




Information needed	What is the quality of the performed Acceptance and System Integration testing																																							
Explanation	If the testing is performed correctly according to the Test Plan, production defects will be prevented. There should not be a significant increase of defects detected after the deployment of a new release or new IT Product. Only releases of newly developed IT Product and adaptive and perfective maintenance will be taken into consideration..																																							
Goal	To get a stable production environment directly after a new release.																																							
Measurements	The number of Critical and High severity defects found in Production, that were not prevented by System Integration and Acceptance testing																																							
Frequency	Weekly																																							
Gathered by	2 <sup>nd</sup> level support																																							
Representation	Line chart																																							
Limit	<i>To be defined</i>																																							
Example	<div><p>Figure 6: Defects in production in relation to major releases</p><table><caption>Production defects in relation to major releases</caption><thead><tr><th>Week</th><th>Critical Defects</th><th>High Severity Defects</th></tr></thead><tbody><tr><td>wk1</td><td>0</td><td>0</td></tr><tr><td>wk2</td><td>0</td><td>0</td></tr><tr><td>wk3</td><td>3</td><td>5</td></tr><tr><td>wk4</td><td>0</td><td>0</td></tr><tr><td>wk5</td><td>0</td><td>0</td></tr><tr><td>wk6</td><td>0</td><td>0</td></tr><tr><td>wk7</td><td>2</td><td>0</td></tr><tr><td>wk8</td><td>0</td><td>0</td></tr><tr><td>wk9</td><td>0</td><td>0</td></tr><tr><td>wk10</td><td>0</td><td>0</td></tr><tr><td>wk11</td><td>3</td><td>4</td></tr><tr><td>wk12</td><td>0</td><td>0</td></tr></tbody></table></div>	Week	Critical Defects	High Severity Defects	wk1	0	0	wk2	0	0	wk3	3	5	wk4	0	0	wk5	0	0	wk6	0	0	wk7	2	0	wk8	0	0	wk9	0	0	wk10	0	0	wk11	3	4	wk12	0	0
Week	Critical Defects	High Severity Defects																																						
wk1	0	0																																						
wk2	0	0																																						
wk3	3	5																																						
wk4	0	0																																						
wk5	0	0																																						
wk6	0	0																																						
wk7	2	0																																						
wk8	0	0																																						
wk9	0	0																																						
wk10	0	0																																						
wk11	3	4																																						
wk12	0	0																																						

Information needed	What is the progress of the ongoing Tests
Explanation	The progress of the ongoing testing should be known (by measuring the number of test cases passed, failed and still open), but there is also a need to know how big the remaining product risk is, as well as how many of the requirements are covered. The answer to all three questions together provides a clear view of what is the status of the testing.
Goal	To have a good overview of the level of remaining risks and readiness for production during the entire project
Measurements	<ol style="list-style-type: none"> <li>1. Number of passed, failed, re-tested, blocked and still to execute test cases overall (both in numbers and in percentages)</li> <li>2. Number of passed, failed, re-tested, blocked and still to execute test cases based on the risks for the critical and high risks (both in numbers and in percentages)</li> <li>3. Number of passed, failed, retested, blocked and still to execute test cases based on the requirements per priority (both in numbers and in percentages)</li> </ol>
Frequency	Weekly
Gathered by	Test manager
Representation	report
Limit	N.A.
Example	

Information needed	How much effort is spent on testing as a percentage of the total effort on project or maintenance
Explanation	In order to improve planning and estimating of testing, it is recommended to collect the data of all effort in order to calculate a percentage of the total project or maintenance activities costs.
Goal	To get a reliable estimation of test effort in an early stage of the project
Measurement	Actual time spent on testing
Frequency	After completion of a project or releases
Gathered by	Test Manager
Representation	Pie chart
Limit	N.A.
Prerequisites	Effort must be filled in correctly

<p>Example</p>	<div><p><i>Figure 7: Testing as a percentage of the total project</i></p><div><p><b>Testing as a percentage of the total project effort</b></p><table border="1"><thead><tr><th>Category</th><th>Percentage</th></tr></thead><tbody><tr><td>Testing effort</td><td>27%</td></tr><tr><td>Non-testing effort project</td><td>73%</td></tr></tbody></table></div><p><i>Figure 8: Effort per test level in percentages</i></p><div><p><b>Effort per test level (in percentages)</b></p><table border="1"><thead><tr><th>Test Level</th><th>Percentage</th></tr></thead><tbody><tr><td>System Test</td><td>51%</td></tr><tr><td>System Integration Test</td><td>26%</td></tr><tr><td>Acceptance Test</td><td>23%</td></tr></tbody></table></div></div>	Category	Percentage	Testing effort	27%	Non-testing effort project	73%	Test Level	Percentage	System Test	51%	System Integration Test	26%	Acceptance Test	23%
Category	Percentage														
Testing effort	27%														
Non-testing effort project	73%														
Test Level	Percentage														
System Test	51%														
System Integration Test	26%														
Acceptance Test	23%														

## 9 Test Tools

The tools currently used for testing in ICT Unit, together with the responsibilities and ownership are described in the table below.

*Table 8: Test Tools*

Tool category	Tool	Responsibility and ownership
Document management system (DMS)	SharePoint	ECDC
Source Control Versioning and Work item Management (SCV/WIM)	Team Foundation Server	Software Development Service Provider
Service Desk, Infrastructure Change Management and Configuration Management database (CMDB)	Ivanti Service Desk	IT Infrastructure Service Provider
Test management	Microsoft Test Manager <sup>14</sup>	Test Manager
Test execution	Microsoft Test Manager	Test Manager
Automated testing <ul style="list-style-type: none"> <li>• UI Automation to test code</li> <li>• creating browser-based automated regression test cases</li> <li>• testing different browsers (Configuration testing)</li> </ul>	Selenium + C# Coded UI (deprecated)  Browser Stack	Software Development Service Provider
Evaluating web site security	Acunetix	IT Infrastructure Provider
Record and playback	Currently MTM Microsoft Test Manager	Software Development Service Provider

<sup>14</sup> Also defect management is covered by this tool

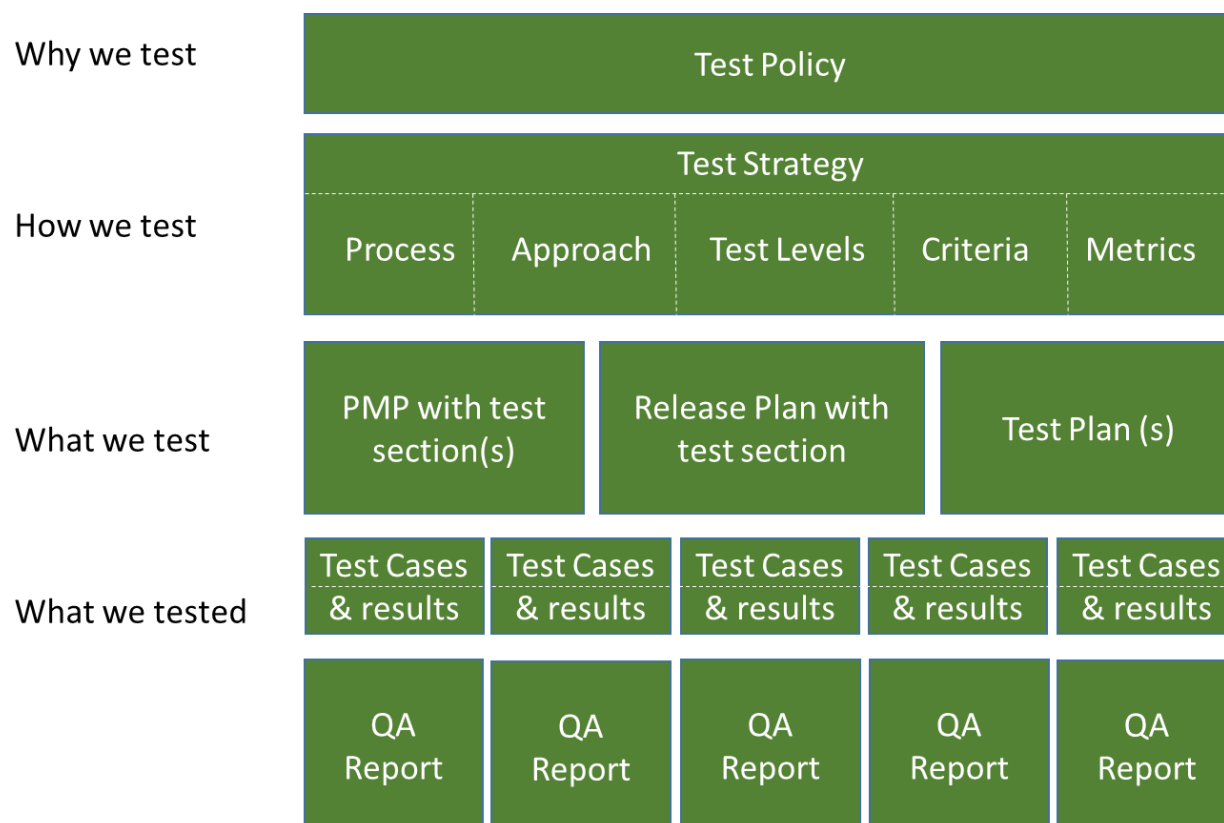
## 10 Documentation

### 10.1 Documentation Structure

Together with the Test Policy and the Test Plans this Test Strategy forms the basis of the process level of the Test Methodology.

It is important that test plans should be lean and should refer to the Test Strategy, and in case of tailoring the Test Strategy a justification needs to be provided in a Master Test Plan.

*Figure 9: Documentation structure*



The testing on the operational level will be described in related documents such as Work Instructions and Templates.

### 10.2 Documents

The following test documentation should be created and maintained:

- **The Test Strategy and Test Policy process documentation:**
  - Test Policy,<sup>15</sup>
  - Test Strategy
- **Test Management process documentation:**

<sup>15</sup> [Link to the test policy](#)

- Test Plan as part of PMP for projects. Based on the size and complexity of the system under test, Test plan can also be defined in a separate document or there can be Master Test plan with separate Test level/test type plans. The decision on test plan is taken during planning of the project.
- Test plan as part of Release plan for Product maintenance.
- QA Report for both test status reporting and test completion reporting
- **Test Level process documentation:**
  - Test Case Specification in TFS<sup>16</sup>
  - Test charter (in TFS)<sup>17</sup>
  - Internal Defect Report<sup>18</sup>
  - Formal acceptance

---

<sup>16</sup> [Link to TFS](#)

<sup>17</sup> [Still in development, link to draft](#)

<sup>18</sup> To be developed as part of the QA report

## Annex A: Quality Attributes

A quality attribute is a feature or characteristic that affects an item's quality.

To help determining the non-functional requirements here is an overview of the quality attributes with their ISTQB definition.

### **Accessibility**

the ease by which users with disabilities can use a component or system.

### **Accuracy**

The capability of the software product to provide the right or agreed results or effects with the needed degree of precision.

### **Adaptability**

The capability of the software product to be adapted for different specified environments without applying actions or means other than those provided for this purpose for the software considered.

### **Analysability**

The capability of the software product to be diagnosed for deficiencies or causes of failures in the software, or for the parts to be modified to be identified.

### **Attractiveness**

The capability of the software product to be attractive to the user.

### **Availability**

The degree to which a component or system is operational and accessible when required for use. Often expressed as a percentage.

### **Changeability**

The capability of the software product to enable specified modifications to be implemented.

### **Co-existence**

The capability of the software product to co-exist with other independent software in a common environment sharing common resources.

### **Compliance**

The capability of the software product to adhere to standards, conventions or regulations in laws and similar prescriptions.

### **Configuration**

The composition of a component or system as defined by the number, nature, and interconnections of its constituent parts.

Example: Configuration testing is done to ensure that the solution behaves correct on different devices or when using different browsers on the same device

### **Effectiveness**

The capability of producing an intended result

### **Efficiency**

1. The capability of the software product to provide appropriate performance, relative to the amount of resources

used, under stated conditions.

2. The capability of a process to produce the intended outcome, relative to the amount of resources used.

### **Error tolerance**

The ability of a system or component to continue normal operation despite the presence of erroneous inputs

### **Fault tolerance**

The capability of the software product to maintain a specified level of performance in cases of software faults (defects) or of infringement of its specified interface.

### **Functionality**

The capability of the software product to provide functions which meet stated and implied needs when the software is used under specified conditions.

### **Installability**

The capability of the software product to be installed in a specified environment.

### **Interoperability**

The capability of the software product to interact with one or more specified components or systems.

### **Learnability**

The capability of the software product to enable the user to learn its application.

### **Maintainability**

The ease with which a software product can be modified to correct defects, modified to meet new requirements, modified to make future maintenance easier, or adapted to a changed environment.

### **Maturity**

1. The capability of an organization with respect to the effectiveness and efficiency of its processes and work practices.

2. The capability of the software product to avoid failure as a result of defects in the software.

### **Operability**

The capability of the software product to enable the user to operate and control it.

### **Performance**

The degree to which a system or component accomplishes its designated functions within given constraints

Performance testing is a testing practice performed to determine how a system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage regarding processing time and throughput rate.

There are three elements that may be considered during testing

**Stress testing:** A type of performance testing conducted to evaluate a system or component at or beyond the limits of its anticipated or specified workloads, or with reduced availability of resources

**Load testing:** A type of performance testing conducted to evaluate the behaviour of a component or system with increasing load, e.g. numbers of parallel users and/or numbers of transactions, to determine what load can be handled by the component of system

**Volume testing:** Testing where the system is subjected to large volumes of data.



**Portability**

The ease with which the software product can be transferred from one hardware or software environment to another.

**Recoverability**

The capability of the software product to re-establish a specified level of performance and recover the data directly affected in case of failure.

**Reliability**

The ability of the software product to perform its required functions under stated conditions for a specified period of time, or for a specified number of operations.

**Replaceability**

The capability of the software product to be used in place of another specified software product for the same purpose in the same environment.

**Robustness**

The degree to which a component or system can function correctly in the presence of invalid inputs or stressful environmental conditions.

**Safety**

The capability of the software product to achieve acceptable levels of risk of harm to people, business, software, property or the environment in a specified context of use.

**Scalability**

The capability of the software product to be upgraded to accommodate increased loads.

**Security**

Attributes of software products that bear on its ability to prevent unauthorized access, whether accidental or deliberate, to programs and data. Security testing is a process intended to reveal flaws in the security mechanisms of an information system that protect data and maintain functionality as intended.

**Stability**

The capability of the software product to avoid unexpected effects from modifications in the software.

**Suitability**

The capability of the software product to provide an appropriate set of functions for specified tasks and user objectives.

**Testability**

The capability of the software product to enable modified software to be tested.

**Traceability**

The ability to identify related items in documentation and software, such as requirements with associated tests.

**Understandability**

The capability of the software product to enable the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use.

**Usability**

The capability of the software to be understood, learned, used and attractive to the user when used under specified conditions.

## Annex B: Definition of testing terms

The definitions presented below will also be added to the ICT Glossary:

### **Acceptance Criteria**

The exit criteria that a component or system must satisfy in order to be accepted by a user, customer, or other authorized entity.

### **Acceptance Testing**

Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system. See also *User Acceptance testing*.

### **Business Process-based Testing**

An approach to testing in which test cases are designed based on descriptions and/or knowledge of business processes.

### **Checklist-based Testing**

An experience-based test design technique whereby the experienced tester uses a high-level list of items to be noted, checked, or remembered, or a set of rules or criteria against which a product has to be verified.

### **Code Coverage**

An analysis method that determines which parts of the software have been executed (covered) by the test suite and which parts have not been executed, e.g., statement coverage, decision coverage or condition coverage

### **Commercial off-the-shelf (COTS)**

A software product that is developed for the general market, i.e. for a large number of customers, and that is delivered to many customers in identical format.

### **Cost of quality**

The total costs incurred on quality activities and issues and often split into prevention costs, appraisal costs, internal failure costs and external failure costs.

### **Defect**

A flaw in a component or system that can cause the component or system to fail to perform its required function, e.g., an incorrect statement or data definition. A defect, if encountered during execution, may cause a failure of the component or system.

### **Dynamic testing**

Testing that involves the execution of the software of a component or system.

### **Entry criteria**

The set of generic and specific conditions for permitting a process to go forward with a defined task, e.g., test phase. The purpose of entry criteria is to prevent a task from starting which would entail more (wasted) effort compared to the effort needed to remove the failed entry criteria.

### **Exit criteria**

The set of generic and specific conditions, agreed upon with the stakeholders for permitting a process to be officially completed. The purpose of exit criteria is to prevent a task from being considered completed when there are still outstanding parts of the task which have not been finished. Exit criteria are used to report against and to plan when to stop testing.

### **Exploratory testing**

An informal test design technique where the tester actively controls the design of the tests as those tests are performed and uses information gained while testing to design new and better tests.

### **Integration Testing**

Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems

### **Master Test Plan**

A test plan that typically addresses multiple test levels.

### **Methodical Test Strategy**

A test strategy whereby the test team uses a pre-determined set of test conditions such as a quality standard, a checklist, or a collection of generalized, logical test conditions which may relate to a particular domain, application or type of testing.

### **Operational Acceptance Testing**

Operational testing in the acceptance test phase, typically performed in a (simulated) operational environment by operations and/or systems administration staff focusing on operational aspects, e.g., recoverability, resource-behaviour, installability and technical compliance

### **Product risk**

A risk directly related to the test object.

### **Project risk**

A risk related to management and control of the (test) project, e.g., lack of staffing, strict deadlines, changing requirements, etc.

### **Quality attribute**

A feature or characteristic that affects an item's quality

### **Regression Testing**

Testing of a previously tested program following modification to ensure that defects have not been introduced or uncovered in unchanged areas of the software, as a result of the changes made. It is performed when the software or its environment is changed.

### **Requirements-based Testing**

An approach to testing in which test cases are designed based on test objectives and test conditions derived from requirements, e.g., tests that exercise specific functions or probe non-functional attributes such as reliability or usability.

### **Risk-based Testing**

An approach to testing to reduce the level of product risks and inform stakeholders of their status, starting in the initial stages of a project. It involves the identification of product risks and the use of risk levels to guide the test process.

### **Session-based Test Management**

A method for measuring and managing session-based testing, e.g., exploratory testing.

### **Session-based Testing**

An approach to testing in which test activities are planned as uninterrupted sessions of test design and execution, often used in conjunction with exploratory testing.

**Statement coverage**

The percentage of executable statements that have been exercised by a test suite.

**Static Testing**

Testing of a software development artefact, e.g., requirements, design or code, without execution of these artefacts, e.g., reviews or static analysis.

**System Testing**

Testing an integrated system to verify that it meets specified requirements.

**Test Approach**

The implementation of the test strategy for a specific project. It typically includes the decisions made that follow based on the (test) project's goal and the risk assessment carried out, starting points regarding the test process, the test design techniques to be applied, exit criteria and test types to be performed.

**Test basis**

All documents from which the requirements of a component or system can be inferred. The documentation on which the test cases are based. If a document can be amended only by way of formal amendment procedure, then the test basis is called a frozen test basis.

**Test case**

A set of input values, execution preconditions, expected results and execution post conditions, developed for a particular objective or test condition, such as to exercise a particular program path or to verify compliance with a specific requirement.

**Test case specification**

A document or form specifying a set of test cases (objective, inputs, test actions, expected results, and execution preconditions) for a test item.

**Test charter**

A statement of test objectives, and possibly test ideas about how to test. Test charters are used in exploratory testing.

**Test closure**

During the test closure phase of a test process data is collected from completed activities to consolidate experience, test ware, facts and numbers. The test closure phase consists of finalizing and archiving the test ware and evaluating the test process, including preparation of a test evaluation report.

**Test condition**

An item or event of a component or system that could be verified by one or more test cases, e.g., a function, transaction, feature, quality attribute, or structural element.

**Test data**

Data that exists (for example, in a database) before a test is executed, and that affects or is affected by the component or system under test.

**Test environment**

An environment containing hardware, instrumentation, simulators, software tools, and other support elements needed to conduct a test.

**Test Level**

A group of test activities that are organized and managed together. A test level is linked to the responsibilities in a project. Examples of test levels are component test, integration test, system test and acceptance test.

**Test object**

The component or system to be tested.

**Test Plan**

A document describing the scope, approach, resources and schedule of intended test activities. It identifies amongst others test items, the features to be tested, the testing tasks, who will do each task, degree of tester independence, the test environment, the test design techniques and entry and exit criteria to be used, and the rationale for their choice, and any risks requiring contingency planning. It is a record of the test planning process

**Test Policy**

A high-level document describing the principles, approach and major objectives of the organization regarding testing.

**Test Strategy**

A high-level description of the test levels to be performed and the testing within those levels for an organization or programme (one or more projects).

**Test suite**

A set of several test cases for a component or system under test, where the post condition of one test is often used as the precondition for the next one.

**Test Type**

A group of test activities aimed at testing a component or system focused on a specific test objective, i.e. functional test, usability test, regression test etc. A test type may take place on one or more test levels or test phases.

**Testing**

The process consisting of all lifecycle activities, both static and dynamic, concerned with planning, preparation and evaluation of software products and related work products to determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects.

**User Acceptance Testing**

Acceptance testing carried out by future users in a (simulated) operational environment focusing on user requirements and needs.