High-Level-Test-Plan
Digital Green Certificate

version	1.0
date	16.04.2021
status	Released
author	T-Systems

Protection category: Intern

Imprint

Issued by

T-Systems International GmbH

File name	document name
High_Level_Test_Plan_Digital_Green_Certificate_V1.0.docx	High-Level-Test-Plan

version	date	status
1.0	16.04.2021	Released

Author

T-Systems

contact person	phone / fax	e-mail
Rudi Klaus	0175 2456027	Rudi.Klaus@T-Systems.com

Brief details

The high-level test concept describes the planned test levels for a “European Digital Green Certificate Gateway and Application Template” from the unit test level to go-live.

Project Profile

project name	project owner	project manager
Digital Green Certificate	Peter Lorenz	Josef Lieven

Foreword

ISO 29119 verification

The determinations and the structure in this document are geared towards ISO 20119.

Summary

list of figures

list of tables

1 Introduction

On 17 March 2021, the European Commission presented a proposal to create a Digital Green Certificate to facilitate the safe free movement of citizens within the EU during the COVID-19 pandemic. This Digital Green Certificates will be valid in all EU Member States.

Many European countries are developing systems which allow their citizens to show their vaccination against COVID-19 and verifications systems to validate these digital certificates. The Goal of this project is, to implement a cross-border “Digital Green Certificate Gateway” to integrate these national solutions. For countries who are not so far in their planning, the EU will provide on open source base templates to build up a national solution.

This document represents a high-level test plan (HLTP) for the Digital Green Certificate Gateway and the templates called Digital Green Certificate Applications. It describes the management aspects of the test organization and the test strategy for the entire project based on the software engineering (SE) book.

The SE book is the standard basis for the development and quality assurance of software at T-Systems. Test management is an integral part of the standard discipline "Quality Assurance (QA)" and has the goal of ensuring product quality both in the phase of product development and in the operation and maintenance phase. It is based on the best practices and procedures defined by the International Software Testing Qualifications Board (ISTQB) and ensures that the QS measures defined in the QS plan are carried out and the quality goals are achieved.

This document essentially describes the following test management aspects for testing and validating the specified and commissioned requirements:

- Definition & delimitation of the test order
- Comprehensive test procedure model
  - Definition & delimitation of test levels
  - Test strategy in the individual test levels
  - Test execution (tasks)
- Test organization & communication
- Deviation management
- Management of test data and test cases
- Tool support

2 Planning of test and acceptance

2.1 Test order and test tasks

The task for the entire test team of the "Testing" workstream is to ensure that all software and software changes in the scope of European Digital Green Certificate Gateway (DGCG) and European Digital Green Certificate Application Templates (DGCA), are integrated and tested correctly.

The following figure shows the IT landscape with the components involved:

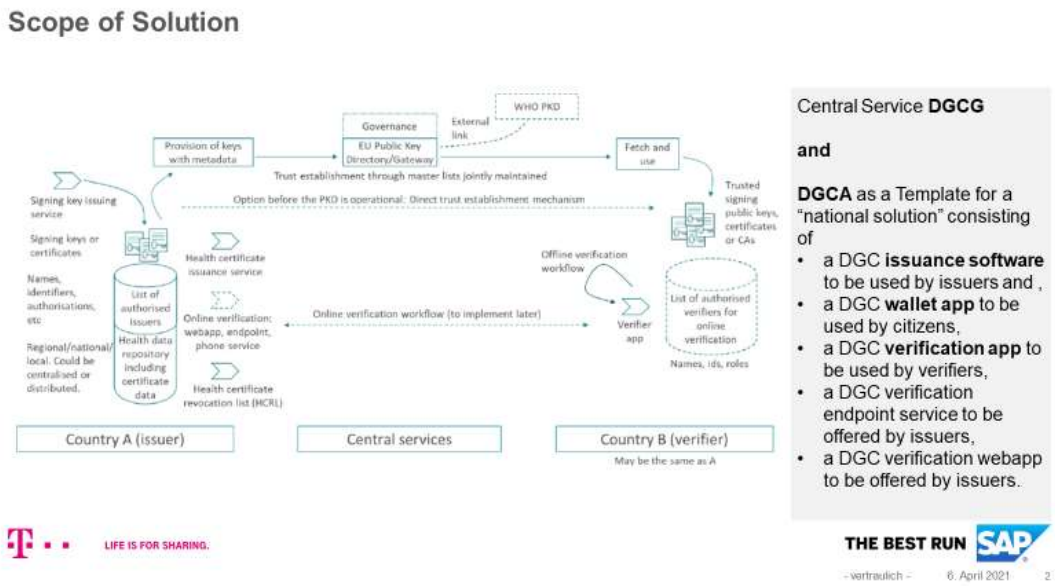

Figure 1: Overview of the Digital Green Certificate Architecture

The QS test team is responsible for the planning, preparation and execution of the tests. The test results are documented transparently and comprehensibly in the test management tool Xray.

This explicitly includes all necessary test management services.

The test object is the software system, which was developed or modified based on the architecture and design concept and installed on the respective defined test environment. It includes following components:

- Digital Green Certificate Gateway (DGCG)
- Digital Green Certificate Application Template (DGCA)

2.2 Test and milestone planning

All planning dates are coordinated between all project participants and sub teams and maintained centrally by Project Management Office and continuously adjusted depending on the project development.

The operational test control and -planning in the individual test level is carried out directly by the person responsible for the test level.

The responsible stream lead testing is involved in project planning. He contributes relevant test planning information and brings the test view into the main project planning. The decisions made there serve as the basis for further detailed planning for the following test levels:

- Unit Test
- Integration Test
- Penetration Test
- Operational Acceptance Test (incl. Dry Run)

2.3 Test organization

2.3.1 Test organization and integration into the entire project

intern

2.4 Communication

The internal communication within the test team in the project is organized as follows:

Daily Sync internal Testing:
- Frequent: daily
- Participants: Test Team

Daily Sync DGC Project
- Frequent: Daily
- Participants: Project Manager and Stream Leads

Weekly Sync EU-Digit
- Frequent: Weekly
- Participants: EU Counter part for Test

Weekly Sync Pilot Community
- Frequent: Weekly
- Participants: Member States

If necessary, dynamic adjustment calls are set up and carried out directly between the participants. The results of the adjustment are recorded in the relevant documents in order to become binding.

If there are overarching problems regarding the processing and implementation of the requirements, the escalation takes place in coordination with the project manager and the stream leads.

The eHealth Network is working in parallel to the project and is defining Use Cases and/or User Stories. We will compare these results with our defined solution. If there are new Features, then they must be brought via Change Process and Project Management into the Project

3 Test strategy

The test activities in the test team run in several test stages, which differ in terms of time and object. According to the ISTQB (International Software Testing Qualifications Board), a test level is a collection of test activities that pursue a specific test goal. They are planned, prepared and then carried out together.

In test preparation, test cases are specified in Jira / X-Ray for each test level and linked to the requirements. The aim is to completely cover the functional and operational requirements with test cases. In the test implementation, the execution of the test cases is also documented in Jira / X-Ray. If defects occur during testing, they are documented and tracked in GitHub

Each test level has a different test focus at the test objects at different levels of abstraction. The respective strategies are described in the following test level specific chapters. The following relevant aspects are discussed:

- Test goal & test characterization
- Test subject / delimitation
- Features to be tested

3.1 Overview Test Level

Test level	Test scope	Responsibility
Unit Test	Functionality within each component	Development
Integration Test	Process between DGCA and DGCG, focused on in the interface and integration of the components as planned	Testing
Penetration Test	Security	Security
Operational Acceptance Test	E2E Process with DGCG and linked a DGCA Template implementation as well as individual national solutions	Testing

Table 3.1: Overview Test Level

3.2 Unit Test

Test Goal	Test of moduls, functionality and Modul integration is part of the build process.
Test Scope	Test Object is the Gateway himself and the templates for DGCA. A static code review will be done with Sonar. National backend components are not linked to the gateway. A simulator will be started during build time of the gateway so that the interfaces can be tested upon software build. All interfaces are tested with positive and negative test cases. Tested interfaces and linked functionality to the Interface are ▪ Upload Certificates ▪ Download Certificates ▪ Signature of Sender ▪ Persistent storage in local in-memory-db
Not Test Scope	Tests with external partner systems is not planned. setup of external security infrastructure (mTLS for system access) not planned

3.3 Integration Test

Test Goal	Test of the interface and the functionality of DGCG. Integration of the DGCA Components and System Test of the DGCA functionality Integration of DGCG and DGCA.
Test Scope	Test Objects are DGCG and DGCA Components. National backend components are not linked to the gateway. The interface of DGCG is verified with DGCA. DGCG Interface Test with negative returns are tested with manual API-calls. Integrations Tests are executed on acceptance environment.
Not Test Scope	Tests with external partner systems is not planned.

Table 3.2 Integration Test

3.4 Performance Test

With the actual scope and the actual architecture there is no need of performance test for the two components DGCG and the DGCA template.

When member states take the template and implement a national solution, we would recommend a performance test. Specially the download of PKIs from the national backend should be in the focus because here you have many devices which access the national backend. The performance at this point, mainly depends on the used infrastructure. Because of this, the member state should run this test on his platform.

3.5 Penetration Test

Test Goal	To identify security leaks in systems, which can be exploited by an attacker to compromise systems or data, get unauthorized access or disturb availability. Penetration tests are simulated attacks against defined system components with defined effort.
Test Scope	T-Systems will execute penetrate tests related to the subject matter of the contract, i. e. DGCG and DGCA, and within the Secure Software Development Lifecyle. I. e., these penetration tests are T-Systems internal penetration tests to fulfill Best Practices like OWASP (Open Web Application Security Project) before going live with the DGCG. Penetration tests will be executed in the White box model.
Not Test Scope	Any complete reference installation of the DGCA, including production-like environments, are out of scope of penetration testing.

Table 3.3 Penetration Test

3.6 Operational Acceptance Test

Test Goal	Test of E2E Process with DGCG and linked a DGCA implementation as well as individual national solutions
Test Scope	The tests can be split in 2 steps. In the first step, the correct connection and usage of the API is tested. In the second step the correct up- and download of the certificates is tested. The tests are running in acceptance environment. The DGCA test implementation is using a Test-CA for signing. The member states which are participating in the test are using official certificates proceeded via CIRCABC
Not Test Scope	The functional correctness of DGCA is not scope of these tests

Table 3.4 Operational Acceptance Test

4 Execution of the tests

4.1 Test process

The present high-level test plan describes the specific test procedure for the test across all applications. The process steps shown in Figure 4: Test Management Process are based on the fundamental test process of the ISTQB (International Software Testing Qualifications Board).

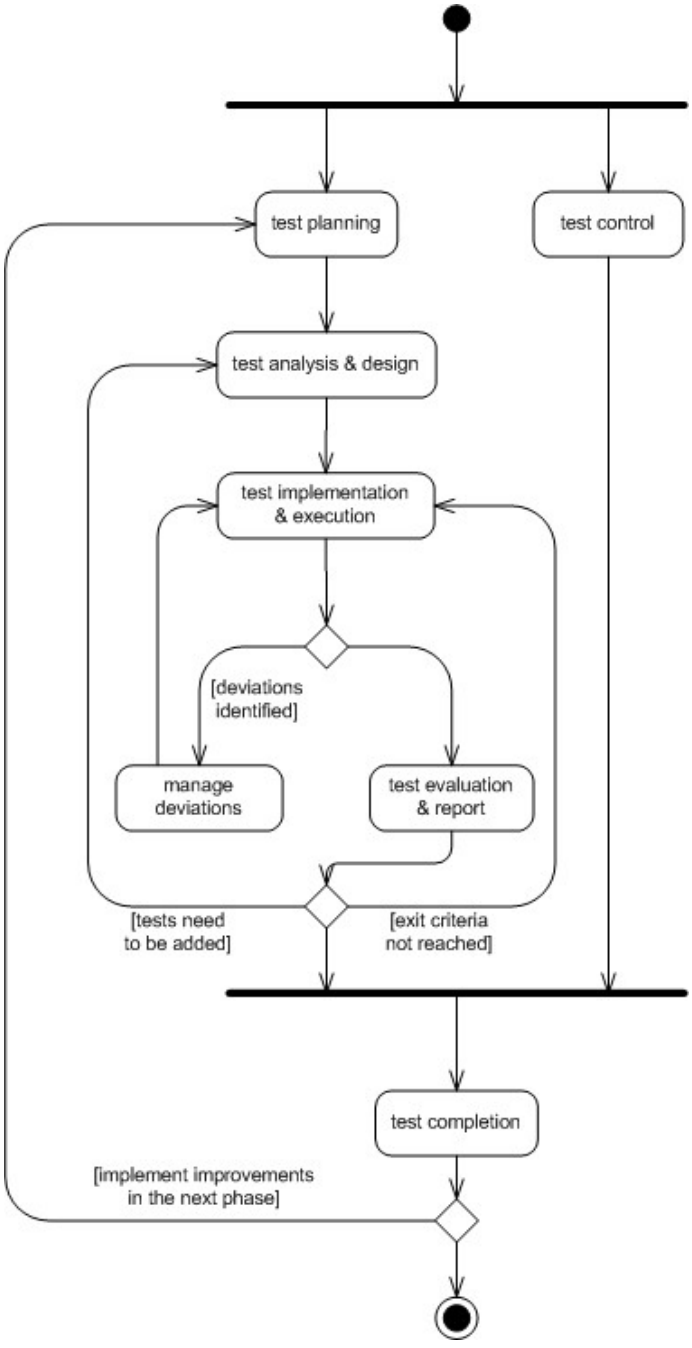

Figure 2 Test Management Process

The fundamental testing process consists of the following main activities:

- Test planning and control
- Test analysis and -design
- Test implementation and -execution
- Test evaluation and report
- Test final

Even if they are listed logically sequentially, all these test process activities in the project can and will take place overlapping in time or coordinated in the individual sub-teams and test levels. Relevant results that affect each other between the test levels are communicated in the daily status meeting and discussed between the teams.

These test activities are briefly described below. The description applies to all test levels.

Test planning and control

For each planned test level, the test objectives and all necessary test activities are identified and defined, which are necessary in order to be able to achieve the task scope and the agreed test objectives. All identified and defined activities are planned and assigned to those responsible. The test is controlled by the test manager. The test progress is tracked daily and compared to the planned value. Any deviations found are recorded and reported. This may include taking corrective action. If necessary, the test planning is adapted in coordination with the project management and relevant project participants.

Test analysis and -design

Based on the technically and professionally specified and coordinated requirements, the test cases and test scenarios to be tested and their test conditions are derived using various test methods and procedures, prioritized accordingly and subjected to an internal review.

The required test data as well as the required test infrastructures such as test environment, hardware, software and possibly technical devices or tools are identified, and their procurement conceptualized.

Test implementation and - execution

All preparatory activities conceived in the analysis and design phase, which are necessary for carrying out the test, are realized and implemented in this phase. The test environment is configured and used accordingly. Tools, possibly any test tools and test scripts for data generation or automation are created and configured to work.

After the prerequisites for the test have been completed, the test cases are carried out and documented according to the planning. Detected deviations are recorded and their corrections are monitored. In addition to error tests, regression tests are also carried out to ensure that troubleshooting or software changes should not have a negative impact on existing functionality, or that no further (previously masked) error states have been exposed.

Test progress and troubleshooting progress are reported daily.

The following information includes can be found in the test progress report:

- Test cases executed,
- The composition of open defects by severity.

Test evaluation and -report - completion

With the rating of the test evaluation, the test activities are examined and evaluated for their goals based on the documented test results.

A final test report is created in which the test results such as test case and error statistics are displayed. Open test cases and open errors as well as identified abnormalities are listed. A risk assessment takes place.

4.2 Results from the test process

The results from the test process are documented and filed.

The following results from the test process are provided:

- The comprehensive high-level test plan (HLTP)
- Test specification (test cases) for the test levels
- The test documentation of the test executions
- The final test report is structured in following chapters
  - Integration
  - Operational Acceptance

4.3 Delivery procedure for the test item

The software of DGCG is developed in the development environment and unit tested. In coordination with the test, operation management will deploy the image into the test environment ACC and later, after successful testing and release, into the production environment PROD.

Analog to DGCG the delivery of DGCA will be developed and unit tested in the development environment. After completing the unit test and in coordination with the test manager it will be deployed into a local test environment INT for the Integration Test, Penetration Test and Operational Test.

Different to DGCG it is not planned to deploy the DGCA software into the production environment PROD or any other environment at Digit.

4.4 Test environment

The requirements for the test environments depend on the test level and their individual test planning.

The following graphic and table gives a general overview of the currently planned environments and their usage in the different test stages:

DGCG-Test level	D*EV on OTC*	TST at Digit	ACC at Digit	PROD at Digit
Unit Test	X			
Integration Test		X	X	
Penetration Test			X	
Operational Test			X	

Table 5 Environments for the tests of DGCG

DGCA-Test level	D*EV on OTC*	INT on OTC
Unit Test	X	
Integration Test		X
Penetration Test		X
Operational Test		X

Table 6: Environments for the tests of DGCA

4.4.1 Monitoring of Test Environments

The test process does not implement a monitoring or logging solution but will use the monitoring methods which are provided for operation.

4.5 Simulators and Test Automation

National Backend Simulators (mock-ups) may be installed on virtual environments to complete the testing environment.

These simulators will make API requests to the DGCG both according to the specification and violating the specification (negative tests).

For those simulators, it is necessary to install a mocked-up trust chain on the testing environment: A CA certificate that is under the control of the test team will allow to sign and manipulate issuer certificates both for integration test and penetration test.

4.6 Test data

The test data required in the respective test levels is created manually in the test environment. The responsibility for the test data generation is directly assigned to the person responsible for the test level.

5 Metrics and bug reporting

5.1 Bug management and bug reports

All deviations found during the test phase are always set and processed as a "defect issue" in GitHub. A simple workflow (KANBAN-Board) for defect issues will visualize the actual status. Due to that, the communication and the exchange between the Development and Test will be efficient during projects.

In order to carry out effective defect tracking and cause analysis, the following information should be recorded for each error:

- Defect ID: unique ID of the defect ticket (automatically assigned by the tool as a suffix to the title of the error issue).
- Error title (mandatory): title of the defect issue in the structure "P_<Severity>_<Component >_<title's description> described as followed:
  - P_<Severity>: evaluated categorized damage class (High, Medium or Low) by the error.
  - Component: the system or the component in which the variance was found
  - Titel's description: short summary, meaningful description of the deviation.

- Error Description (mandatory): Sufficient description of the deviation including the process until the error occurs (for the reproducibility of the error). It contains the additionally following datas:
  - Test level (mandatory): test level in which the variance was found (Integration Test, Operational Acceptance Test, Performance Test).
  - Test Case Reference (optional): Link to the specific test case with which the variance was found.
- Comment (optional): chronological comments/remarks of the agent assigned to the ticket.

5.1.1 Life cycle for error handling

The life cycle for T-Systems internal defect handling is to be applied as a standard process for defect issues in a Kanban Board.

The following workflow is planned for defect handling:

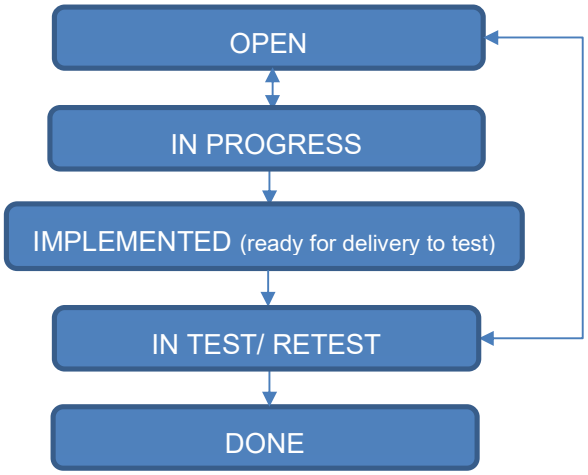

Figure 3: Workflow of an error issue

5.1.2 Definition of defect classifications

The following error classifications (severity) are currently valid:

Defect class High:

The appropriate or economically reasonable use of the overall system in accordance with the specification is not possible or is restricted by non-function or malfunction(s). The day-to-day business cannot reasonably be continued (exemplary defect: the public key packages could not be up- or downloaded).

Defect class Medium:

The appropriate or economically reasonable use of the overall system is considerably restricted or hindered according to the specification.

The core functionality is guaranteed, but there is a serious error in an essential sub-functionality which considerably hinders or makes work unreasonable.

Defect class Low:

According to the specification, the appropriate or economically reasonable use of the overall system is only insignificant, i.e. it is restricted or prevented without any significant effect on the functionality of the overall system.

5.1.3 Tool use for defect tracking

All defects are tracked in GitHub.

If deviations from the specified requirements are detected during the test phases of the respective test level, the defect finder is obliged to set the deviation as a defect issue promptly and completely in GitHub according to workflow defined in KANBAN Board (status "OPEN"). This issue is such titled with prefix in a structure, so that everyone can immediately note it as a defect ticket.

- A developer of the agile developing team checks the ticket and analyzes the defect. He clarifies open topics with the defect finder in case of need and starts to fix the defect (status "IN PROGRESS").
- As soon as the ticket is resolved, successfully unit tested and ready for delivery to test, it will be shifted to status "IMPLEMENTED“.
- As soon as the image with the fix is delivered to the test environment, the ticket is shifted to status "IN TEST/RETEST " and assigned to the test team or the defect finder for retesting.
- After successful retesting, the ticket is documented accordingly and will be shifted to status “DONE” by the tester. If the defect is still present, the ticket should be returned to development for renewed processing by shifting it back to status "OPEN").
- If an issue is not found to be an error during analysis, it is set to "In TEST/RETEST" and resubmitted to the ticket creator for checking and closing.

5.1.4 Test status report

The test status report consists of three parts

- Overview and current risks.
- Test case status shows the current execution status of the planned test cases
- Error report contains the current status of the errors grouped by their respective severity or the affected system components.

6 Management of test cases

The test cases for the test levels

- Integration Test and,
- Operational Test

are created in JIRA-XRAY. This tool is used as a test management tool for test case administration and reporting.

Lifecycle model for the test execution of a test case

The lifecycle model for the test execution of a test case is shown in Figure 13 and the corresponding status values are described in Table 4 4.

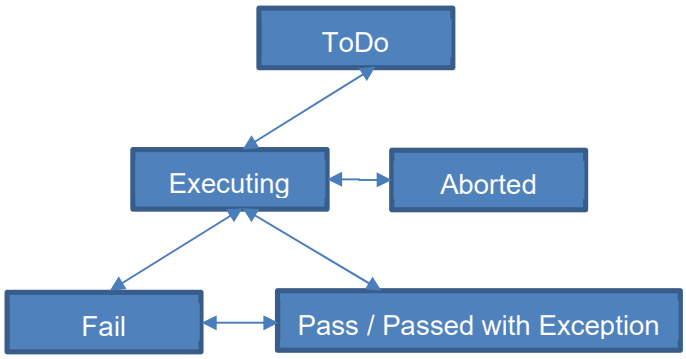

Figure 4 Lifecycle test execution status

Status	description
ToDo	The test execution is planned, but so far, no test execution has been done.
Pass	The test case was executed successfully (also under consideration of the above-mentioned conditions with successful in terms of the test idea).
Passed with Exception	The test case was successfully executed but a Prio Low error was found during execution, but this does not restrict the required functionality.
Fail	The test execution failed; errors have occurred.
Executing	The test execution is in progress and has not yet been completed. There are test steps that are still open or planned.
Aborted	The following semantic cases are represented by the status "Aborted":

Status	description
	- TF " Shifted": The test has been removed from the test program because of missing test prerequisites (e.g. if functionality has not been implemented despite assignment to a release, or has not been released due to lack of quality, or if the error correction for a blocking error is no longer implemented in the current release) and must be rescheduled in a subsequent release or test cycle. - TF "not applicable": The test case was planned. However, the test case is not relevant (e.g. because changes such as CRs have occurred in the meantime or the test was based on incorrect prerequisites).

Table 6.1 Test execution status

7 Used tools

The following table lists the test tools used in this overall service for testing.

Tool	Application and explanation
GitHub	Defect management tool
Jira/XRAY	Test management Tool for Interface, Operational and Integration Tests

Table 7.1 Used tools