**Test**

**Discipline Document**

**I-3 Analysis and Software Design**

**Project: HHS Term Planning**

**Date: 19-03-2014**

**Owner: Gideon Kuijpers, Gary Haime**

**Version: 0.1**

**Document History**

**Revision History**

|  |  |  |
| --- | --- | --- |
| **Version** | **Revision date** | **Summary of changes** |
| 0.1 | 19-03-2014 | Introduction of the test discipline is added. |
| 0.2 | 20-03-2014 | TRA is added. |
| 0.3 | 25-03-2014 | Master test plan is added. |
| 0.4 | 02-04-2014 | Detail Test plan is added. |

**Approvals**

This document requires the following approvals.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Name** | **Function** | **Date of approval** | **Signature** |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Distribution**

This document is intended for:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Function** | **Date of issue** | **Version** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1. Introduction 5](#_Toc384844472)

[2. Test Risk Analysis 6](#_Toc384844473)

[2.1 Identifying the stakeholders and the kick-off 6](#_Toc384844474)

[2.1.1 Stakeholders 6](#_Toc384844475)

[2.1.2 Kick-off 6](#_Toc384844476)

[2.2. Determine functions and areas of importance 6](#_Toc384844477)

[2.3 Determine relative importance 7](#_Toc384844478)

[2.4 Process Data 9](#_Toc384844479)

[3. Master Test Plan 10](#_Toc384844480)

[3.1 Task Description 10](#_Toc384844481)

[3.1.1 Test project name 10](#_Toc384844482)

[3.1.2 Project summary 10](#_Toc384844483)

[3.1.3 Anticipated Results 10](#_Toc384844484)

[3.1.4 Customer 10](#_Toc384844485)

[3.1.5 Accepter 10](#_Toc384844486)

[3.1.6 Scope 11](#_Toc384844487)

[3.1.7 Test level 12](#_Toc384844488)

[3.1.8 Expected end date / available budget 12](#_Toc384844489)

[3.1.9 Task description 12](#_Toc384844490)

[3.1.10 Agreements on reporting 12](#_Toc384844491)

[3.2 Test base 13](#_Toc384844492)

[3.2.1 Introduction 13](#_Toc384844493)

[3.2.2 Documents 13](#_Toc384844494)

[3.2.3 Application 14](#_Toc384844495)

[3.2.4 Test ware 14](#_Toc384844496)

[3.3 Test Strategy 14](#_Toc384844497)

[3.3.1 Test approach 14](#_Toc384844498)

[3.3.2 Risk analysis 14](#_Toc384844499)

[3.3.3 Quality attributes 16](#_Toc384844500)

[3.3.4 Strategy matrix 17](#_Toc384844501)

[3.3.5 Test environment 18](#_Toc384844502)

[3.3.6 Assuring the quality of the test project 19](#_Toc384844503)

[3.3.7 Release advice 19](#_Toc384844504)

[3.3.8 Change and Error management 19](#_Toc384844505)

[3.4 Planning 20](#_Toc384844506)

[4. Detailed test plan for system test 21](#_Toc384844507)

[4.1 Task Description 21](#_Toc384844508)

[4.1.1 Test project name 21](#_Toc384844509)

[4.1.2 Anticipated Goal 21](#_Toc384844510)

[4.1.3 Customer 21](#_Toc384844511)

[4.1.4 Accepter 21](#_Toc384844512)

[4.1.5 Scope 21](#_Toc384844513)

[4.1.6 Test level 22](#_Toc384844514)

[4.1.7 Expected end date / available budget 22](#_Toc384844515)

[4.1.8 Task description 22](#_Toc384844516)

[4.1.9 Agreements on reporting 22](#_Toc384844517)

[4.2 Test Base 22](#_Toc384844518)

[4.2.1 Introduction 22](#_Toc384844519)

[4.2.2 Documents 22](#_Toc384844520)

[4.2.3 Application 22](#_Toc384844521)

[4.2.4 Test ware 22](#_Toc384844522)

[4.3 Test Strategy 23](#_Toc384844523)

[4.3.1 Test approach 23](#_Toc384844524)

[4.3.2 Risk analyses 23](#_Toc384844525)

[4.3.3 Quality attributes 24](#_Toc384844526)

[4.3.4 Strategy matrix 24](#_Toc384844527)

[4.3.5 Test techniques matrix 26](#_Toc384844528)

[4.3.6 Test environment 28](#_Toc384844529)

[4.3.7 Assuring the quality of the test project 28](#_Toc384844530)

[4.3.8 Release advice 28](#_Toc384844531)

[4.3.9 Change and Error management 28](#_Toc384844532)

[4.4 Planning 29](#_Toc384844533)

[4.5 Sanity check report 29](#_Toc384844534)

[5. Test design 31](#_Toc384844535)

1. Introduction

The purpose of this document is to describe the test approach and test methods chosen for this project. This document consists of five different components:

* The risk analysis: an overview of the components contained within the system and an analysis of their risk-sensitivity.
* The master test plan: outlines the test approach and gives a global view on how the tests will be enacted.
* The detailed test plan: deepens the master test plan. It gives a more complete description of how the system will be tested, which tests will be used and how they will be executed.
* The sanity check report: contains the data used as the test base.
* Test design: contains all high priority test designs.

2. Test Risk Analysis

2.1 Identifying the stakeholders and the kick-off

2.1.1 Stakeholders

The client expects a term planning application that is less time consuming and more efficient that the currently used system. For more information, check the PID and Requirements Discipline document.

By performing a 1-D test risk analysis, the important areas of the system are identified, after which the test depth and the order of performing tests are determined.

The project’s stakeholders:

* Jos van Aalten (general manager Academy for IT & Media)
* Erik van Dordrecht (team manager at the Academy for IT & Media)
* Josine (course manager at the Academy for IT & Media)
* Thea (teacher at the Academy for IT & Media)

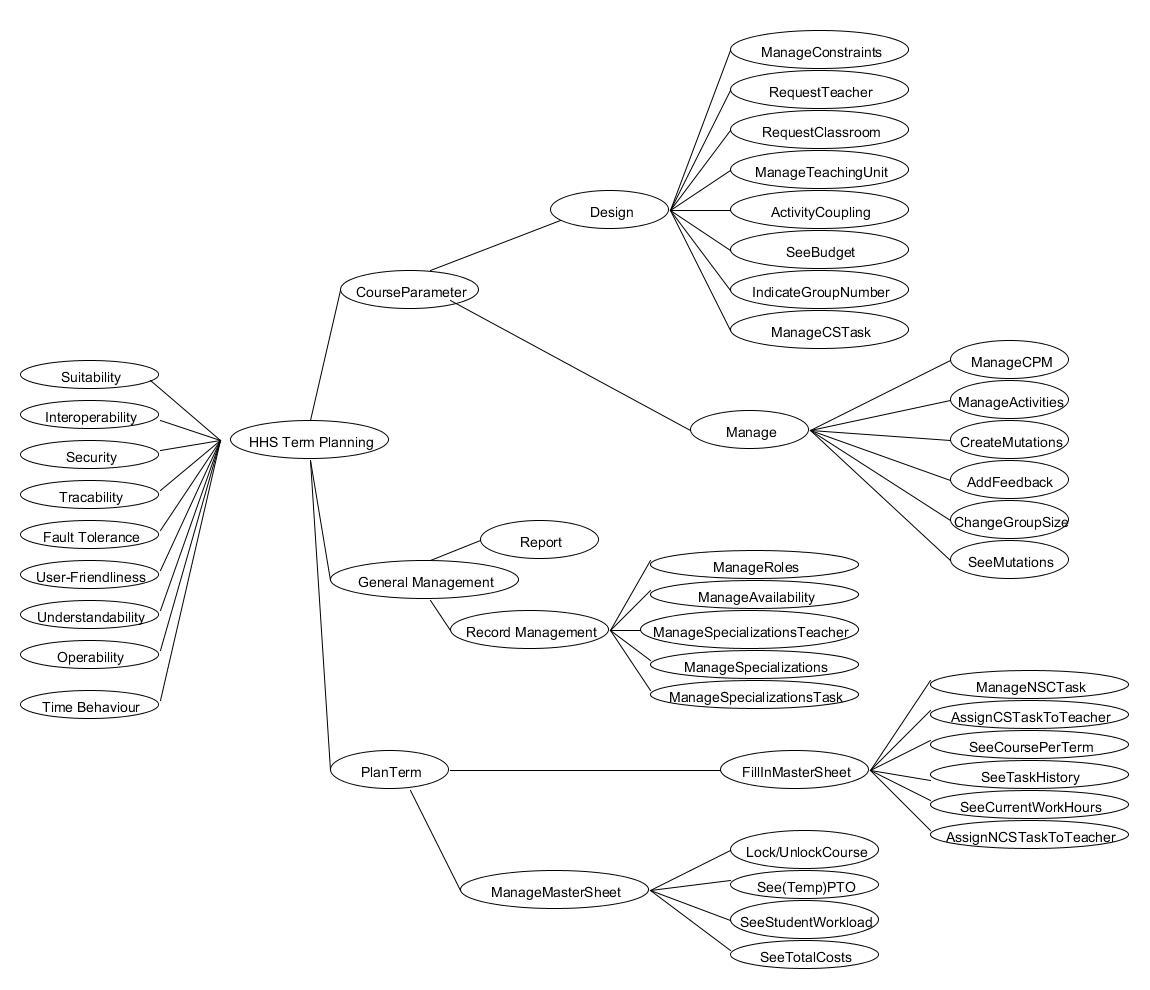
2.1.2 Kick-off

To determine the test depth and the order of performing the tests a test risk analysis (TRA) is needed. Usually the stakeholders would be invited for a TRA session, but this project is organised in such a way that would make this impossible. The information needed to perform a TRA session was given too late. We only had one day to prepare for the TRA. Therefore the following input is used instead of the output of the TRA:

* Prioritized requirements as described in the Requirements Discipline.
* Use case specifications as described in the Requirements Discipline.
* Processes described in the Business Modelling Discipline.

2.2. Determine functions and areas of importance

To determine the risk areas of the system and its functions a test tree has been made. The left branch of the test tree describes the most important quality attributes of the system, in accordance with the ISO 9126 norms. The right side describes the functions of the system grouped by subsystem.



2.3 Determine relative importance

In this chapter the importance of each risk area from the test tree will be prioritized by the test manager and test coordinator. Both will divide 110 points among the risk areas to prioritize them.

The following point system is used:

* 9 points: crucial, the system cannot function without these elements
* 5 points: important, the system can still function if a workaround or quick fix is available
* 3 points: non crucial, the system can still function without these elements but testing is preferred
* 1 point: non important, the system will fully work without these elements

|  |  |  |
| --- | --- | --- |
| **Risk area** | **Test Coordinator** | **Test Manager** |
| CourseParameter -> Design -> ManageConstraints | 5 | 3 |
| CourseParameter -> Design -> RequestTeacher | 3 | 1 |
| CourseParameter -> Design -> RequestClassroom | 3 | 1 |
| CourseParameter -> Design -> ManageTeachingUnit | 3 | 3 |
| CourseParameter -> Design -> ActivityCoupling | 1 | 1 |
| CourseParameter -> Design -> SeeBudget | 1 | 3 |
| CourseParameter -> Design -> ManageCSTask | 5 | 5 |
| CourseParameter -> Design -> IndicateGroupNumber | 1 | 3 |
| CourseParameter -> Manage -> ManageCPM | 9 | 9 |
| CourseParameter -> Manage -> ManageActivities | 5 | 5 |
| CourseParameter -> Manage -> CreateMutations | 3 | 5 |
| CourseParameter -> Manage -> AddFeedback | 1 | 3 |
| CourseParameter -> Manage -> ChangeGroupSize | 3 | 3 |
| CourseParameter -> Manage -> SeeMutations | 1 | 5 |
| General Management -> Record Management -> ManageSpecilizationsTeacher | 5 | 3 |
| General Management -> Record Management -> ManageSpecilizations | 1 | 1 |
| General Management -> Record Management -> ManageSpecilizationsTask | 5 | 3 |
| General Management -> Record Management -> ManageAvailability | 3 | 5 |
| General Management -> Record Management -> ManageRoles | 5 | 5 |
| PlanTerm -> FillInMasterSheet -> ManageNCSTask | 3 | 5 |
| PlanTerm -> FillInMasterSheet -> AssignCSTaskToTeacher | 9 | 9 |
| PlanTerm -> FillInMasterSheet -> SeeCoursePerTerm | 3 | 3 |
| PlanTerm -> FillInMasterSheet -> AssignNCPTaskToTeacher | 9 | 9 |
| PlanTerm -> FillInMasterSheet ->  SeeCurrentWorkHours | 5 | 3 |
| PlanTerm -> FillInMasterSheet -> SeeTaskHistory | 3 | 5 |
| PlanTerm -> ManageMasterSheet -> See(temp)PTO | 5 | 5 |
| PlanTerm -> ManageMasterSheet -> Lock/UnlockCourse | 3 | 3 |
| PlanTerm -> ManageMasterSheet -> SeeStudentWorkload | 3 | 1 |
| PlanTerm -> ManageMasterSheet -> SeeTotalCosts | 3 | 1 |

2.4 Process Data

The risk areas of the system are categorized by the points given in the risk area analysis. These categories are as follows:

* Critical, the 10% extremely important risk fields
* High, the 20% more important risk fields
* Medium, the 30% less important risk fields
* Low, the 40% least important risk fields

The calculation is specified in the Test Risk Analysis of the MTP and in their respective detailed test plans.

3. Master Test Plan

3.1 Task Description

3.1.1 Test project name

Term planning Test project (TP2).

3.1.2 Project summary

The objective of the master test plan (MTP) is to give structure to the test process and make sure everyone involved in the test process is on the same page.

The master test plan describes the test approach, activities and (end) products that will require further elaboration in the detailed test plans.

3.1.3 Anticipated Results

The client expects that the currently used system to design courses and plan terms is replaced by a new system that meets the following criteria:

* Increased efficiency of data processing.
* Reduction in amount of errors made by users.
* Friendlier user interface.
* Data input by the users is semi-automatic.

By meeting the criteria mentioned above, the users of the new system should be able to plan terms in less time than before *and* with less errors due to manual input.

The test process will guarantee that the system meets the system specifications, as specified within the requirements, by giving insight about approach, activities and products to be delivered.

The Master Test Plan delivers the activities needed to ascertain these specifications are met. This concludes in the ultimate goal; Assuring the quality of the product.

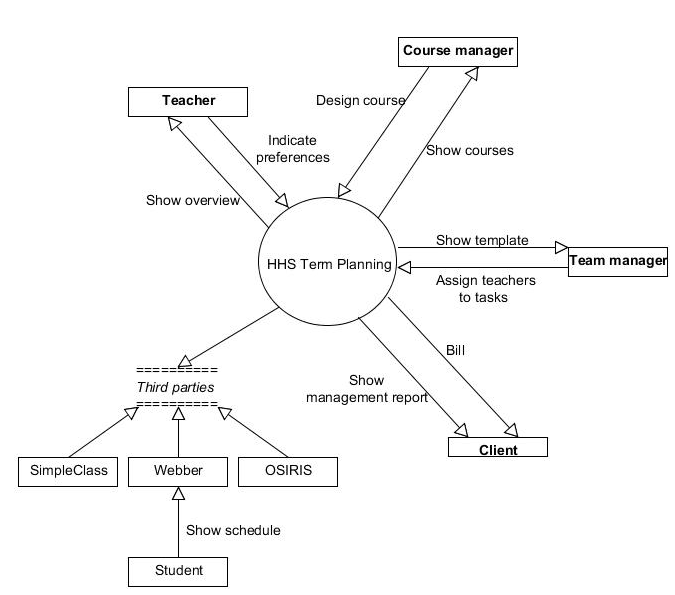
3.1.4 Customer

The customer of this project is the client, Jos van Aalten.

3.1.5 Accepter

The acceptors of this project are Gideon Kuijpers (test manager) and Gary Haime (test coordinator).

3.1.6 Scope



The system's main aim is to plan courses within specified terms. When which course will run is specified in the year plans of the academy for IT & Media.

The process can be divided in five processes: create a year plan, design a course, plan a term, create a schedule and realise a course, as described in the business modelling discipline.

The system focus lays with two of these processes: design a course plan and plan a term. These two will be fully evaluated for testing.

Creating a year plan falls fully outside of scope. The scheduling process is done externally but the communication needed, will be tested.

Realising a course results in adding feedback which is done in our system and will therefore be also evaluated for testing.

3.1.7 Test level

The test levels that are used within the project are Module test, Module-integration test, System test, chain test and acceptance tests concerning the user and functionality. The mentioned test are used because of the following reasons:

* Module/module-integration test must ensure that the modules to design a course plan and to plan a term work according to technical design and work together correctly.
* System test will ensure that the HHS Term Planning system works as functional designed.
* Chain test must ensure that the HHS Term Planning system works together with the scheduling department.
* Acceptance test must be performed to ensure that the system meets the specifications of the requirements and is in accordance with the wishes and demands of the client.

3.1.8 Expected end date / available budget

The deadline is 10-04-2014. By then the documentation is to be delivered to the client of this project and the tests must be ready to be executed. The available budget(in hours) for test-planning, divided over a three week period is approximately 60 hours.

These 60 hours are taken with the multiple disciplines of team members in consideration.

After test planning, the coding team needs to design tests. Designing the several tests is a time-consuming activity which by could take up to 200 hours. These hours are taken based on assumption and should be evaluated between the coding team and the client.

Advice for a realisation-budget of tests is 250 hours, which should be executed before the construction phase end-date. Reaching a total test budget of 510 hours.

3.1.9 Task description

The client has an application, which is responsible for planning, assigning and scheduling the tasks of the teachers of the Academy for IT & Media. The whole process takes about 900 hours. The client thinks this could be done more efficiently. He wants it to be optimized. The new application should be able to do the process within 780 hours.

According to customer the project can be called a success when the following expectations are acquired:

* The client can perform the process of planning assigning, scheduling and evaluating terms in 780 hours per term.
* The client can generate management reports, which contain information about deploying of teachers and data on the available expertise and expertise needed in the future.
* The above-mentioned process contains less faults.

For the MTP to be called a success:

* The MTP provides sufficient information to form detailed test plans.
* MTP contains guidelines on all tests to be executed.

3.1.10 Agreements on reporting

The project lasts eight weeks. At the end of each iteration in the elaboration phase, which is every two weeks, reports per different test level is to be delivered to the client. The deliverance to the client will be done per e-mail or will be placed on the file exchange forum. The update contains information on the following data:

* results of the tests per risk category that are performed successful.
* results of the tests per risk category that are delayed or are a failure.
* Description of a test: this includes a date, costs and responsible tester.
* Quality of the test: insight is given by looking at the results of the tests per risk

risk category.

* Release advice

To ensure the client’s satisfaction, delivering data on tests is a requisite. This serves as a measurement for the project’s progress. This kind of data includes the functionality.

3.2 Test base

3.2.1 Introduction

The test base describes the documents that are needed as a base for testing the test object and gives a description of on what it consist of. The documents for testing are used on different test levels. The following documents are used on the following test levels:

* Module/Module integration test: Design discipline
* System test: Analysis discipline
* FAT: Requirements & Analysis discipline
* UAT: Requirements discipline
* Chain test: Design Discipline

The various tests use the following elements to form test objects:

* GUI: the interface to access the functionalities of the new system.
* Business logic: All the core functionality within the new system.
* Data logic: All functionality concerning the database.

Together these elements form the HHS Term Planning application, which are used for testing.

3.2.2 Documents

|  |  |  |  |
| --- | --- | --- | --- |
| **Document name** | **version** | **date** | **status/remarks** |
| Requirements discipline | 1.3 | 28-03-2014 | UAT, final  FAT, final |
| Business Modelling discipline | 1.0 | 14-03-2014 | Final (as reference during any test) |
| Analysis discipline | 0.4 | 25-03-2014 | System test (class diagram, Sequence diagrams)  FAT(final) |
| Design discipline | 0.1 | 28-03-2014 | Module/integration (final)  Chain (final) |

3.2.3 Application

|  |  |  |  |
| --- | --- | --- | --- |
| **Application name** | **version** | **date** | **status/remarks** |
| HHS Term Planning |  |  | system to be build |
| Term Planning mail plug-in Scheduling Department |  |  | During and after construction of code that handles scheduling department connection. |
| SQL Database | 1.0 | t.b.d. |  |

3.2.4 Test ware

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ware** | **version** | **date** | **status/remarks** |
| Detailed Test plan system test | 1.0 |  |  |
| Master Test plan | 1.0 |  |  |

### 

3.3 Test Strategy

3.3.1 Test approach

The time to perform testing on the system is limited; not everything can be tested because of this reason. So choices have to be made. There has been striven to divide the test capacity as effective and efficiently as possible about the whole test section. This is further described in the remaining test strategy.

The test strategy is risk-based. This means that the new system is built well enough to guarantee that there will be no major risks for the Hague University. Areas of the system that are categorized as major risks will be tested thoroughly.

The first step of determining the test strategy was the above 1-D risk analysis. Afterwards the test strategy is determined on the outcome of this analysis. The strategy describes what and how there will be tested by the team and is focused on finding major risks at an earlier stage of the project so it will cost the client less money.

3.3.2 Risk analysis

On 20-03-2014 the following people carried out a risk analysis:

* Gideon Kuijpers, Test manager
* Gary Haime, Test coordinator

The following risks and their significance are acknowledged:

|  |  |  |
| --- | --- | --- |
| **Risk category** | Risk area | Relative importance |
| **Critical** | CourseParameter-Manage-ManageCPM | 18 |
|  | PlanTerm-FillInMasterSheet-AssignCSTaskToTeacher | 18 |
|  | PlanTerm-FillInMasterSheet-AssignNCPTaskToTeacher | 18 |
| **High** | CourseParameter-Manage-ManageActivities | 10 |
|  | CourseParameter-Design-ManageCSTask | 10 |
|  | General Management-Record Management-ManageRoles | 10 |
|  | PlanTerm-ManageMasterSheet-See(temp)PTO | 10 |
| **Medium** | CourseParameter-Design-ManageConstraints | 8 |
|  | General Management-Record Management-ManageSpecilizationsTeacher | 8 |
|  | General Management-Record Management-ManageAvailability | 8 |
|  | PlanTerm-FillInMasterSheet-ManageNCSTask | 8 |
|  | PlanTerm-FillInMasterSheet-SeeCurrentWorkHours | 8 |
|  | PlanTerm-FillInMasterSheet-SeeTaskHistory | 8 |
|  | General Management-Record Management-ManageSpecilizationsTask | 8 |
|  | CourseParameter-Design-ManageTeachingUnit | 6 |
|  | CourseParameter-Manage-CreateMutations | 6 |
|  | CourseParameter-Manage-ChangeGroupSize | 6 |
|  | CourseParameter-Manage-SeeMutations | 6 |
|  | PlanTerm-FillInMasterSheet-SeeCoursePerTerm | 6 |
| **Low** | CourseParameter-Design-RequestClassroom | 4 |
|  | CourseParameter-Design-SeeBudget | 4 |
|  | CourseParameter-Design-IndicateGroupNumber | 4 |
|  | CourseParameter-Manage-AddFeedback | 4 |
|  | PlanTerm-ManageMasterSheet-Lock/UnlockCourse | 4 |
|  | PlanTerm-ManageMasterSheet-SeeStudentWorkload | 4 |
|  | PlanTerm-ManageMasterSheet-SeeTotalCosts | 4 |
|  | General Management-Record Management-ManageSpecilizations | 2 |
|  | CourseParameter-Design-ActivityCoupling | 2 |

3.3.3 Quality attributes

|  |  |
| --- | --- |
| **Quality attributes** | **Significance**  **H M L N** |
| **Functionality**  Suitability  Accuracy  Interoperability  Compliance  Security  Traceability | * + x M x x   + x x x N   + x x L x   + x x x N   + H x x x   + x M x x |
| **Reliability**  Maturity  Fault tolerance  Recoverability  Availability  Degradability | * + x x x N   + x M x x   + x x x N   + x x x N   + x x x N |
| **Usability**  Understandability  Learnability  Operability  Explicitness  Customisability  Attractivity  Helpfulness  User-friendliness | * + x M x x   + x x x N   + x M x x   + x x x N   + x x x N   + x x x N   + x x x N   + H x x x |
| **Efficiency**  Time behaviour  Resource behaviour | * + x x L x   + x x x N |
| **Maintainability**  Analysability  Changeability  Stability  Testability  Manageability  Reusability | * + x x x N   + x x x N   + x x x N   + x x x N   + x x x N   + x x x N |
| **Transferability**  Adaptability  Installability  Conformance  Replaceability | * + x x x N   + x x x N   + x x x N   + x x x N |

3.3.4 Strategy matrix

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk area** | **User-Friendliness** | **Fault Tolerance** | **Security** |
| *From “Quality attributes” table* | H | H | M |
| Module test/integration test | - | H | - |
| System test | - | M | M |
| Functional acceptance test | - | - | M |
| User acceptance test | H | - | - |
| Chain test | - | M | - |

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk area** | **Suitability** | **Operability** | **Interoperability** |
| *From “Quality attributes” table* | M | M | M |
| Module test/integration test | - | L | M |
| System test | M | - | - |
| Functional acceptance test | H | - | - |
| User acceptance test | - | H | - |
| Chain test | - | - | H |

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk area** | **Traceability** | **Understandability** | **Time Behaviour** |
| *From “Quality attributes” table* | M | M | L |
| Module test/integration test | - | - | - |
| System test | M | L | - |
| Functional acceptance test | H | - | L |
| User acceptance test | - | H | - |
| Chain test | - | - | - |

H=high, M=medium, L=low, N= not relevant

The subcategories of the quality attributes, maintainability and transferability are not taken into account as the stakeholders did not mention any of the points belonging to this two points. As the previous sections show, the most important aspects are that the key functionalities work without flaws and that the new system is usable. As such user-friendliness, understandability and operability get a high priority in the user acceptance test and fault tolerance gets a high priority in the module/integration and medium priority in the system and chain test to discover faults at an early stage to reduce the costs. Security is to gets a medium priority because other people than the stakeholders should not be able to use the system. Also the speed of the system is to be tested. Though this has a low priority due the fact speed is not an issue nowadays.

3.3.5 Test environment

The Hague University employs several windows operated pc’s. These pc’s are connected to the internet and have no specific intranet. These pc’s are connected to a main server, all software desktops are launched through these servers through a login system.

Teachers and students all have individual logins, with corresponding roles attached to them. Their respective desktops have restraints to downloading and installing software.

The test environment for the new system should be similar to the environment the Hague University uses as described above.

Furthermore the environment should be able to adapt to the different kinds of tests that will be performed. The following section describes which tools are needed for which kind of test:

* Module /integration test : The environment for this test should at least contain a database, Unit test-tool, stubs and drivers to simulate missing modules needed for the test, test data, data registration tool and version control tool.
* System test: the environment for this test should contain a database, write-only and login rights to the used database, test data, stubs and drivers, test tool, and a data registration tool.
* Functional acceptation test: the environment should contain a database, a test tool , a data registration tool and test data.
* User acceptance test; the environment should contain a simulator the simulate the new system, the data registration tool and the test data.

To maintain the adaptable environments the test team made agreements about the configuration, release and back-ups of a test object. The configuration item of the different test objects must have a version number and history. Before a test object can be released it must have at least a conditional release advice and there are no major concerns described in the release notes. Also, before a new version can be uploaded to the test environment, a back-up has to be made.

3.3.6 Assuring the quality of the test project

The quality of the test project shall be assured in the following ways:

* Intakes: In order to ascertain the quality of the test bases a sanity check shall be performed. The results of this check shall either show that the test bases have enough quality to be used as test bases or whether they should be modified and, if so, in what way.
* Review of the test designs: by reviewing the test design documents, the stakeholders can be given insight about the test procedure. By doing this the team can make sure the stakeholders are aware of the choices made and if needed will intervene. This will also make sure that any forgotten risks will be brought forward if the stakeholders miss them.
* Reviewing test ware: once a certain type of test has been done certain parts can be reused for other tests. These are called the test ware and are used to increase efficiency. However in order to maintain an effective way of testing the test ware needs to be reviewed if it is indeed applicable for the next test(s). The development team can advice if this is necessary.

3.3.7 Release advice

In the project there are three possible release advices. A release advice can be positive, conditional or negative. These can be characterized as followed:

* positive: is given when the functions, defined as a high or middle risk contain no major risks and ones with a low risk are at least functional.
* conditional: is given when there are still bugs in some functions of the tested area. but they do not possess a threat to the system. This bugs can be fixed a next time.
* negative: is given when multiple function, which were tested, contain majors issues. In this case the test team mentions to the developers which parts were tested negative.

A release advice is given at the end of a test cluster by the test manager, which the name indicates, is merely an advice. This advice will be based on the product status in proportion to the envisioned result. Relevant factors in this comparison are remaining product risks, the yet to be executed tests and the gap between the striving quality and the realised quality.

Afterwards the test manager comes with recommendations to get a positive advice. The final decision of whether a function should be implemented will be made by the client.

3.3.8 Change and Error management

To guarantee the quality of the system adaptation and how changes in the sanity check are handled. This also applies to findings and new releases.

Adaptations in the sanity check are handled by a meeting between the responsible parties.

Minor changes can be made directly in the test design. For changes of bigger impact, a change request has to be submitted. The project manager will then weigh the pros and cons and come to his decision. This will be done by the hand of the reason, benefits, details and costs.

To handle findings the tester that finds, one is asked to register the findings in the findings registration tool. This registration contains the following data:

* identifier
* date
* name of tester
* status
* risk
* release version
* summary
* description

After registration, the test coordinator within the team in India checks if the above-mentioned process has been completed correctly. Then a review is hold to discuss the findings, which are open, new or solved, and the team comes up with a solution for certain findings. Afterwards a developer is to program the solution and the tester that did the finding must test if the solution is successful.

After the findings and changes/errors are processed, releases are planned through a dialog between the test coordinator and the project manager. In this dialog the test coordinator of the team gives an indication of when the test object should be released. The project manager gives a time indication of how long it takes to bring out the release. Afterwards the release is reviewed with the stakeholder.

3.4 Planning

Throughout the Inception and Elaboration phases, documentation and designs are tested by the client and the stakeholders to ensure all documents are sufficient.

During the construction phase, the code is divided into six packages. At the end of construction of such a package, the coding team has 4 hours at the end or construction to perform the module test and another 4 hours to perform integration test. As results of these test can be negative or inconclusive, these tests may have to be redone after code has been edited.

The module tests may be done three times and the integration tests five times resulting in a budget of (4x3 + 4\*5) 32 hours. The project group advices to save time for after the system test as coding may need to be edited after the system test.

System testing is performed when the code in alpha phase and is expected to take 20 hours. After the system test, the functional acceptance test can be performed. As much of the functional acceptance testing is performed during the system test, this should take 10 hours total.

The user acceptance test takes 2 hours per iteration, and is performed at the end of the construction phase. This can be performed up to 8 times resulting in a budget of (2x8) 16 hours.

The total budget concludes as (32+20+10+16) 78 hours. And is performed during the construction phase.

4. Detailed test plan for system test

4.1 Task Description

4.1.1 Test project name

TP2 system test

4.1.2 Anticipated Goal

As mentioned in the master plan, the goal for the system test is to ascertain the application works as designed. This test is executed in the construction phase and will be used to test the completed code and to ensure said code is cohesive and functional.

With code fully functional and tested, the results of this test is used in conjunction with the functional acceptance test to ensure the application works as intended by the project team, with approval and understanding from the HHS.

The results of these test help to complete two of the goals from the master test plan:

* Increased efficiency of data processing and reducing errors.
* Data input by the users is semi-automatic.

As the master test accumulated in assuring the quality of the product, we must test the quality of the requirements and design documents. When these requirements are met, we can conclude that the TP2 System test base is of sufficient quality. As a result, the TP2 system test ascertain the quality of the system. This means that the code and functionalities meet the quality expectations.

The goal for TP2 System test can be described as ascertaining this quality, helping the goal for the master test plan.

4.1.3 Customer

The customer is of the project is client, Jos van Aalten.

4.1.4 Accepter

Gideon Kuijper(Test Manager), Gary Haime(Test Coordinator).

4.1.5 Scope

The system will be tested from it’s first alpha phase. The system test wil be based on the requirements as specified in the requirements discipline document.

Code and functionality dealing with external communications are part of chain testing and falls out of scope of system testing. As such, the coupling with the scheduling department will not be tested with the system test. Likewise for acceptance of the year planning.

Feedback functions as described in realising a course from the master plan will be tested.

4.1.6 Test level

System test

4.1.7 Expected end date / available budget

System test should costs 20 hours(times modal Indian programming wages) maximum with the end date in mind. This plan has been conceived on the basis of two test workers(test manager, and test coordinator).

4.1.8 Task description

The system test starts when the code reaches alpha phase. Using the requirements discipline document as a base, all elements will be checked to see if the built application produces the required results. Checked elements can result in a positive result when the requirements are available and functional as described.

If any functionality is missing, the test will be considered to have negative results and the construction should continue to add the missing feature(s).

TP2 system test can be called a success when all requirements and documents have been tested. However, TP2 system test needs to be reused until all results are positive in order. As such, the goals of the test are met when all results are positive.

4.1.9 Agreements on reporting

Reporting will be happening as stated in the description of the Agreements on reporting in the Master test plan. But instead of reporting to the client there will be reported to the test coordinator responsible for the detailed test plan.

4.2 Test Base

4.2.1 Introduction

The system test will be based on the analysis discipline document. The analysis discipline document describes the general workings of the system. These general workings describe all needed functionalities in the term planning application.

Using the analysis as a base, means having all needed functionalities as reference to test the system with. Which is sufficient to clear the system tests’ goal.

4.2.2 Documents

|  |  |  |  |
| --- | --- | --- | --- |
| **Document name** | **version** | **status** | **Remarks** |
| Analysis discipline | 1.0 | 07-04-2014 |  |

4.2.3 Application

|  |  |  |  |
| --- | --- | --- | --- |
| **Application name** | **version** | **date** | **status/remarks** |
| HHS Term Planning | 1.0 | t.b.d |  |
| SQL Database | 1.0 | t.b.d. |  |

4.2.4 Test ware

|  |  |  |  |
| --- | --- | --- | --- |
| **Test ware** | **version** | **status** | **Remarks** |
| Module/integration test | t.b.d | executed | positive results of both |
| Master test plan | 1.0 | final | Used for general guidelines and information. |

4.3 Test Strategy

4.3.1 Test approach

The time to perform testing on the system is limited; not everything can be tested because of this reason. So choices have to be made. Because of the mentioned reason all the functionalities, within the design course, plan a term and realise a course, and quality criteria will be tested whenever they have a critical, high or medium category.

The risk categories for each functionality of quality criteria are based on the results of the test risk analysis. Functionalities and criteria are tested more intensive and with a deeper test-depth when the risk category is higher.

4.3.2 Risk analyses

On 20-03-2014 the risk analyse is performed by:

* Gideon Kuijpers, Test manager
* Gary Hame, Test engineer

The following risks and their significance are acknowledged:

|  |  |  |
| --- | --- | --- |
| **Risk category** | **Risk area** | **Significance** |
| **Critical** | CourseParameter-Manage-ManageCPM | 18 |
|  | PlanTerm-FillInMasterSheet-AssignCSTaskToTeacher | 18 |
|  | PlanTerm-FillInMasterSheet-AssignNCPTaskToTeacher | 18 |
| **High** | CourseParameter-Manage-ManageActivities | 10 |
|  | CourseParameter-Design-ManageCSTask | 10 |
|  | General Management-Record Management-ManageRoles | 10 |
| **Medium** | CourseParameter-Design-ManageConstraints | 8 |
|  | General Management-Record Management-ManageSpecilizationsTeacher | 8 |
|  | General Management-Record Management-ManageAvailability | 8 |
|  | PlanTerm-FillInMasterSheet-ManageNCSTask | 8 |
|  | PlanTerm-FillInMasterSheet-SeeCurrentWorkHours | 8 |
|  | PlanTerm-FillInMasterSheet-SeeTaskHistory | 8 |
|  | CourseParameter-Design-ManageTeachingUni  t | 6 |
|  | CourseParameter-Manage-CreateMutations | 6 |
|  | CourseParameter-Manage-ChangeGroupSize | 6 |
|  | CourseParameter-Manage-SeeMutations | 6 |
|  | General Management-Record Management-ManageSpecilizations | 6 |
|  | General Management-Record Management-ManageSpecilizationsTask | 6 |
|  | PlanTerm-FillInMasterSheet-SeeCoursePerTerm | 6 |

4.3.3 Quality attributes

In the MTP, various attributes have been evaluated out of the ISO 9126. Not all quality attributes are necessary for the DTP. The following have been selected:

* Suitability: H
* Security: H
* Traceability: M
* Fault Tolerance: M
* Understandability: L

4.3.4 Strategy matrix

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Risk area** | **from TRA** | **Security** | **Fault tolerance** | **Traceability** | Suitability | Understandability |
| from MTP |  | H | M | M | M | L |
| CourseParameter-Manage-ManageCPM | C |  | M | L | M | H |
| PlanTerm-FillInMasterSheet-AssignCSTaskToTeacher | C |  | H |  | M | L |
| PlanTerm-FillInMasterSheet-AssignNCPTaskToTeacher | C |  | H |  | M | L |
| CourseParameter-Manage-ManageActivities | H |  | L |  | M |  |
| CourseParameter-Design-ManageCSTask | H |  | L |  | M |  |
| General Management-Record Management-ManageRoles | H | H |  | L |  |  |
| CourseParameter-Design-ManageConstraints | M |  |  |  | M |  |
| General Management-Record Management-ManageSpecilizationsTeacher | M |  | L | L | M |  |
| General Management-Record Management-ManageAvailability | M |  | L |  | M |  |
| PlanTerm-FillInMasterSheet-ManageNCSTask | M |  | L |  | M | L |
| PlanTerm-FillInMasterSheet-SeeCurrentWorkHours | M |  |  | L |  |  |
| PlanTerm-FillInMasterSheet-SeeTaskHistory | M |  |  | M | M |  |
| CourseParameter-Design-ManageTeachingUni  t | M |  | L |  | M |  |
| CourseParameter-Manage-CreateMutations | M |  | L | M | M |  |
| CourseParameter-Manage-ChangeGroupSize | M |  |  |  | M |  |
| CourseParameter-Manage-SeeMutations | M |  | L | M | M |  |
| General Management-Record Management-ManageSpecilizations | M |  | L |  | M |  |
| General Management-Record Management-ManageSpecilizationsTask | M |  | L |  | M |  |
| PlanTerm-FillInMasterSheet-SeeCoursePerTerm | M |  |  | M | M |  |

C=critical, H=high, M=medium, L=low, N= not relevant

4.3.5 Test techniques matrix

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Risk area** | **Security** | **Fault Tolerance** | **Traceability** | **Suitability** | **Understandability** |
| CourseParameter-Manage-ManageCPM |  | Gegevens cyclustest | Exploratory testen | CRUD | Real Life Test |
| PlanTerm-FillInMasterSheet-AssignCSTaskToTeacher |  | Algoritmetest |  | Proces cyclus test | Real Life test |
| PlanTerm-FillInMasterSheet-AssignNCPTaskToTeacher |  | Algoritmetest |  | Proces cyclus test | Real Life test |
| CourseParameter-Manage-ManageActivities |  | CRUD |  | Gegegevens  cyclustest |  |
| CourseParameter-Design-ManageCSTask |  | CRUD |  | Gegegevens  cyclustest |  |
| General Management-Record Management-ManageRoles | C/E Graphing + Syntax |  | Exploratory test |  |  |
| CourseParameter-Design-ManageConstraints |  |  |  | Exploratory test |  |
| General Management-Record Management-ManageSpecilizationsTeacher |  | CRUD | Exploratory test | Gegevens cyclustest |  |
| General Management-Record Management-ManageAvailability |  | Semantic test |  | Exploratory testing |  |
| PlanTerm-FillInMasterSheet-ManageNCSTask |  | CRUD |  | Gegevens cyclustest | Use case test |
| PlanTerm-FillInMasterSheet-SeeCurrentWorkHours |  |  | Exploratory test |  |  |
| PlanTerm-FillInMasterSheet-SeeTaskHistory |  |  | Exploratory test | Real life test |  |
| CourseParameter-Design-ManageTeachingUni  t |  | CRUD |  | Gegevenscyclustest |  |
| CourseParameter-Manage-CreateMutations |  | Semantic test | Exploratory test | Exploratory test |  |
| CourseParameter-Manage-ChangeGroupSize |  |  |  | real life test |  |
| CourseParameter-Manage-SeeMutations |  | Error guessing | Exploratory | Exploratory |  |
| General Management-Record Management-ManageSpecilizations |  | CRUD |  | gegevens cyclustest |  |
| General Management-Record Management-ManageSpecilizationTask |  | CRUD |  | gegevens cyclustest |  |
| PlanTerm-FillInMasterSheet-SeeCoursePerTerm |  |  | exploratory  Testing | Real life test |  |

4.3.6 Test environment

For the preconditions of the test environment, look at the preconditions for the system test, described in the master test plan. To maintain this environment the configuration item record data and status of the test object, described in the scope of this plan, must updated.

Before a release can be shipped a positive release advice is to be given by the test manager and the moment for the release to be shipped must be discussed with the test coordinator. At the moment for shipping a back-up of the new version of the system is to be made and archived.

4.3.7 Assuring the quality of the test project

The assurance of quality of the test trajectory is guaranteed as described in the Master test plan.

4.3.8 Release advice

The release advice is based on the difference between the expected result and the actual that are the outcome of testing the designing, planning and realising process.

If the output between those two match and no new major risks are found in the functionalities of the design a course and plan a term process, a positive advice is given for the release system.

Under certain circumstances when the test result differ, a conditional advice is given depending on the category of the founded risk(s).

When a negative advice is given the founded risks are discussed and a solution for the risks is founded.

4.3.9 Change and Error management

In the test designs there will be expected results. If these results are not realised in the actual testing, that particular test will be marked as failed. In some cases the tester evaluates this failing aspect and has the ability to successfully repair it, but this will not be preferred. At such a moment, the change is not registered by a tester nor a developer, which will lead to miscommunication. However, this does not occur often. Many times it will require skilled developers to find and fix the failed part of the test by the hand of the registration tool. But these developers can’t start programming right away. The project manager will receive the failed test outcome and evaluates if it is cost-effective. When sufficiently cost-effective, the project manager gives a thumbs up, and the developers can start.

4.4 Planning

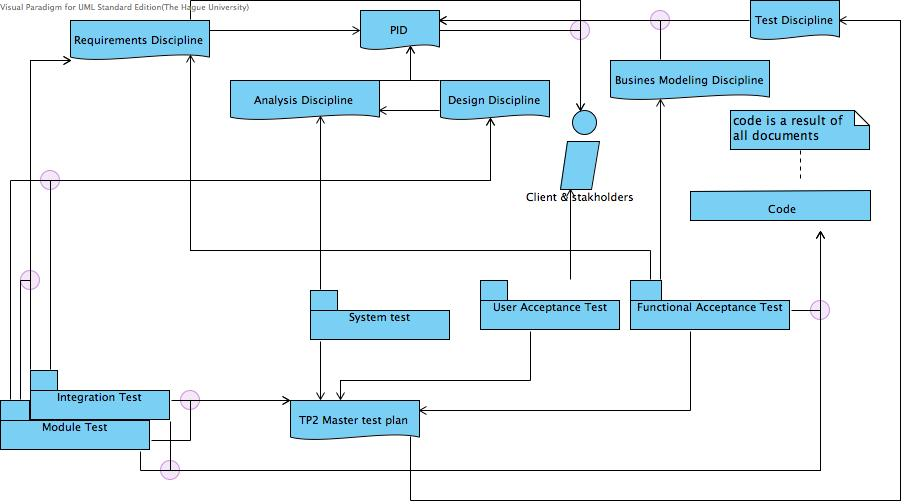
The master test plan prescribes 20 hours total for the system test, before 10 hours of the functional acceptance test.

Simply checking the delivered code on the presence of the requirements and the designs and sequence diagrams should take eight hours maximum. Resulting in a possible three runs of the test.

When the code first reaches alpha phase, the coding should immediately be system-tested. Results can then be handled, this will most likely result in alteration of the code and the need for new module and integration tests.

When the second run of module and integration tests have concluded, all results should prove positive. A rerun of coding and module/integration tests could be needed, making the third the definitive final run. Delivering the perfect base for the functional acceptance test.

4.5 Sanity check report



The above diagram shows the needed documents for the various tests. The PID is the basis for all documents and has been created in collaboration with the client. The PID results in five documents which have been created by the project-group and checked and approved by the client and stakeholders.

To improve the readability of the diagram, grey circles have been added to show where lines come together.

We can assume the quality of the documents is sufficient because of interviews and reviews with the client and stakeholders. As a result, the basis for the tests are assumed sufficient.

RUP system development ensures that by designing tests, any shortcomings in base documents can still be added or salvaged during construction, even though the team carefully constructed all documents and tests to avoid alteration during the construction phase.

The necessary states of the documents to be suitable for testing are described in chapter test base of the TP2 master test plan.

5. Test design

[Select for each requirement a test design technique from the technique matrix of the test plan and specify the related logical and physical test cases. Specify this at least for the requirements that are part of the architectural proof of concept and for the requirements with the highest priority.]

**ManageCPM**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Action** | Course | Teaching Unit | Task | Activity |  |  |
| New course | C |  |  |  |  |  |
| Edit course | U |  |  |  |  |  |
| New TU |  | C |  |  |  |  |
| Delete TU |  | D |  |  |  |  |
| New task |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |