|  |
| --- |
| FONTYS UNIVERSITY OF APPLIED SCIENCE |
| Test Plan |
| Parcel Handling Simulation |
|  |
| **GDS - Group 3** |
| **10/11/2010** |

|  |  |  |
| --- | --- | --- |
| Document name |  | Test Plan |
| Project name |  | Parcel Handling Simulation |
| Version |  | 3.0 |
| Department |  | ICT |
| Client |  | Casper Schellekens |
| School tutor |  | Peter Boots |

|  |  |  |
| --- | --- | --- |
| Group leader |  | Qian Li |
| Group members |  | Antoine Girard |
|  |  | John Ibeagha |
|  |  | Kritian Kolev |
|  |  | Sebastien Lepage |

**Document Change Control**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Related pages** | **Brief Description of Change** |
| 1.0 | 2010-09-21 | All |  |
| 2.0 | 2010-10-05 | 2, 3 | Add approach, test case overview |
| Final |  |  |  |

**Review and Approval**

|  |  |  |  |
| --- | --- | --- | --- |
| **Review** | | | |
| **Version** | **Date** | **Reviewed by** | **Position/Role** |
| 1.0 | 2010-10-04 | Casper Schellekens | Client |
| 2.0 |  | Casper Schellekens | Client |

|  |  |  |  |
| --- | --- | --- | --- |
| **Approval** | | | |
| **Version** | **Date** | **Approved by** | **Position/Role** |
| Final |  | Casper Schellekens | School tutor |

**Table of Contents**

[1. Introduction 1](#_Toc274385049)

[1.1 Goal 1](#_Toc274385050)

[1.2 Scope of the Test Plan 1](#_Toc274385051)

[2. Test Approach 2](#_Toc274385052)

[3. Test Cases 3](#_Toc274385053)

[3.1 Overview 3](#_Toc274385054)

[3.2 Details 3](#_Toc274385055)

[4. Test Scenarios 12](#_Toc274385056)

[4.1 Test scenario 1 12](#_Toc274385057)

[4.2 Test scenario 2 12](#_Toc274385058)

[4.3 Test scenario 3 13](#_Toc274385059)

# Introduction

This Test Plan document describes the appropriate strategies, process and methodologies which are chosen to plan, organize and execute the testing of the Parcel Handling Simulation application.

## Goal

A primary objective is to: assure that the system meets the full requirements, satisfies the use case scenarios and maintains the quality of the product. At the end of the project development cycle, the user should find that the project has fulfilled all of their expectations as detailed in the requirements.

The secondary goal of this testing will be to identify and expose all issues and associated risks, and ensure that all issues are addressed in an appropriate matter before released.

## Scope of the Test Plan

The test scope includes the following:

* Use case requirements listed in the **URS** final version
* Application performance

# Test Approach

According to Project Plan, there are 3 iterations during Realization Phase as below:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Start | End | Duration | Note |
| Realization Phase   1. 1st iteration (app with basic functions) 2. 2nd iteration (add distributed system) 3. 3rd iteration (final deliverables) | 2010-09-27  2010-09-27  2010-11-15  2010-12-13 | 2011-01-16  2010-10-17  2010-12-12  2011-01-16 | 10w  3w  4w  3w | Exams after  Incl. holidays |

Based on the different tasks of each iteration, a test approach was set up for both the group and the client to test the application.

For the group, all the functions and its related codes will be tested before handing the application to the client. The functions will be tested in the group and the codes will be checked individually. The school tutor will also check the codes of each function and the distributed system. In the end of each iteration, the client will be able to test the functions according to the test cases, in order to give feedback on the acceptance of our work.

To sum up, a test schedule is set up for the group, the tutor and the client.

|  |  |  |
| --- | --- | --- |
| **Functional Test Plan – the Group and the Tutor** | | |
| Period | Test Content (the group) | Feedback (the tutor) |
| 1st iteration | Code for basic functions, performance of basic functions like New, Save, Add component, etc | Guidance to correct the codes and how to improve |
| 2nd iteration | Code for distributed system and the algorithm for sorting parcels, how the sorters and storages work | Guidance to improve the distributed system and algorithms |
| 3rd iteration | All the codes and whether distributed system functions well, the overall performance of the application | Remarks on the whole project and group work |

|  |  |  |
| --- | --- | --- |
| **Acceptance Test Plan - Client** | | |
| Date | Test Content | Expected feedback |
| End of 1st iteration  2010-10-18 | Basic functions like New, Save, Add component, etc  Test case 1, 2, 3, 4, 5, 9, 10, 11, 12 (see chapter 3) | Acceptance of basic functions and what are still expected |
| End of 2nd iteration  2010-12-13 | Sorting functions and simulation workflow  Test case 6, 7, 8, 13, 14, 15(see chapter 3) | Acceptance of distributed system and what to improve for the whole |
| End of 3rd iteration  2011-01-17 | The whole application and its overall performance | Remarks on the complete application and mark for the project |

# Test Cases

## Overview

|  |  |  |
| --- | --- | --- |
| **Test case ID** | **Description** | **Use Case ID** |
| 1 | New simulation | 1 |
| 2 | Save as | 2 |
| 3 | Save | 3 |
| 4 | Open | 4 |
| 5 | Exit | 5 |
| 6 | Start simulation | 6 |
| 7 | Pause simulation | 7 |
| 8 | Stop simulation | 8 |
| 9 | Add component | 9 |
| 10 | Draw conveyors | 10 |
| 11 | Add storage | 11 |
| 12 | Remove object | 12 |
| 13 | Set parcel simulation prosperities | 13 |
| 14 | Set conveyor belt speed | 14 |
| 15 | Set simulation speed | 15 |

## Details

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **1** | |
| **Use Case - ID** | 1 | |
| **Test Case - Title** | New Simulation | |
| **Author** | Qian Li | |
| **Pre-condition** | The client program is open.  A simulation is not running. | |
| **Description** | To start a new simulation design | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Select File->New or click on “New simulation” button | 1. The system creates a new workspace if there’s no unsaved change. 2. The system shows a form with “Save”, “Don’t save” and “Cancel” choices if there are unsaved changes. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **2** | |
| **Use Case - ID** | 2 | |
| **Test Case - Title** | Save as | |
| **Author** | Qian Li | |
| **Pre-condition** | The client program is open.  A simulation is open and not running. | |
| **Description** | To save the changes of a simulation | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Select File->Save as or click on “Save as” button | The system shows a window with locations. |  |
| 1. Find the location or enter the location in “File name” and choose the file format | The system shows the information just entered. |  |
| 1. Click on “Save” button | The system closes the window and saves the simulation/changes. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **3** | |
| **Use Case - ID** | 3 | |
| **Test Case - Title** | Save | |
| **Author** | Qian Li | |
| **Pre-condition** | The client program is open.  A simulation is open and not running. | |
| **Description** | To save the changes of a simulation | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Select File->Save or click on “Save simulation” button | 1. The system saves the changes and shows nothing, if the simulation is already saved before. 2. The system shows a window of the locations for saving a new simulation. Go to Test Case 2 action 2 and 3. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **4** | |
| **Use Case - ID** | 4 | |
| **Test Case - Title** | Open | |
| **Author** | Qian Li | |
| **Pre-condition** | The client program is open.  A simulation is not running. | |
| **Description** | To open an existing simulation | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Select File->Open or click on “Open” simulation” button | The system shows a window form with locations and files. |  |
| 1. Select one existing simulation and click “Open” | The system shows the selected simulation. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **5** | |
| **Use Case - ID** | 5 | |
| **Test Case - Title** | Exit | |
| **Author** | Qian Li | |
| **Pre-condition** | The client program is open.  A simulation is open and not running. | |
| **Description** | To exit the client program | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Select File->Exit on the toolbar | 1. The system exits if there’s nothing unsaved. 2. The system goes to “Save” test case if there are unsaved changes. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **6** | |
| **Use Case - ID** | 6 | |
| **Test Case - Title** | Start Simulation | |
| **Author** | Qian Li | |
| **Pre-condition** | The program is open.  The edit part is done and the simulation mode is launched. | |
| **Description** | To start the simulation | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Click on “Start” button | The simulation is running. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **7** | |
| **Use Case - ID** | 7 | |
| **Test Case - Title** | Pause Simulation | |
| **Author** | Qian Li | |
| **Pre-condition** | The client program is open.  Simulation mode is launched and a simulation is running. | |
| **Description** | To pause the running simulation | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Click on “Pause” button | The simulation pauses. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **8** | |
| **Use Case - ID** | 8 | |
| **Test Case - Title** | Stop Simulation | |
| **Author** | Qian Li | |
| **Pre-condition** | The client program is open.  Simulation mode is launched and a simulation is running or paused. | |
| **Description** | To stop the simulation | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Click on “Stop” button | The simulation stops. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **9** | |
| **Use Case - ID** | 9 | |
| **Test Case - Title** | Add component | |
| **Author** | Qian Li | |
| **Pre-condition** | The client program is open.  Simulation mode is launched and a simulation is running or paused. | |
| **Description** | To stop the simulation | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Click on one component, then click on the workspace | The component is shown in the workspace, where the user clicked. |  |
| 1. Click on one component, accidentally click outside the workspace | Nothing happens; the system doesn’t show the component. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **10** | |
| **Use Case - ID** | 10 | |
| **Test Case - Title** | Draw conveyors | |
| **Author** | Qian Li | |
| **Pre-condition** | The application is open and is in edit status. | |
| **Description** | To draw conveyors on the workspace | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Click on “Conveyor” button, hold the mouse from one component, dragging to another one and release | The system shows a line between two conveyors on the workspace with no error. |  |
| 1. Click on “Conveyor” button, hold the mouse from one component and drop it where there’s no component | The system shows nothing. |  |
| 1. Click on “Conveyor” button, hold the mouse from one component, dragging to another one and release, but the path is invalid (like two gates already connected) | The system shows nothing. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **11** | |
| **Use Case - ID** | 11 | |
| **Test Case - Title** | Add storage | |
| **Author** | Kristian Kolev | |
| **Pre-condition** | Client application is running in editor mode. | |
| **Description** | To test the “Add storage” functionality in editor mode | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Click the “Storage” icon 2. Click on a random sector in the simulation grid 3. Click on the disabled “Storage” icon | 1. The system shows highlights the “Storage” icon 2. The system places a storage component at the selected sector. The “Storage” icon is disabled 3. The system does not react. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **12** | |
| **Use Case - ID** | 12 | |
| **Test Case - Title** | Remove object | |
| **Author** | Kristian Kolev | |
| **Pre-condition** | Client application is running in editor mode | |
| **Description** | To test the “Remove object” functionality in editor mode and re-enabling the “Add Storage” icon | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Click on the previously placed storage component in the simulation grid | The system highlights the selected component. |  |
| 1. Click on the “Remove Object” icon | The system removes the storage component and re-enables the “Storage” icon |  |
| 1. Click on the “Remove Object” icon again | The system doesn’t respond (nothing selected). |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **13** | |
| **Use Case - ID** | 13 | |
| **Test Case - Title** | Set parcel simulation properties | |
| **Author** | Kristian Kolev | |
| **Pre-condition** | Client application is running in editor mode. The simulation grid contains the following:   * 1 or more check-in desks * 1 or more gates | |
| **Description** | To test the parcel generation preferences in editor mode | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Select a check-in desk in the simulation grid | The system highlights the selected component and enables the “Parcel generation preferences” icon. |  |
| 1. Click on the “Parcel generation preferences” icon | The system opens the “Parcel generation preferences” dialog. |  |
| 1. Enter a random parcel generation rate. Enter a destination percentage split that exceeds 100% (E.g., two gates: 60%, 45%) | The system shows the set values. |  |
| 1. Click OK | The system displays a warning message, prompting the user to correct the percentage values. |  |
| 1. Modify the percentage values to a sum under 100% (E.g., two gates: 50%, 45%). | The system shows the set values. |  |
| 1. Click OK | The system displays a warning message, prompting the user to correct the percentage values. |  |
| 1. Modify the percentage values to a 100% sum (E.g., two gates – 60%, 40%) | The system shows the set values. |  |
| 1. Click OK | The system saves the new values. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **14** | |
| **Use Case - ID** | 14 | |
| **Test Case - Title** | Set conveyor belt speed | |
| **Author** | Kristian Kolev | |
| **Pre-condition** | Client application is running in editor mode | |
| **Description** | To test conveyor belt speed preference in editor and simulation mode. | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Click the “Conveyor belt speed” icon | The system opens a dialog. |  |
| 1. Modify the conveyor belt speed to a value within the accepted range, click OK. | The system changes the speed to a new value. |  |
| 1. Click the “Conveyor belt speed” icon | The system opens a dialog. |  |
| 1. Modify the conveyor belt speed to a value outside the accepted range, click OK. | The system shows a warning and prompts the user to modify the value. |  |
| 1. Start the simulation | The system switches to simulation mode. |  |
| 1. Click the “Conveyor belt speed” icon | The system opens a dialog. |  |
| 1. Modify the conveyor belt speed value, click OK. | The system changes the speed to a new value. The conveyor belts speed up or down accordingly. |  |

|  |  |  |
| --- | --- | --- |
| **Test Case - ID** | **15** | |
| **Use Case - ID** | 15 | |
| **Test Case - Title** | Set simulation speed | |
| **Author** | Kristian Kolev | |
| **Pre-condition** | Client application is running in editor mode | |
| **Description** | To test simulation speed preference in editor and simulation mode | |
| **Action** | **Expected Result** | **Check – Actual Output** |
| 1. Click the “Simulation speed” icon | The system opens a dialog. |  |
| 1. Modify the simulation speed to a value within the accepted range, click OK. | The system changes the speed to a new value. |  |
| 1. Click the “Simulation speed” icon | The system opens a dialog. |  |
| 1. Modify the simulation speed to a value outside the accepted range, click OK. | The system shows a warning and prompts the user to modify the value. |  |
| 1. Start the simulation | The system switches to the simulation mode. |  |
| 1. Click the “Simulation speed” icon | The system opens a dialog. |  |
| 1. Modify the simulation speed value, click OK. | The system changes the speed to a new value.The entire simulation speeds up or down accordingly (both conveyor speed and simulation time). |  |

# Test Scenarios

|  |  |  |
| --- | --- | --- |
| **Test Scenario** | **1** | |
| **Functional Requirements** | Program can be used to edit simulation layouts, adding and removing objects from them. | |
| **Goal** | File new, open, save, save as functionality is working as expected. | |
| **Procedure** | | **Remark** |
| 1. Create new file | |  |
| 1. Place some random objects in the grid | |  |
| 1. Connect them with conveyor lines. | |  |
| 1. Adjust settings in at least one object | |  |
| 1. Save file | |  |
| 1. Close file | |  |
| 1. Close program | |  |
| 1. Re-open program | |  |
| 1. Open file | |  |
| 1. Verify that everything is the same | |  |
| 1. Remove some objects | |  |
| 1. Attempt to close program. Program should ask whether to save the changes. | |  |
| 1. Choose to save changes | |  |
| 1. Reopen program | |  |
| 1. Open file | |  |
| 1. Verify that the objects have been removed | |  |

|  |  |  |
| --- | --- | --- |
| **Test Scenario** | **2** | |
| **Functional Requirements** | Program can locally run a simulation. | |
| **Goal** | Test that basic simulation functionality is running correctly. | |
| **Procedure** | | **Remark** |
| 1. Design a simulation layout that includes one check-in gate, one exit gate and a connecting conveyor line. | |  |
| 1. Adjust the layout's setting as follows:    1. Add one airplane to the fleet. Set an arrival time of 60 minutes after start of simulation. Specify that it lands on gate 1.    2. Set the check-in gate to generate a bag every 15 minutes. Verify that the destination plane is correct.    3. Set the simulation speed to 1 hour of simulation time in 60 seconds real time. | |  |
| 1. Start the simulation. | |  |
| 1. Observe bags at minutes 15, 30, 45 being generated from the check-in gate. Those proceed to the exit-gate's storage. The counter showing the number of stored bags increases by one every time a bag reaches the gate/storage component. | |  |
| 1. Observe that the plane arrives at minute 60. | |  |
| 1. Observe that the currently stored bags all go in the plane (storage counter goes to 0). | |  |
| 1. Observe that the bags created at minutes 60, 75, 90, 105 all go from the check-in gate to gate A, directly into the still-present plane without affecting the gate's storage counter. Additionally, the plane stay is fixed to one hour of simulation time. | |  |
| 1. Observe that the plane departs at minute 120. Bag 120 (created at minute 120), and all subsequent bags go into the storage part of the now empty gate A. The counter increases accordingly. | |  |
| 1. Stop simulation. Observe that everything goes back to the state that was seen before starting the simulation. | |  |

|  |  |  |
| --- | --- | --- |
| **Test Scenario** | **3** | |
| **Functional Requirements** | Program can locally run a simulation. | |
| **Goal** | Verify correct behavior of the simulator with more complex scenarios. | |
| **Procedure** | | **Remark** |
| 1. Design a simulation layout that includes one check-in gate, one sorter and two exit gates. The check-in desk is connected to the sorter, which connects to both gates. | |  |
| 1. Adjust the layout's setting as follows:    1. Add two airplanes to the fleet. Set an arrival time of 60 minutes after start of simulation for the first one. Arrival of the second one to 75 minutes in. Specify landing at gates 1 and 2, respectively.    2. Set the check-in gate to generate a bag every 15 minutes. Verify that the generated bags will have a destination plane randomly assigned to them out of both available planes.    3. Set the simulation speed to 1 hour of simulation time in 60 seconds real time. | |  |
| 1. Start the simulation. | |  |
| 1. Observe bags at minutes 15, 30, 45 being generated from the check-in gate. Those proceed to the correct exit-gate's storage. The counter showing the number of stored bags increases by one every time a bag reaches the gate/storage component. | |  |
| 1. Observe that plane 1 arrives at minute 60. | |  |
| 1. Observe that the currently stored bags all go in the plane (storage counter goes to 0). | |  |
| 1. Observe that the bags created at minutes 60, 75, 90, 105 with a plane 1 destination all go from the check-in desk to gate A, directly into the still-present plane without affecting the gate's storage counter. Additionally, the plane stay is fixed to one hour of simulation time. | |  |
| 1. Observe that plane 2 arrives at minute 75. | |  |
| 1. Observe that all bags with a destination of plane 2 are transferred from storage to the plane. Additionally, all other bags go to their respective planes. | |  |
| 1. Observe that both planes depart after 1 hour of stay. | |  |
| 1. Observe that subsequent bags are routed to the storage facilities of the exit gates. | |  |
| 1. Stop simulation. Observe that everything goes back to the state that was seen before starting the simulation. | |  |