|  |
| --- |
| FONTYS UNIVERSITY OF APPLIED SCIENCE |
| User Requirements Specification |
| Parcel Handling Simulation |
|  |
| **GDS - Group 3** |
| **9/28/2010** |

|  |  |  |
| --- | --- | --- |
| Document name |  | User Requirements Specification |
| Project name |  | Parcel Handling Simulation |
| Version |  | Final |
| Department |  | ICT |
| Client |  | Peter Boots |
| School tutor |  | Casper Schellekens |

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

**Approval Signatures**

|  |
| --- |
| Approved by: School tutor  Date: |
| Approved by: Client  Date: |

**Document Change Control**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Related pages** | **Brief Description of Change** |
| 1.0 | 2010-09-01 | All |  |
| 2.0 | 2010-09-07 |  | Corrections made in Last meeting with Tutor |
| Final | 2010-09-28 |  | Use case “Remove object”, storage |
|  |  |  |  |

**Review and Approval**

|  |  |  |  |
| --- | --- | --- | --- |
| **Review** | | | |
| **Version** | **Date** | **Reviewed by** | **Position/Role** |
| 1.0 | 2010-09-06 | Peter Boots | Client |
| 1.0 | 2010-09-06 | Casper Schellekens | School tutor |
| 2.0 | 2010-09-13 | Peter Boots | Client |
| 2.0 | 2010-09-13 | Casper Schellekens | School tutor |

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

**Table of Contents**

[1. Introduction 1](#_Toc273528477)

[1.1 Purpose 1](#_Toc273528478)

[1.2 Assignment: Parcel handling simulation 1](#_Toc273528479)

[1.3 Users 1](#_Toc273528480)

[1.4 Cyclic Development of Application 1](#_Toc273528481)

[1.5 Components 2](#_Toc273528482)

[1.6 Non Functional Requirements 3](#_Toc273528483)

[2. Use Cases 4](#_Toc273528484)

[3. User Interface 13](#_Toc273528485)

[4. MoSCoW Matrix 15](#_Toc273528486)

# Introduction

## Purpose

This document represents the analysis of the functionalities and user requirements of the Parcel Handling Simulation application that we intend to develop for the client. We will here explain some of the basic requirements that would be made available in the first edition of our application.

## Assignment: Parcel handling simulation

The Parcel Handling Simulation is a distributed application that is intended to resemble the parcel handling systems at the airports. It includes the simulation of the conveyors used to transport the parcels, the inputs, which are the check-in or parcel drop-off gates at the airport, sorters that route the items to their destination and the outputs, which are the destination gates of the parcels.

The user will be able to build the conveyors by drawing lines on the working area. Each parcel will have a set of properties such as destination, ID, priority based on urgency of delivery, etc, which would allow the sorters to navigate the parcel to its destination.

Finally, additional features as parcel dimension diversity, storage facility, belt speed etc. can be added to the simulation.

## Users

The users of the fully functioned application would be the companies like airports, large delivering companies, warehouses, etc.

## Cyclic Development of Application

The way our team will build this application is through the Iterative Application Development (IAD) approach. This means that when developing this application, we will have to repeat certain stages (design, and development) in a cyclic manner.

There would be 3 cycles of development. These would be preceded with the Project Plan and Design Document clarifying the exact technical nature of the application.

Each cycle would involve the addition of functional elements to the application i.e. we would have the realization of the full application with certain functions working in a cycle and then we would add more functional elements to the application in the next IAD. As mentioned above, the idea would be to use 3 IAD’s to achieve our target of a fully functioned parcel handling the simulation application.

Further details on the exact constituents of the various phases are stated in the Project Plan.

## Components

Here is a short description of each component from which the user can create the simulation layout.

The user can add or remove all of the following elements in the simulation with the exception of the actual payload - the parcels. Those are generated automatically at the check-in gates.

**Sorter:**

An object in the application that has a function of getting parcel information and guiding the parcel further to its corresponding destination gate.

**Check-in gate:**

An application component with the function of getting the end destination information and other needed information used in handling the parcels. Entry point for all parcels will enter in the simulation.

**Destination gate (with storage):**

A point of the exit of the parcels from the simulation, sorter uses the position of the destination gates to determine the direction of parcels. Each destination gate is automatically generated with a storage which is used for storing parcels for later flights. If there are more flights at that destination gate, the parcels will be put to the storage first then go to the flights.

**Parcels:**

The objects being moved from one check-in gate to one destination gate. When checked in the parcels are given random values that determine its destination gate. On their creation by the check-in desks they are assigned a destination gate, after which they are routed directly to there by the sorters.

**Conveyor segment (straight):**

This component is available in 4 varieties - horizontal, vertical and two diagonals. It connects all other components together into a logical parcel handling simulation.

**Conveyor segment (90 degree turn):**

This component serves the same function as the straight conveyor and allows the user to build more topologically complex conveyors. There are 4 varieties - the four rotations of the base element based on the direction of the movement.

**Conveyor segment (45 degree turn):**

This component facilitates include of the sorters in the conveyor system. Again, it has 4 rotations.

## Non Functional Requirements

1. Interface requirements:

- The system must be intuitive. The most common functions should be one click away.

- The user interface should be responsive. It should immediately acknowledge the user input.

- English is used as the language.

2. Operating requirements:

- The system should be a turn-key solution, not requiring an installation or additional configurations before it can be used, as much as its distributed nature allows that.

- The application should be able to work on the window OS platform.

3. Performance requirements:

- The system should not limit the number of elements in the simulation.

- The application must be able to work as a distributed application.

- The system should strike an acceptable balance between smoothness of the simulation (refresh rate) and network utilization.

- Visualization of parcels in movements is necessary.

4. Lifecycle requirements:

- The development time specified in the project phasing should not be exceeded.

- The system should require zero maintenance.

# Use Cases

|  |  |
| --- | --- |
| ID | **1** |
| Name | **New simulation** |
| Goal | To create a new simulation file |
| Pre-condition | The client program is open.  The simulation is not running. |
| Main  Success  Scenario | 1. The user selects File->New on the toolbar. 2. The system creates a new empty workspace. |
| Extensions | 1a. A simulation file is already open and there is unsaved change.  1. The system presents the user with the choices: Save, Don’t Save, Cancel  2. The user selects one of the three and the system responds accordingly:   |  |  | | --- | --- | | Save | * 1. The system goes to "Save" use case   2. The system returns to MSS, step 2 | | Don't save | * 1. The system closes the current simulation, discarding unsaved changes   2. The system returns to MSS, step 2 | | Cancel | The system exits the use case | |
| Post-condition | A new empty workspace is created. |
| Author | Kristian Kolev |

|  |  |
| --- | --- |
| ID | **2** |
| Name | **Save As** |
| Goal | To save the currently open simulation under a user-chosen name and location |
| Pre-condition | The client program is open.  A simulation is open and not running |
| Main  Success  Scenario | 1. The user selects File->Save As on the toolbar. 2. The system prompts the user to enter the directory and file name under which to save the simulation. 3. The user enters the information. 4. The system saves the simulation file. |
| Extensions | 1a. The file already exists.   1. The system prompts the user if they wish to overwrite the existing file. 2. The user makes a choice and the system responds accordingly:  |  |  | | --- | --- | | Yes | The system overwrites the existing file | | No | The system goes back to MSS, step 2 | |
| Post-condition | All changes to the simulation file are saved. |
| Author | Kristian Kolev |

|  |  |
| --- | --- |
| ID | **3** |
| Name | **Save** |
| Goal | To save the currently open simulation |
| Pre-condition | The client program is open.  A simulation is open and not running. |
| Main  Success  Scenario | 1. The user selects File->Save on the toolbar. 2. The system saves all changes to the simulation file. |
| Extensions | 1a. The user has not previously saved the current simulation.  The system goes to the “Save as” use case, MSS, step 2. |
| Post-condition | All changes to the simulation file are saved. |
| Author | Kristian Kolev |

|  |  |
| --- | --- |
| ID | **4** |
| Name | **Open Simulation** |
| Goal | To open an existing simulation |
| Pre-condition | The client program is open.  The simulation is not running. |
| Main  Success  Scenario | 1. The user selects File->Open on the toolbar. 2. The system shows a window form which contains files and locations. 3. The user chooses a simulation file and click “OK” button. |
| Extensions | 3a. The simulation file doesn’t exist.  The user clicks “Cancel” button and go to MSS 1.  3b. There is an existing open simulation already.  If there’s no change to the open simulation, the system closes the existing open simulation and opens the one that the user selects.  If there are changes to the existing open simulation, go to use case “Save” or “Save as”. |
| Post-condition | The system opens the selected simulation. |
| Author | Qian Li |

|  |  |
| --- | --- |
| ID | **5** |
| Name | **Exit** |
| Goal | To exit the client program |
| Pre-condition | The client program is open.  A simulation is open and not running. |
| Main  Success  Scenario | 1. The user selects File->Exit on the toolbar. 2. The system exits. |
| Extensions | 1a. A simulation with unsaved changes is open.  The system goes to the “Save” use case, MSS, step 2. |
| Post-condition | The client program exits successfully. |
| Author | Kristian Kolev |

|  |  |
| --- | --- |
| ID | **6** |
| Name | **Start Simulation** |
| Goal | To start the simulation |
| Pre-condition | The program is open.  The edit part is done and the simulation mode is launched. |
| Main  Success  Scenario | 1. The user clicks on “Start” button. 2. The simulation starts. |
| Extensions |  |
| Post-condition | A simulation is running |
| Author | Antoine Girard |

|  |  |
| --- | --- |
| ID | **7** |
| Name | **Pause Simulation** |
| Goal | To pause the simulation |
| Pre-condition | The client program is open.  Simulation mode is launched and a simulation is running. |
| Main  Success  Scenario | 1. The user clicks on “Pause” button. 2. The simulation pauses. |
| Extensions |  |
| Post-condition | Simulation paused |
| Author | Antoine Girard |

|  |  |
| --- | --- |
| ID | **8** |
| Name | **Stop Simulation** |
| Goal | To stop the simulation |
| Pre-condition | The program is open.  Simulation mode is launched and a simulation is running or paused |
| Main  Success  Scenario | 1. The user clicks on “Stop” button. 2. The simulation stops. |
| Extensions |  |
| Post-condition | The simulation stopped. |
| Author | Antoine Girard |

|  |  |
| --- | --- |
| ID | **9** |
| Name | **Add component** |
| Goal | To add a component to the editor. It can be: a sorter, a check-in gate, a destination gate with storage. There is a special use case for the conveyor component. |
| Pre-condition | The application is open and is in editor state. |
| Main  Success  Scenario | 1. The user selects the component icon he wants. 2. The user puts the chosen component icon on the workspace. 3. The application snaps the component to the nearest square on the grids. |
| Extensions | 1a. User doesn’t select the component icon correctly  Nothing happens in the workspace. Go to MSS step 1.  2a. User puts the component icon outside the workspace  Nothing happens in the workspace. Go to MSS step 1. |
| Post-condition | The component has been added on the workspace. |
| Author | Antoine Girard |

|  |  |
| --- | --- |
| ID | **10** |
| Name | **Draw conveyors** |
| Goal | To draw the conveyor belts between all the other components |
| Pre-condition | The application is open and is in edit status. |
| Main  Success  Scenario | 1. The user selects the conveyor icon he wants. 2. The user draws a line in the workspace from one component to another (by holding and releasing the mouse). 3. The application checks if the path contains no error. 4. The application draws the conveyor between requested points. |
| Extensions | 1a. The user draws a line from one component, and drops it where the component is not present.  The line is simply deleted and the user has to do it again.  Go to MSS step 2.  2a. The user draws a line from one component to another but it makes an invalid path (like two check-in gates connected).  The line is deleted with an alert which indicates the mistake.  Go to MSS step 2. |
| Post-condition | The conveyor has been drawn. |
| Author | Antoine Girard |

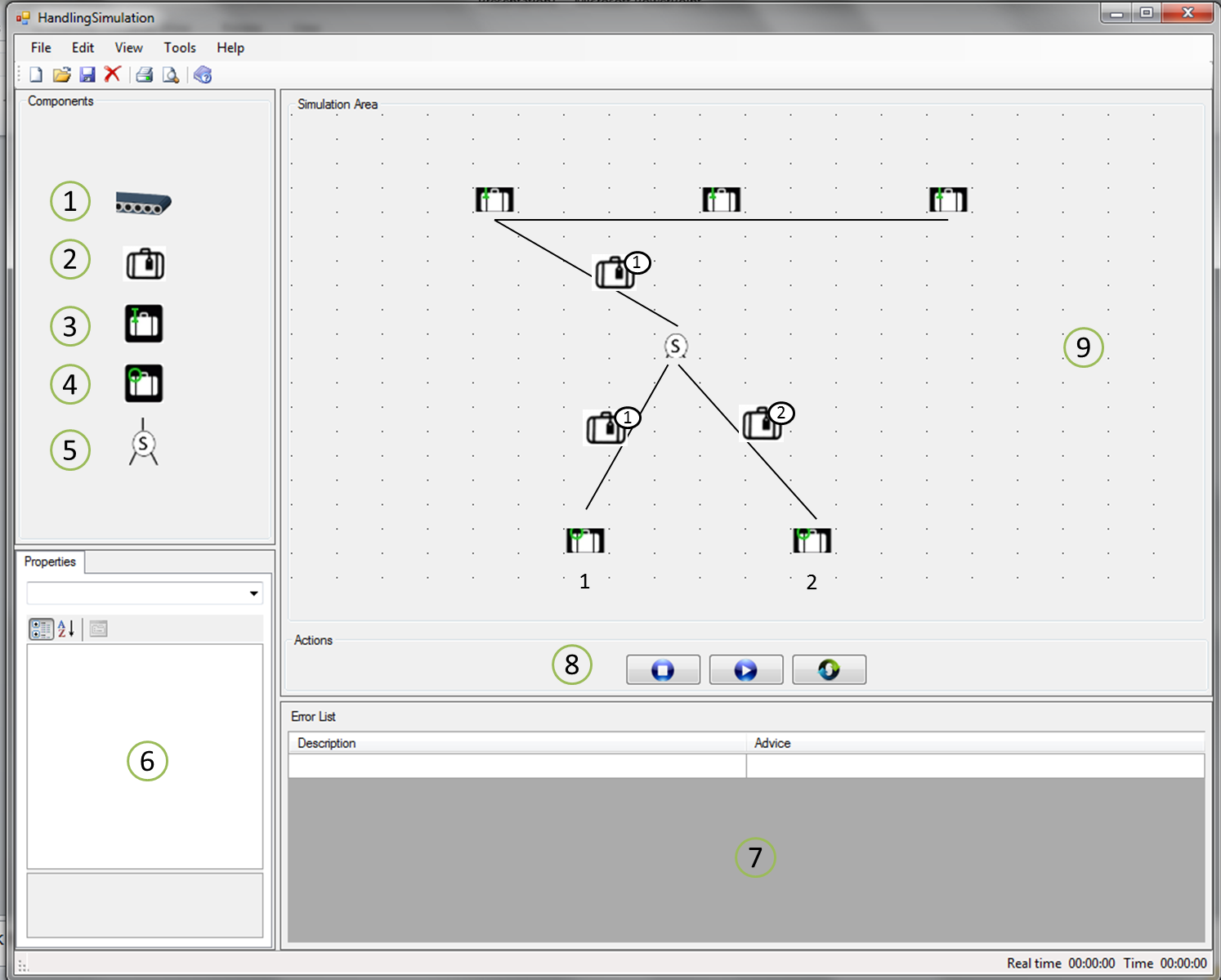
|  |  |
| --- | --- |
| ID | **11** |
| Name | **Remove Object** |
| Goal | To remove an added item in the simulation |
| Pre-condition | The application is in edit status. |
| Main  Success  Scenario | 1. The user selects a component on the workspace. 2. The user clicks on “Remove” button. |
| Extensions | 2a. The component is only connected to one other component  The system removes the component without affecting the other component.  2b. The component is connected to more than one component.  The system shows a message “This component is connected to multiple components and cannot be removed.” and doesn’t remove the selected object. |
| Post-condition | The system removes the selected component and it’s not shown anymore. |
| Author | Qian Li |

|  |  |
| --- | --- |
| ID | **12** |
| Name | **Set parcel simulation properties** |
| Goal | To set the properties of parcels, such as which parcel goes to which destination gate |
| Pre-condition | The application in edit status. |
| Main  Success  Scenario | 1. The user selects the Set Parcel Properties icon 2. System gives a form to fill in plane destination percentages and check-in gate 3. When form is filled user accepts choices filled 4. Parcel simulation properties set. |
| Extensions | 2a. User doesn’t fill the parcel simulation properties form correctly.  The system gives a message that the parcel simulation properties form should be correctly filled.  Go to MSS step 2 |
| Post-condition | Parcel Simulation is set. |
| Author | Ibeagha Ginika |

|  |  |
| --- | --- |
| ID | **13** |
| Name | **Set conveyor belt speed** |
| Goal | To set the speed of the conveyors for the whole simulation. |
| Pre-condition | The application is in edit status. |
| Main  Success  Scenario | 1. The user chooses to increase or decrease the conveyor belt speed icon. 2. The system increases/decreases the value in the speed label. 3. When simulation is running, the conveyor belt speeds up or slows down. In edit status only value changes |
| Extensions | 1a. User tries to decrease speed when speed value is on its 1st value  The system gives a message that speed value cannot be decreased any more.  Go to MSS step 1 |
| Post-condition | Conveyor belt speed has been set |
| Author | Ibeagha Ginika |

|  |  |
| --- | --- |
| ID | **14** |
| Name | **Set simulation speed** |
| Goal | To set the speed of the whole simulation. |
| Pre-condition | The application is in simulation status or edit status. |
| Main  Success  Scenario | 1. The user chooses to increase or decrease the simulation speed icon. 2. System increases/decreases the value in the speed label 3. When the simulation is running, the belt speeds up or slows down. In edit status only value changes. |
| Extensions | 1a. User tries to decrease the speed when speed value is on its 1st value  The system gives a message that the speed value cannot be decreased any more.  Go to MSS step 1 |
| Post-condition | The simulation speed has been set. |
| Author | Ibeagha Ginika |

# User Interface



**Simulation Area**

**Output Control**

**Property Bar**

**Component Bar**

**Components Bar:**

1 – Add a conveyor, the user click on the component and draw a path between the other conveyors, sorters, destination gate.

2 – Add a luggage

3 – Add a check-in gate (Drag & Drop)

4 – Add a destination gate (Drag & Drop)

5 – Add a sorter

**Properties Bar:**

6 – Will display the properties of the selected component in the edition/simulation area. All of component will be display in a list, selecting a name; the properties will be displayed as well.

**Output Control:**

7 – Display errors if there exists some inconsistencies

**Simulation Area:**

8 – Navigation bar, stop and play the simulation, check the configuration of the simulation

# MoSCoW Matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Functions | M | S | C | W |
| Distributed System Client-Server-Worker Functionality |  |  |  |  |
| Distributed System, Worker mesh Network |  |  |  |  |
| New Simulation |  |  |  |  |
| Open Simulation |  |  |  |  |
| Save |  |  |  |  |
| Save as |  |  |  |  |
| Exit |  |  |  |  |
| Start Simulation |  |  |  |  |
| Stop Simulation |  |  |  |  |
| Pause Simulation |  |  |  |  |
| Add components |  |  |  |  |
| Draw conveyors |  |  |  |  |
| Remove object |  |  |  |  |
| Set parcel simulation properties |  |  |  |  |
| Set conveyor belt speed |  |  |  |  |
| Set simulation speed |  |  |  |  |

\* **M** - MUST have this.

\* **S** - SHOULD have this if at all possible.

\* **C** - COULD have this if it does not affect anything else.

\* **W** - WON'T have this time but WOULD like in the future