**Software Requirements Specification**

**SimpleMerge Project**

**Team 17**

**Project team member:**

**Lukas Gužauskas**

**Julian**

**Celestin**

**Table of contents**

1. Introduction project team 3

2. Introduction 4

3. USE CASE 4

4. Nonfunctional Requirements 4

5. Constraints 4

6. Development and Target Platforms 4

7. Project Glossary 4

Introduction project team

Project team 17:

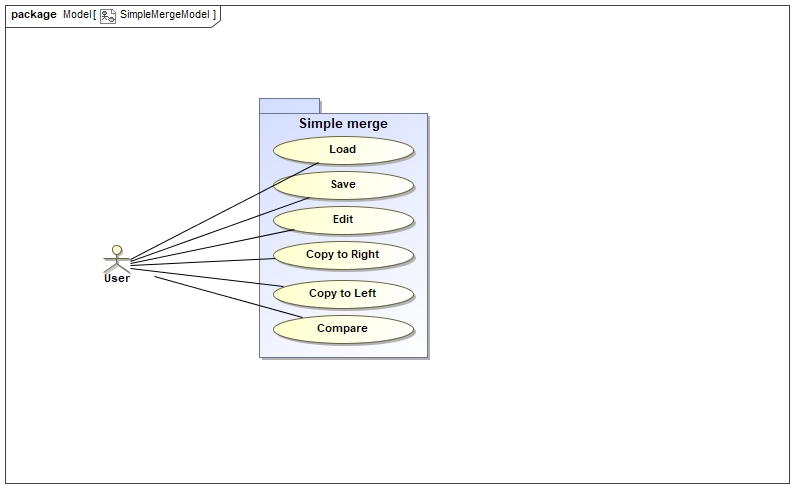
Lukas Gužauskas

Julian

Celestin

1. Introduction

The goal of this project is to create a SimpleMerge information system using Java language. The SimpleMerge displays two text files next to edit panel (side by side) and the user can use several functionalities: load, edit, save, compare 2 files and merge. This document describes the requirements of this program.



1. USE CASE

UC1 Load

UC2 Edit

UC3 Save

UC4 Copy to Right

UC5 Copy to Left

UC6 Compare

1. Flow of Events for *Load* Use Case
   1. Preconditions:

The user must have text files.

* 1. Main flow:

The user can press “LOAD” button [S1-S2] [E1-E2] on the top of the main window. The edit panels on the left and on right display the contents in the corresponding edit panel.

* 1. Subflows:

[S1]. When the “LOAD” button is pressed, the small window shows up. The user selects two text files he wants to display on edit panel on the left and the right in the system. If the user wants to cancel the input file then he can press “Cancel” button [S2].

[S2]. When the Cancel button is pressed, the small window is closed.

* 1. Alternative flows:

[E1]. If user loads text files which is not supported file format, then the system reports error message.

[E2]. If user loads text files which is a blank text file, then the system could read it.

1. Flow of Events for *Edit* Use Case
   1. Preconditions:

The user has uploaded text files into the system and displays the contents in the corresponding edit panels.

* 1. Main flow:

The user can click Edit button [S1] on top of the main window. The user could edit the text files. After editing of content is finished, the user could save the edited content into the file [UC3].

* 1. Subflows:

[S1]. When the Edit button is clicked. The program unlocks the edit panel on the left or the right depending what side the user selects by pressing the left or the right button.

* 1. Alternative flows:

None.

1. Flow of Events for *Save* Use Case
   1. Preconditions:
2. The user has uploaded text files into the system and displays the contents in the corresponding edit panels.
3. The user has edited text files.
   1. Main flow:

The user could click in two ways: Save As button [S1] or Save button [S2] on top of the main window [E1-E3], then the edited content of the file turns saved.

* 1. Subflows:

[S1]. If the Save As button is clicked, a directory window shows up asking the user for where the file could be saved. The user specifies name of the directory to place the file and clicks Save button.

[S2]. If the Save button is clicked, the edited content of the panel is saved to the same file.

* 1. Alternative flows:

[E1]. If the user specifies an invalid name (e.g. space symbol inserted) of the file, then the program shows up a pop-up window which is reporting an input error.

[E2]. If the user requests Save As that the file name is the same as the original file in the specified folder, the program rewrites the previous file.

[E3]. If the user hasn’t uploaded [UC1] the file into the panel and attempts requesting to run [S1-S2], then the program reports error message with pop-up window for the user.

1. Flow of Events for *Compare* Use Case
   1. Preconditions:

The user has uploaded text files into the system and displays the contents in the corresponding edit panels.

* 1. Main flow:

The user could click Compare button [S1][E1-E3]. Then the program displays the different lines highlighted with color.

* 1. Subflows:

[S1]. When the Compare button is clicked, the program executes comparison of two text files.

* 1. Alternative flows:

[E1]. If the user attempts to click Compare button without uploaded 2 files into both panels, nothing occurs and none information is displayed.

[E2]. If the left panel is a blank text, other panel has some lines of the text then the comparison results will make up non-highlighted text on the right panel. So, nothing occurs and none information is displayed.

[E3]. If the both panels have the same contents, pop-up window shows up that the new content is still the same as the previous content.

1. Flow of Events for *Copy to Left* Use Case
   1. Preconditions:
2. The user has uploaded text files into the system and displays the contents in the corresponding edit panels.
3. The compare feature must be executed to compare 2 contents and must be viewed different lines.
   1. Main flow:

The user could select to switch the lines that are different. Further, the user could click the Copy to Left button [S1] [E1] on top of the main window. Then the selected line will show up in the left panel.

* 1. Subflows:

[S1]. After the selected line, the marking is moving to the left panel of the next line by clicking Copy to Left button.

* 1. Alternative flows:

[E1]. If the panel on the right contents blank text, then [UC5] results to delete lines in the left panel.

1. Flow of Events for *Copy to Right* Use Case
   1. Preconditions:
2. The user has uploaded text files into the system and displays the contents in the corresponding edit panels.
3. The compare feature must be executed to compare 2 contents and must be viewed different lines.
   1. Main flow:

The user could select to switch the lines that are different. Further, the user could click the Copy to Right button [S1] [E1] on top of the main window. Then the selected line will show up in the right panel.

* 1. Subflows:

[S1]. After the selected line, the marking is moving to the right panel of the next line by clicking Copy to Right button.

* 1. Alternative flows:

[E1]. If the panel on the left contents blank text, then [UC5] results to delete lines in the right panel.

1. Misuse Cases

**UC7 Flow of Events for *Upload Text File* Use case**

**7.1 Preconditions:**

**7.2 Main flow:**

**7.3 Subflows:**

None.

**7.4 Alternative flows:**

None.

1. System Sequential Diagrams
2. Nonfunctional Requirements
3. Constraints
4. Requirements Dependency Traceability Table

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | UC1 | UC2 | UC3 | UC4 | UC5 | UC6 |
| UC1 |  |  |  |  |  |  |
| UC2 |  |  |  |  |  |  |
| UC3 |  |  |  |  |  |  |
| UC4 |  |  |  |  |  |  |
| UC5 |  |  |  |  |  |  |
| UC6 |  |  |  |  |  |  |

1. Development and Target Platforms
2. Eclipse development MVC
3. MagicDraws
4. Junit 5
5. Project Glossary
6. Document Revision History

|  |  |
| --- | --- |
| Version | 1.1 |
| Name(s) | Lukas Gužauskas |
| Date | 5/29/2018 |
| Change Description | Original creation of the SRS |