SE\_Software\_Requirement\_Specification

SimpleMerge

20110138 강원석

20133959 공찬형

20130895 성호준

20160030 심대범

20161493 하태윤

Introduction

The goal of this project is to create a Java-version of SimpleMerge program. The program provides the functions which are similar to ‘WinMerge’. The project will be divided into several phases and will be done on Github. This document describes the requirements of the program.

(Github Address : <https://github.com/lenny-kong/se_termproj>)

Functional Requirements

1. Viewing, Editing, Saving files
   1. When user start the program, a main window with 2 panels is displayed.
   2. On top of each panel, there are “Load” , ”Edit” , ”Save” buttons.
      1. When user clicks “Load” button, user can choose a file, load the contents and display them in the edit panel.
      2. When user clicks “Edit” button, user can edit the strings in the edit panel
      3. When user clicks “Save” button, the program should save the edited contents into the file.
2. Comparing Function.
   1. When user presses “Compare” button, the program displays the different lines with a colored font/background.
   2. Comparison is done line by line. (Result should be same as the one resulted from ‘Diff’)
3. Merging Function
   1. When user clicks “Copy to Right” button, the selected block in the left panel is copied to the file shown in the right panel.
   2. When user presses “Copy to Left” button, the selected block in the right panel is copied to the file shown in the left panel.
   3. Two functions will make the corresponding blocks be identical.

Non-functional Requirement & constraints

1. The project should be done on Github.
2. All progress of the project should be trackable.
3. Easy to test the code.
4. GUI components should be designed by using MVC architecture pattern.

Use case name : Simple Merge Program

Level : User goal level

Primary actor : User / Supporting actor : Operating System

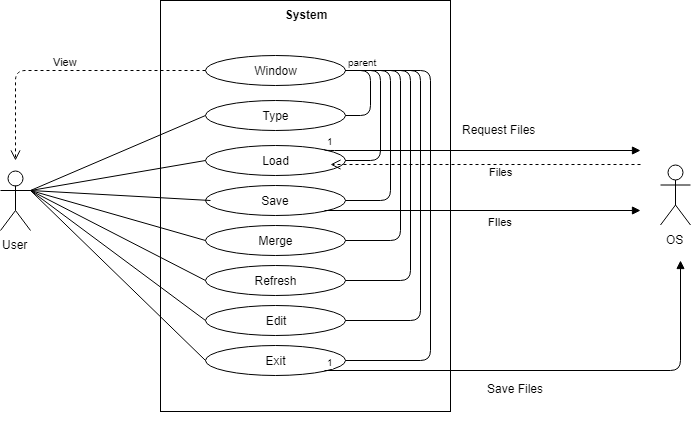
Main Success Scenario

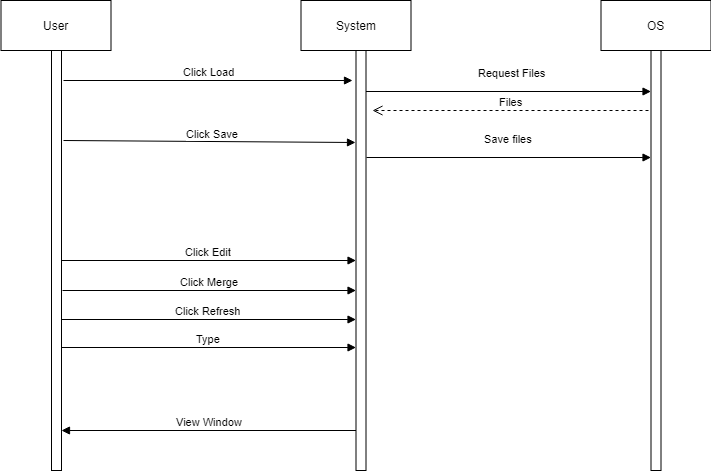
1. User starts the program. The program shows main window including few panels.
2. User clicks ‘Load’ button. A new window pops up, and user can choose two text files.
3. User chooses two text files. The program loads the files and pops them in the window.
4. User clicks Refresh button. The program compares two text files and show differences between two files.
5. User clicks ‘Merge’ buttons. The program does ‘Left to Right’ or ‘Right to Left merge’ function.
6. User clicks ‘Edit’ button. The program enters into edit mode.
7. User clicks ‘Save’ button. Operating System saves the files.
8. User clicks ‘Refresh’ button. The program compares the files again.

Alternative Scenario

1. User chooses files which is not txt files. The program does not recognize the files.
2. User tries to load new files after the first file loading is completed. The program shows warning message to user.
3. When user tries to save text file, user can overwrite original file or create a new file with new name.

UseCase Diagram & System Sequence Diagram





Development and Target Platforms

1. Windows Operating System
2. Eclipse IDE

Project Glossary

1. Merge : Change one of the files using the other (Insert, Copy etc.)
2. Refresh : Re-compare the two files after editing the files (comparing)

Document Revision History

|  |  |
| --- | --- |
| Version | Final |
| Name | SE\_Team\_18 |
| Date | June 8, 2018 |
| Change Description | Final check |

1. 4/30 : Check functional requirements & make a rough use case model
2. 5/3 : Make a use case diagram and system sequence diagram
3. 5/11 : Use case & Scenario revision
4. 5/16 : Minor fix
5. 5/20 : minor fix
6. 6/4 : minor fix
7. 6/8 : Final check