**Software Requirement Specification**

|  |  |
| --- | --- |
| **과 목** | **휴먼 ICT 소프트웨어 공학** |
| **팀 구성원** | **20123599 이동직**  **20123421 이현민**  **20146215 지무근**  **20121067 한재현**  **20164891 한윤진** |

[**1. Introduction to the system** 2](#_Toc514343517)

[**2. Use case diagrams** 3](#_Toc514343518)

[**3. Use case descriptions** 4](#_Toc514343519)

[**UC1 Load text file** 4](#_Toc514343520)

[**UC2 Edit** 4](#_Toc514343521)

[**UC3 Save** 5](#_Toc514343522)

[**UC4 Compare** 5](#_Toc514343523)

[**UC5 Merge** 6](#_Toc514343524)

[**4. Misuse cases** 7](#_Toc514343525)

[**UC6 Pressing <Edit> button without loaded file** 7](#_Toc514343526)

[**UC7 Compare or Merge when files are not loaded on both sides** 7](#_Toc514343527)

[**UC8 Merge before compare** 8](#_Toc514343528)

[**UC9 Merge with no selected section** 8](#_Toc514343529)

[**5. System Sequence Diagram** 9](#_Toc514343530)

[**6. Non-functional requirements** 10](#_Toc514343531)

[**NR1. Performance** 10](#_Toc514343532)

[**NR2. Usability** 10](#_Toc514343533)

[**Constraints** 10](#_Toc514343534)

[**7. Requirement Dependency Traceability** 11](#_Toc514343535)

[**8. Development and Target Platforms** 11](#_Toc514343536)

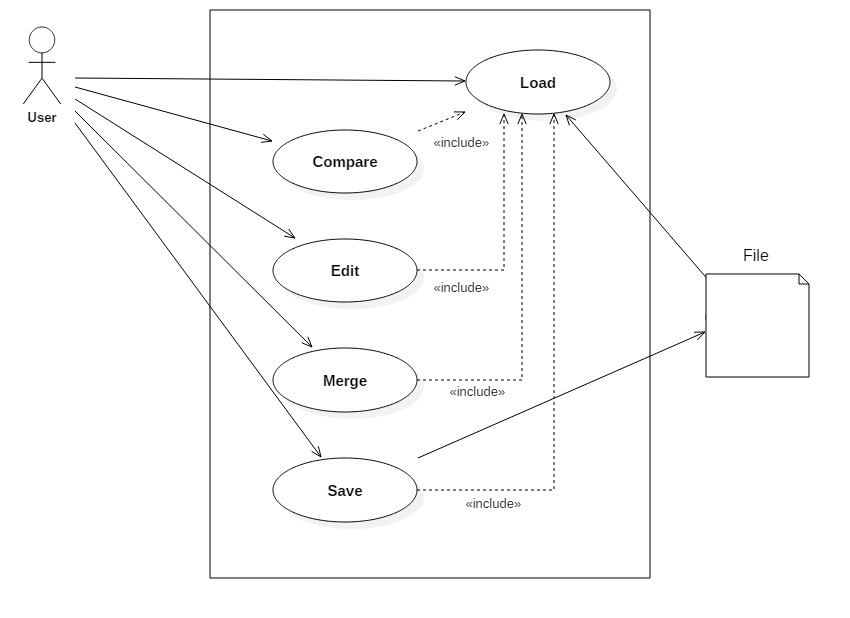
[**9. Project Glossary** 11](#_Toc514343537)

[**10. Document Revision History** 11](#_Toc514343538)

**1. Introduction to the system**

The goal of this project is to create SimpleMerge of which main functionality is to compare and merge the files. This program compares two files based on the LCS algorithm. This document describes the requirements of this program.

**2. Use case diagrams**



**3. Use case descriptions**

**UC1 Load text file**

1.1 Precondition

* User pushed <Load>button

1.2 Main Flow

* When the <Load> button pressed, a file dialog that can select text file is opened. User can choose file what he or she wants to load. The content of selected file is displayed in edit panel. [S1] [S2]

1.3 Subflows

* [S1] If two files are opened from the file dialog at the same time, they are opened in the order in which they are selected. And two files are automatically compared [UC4].
* [S2] If files are loaded on both sides by loading current file, two files are automatically compared [UC4].

1.4 Alternative Flows

* [E1] When there already exists a loaded file on the side that user press <Load> button and the loaded file has been edited, ask user to save existing file [UC3] before loading new file.

**UC2 Edit**

2.1 Preconditions

* At least one file is loaded on one of two edit panels User pressed <Edit> button

2.2 Main Flow

* User can move focus to one of the two edit panels by clicking the panel. After that, user can edit, copy, paste, cut, find and select text.

2.3 Subflows None

2.4 Alternative flows None

**UC3 Save**

3.1 Precondition

* User pushed <Save> button

3.2 Main Flow

* If the user presses the <Save> button, save the modified part of the loaded file.

3.3 Subflows None.

3.4 Alternative Flow

* [E1] If user tries to close the window without saving, ask user to save the opened file before finish this program.

**UC4 Compare**

4.1 Precondition:

* Two files are loaded on both sides.
* User pushed <Compare> button.

4.2 Main Flow

* Using “Longest Common Subsequence” algorithm, compares two loaded files on both sides line by line. [E1] Different lines are displayed with background color so that the differences between two files are easily recognizable.

4.3 Subflows None

4.4 Alternative Flow

* [E1] If two files are exactly the same, show a message box that says “No different line found”

**UC5 Merge**

5.1 Precondition

* Two files are already compared by <Compare> button.

5.2 Main Flow

* If user select colored section, it changes its background color showing that it has been selected. And either <Copy to right> button or <Copy to left> button clicked, merge selected section into designated side. Then, eliminate background color of merged section on both sides.

5.3 Subflows None

5.4 Alternative Flow None

**4. Misuse cases**

**UC6 Pressing <Edit> button without loaded file**

6.1 Preconditions

* Any file is not loaded

6.2 Main Flow

* User tries to edit by pressing <Edit> button without loaded file [E1].

6.3 Subflows None.

6.4 Alternative Flows

* [E1] If no file is loaded, disable <Edit> button to prevent from editing without file.

**UC7 Compare or Merge when files are not loaded on both sides**

7.1 Preconditions

* one or no file is loaded

7.2 Main Flow

* User tries to compare or merge when files are not loaded on both sides [E1].

7.3 Subflows None.

7.4 Alternative Flows:

* [E1] If files are not loaded on both sides, disable <Compare> and <Merge> buttons. User can compare or merge only if two files are loaded.

**UC8 Merge before compare**

8.1 Preconditions

* Two files are loaded on both sides.
* Two files are never compared or are not been compared after edit.

8.2 Main Flow

* User tries to merge before comparing two files.

8.3 Subflows None.

8.4 Alternative Flows

* [E1] Show a message box saying that “Compare before merge”

**UC9 Merge with no selected section**

9.1 Preconditions

* Two files are loaded on both sides.

9.2 Main Flow

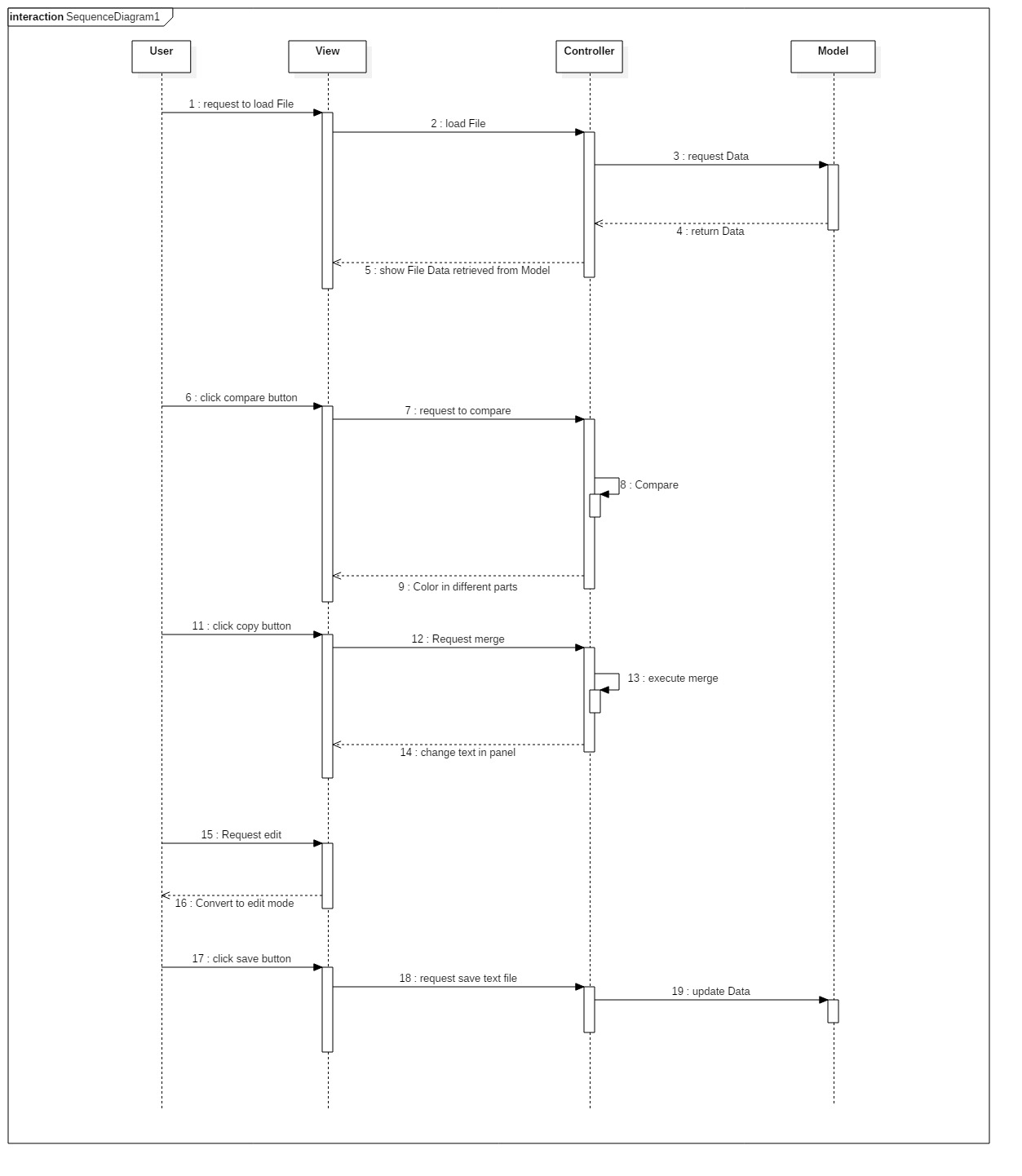
* User tries to merge with no section selected.

9.3 Subflows None.

9.4 Alternative Flows

* [E1] Show a message box saying that “Please select section to merge”

**5. System Sequence Diagram**



**6. Non-functional requirements**

**NR1. Performance**

* Compare and Merge finish in reasonable time.

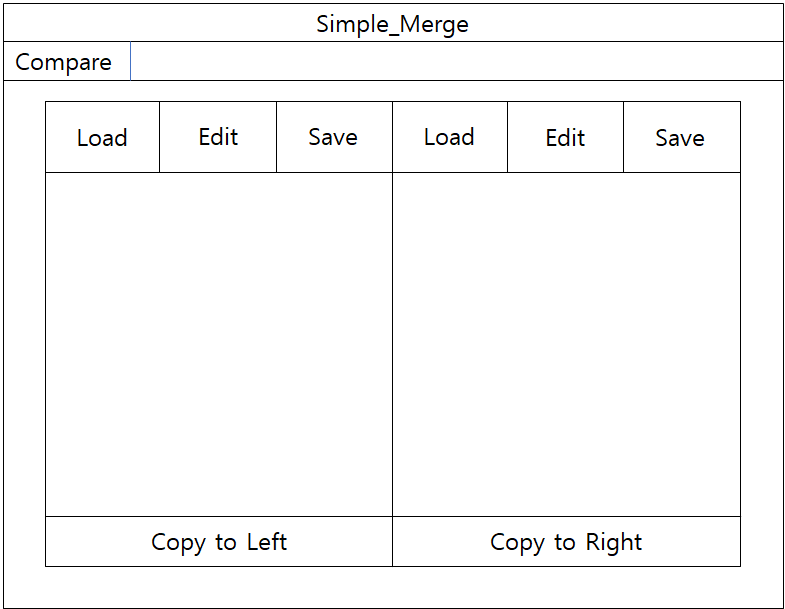
**NR2. Usability**

NR2.1 file option

* incoding type must be UNICODE

NR2.2 User Interface

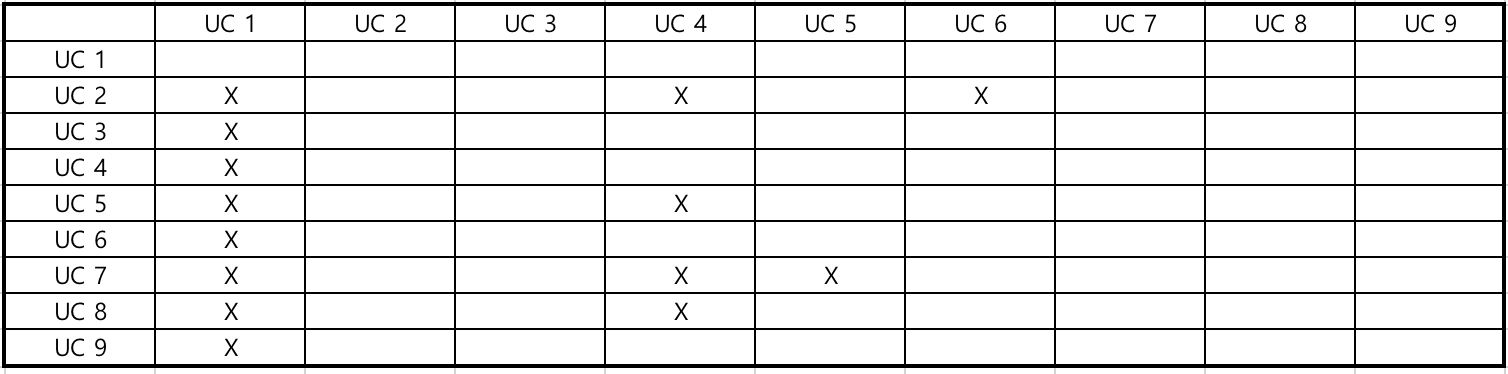
* mouse event, button clicked keyboard event



**Constraints**

* All code development shall be done with the Java programming language.
* All testing shall be done using JUnit and EasyMock.
* MVC model
* test case must contain happy path & bad case

**7. Requirement Dependency Traceability**



**8. Development and Target Platforms**

* Any systems that support Java application
* Development: IntelliJ IDEA

**9. Project Glossary**

* UC: Use Case
* NR: Non-function requirements
* LCS: Longest Common Sequence

**10. Document Revision History**

* 2018/05/09: Generating document
* 2018/05/12: Add categories and re-building
* 2018/05/15: Update Use-case scenarios
* 2018/05/16: Make a draft document