

**Software Engineering Class01 term project**

**SimpleMerge Team7**

**Software Requirements Specification**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Team Leader 김정민 |  | Team Member  김상헌, 양종만, 이동현, 임기찬, 황선준 |  | Writer  김상헌, 이동현 |

COTENTS

1. [Introduction to the System 1](#_Toc349734314)

[Definition](#_Toc349734316)

[Target Platforms](#_Toc349734317)

[Development Platforms](#_Toc349734318)

1. [Usecase Event Flow 2](#_Toc349734319)

[Usecase](#_Toc349734320) Diagram

[Usecase Descriptions](#_Toc349734324)

1. System Sequence Diagrams [3](#_Toc349734331)
2. [Non-Functional requirements 4](#_Toc349734336)

[Quality Requirements](#_Toc349734337)

[Constraints](#_Toc349734338)

1. [Appendix 5](#_Toc349734336)

[Project Glossary](#_Toc349734337)

[Requirement Dependecy](#_Toc349734338) Traceability Table

[Document Revision History](#_Toc349734338)

# Introduction to the System

## Definition

User는 프로그램을 통해 기존에 있는 텍스트 파일을 불러오거나, 수정하고, 파일을 저장할 수 있다. 만약 2개의 파일에 내용이 다른 줄이 존재하면, 차이를 나타내고 한쪽으로 병합할 수 있는 프로그램이다.

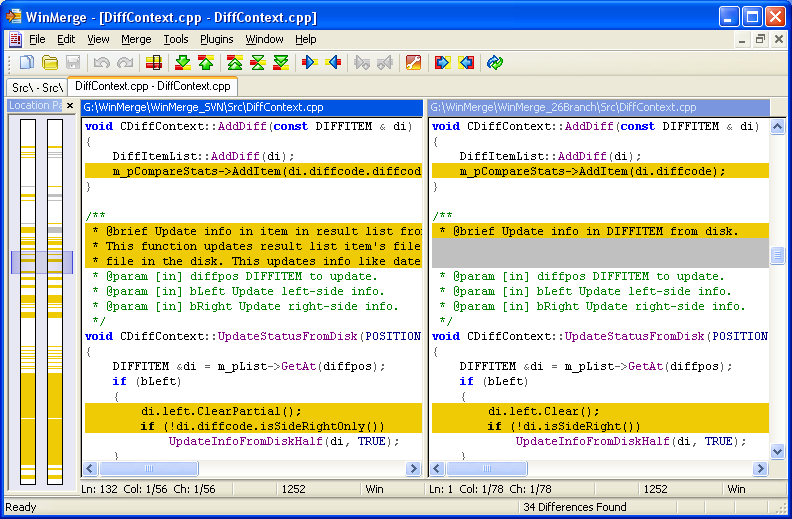


Figure 1 비슷한 프로그램의 사용 예시(WinMerge)

## Target Platforms

Windows 7, 10

Ubuntu 16.04.4

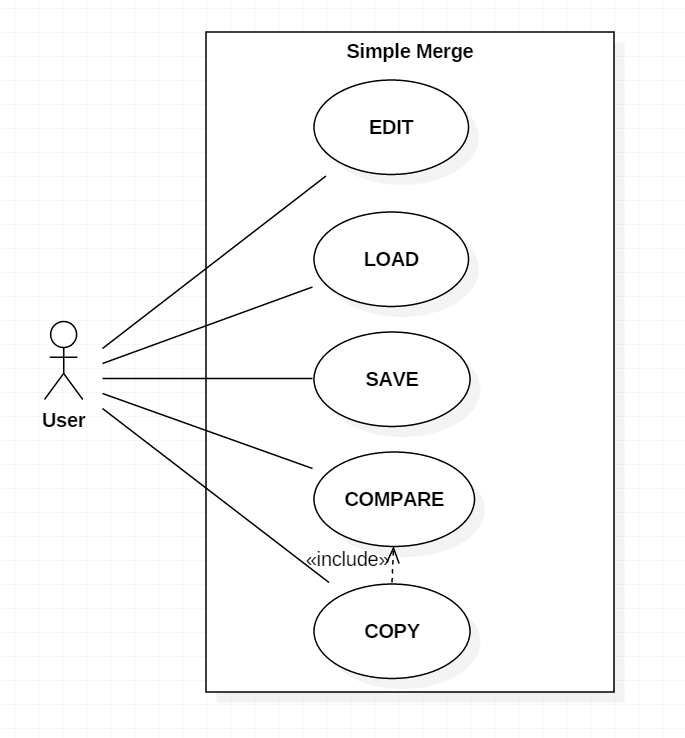
IOS

## Development Platforms

|  |  |
| --- | --- |
| JDK | 8.0.1 |
| Graddle | 4.7 |
| Eclipse | Oxygen 4.7.3a |
| JavaFX  Scene Builder | 8.4.1 |
| Junit | 4.12 |
| EasyMock | 3.6 |

# UseCase Event Flow

## UseCase Diagram



## UseCase Description

**UC1 : Load**

**UC2 : Edit**

**UC3 : Save**

**UC4 : Compare**

**UC5 : Copy**

**UC1 Flow of Events for the *Load* Use Case**

* 1. Preconditions :

None.

* 1. Main Flows :

2개의 text field에 대하여 각각의 Load버튼이 존재한다. Load버튼을 누르면 원하는 파일을 선택할 수 있는 파일 탐색기 창이 뜬다. 파일 탐색기 내에서 load하고 싶은 파일을 선택하여 열면, 해당 text field에 파일이 string형태로 load되고[S1][E1], 해당 text file의 경로를 text page의 상단에 표시해준다.

* 1. Subflows :

[S1] : 해당 Load버튼이 포함된 text field에 이미 다른 파일이 Load되어 있거나, Edit 기능이 수행된 경우, 즉 Modified 상태일 경우 기존 파일에 대하여 Save를 먼저 진행할지[UC3] 여부를 묻는 창을 띄우고, 새로운 파일을 Load해올 수 있도록 한다.

* 1. Alternative Flows :

[E1] String형식으로 표현되는 파일은 여기서는 .txt 파일을 비롯하여 .c, .cpp, .java, .py 확장자를 가지는 파일을 의미하며 해당 파일들을 UTF-8의 인코딩 방식으로 불러들인다. 그 외의 타입의 파일들은 파일 탐색기에서 표시되지 않는다.

**UC2 Flow of Events for the *Edit*  Use Case**

* 1. Preconditions :

None.

* 1. Main Flows :

2개의 text field에 대하여 각각의 Edit버튼이 존재한다. Edit버튼을 누르면, Edit 버튼이 눌린 상태로 표시되며 해당 Edit버튼이 속한 text field의 내용을 편집할 수 있다. 동시에 modified상태로 지정된다.[S1] Edit 버튼을 누른 후 수정을 원하는 라인을 더블 클릭하여 해당 line의 text를 수정할 수 있다. [E1]

* 1. Subflows :

[S1] : 이미 Compare[UC4]가 실행되어 음영 처리되어있는 text field에 대하여 edit을 진행하게 되면, edit한 부분이 포함되는 음영 영역을 제외하고 위, 아래의 음영은 유지된다. Edit이 진행된 부분은 배경색이 Compar[UC4] 진행 전인 흰색 배경으로 변환된다.

* 1. Alternative Flows :

[E1] : 해당 라인의 수정이 끝나고 Enter를 치면 해당라인의 수정이 완료되고 해당 Line밑에 빈 Line이 추가된다. 해당 라인의 Text를 모두 지우고 backspace를 누름으로써 해당 Line을 삭제할 수도 있다.

**UC3 Flow of Events for the *Save*  Use Case**

* 1. Preconditions :

None.

* 1. Main Flows :

2개의 text field에 대하여 각각의 Save 버튼이 존재한다. Save버튼을 누르면, 파일 저장을 위한 파일 탐색기 창이 뜬다. 파일 탐색기를 이용해 파일의 저장위치, 이름, 확장자를 지정해줄 수 있다. [S1] 파일이 성공적으로 저장되면, 해당 text field의 Edit버튼이 비활성화되어 편집이 불가능해진다.

* 1. Subflows :

None.

* 1. Alternative Flows :

[E1] : 이미 지정된 파일 경로가 존재하지 않는 경우, 예를 들어 프로그램 실행 이후 Load나 Save 과정을 거치지 않고 Edit만 진행한 경우, 반드시 파일의 저장 관련 설정을 지정해주는 과정을 통해 Save가 진행되어야 한다.

**UC4 Flow of Events for the *Compare*  Use Case**

* 1. Preconditions :

None.

* 1. Main Flows :

Compare 버튼은 두 text field의 내용을 비교해주는 기능으로 프로그램에 하나만 존재한다. Compare버튼을 누르게 되면, 두 개의 text field 내의 다른 부분에 대해 노란색으로 음영처리를 한다. 두 개의 text field를 비교할 때는 line-by-line으로 다른 부분이 있는지를 찾는다. 중간에 공백을 제외하면 문자열이 같아지는 줄에 대해서는 회색으로 해당 부분을 음영 처리할 수 있도록 한다.

* 1. Subflows :

None.

* 1. Alternative Flows :

None.

**UC5 Flow of Events for the *Copy* Use Case**

* 1. Preconditions :

Compare기능이 선행되어 두 text field의 다른 부분이 음영처리 되어있어야 한다. 또한 음영 부분 위에 커서가 놓여있거나, 선택된 영역 내에 음영 부분이 포함되어 있어야 한다.

* 1. Main Flows :

커서가 포함되어 있는 음영 부분에 대해서만 맞은편 text field로 내용이 복사된다. 특정 영역을 선택한 상태라면, 선택한 영역 내에 포함되는 음영 부분에 대해서 맞은편 text field로 내용이 복사된다. 이 때 음영 부분은 내용이 다른 한 줄을 나타내는 것이 아니라, 연속적인 음영 부분을 나타낸다. 연속적인 음영 부분에 대하여 커서가 포함되어 있으면 해당 음영 부분이 모두 복사된다. 복사된 부분에 대해 음영 부분은 초기화된다.

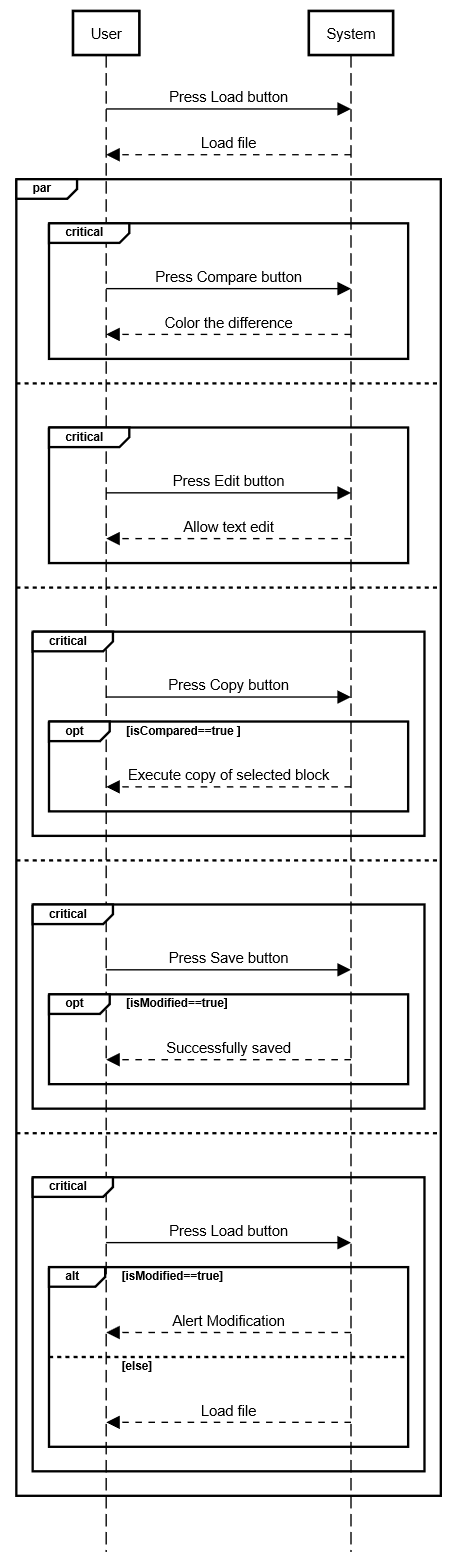
* 1. Subflows :

None.

* 1. Alternative Flows :

None

# System Sequence Diagrams



# Non-Functional Requirements

## NR1. Performance

시스템은 사용자의 입력을 기다려야 하며 시스템에 들어오는 입력에 따라 필요한 최소한의 function만을 실행한다. 모든 function은 신속하게 동작해야 한다.

NR1.1 User response

시스템은 사용자의 입력에 대해 최대 1초안에 반응해줘야 한다.

Origin : artifacts/회의록/2018\_05\_14

Priority : 3

Implementation Completed Data: 2018/05/31

## NR2. Usability

TEXT 파일들을 modify할 수 있어야 한다.

NR2.1 File Option

사용자는 text로 표현되는 파일 인 것에 관해서 인코딩 타입이 UTF-8형태로만 modify할 수 있다.

Origin : artifacts/회의록/2018\_06\_04

Priority : 3

Implementation Completed Data: 2018/06/05

NR2.2 User Interface

시스템은 mouse event, button, keyboard event on text field 와 같은 동작을 허용해야 한다.

Origin : artifacts/회의록/2018\_05\_14

Priority : 1

Implementation Completed Data: 2018/06/01

## Constraints

모든 코드 개발은 JAVA 프로그래밍 언어를 통해서 이루어 진다.

Testing Framework로 Junit과 EasyMock를 이용한다.

리눅스, MAC IOS 에서도 프로그램이 정상 작동하여야 한다.

# Appendix

## Project Glossary

|  |  |
| --- | --- |
| 파일 | 일반적인 문서파일들을 의미한다. |
| modify | Load와 Edit과 같이 기존의 파일을 수정하는 경우들을 의미한다. Copy는 해당되지 않는다. |
| Text field | modify가능한 파일의 글자들이 표시되는 영역을 의미한다. |
| Testing Framework | 유닛 테스트를 위해 쓰이는 틀을 지칭한다. 현 프로그램에선 Junit과 EasyMock를 사용할 예정이다. |
| 유닛 테스트 | 전체프로그램이 아닌 프로그램의 작은 단위들의 정상 동작 여부를 확인하는 과정을 의미한다. |
| MVC | Model/View/Controller를 의미하며 모델은 프로그램의 정보를, 뷰는 화면과 같은 사용자 인터페이스를, 컨트롤러는 뷰와 모델 간의 상호동작을 관리한다. 영역을 구분함으로써 다른 영역의 영향없이 한 영역을 쉽게 고칠 수 있게 해주는 개발 방법이다. |
| LCS 알고리즘 | Longest Common Subsequence의 약어로 두 개의 문서파일을 비교할 때 자주 쓰이는 방식이다. |
| UNICODE | 전 세계의 모든 문자를 컴퓨터에서 일관되게 표현하고 다룰 수 있도록 설계된 산업 표준이며, 유니코드 협회(UNICODE CONSORTIUM)가 제정한다. |
| UTF-8 | UTF-8은 Universal coded character set + transformation format -8bit의 약자이다. ascii 문자가 아닌 값들에 투명한 파일시스템이나 다른 소프트웨어와 호환성을 가지며 다른 인코딩과의 왕복 변환이 간단하다. |

## Requirements Dependency Traceability Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | UC1 | UC2 | UC3 | UC4 | UC5 |
| UC1 |  |  |  |  |  |
| UC2 |  |  |  |  |  |
| UC3 |  |  |  |  |  |
| UC4 |  |  |  |  |  |
| UC5 |  |  |  | X |  |

## Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| 작성날짜 | 상세 내용 | 작성자 | 버전 |
| 2018/05/06 | 최초 작성 | 김상헌 | V0.1 |
| 2018/05/07 | 문서 형식 수정 | 김상헌 | V0.15 |
| 2018/05/10 | Use case 작성 | 이동현 | V0.3 |
| 2018/05/14 | Use case 완성 | 이동현 | V0.5 |
| 2018/05/14 | 로고 수정, 페이지 정렬, Glosaary 추가 | 김상헌 | V0.55 |
| 2018/05/15 | Use case diagram/Sequence Diagram 추가, NR 수정 | 김상헌 | V0.8 |
| 2018/05/16 | Requirement Dependency Traceability Table 완성 | 김상헌 | V0.9 |
| 2018/05/17 | 제출 전 확인 | 김정민 | V1.0 |
| 2018/06/04 | 개발 후 확인 | 이동현 | V1.1 |
| 2018/06/06 | Sequence Diagram 최신화 / Usecase, Glossary 최신화 | 김상헌 | V1.2 |