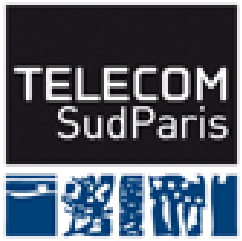


Software Engineering
Project Design

Version 1.0.1

BERRIRI, Asma

FATEEN, Mostafa

MACHADO, Rebeca

SELLAMI, Ramzi

Revision History

Date	Version	Description	Author
12/01/2015	1.0.0	Requirements specification.	Awesome Team
19/01/2015	1.0.1	Architecture design.	Awesome Team

Contents

Project Design

1. Introduction

This document describes the requirements and design of a Software Engineering project consisting in a simple program that finds the longest common substring between two ASCII text files given as input. This functionality is a common requirement in information retrieval and text editing applications.

The aim of the project is to improve the Software Engineering skills of the working team -with the assistance of professor J. Paul Gibson- going through a whole cycle of requirements specification, design, modeling, implementation and testing.

2. Requirements Specification

2.1 Functional Requirements

- The program will not have a user interface and will be executed from the command line.
- The program must accept two ASCII text (.txt) files as input. Files could be the same or even empty files.
- The program must analyze both files and retrieve the longest substring which is common in their content (including blank spaces in such a case).
- This substring –if not empty- could be a word, a single letter, a blank space, a phrase or even the whole file.
- The output should be a message displayed by command line in the form of:
  The longest substring between [Input File 1] *and* [Input File 2] *is "*[longest substring]*".*
- In case of having many common substrings with the same length, the program will return just one of them.
- In case of having no common substrings, the program must display a "*No common substrings*" message.
- There will not be any restriction for the execution time.
- There will not be any file size restriction besides from those of the machine.

2.2 Implementation Requirements

- The program must have a Java version and a C version with the same functionality.
- All inputs must return the same output in both versions.

2.3 Tests Requirements

- The tests should include at least one case with empty files as input.
- The tests should include at least one case with the same file as input.
- The tests should include at least one case with randomly generated files as input.

2.4 Example of Execution

Input

File 1	**File 2**
Hello, my name is Oompa Loompa and I love to sing in the Chocolate Factory.	I once saw an Oompa Loompa singing and dancing.

Output

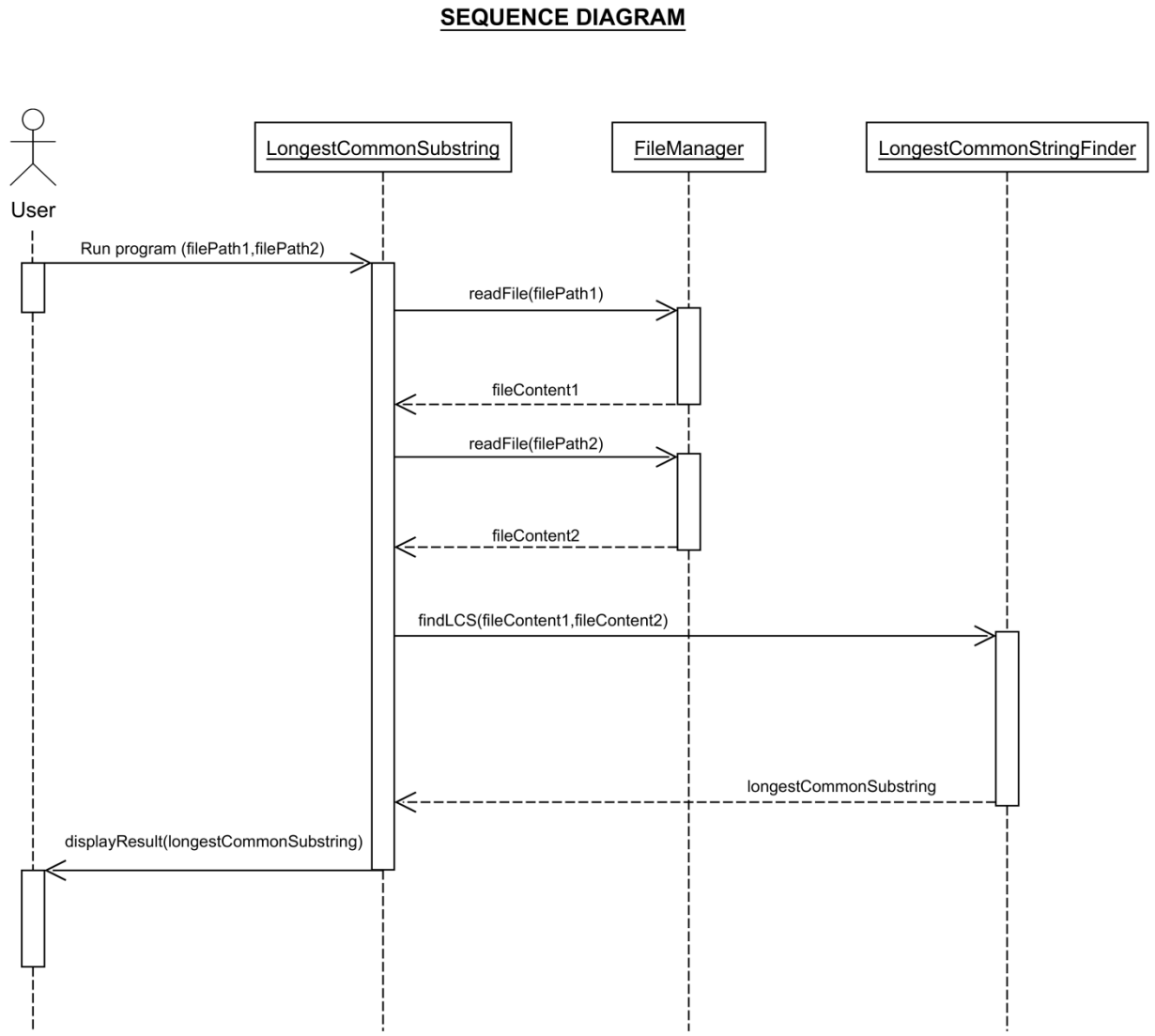
The longest substring between File 1 and File 2 is " Oompa Loompa ".

3. Architecture Design

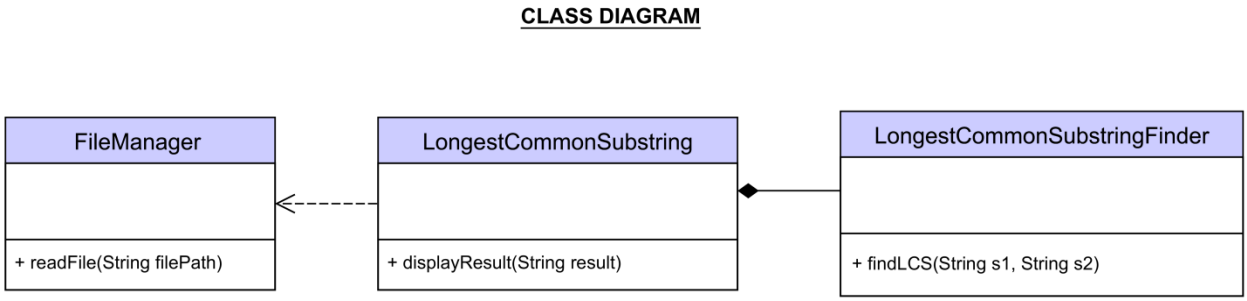
Considering that the complexity of the project relies in the algorithm to achieve the functionality and not in the structure of the software; the design is quite simple with only three structures. Java version can easily represent this as classes while C version can represent them as procedures. The concept remains the same for both versions.

We show here two diagrams to exemplify and illustrate the conceived structure and functioning of the project; without including the tests.

3.1 Sequence Diagram



3.2 Class Diagram



4. Test Cases Specification

In the java testing, we are using "junit" for unit testing. We made a test suite that calls all the other test cases.

Our test cases are:

1) Random String Generation Test:
    a. We generated an empty string
    b. We generated a 500 word random string and made sure it was alphanumeric and of the correct length
    c. We generated two strings each of a 500 characters and then we made sure of the length, alphanumeric, and that they are different
2) Longest Common Substring finder
    a. We made a simple test between "aa bb cc dd" and "aa"
    b. 2 null strings
    c. 2 strings with but letters of different case (lcs is case sensitive)
    d. No common substring
3) File Reader
    a. We read a one line file
    b. We read a multiple line file
    c. We try to read a directory
    d. We try to read a non-existing file

4) File Writer
    a. Test when we send one file to write
    b. Test when we send multiple (2) files to write