SOFTWARE REQUIREMENTS SPECIFICATION

for

CMPS 4113 - Software Engineering

Contents

Revision History

February 4, 2017 Initial Draft - Christopher Silva, Anthony Enem, Nathan Durst, Da Dong, and Shujing Zhang.

1 Introduction

Dr. Stringfellow (hereafter referred to as the client), is interested in software that will help ensure that her computer science students are writing programs that fit her specifications. This software should calculate and display metrics about the users source code such as line of code, lines of documentation, and the ratio of the two.

1.1 Purpose

This document details the Project Plan for the Software Metrics Calculation System (hereafter referred to as SCMS), which the Software Engineering group ID-10-T (hereafter also referred to as the team) has devised to assist in the software development process. The plan outlines the different areas of the project that must be addressed for successful development of the software. It establishes guidelines for resources that will be used in the project, and also points out additional resources that are needed. The Project Plan shows how the team is comprised and states the means of reporting. This plan addresses some of the risks involved in the project and the steps to correct those risks, if they occur. Also, quality assurance will be mentioned, and a glossary of terms used in this document is included.

1.2 Scope

The client wants SMCS to quickly calculate code metrics on student source code. The client currently spends an excessive amount of time looking for issues that could be solved if students had software to point them out. The client would like SMCS to support C++ and Java source code. The client would like SMCS to be easily extensible in the future to allow for more types of metrics or languages.

1.3 Main Objective

The main objective of SMCS is to help first and second year computer science students become better programmers by giving them a tool that will point out some frequent simple mistakes that they make.

1.4 Overview of Document

The remainder of the document is intended to inform the client of the intended system. Hardware and software requirements, major users, both major and minor functions,

constraints, and intended user interface are described.

2 Users

2.1 Who are the Users?

The principal users of the SMCS are students taken freshmen or sophomore level computer science courses. Due to the SMCS being able to analyze only C++ and/or Java source code files, the users would also need to be familiar with the language(s) accepted by the SMCS. Other users might include upper level CS students who need some analysis on their source code written in the specified language, or professors who might want to integrate the SMCS as part of their grading system to analyze students source code written in programming languages accepted by SMCS.

2.2 Use Cases

User analyzes a source code file :

1. User double clicks the SMCS executable file and that launches the SMCS web interface on the computers default browser.
2. User loads file(s) by clicking on an open file button and selecting file(s) from an Open file dialog OR by dragging and dropping source code file(s) onto a drag-and-drop panel on the SMCS interface.
3. User chooses programming language of uploaded source code. file(s) in one of two ways:
   a) User accepts SMCS automatically detected programming language from the file(s) extension.
   b) User selects file(s) extension of source code file(s) from a drop-down menu of accepted languages if automatically detected language is incorrect.
4. User clicks a button for SMCS to begin source code analysis.
5. SMCS analysis
   a) Main success scenario:
      i. SMCS successfully completes analysis of uploaded source code file(s).
      ii. SMCS displays results from analysis of uploaded file(s) to the user
   b) Incorrect source code file scenario :

i. File(s) uploaded by user is not of the selected programming language. SMCS displays an error of this scenario and prompts the user to select the correct programming language.

6. User closes the SMCS application.

3 System

3.1 Development Environment

The team will program SMCS in Go using JetBrains Gogland as the IDE. The team will use Git for version control, Slack for communications, and LaTeX for documentation.

3.2 Target Environment

SMCS must run on Windows XP or later, Linux, and Mac OS X 10.7 or later.

3.3 Functional Requirements

SMCS must allow the user to load a source code file and view metrics about their code.
SMCS must initially support C++ and/or Java.
SMCS must support the following metrics:

- Lines of Code (LOC) - The number of lines of code.
- Lines of Documentation (LOD) - The number of lines of documentation.
- Ration of LOC to LOD - This will be used to tell the user if there is too little documentation.
- Blank Lines - The number of blank lines. This will be used to tell a user if there is not enough whitespace.
- Total Lines - The total number of lines.
- Number of Functions - The number of functions in a file.
- Number of Function Parameters - The number of parameters in each function. This will be used to warn a user if there is an excessive amount of parameters in a function.
- Number of Non-void Functions - The number of functions that do not have a void return type.
- Methods Per Class - The number of methods in each class.
- Lines Per Function - The number of lines in each function.
- Cyclomatic complexity - The number of linearly independent paths within a program.

3.4 Non-functional Requirements