

Software Metrics Calculation System

Test plan

Version 1.0

Prepared by
Christopher Silva
Shujing Zhang
Anthony Enem
Nathan Durst
Da Dong

Contents

1	Introduction	1
1.1	Purpose	2
1.2	Product Overview	2
1.3	Test Types	2
1.3.1	Unit testing	2
1.3.2	Validation testing	2
1.3.3	System testing	2
2	Scope and Objectives	2
2.1	Unit Testing	3
2.2	Validation Testing	3
2.3	System Testing	3
2.4	Testing Strategy	3
3	Testing Requirements	3
3.1	Hardware	4
3.2	Software	4
4	Recording Procedures	4
4.1	Unit Test Procedures	4
4.2	Validation Test Procedures	4
4.3	System Test Procedures	5
5	Test Results	5
6	Appendix	5
6.1	How to Report a Bug	6
6.2	Contacts	6
7	Glossary	6

1 Introduction

1.1 Purpose

This document describes the plan for testing the Software Metrics Calculation System (SMCS). This Test Plan document supports the following objectives: Identify the required resources and provide an estimate of the test efforts. List the deliverable elements of the test activities. Recommend and describe the testing strategies to be employed.

1.2 Product Overview

The main objective of SMCS is to help first and second year computer science students become better programmers. This software will calculate and display metrics about the users source code such as line of code, lines of documentation, cyclical complexity, etc. These metrics will be used to give the user feedback on their code and to correct commonly made mistakes.

1.3 Test Types

The following is a list of test types we are going to follow

1.3.1 Unit testing

The purpose of Unit tests is to discover any incorrect or insufficient code. This will be achieved by focusing on the operations encapsulated by the class and the state behavior of the class.

1.3.2 Validation testing

Once the individual modules and their relative data structures and methods have been tested, the validation phase will check that SMCS meets specifications defined in the requirements document.

1.3.3 System testing

Once the unit and validation testing are completed, the software then undergoes system tests which interacts with input and tests the successfulness of the browsers and operating systems.

2 Scope and Objectives

Outlined below are the main test types that will be performed. All test plans and conditions will be developed from the requirements document and design diagram.

2.1 Unit Testing

The SMCS development progress will be reviewed at regularly scheduled meetings. Each review will correspond to one system module. The objective of the reviews is to ensure correctness and functional integrity within each module. Issues to consider are the correctness of tokenization, parameters and arguments as well as expected returned values from modules. The testing method used for this phase is white box testing.

Entrance Criteria - A new module should be coded in time for each peer review. As the group meets every Saturday, a new module should be completed so that it can be reviewed and tested.

Exit Criteria - All errors identified, during formal reviews and unit testing are fixed and tested.

2.2 Validation Testing

The validation testing will prove that the minimum metrics specified from the requirements document will be implemented and their results will be correct. The actual testing method used for this phase is alpha testing.

Entrance Criteria - Enough code is provided and unit tested so that the user can complete the alpha testing successfully.

Exit Criteria - All errors from validation tests must be fixed or otherwise documented.

2.3 System Testing

This test phase proves that the system requirements are met as defined in the requirements document. This should prove that the SMCS will be successful on the specified operating systems as well as different internet browsers.

Entrance Criteria - All modules and classes are implemented, unit tested and validation tested.

Exit Criteria - All errors from the system test must be fixed or otherwise documented.

2.4 Testing Strategy

The testing strategy will utilize predominantly white box testing and alpha testing.

3 Testing Requirements

The testing phases will require atleast five PCs with the following specifications:

3.1 Hardware

- 32 bit architecture
- at least 500 MHz processor speed
- at least 64 MB RAM

3.2 Software

- Windows XP, OS X 10.7, Linux 2.6.23 operating systems
- Firefox, Chrome, Edge, Safari web browser

4 Recording Procedures

During each test phase, errors will be recorded as they are detected. Errors will be categorized as high or low priority based on their impact on system performance.

4.1 Unit Test Procedures

All errors recorded during the unit test phase of each module along with the action taken to correct the error will be documented as a Unit test result.

4.2 Validation Test Procedures

All errors recorded during the validation test phase along with the category of the error, status of the error and action taken will be documented as a validation test result.

4.3 System Test Procedures

All errors recorded during the system test phase along with the category of the error, status of the error and action taken will be documented as a system test result.

5 Test Results

This chapter defines and describes the results of the testing process.

Unit Testing

test	Input	Expected Output	Actual Output	Result
1	+	ADD	ADD	true
2	-	SUB	SUB	true
3	int main()	FUNCTION	FUNCTION	true
4	#include <iostream >(.cpp)	IMPORT	IMPORT	true
5	import java.util.Date (.java)	IMPORT	IMPORT	true
6	//comment	LINECOMMENT	LINECOMMENT	true
7	/*comment*/	BLOCKCOMMENT	BLOCKCOMMENT	true

Validation Testing

test	Input	Expected Output	Actual Output	Result
1	empty file	0 LOC, 0 LOD	TBD	TBD
2	15 lines without comments	15 LOC, 0 LOD	TBD	TBD
3	21 lines of just comments	0 LOC, 21 LOD	TBD	TBD
4	7 lines of comments 8 lines of code	8 LOC, 7 LOD	TBD	TBD
5	2 lines of code 12 lines of comments 53 lines of code	55 LOC, 12 LOD	TBD	TBD

6 Appendix

6.1 How to Report a Bug

If you find any bugs, please tell us by sending an email to ID10T.BUG@example.com
Please include the following information when reporting a bug:

- A complete description of the problem what led to it.
- What operating system and web browser you are using.
- What error messages are displayed.

6.2 Contacts

Contact information for ID-10-T team members.

Member Name	Phone Number
Christopher Silva	940-782-1234
Anthony Enem	940-782-2345
Nathan Durst	940-782-3456
Da Dong	940-782-4567
Shujing Zhang	940-782-6789

7 Glossary

Alpha testing - A trial of software carried out by a developer before a product is made available for beta testing.

White-box testing - A method of testing software that tests internal structures or workings of an application.