HubListener: Verification & Validation Plan

Zed Ahmad, Prakhar Jalan, Pedro Oliveira, Piranaven Selvathayabaran

December 3, 2018

1 Revision History

Date	Version	Notes
November,21 2018	1.0	Template Setup

2 Symbols, Abbreviations and Acronyms

The symbols, abbreviations and acronyms used in this document include those defined in the table below.

symbol	**description**
MIS	Module Interface Specification
MG	Module Guide
TC	Test Case
VnV	Verification and Validation

Contents

1 Revision History **i**

2 Symbols, Abbreviations and Acronyms **ii**

3 General Information **1**
- 3.1 Summary . . . 1
- 3.2 Objectives . . . 1
- 3.3 References . . . 1

4 Plan **2**
- 4.1 Verification and Validation Team . . . 2
- 4.2 SRS Verification . . . 2
- 4.3 Design Verification Plan . . . 2
- 4.4 Implementation Verification Plan . . . 3

5 System Test Description **3**
- 5.1 Tests for Functional Requirements . . . 3
  - 5.1.1 Valid User Input . . . 3
  - 5.1.2 Requirements Testing . . . 4
  - 5.1.3 Error Handling Testing . . . 5
- 5.2 Nonfunctional Requirements Evaluation . . . 5
  - 5.2.1 Usability Testing . . . 5
  - 5.2.2 Maintainability Testing . . . 6
  - 5.2.3 Performance Measurement . . . 6
- 5.3 Trace to Requirements . . . 7
- 5.4 Trace to Modules . . . 7

6 Unit Testing **7**

7 Automated Testing **7**

8 Appendix **7**
- 8.1 Usability Survey Questions . . . 7

List of Tables

List of Figures

This document outlines the system verification and validation plan for the HubListener software. General information regarding the system under test and the objectives of the verification and validation activities are provided in Section 3. Overviews of the verification plans for the SRS, design, and implementation are given in Section 4, along with a summary of the validation plan for the software.

3 General Information

3.1 Summary

The software being tested is the open-source software, HubListener. HubListener provides users with relevant metrics, trends, and information regarding their GitHub project, and in some cases, push the user to make appropriate changes (can be organizational) that are intended to lean the project towards a more 'successful' one.

3.2 Objectives

The purpose of the verification and validation activities is to confirm that HubListener exhibits the desired software qualities. The primary objective is to build confidence in the correctness of the software. The tests described in this document cannot definitely prove correctness, but they can build confidence by verifying that the software is correct for the cases covered by test.

3.3 References

Information about the purpose and requirements of HubListener can be found in the SRS document. The latest documentation for HubListener can be found on GitHub: HubListener

4 Plan

4.1 Verification and Validation Team

The HubListener Team, which includes all the authors of this document will be responsible for the verification and validation of HubListener. Input from our supervisor, Dr. Smith will also contribute to the VnV Plan and he will ensure proper procedures take place.

4.2 SRS Verification

SRS Verification will be carried out by reviews. The HubListener team will do two reviews. One review will take place on January 31st, 2019, the expected launch date of Version 1. A second review will occur at the launch date of Version 2 which will be determined at a later date. These reviews will be done as a collective in separate meetings to confirm theories and models in the SRS are correct. Any issues identified during this review will be noted down and new Issues on GitHub will be created. These issues will have top priority and will be completed in the next possible Sprint by one member of the HubListener team. This review will be followed up by additional reviews by Dr. Spencer Smith and Dr. Anand (if he wishes). Again, any issues identified by these reviewers will be recorded through the issues tracker on GitHub. A focus of these reviews will be to verify the functional and non-functional requirements of correctness and understandability by identifying information from the SRS that is incorrect or ambiguous.

4.3 Design Verification Plan

The design of HubListener will be outlined in the Module Guide (MG) and Module Interface Specification (MIS) documents. The design will be verified by review of these documents. Dr. Smith will review this documents. To verify correctness, part of this review til be to ensure that every module traces to a requirement and that every requirement is traced to by a module. Any issues identified during this review will be noted down and new Issues on GitHub will be created. These issues will have top priority and will be completed in the next possible Sprint by one member of the HubListener team. Reviews of these design documents will also focus on ensuring

the understandability by identifying descriptions and specifications that are ambiguous.

4.4 Implementation Verification Plan

The implementation of HubListener will be verified by review and by testing. The HubListener team will extensively review the implementation. This will be done via bi-weekly code-reviews. In the context of agile methodologies, every sprint, each member will be assigned code from a peers to review. This code is typically from the previous sprint. If the code passes code review, it will be merged into the master-branch. Any issues identified during this review will be noted down and new Issues on GitHub will be created. These issues will have top priority and will be completed in the next possible Sprint by one member of the HubListener team. These reviews will contribute to verifying correctness and understandability of the software by identifying code that is not traceable to any specifications described in the SRS , MG or MIS.

The implementation will also be verified through testing. Specific test cases are outline in Section 5 of this document. Test cases that are directly dependent on implementation details will be outlined at a later date. All tests will be written(where applicable), reviewed and executed by the HubListener Team.

5 System Test Description

In this section, we will describe and define the tests for functional and non-functional requirements as well as for HubListener's design module.

5.1 Tests for Functional Requirements

This section will describe the test cases that will be used for testing the functional requirements.

5.1.1 Valid User Input

The following set of test cases are intended to cover different forms of valid user input

1. test-id1

   Control: Manual versus Automatic

   Initial State:

   Input:

   Output: [The expected result for the given inputs —SS]

   Test Case Derivation: [Justify the expected value given in the Output field —SS]

   How test will be performed:

5.1.2 Requirements Testing

The following set of test cases are intended to cover testing of each functional requirement

Requirement 1

1. test-id1

   Control: Manual versus Automatic

   Initial State:

   Input:

   Output: [The expected result for the given inputs —SS]

   Test Case Derivation: [Justify the expected value given in the Output field —SS]

   How test will be performed:

Requirement 2

1. test-id1

   Control: Manual versus Automatic

   Initial State:

Input:

Output: [The expected result for the given inputs —SS]

Test Case Derivation: [Justify the expected value given in the Output field —SS]

How test will be performed:

Requirement 3 ...

5.1.3 Error Handling Testing

The following set of test cases are intended to cover error handling situations.

1. test-id1

   Control: Manual versus Automatic

   Initial State:

   Input:

   Output: [The expected result for the given inputs —SS]

   Test Case Derivation: [Justify the expected value given in the Output field —SS]

   How test will be performed:

5.2 Nonfunctional Requirements Evaluation

5.2.1 Usability Testing

In this section, we will explore the manual test cases needed for testing usability.

1. test-id1

   Control: Manual versus Automatic

   Initial State:

   Input:

   Output: [The expected result for the given inputs —SS]

Test Case Derivation: [Justify the expected value given in the Output field —SS]

How test will be performed:

5.2.2 Maintainability Testing

In this section, we will explore the manual test cases needed for testing the maintainability of HubListener.

1. test-id1

   Control: Manual versus Automatic

   Initial State:

   Input:

   Output: [The expected result for the given inputs —SS]

   Test Case Derivation: [Justify the expected value given in the Output field —SS]

   How test will be performed:

5.2.3 Performance Measurement

In this section, we will explore the manual test cases needed for testing the performance of HubListener.

1. test-id1

   Control: Manual versus Automatic

   Initial State:

   Input:

   Output: [The expected result for the given inputs —SS]

   Test Case Derivation: [Justify the expected value given in the Output field —SS]

   How test will be performed:

5.3 Trace to Requirements

Will include matrix here. The purpose of the traceability matrix is provide easy reference on which requirement are verified by which test cases,and which test cases need to be updated if a requirement changes.

5.4 Trace to Modules

In this section, a matrix will be included to trace which modules are verified by which test cases, and which test cases need to be updated

6 Unit Testing

Unit Testing will be completed using either Mocha.js or Jasmine.js . Both tools provide the resources required to get thorough unit testing complete. We are still in the research phase and not have decided on a set tool. Jasmine.js remains the front-runner as one of members on the HubListener team has experience with this tool.

7 Automated Testing

Automated testing will further enhance our testing efforts. We are investigating Travis-CI which is a free web based service that allows to register a trigger on GitHub so that every time a commit is pushed to GitHub an isolated Ubuntu container with the correct container that we want to test, builds the software (if needed) and then runs the test.

8 Appendix

This is where you can place additional information.

8.1 Usability Survey Questions

Here is where the Usability Survey Questions will go.