Kai Tindall

k.tindall-2018@hull.ac.uk

Abstract

This document will outline the timelines and procedures used in the project to try to ensure success.

Software Development Plan

Security in Internet of Things Devices

Table of Contents

[1.1 Revision History 2](#_Toc57039938)

[1.2 Document References 2](#_Toc57039939)

[1.3 Acronyms 2](#_Toc57039940)

[1.4 Glossary 2](#_Toc57039941)

[2 Software Development Activities 3](#_Toc57039942)

[2.1 Requirements Engineering 3](#_Toc57039943)

[2.2 Software Design and Implementation 3](#_Toc57039944)

[2.3 Software Verification and Validation 3](#_Toc57039945)

[2.4 Software Maintenance 3](#_Toc57039946)

## Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version Number** | **Date** | **Author** | **Comment** |
| 1.0 | 23/11/2020 | Kai Tindall | Initial draft of the document |
|  |  |  |  |
|  |  |  |  |

## Document References

|  |  |  |  |
| --- | --- | --- | --- |
| **Identification Number** | **Description** | **Version** | **Reference Number** |
| SITD-0001 | The process initiation document (PID). Outlines the project in broad terms. | 1.1 | 1 |
|  |  |  |  |
|  |  |  |  |

## Acronyms

|  |  |
| --- | --- |
| **Acronym** | **Definition** |
| PID | Process Initiation Document |
| UML | Unified Modelling Language |
| SRS | Software Requirements Specification |
| API | Application Programming Interface |
| JRE | Java Runtime Environment |

## Glossary

|  |  |
| --- | --- |
| **Word** | **Definition** |
| Pseudocode | Code that is easy to understand but is not actually written in a real language. Normally written to convey the semantics of a design rather than any syntax. |
|  |  |
|  |  |

# Software Development Activities

The project will use a list of activities defined by (Elgabry, 2017).

## Requirements Engineering

This is the process of creating a list of statements that outline what is expected of the project. These include statements that relate to goals (primary, secondary, and “nice to haves”) and legislation that legally must be met.

There will be a “Software Requirements Specification Document” produced by the project that will outline all of these and will act as the main document that any testing will reference to be testing against. This document will be validated against section 2 of the PID [1].

## Software Design and Implementation

The design activity is where the project will be modelled in a combination of pseudocode and UML that describe the software that will fulfil the requirements specified in the SRS. There will be absolutely no implementation details within the design as it is not good practise. The design and implementation of the software should be separate. This activity will culminate in the creation of a software design document.

As a specific activity within the software design activity for this project will be the design of the API that will be used to expose the communications between the client and the command and control server.

Implementation will aim to take the design that has been produced and turn it into working C++ code. If the implementation is true to the design and the design fulfils the requirements, then by the time the design has been fully implemented the software should reflect the requirements. This activity will produce the C++ code.

## Software Verification and Validation

Verification is the process of checking that the software meets the specification laid out in the SRS. The project will aim to achieve this by using a combination of automated and manual testing. Automated unit and integration testing will be employed as a way of constantly checking code is working as it is intended to be; however, the main verification will be completed by running through manual system testing that will be defined in a software testing document.

Validation is the process of checking the project against the needs of the stakeholders and the goals of the project. This will be done continuously by analysing the codebase at each minor review for diversion from the project goals.

## Software Maintenance

Software maintenance is about the changing of requirements as well as fixing bugs. The former shall be handled with an agile project management style that will allow for requirements to change midway through the project without causing too many problems. Each major review will also assess if any requirements have changed and make the proper arrangements if it is so.

The process of raising and tracking bugs will take place through GitHub issues. This will allow for a central place where the bugs will be documented. Through using this the project members will be able to quickly see if there are any bugs that need to be dealt with.

# Software development tools

## Workstation

The workstation for the project will be a windows PC that has VirtualBox installed with a virtual machine using a Debian 10 (buster) image created. The JRE must be installed on both the Windows and Debian machines.

## Requirements management and documentation

## Software Design

The tool that will be used for software design is ArgoUML. ArgoUML is an open source UML diagramming application. It is written in Java which means it can be opened on any device that has the JRE installed. This is useful to the project as it means the design can be opened on whichever the developer is currently using and doesn’t have to switch between them.

## Coding and automated tests

## Configuration management

## Development rules and standards