Plan for Software Aspects of Certification

For the Linphone group chat project (Agile approach)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

Patience Mtsweni, Date

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

Lerato Molokomme, Date

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

Tsepo Ntsaba, Date

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

Mpedi Mello, Date

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

Lutfiyya Razak, Date

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

Ephiphania Munava, Date

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

Kobus Coetzee, Client Date

# Contents

[1 Contents 2](#_Toc419923063)

[2 Change control 3](#_Toc419923064)

[3 Introduction 3](#_Toc419923065)

[3.1 Purpose 3](#_Toc419923066)

[3.2 Scope 3](#_Toc419923067)

[3.3 Documentation conventions 4](#_Toc419923068)

[4 System overview 5](#_Toc419923069)

[4.1 Server Architecture 5](#_Toc419923070)

[5 Software overview 5](#_Toc419923071)

[5.1 Software architecture 5](#_Toc419923072)

[6 Certification considerations 5](#_Toc419923073)

[7 Software lifecycle 5](#_Toc419923074)

[7.1 Scrum 6](#_Toc419923075)

[7.2 Continuous integration (CI) and Continuous deployment (CD) 7](#_Toc419923076)

[7.3 Test driven development 8](#_Toc419923077)

[8 Software development environment 9](#_Toc419923078)

[9 Software lifecycle data 9](#_Toc419923079)

[10 Schedule / Software development plan 10](#_Toc419923080)

[11 Licencing and ownership 10](#_Toc419923081)

# Change control

|  |  |  |
| --- | --- | --- |
| **Author** | **Reason for change** | **Date** |
| KC | Document creation | 2015/05/20 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Introduction

## Purpose

The purpose of this project is to extend Linphone's Instant Messaging (IM) implementation on Android platforms to include group chat and to implement other minor improvements to Linphones IM capabilities and user interface.

This project will give the student group exposure to:

* Android development (Java and C)
* Open source contribution
* Cryptography
* Session initiation protocol (SIP)
* DO-178 certification

This project also forms part of a larger Masters study on development methodologies for projects seeking DO-178 certification.

## Scope

The students are required to develop the following functionality for the Linphone project

* Group chat (Invite additional members to a chat, all members receive chats)
* Secure group chat (AES256)
  + A basic message encryption implementation will be provided
* Creation and deletion of groups
* Voice record and send over IM
* Rework the messaging user interface
  + Spacing between words are terrible
  + Make the text bigger
  + Block indents required to better specify who said what
  + Presence indication to show a remote user is typing
  + User picture portraits

## Documentation conventions

DO-178B does not specify the documentation standard to be followed, but most projects do follow some or other documentation standard. The following figure is loosely based on [MIL-STD-490A](http://en.wikipedia.org/wiki/MIL-STD-498), although sometimes “Detail design” and “Notes” are changed into some other topic of discussion.

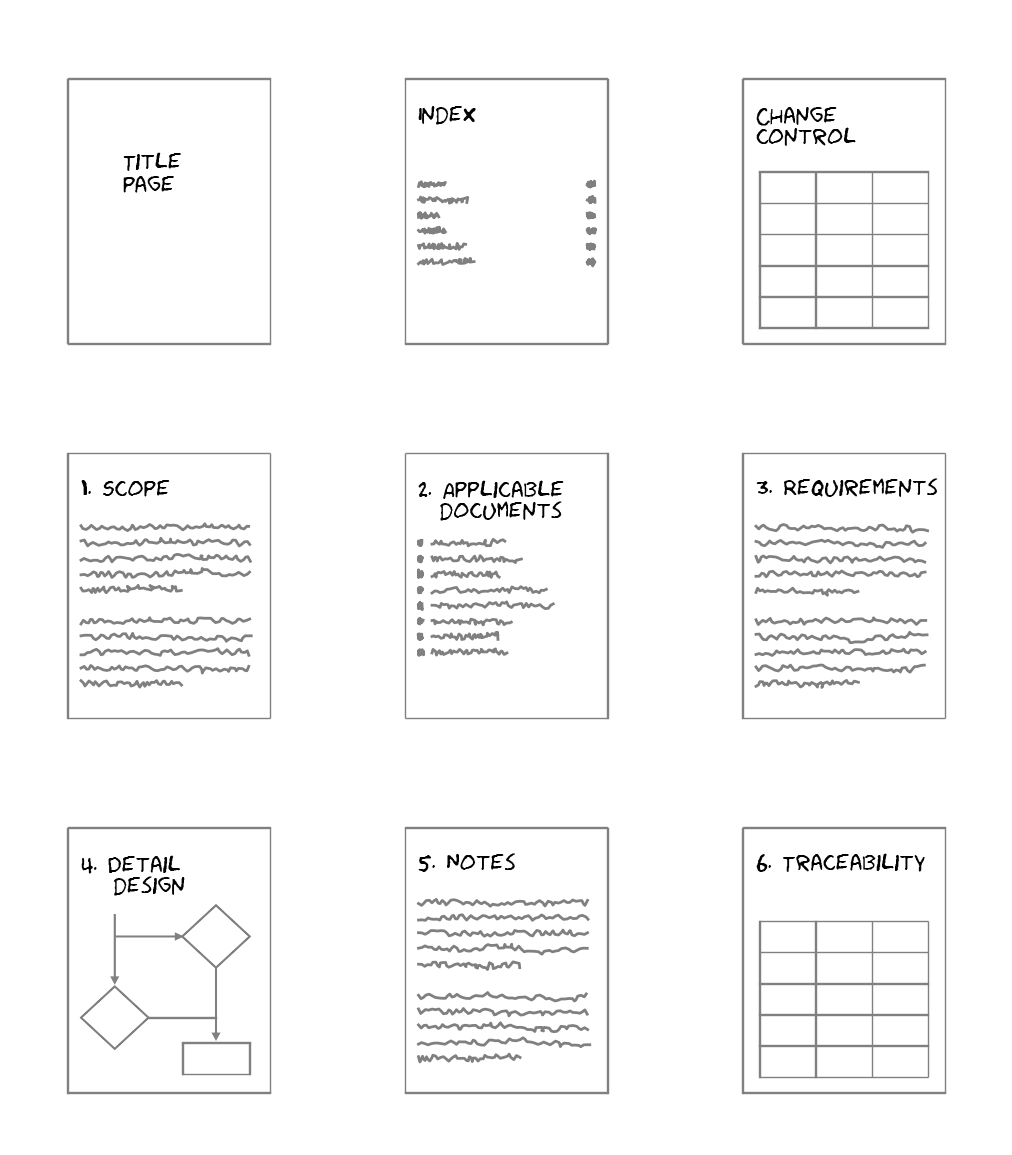


Figure : Documentation convention

In the documentation, especially when talking about requirements and specifications, certain words convey additional meaning apart from their linguistic use. These words are usually capitalized.

**SHALL** and **SHALL NOT** - Indicates a mandatory requirement.

**WILL** and **WILL NOT** - Indicates a declaration of purpose or an expression of simple futurity.

**SHOULD** and **SHOULD NOT** - Indicates a non-mandatory desire, preference or recommendation.

**MAY** and **MAY NOT** - Indicates a non-mandatory suggestion or permission.

**MUST** and **MUST NOT** should be avoided as it causes confusion with the above terms.

# System overview

## Server Architecture

# Software overview

## Software architecture

# Certification considerations

The project will be developed according to DO-178 certification requirements, for level D certification. Justification for level D is not dependant on safety considerations as is usually the case, but rather on limited time and the limited experience the team has with DO-178.

This project is not intended for avionics applications and is not safety critical, but rather serves as a case study on DO-178 software development.

# Software lifecycle

An agile software development methodology will be followed with this project:

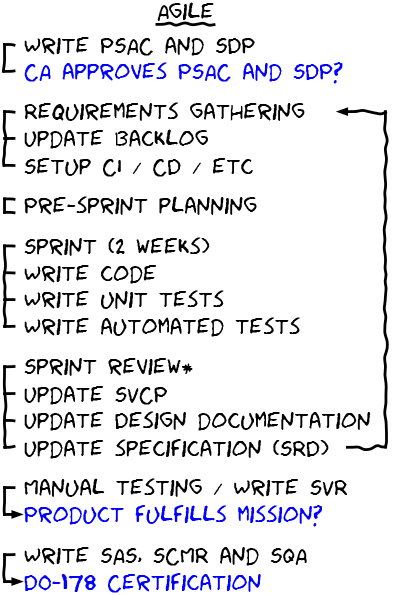


Figure : Agile development

## Scrum

During sprint planning and review sessions, we can review and update the Design Description (DD) document detailing the software architecture. At the end of a sprint, the implemented user stories will form the Software Requirements Data (SRD’s). It will look something like this…

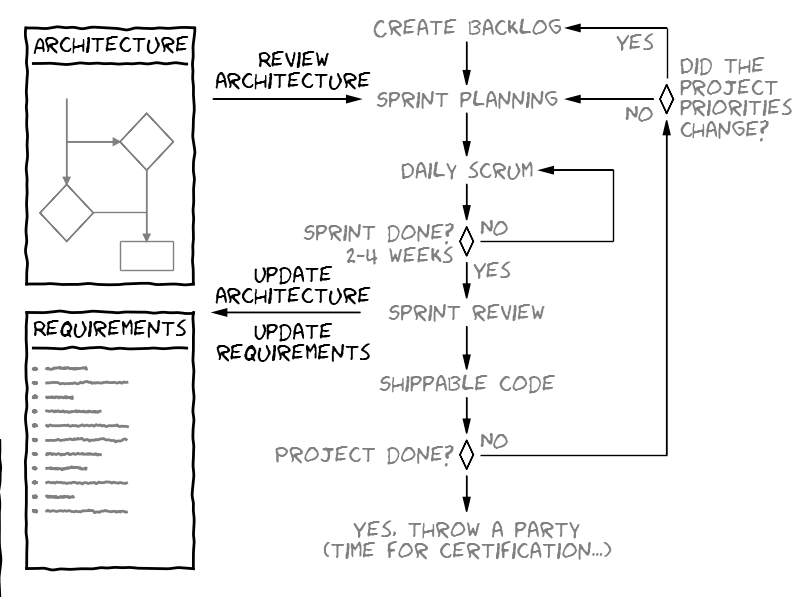


Figure : Scrum outputs

During sprint planning we ensure that the user stories i.e. the high level requirements we are planning to implement is consistent with previous implemented requirements. During sprint review we update the requirements with the implemented user stories, and ensure the created functional tests and unit tests i.e. the low level requirements are consistent with the high level requirements (user stories).

## Continuous integration (CI) and Continuous deployment (CD)

The CI and CD servers themselves are effectively the Software Life Cycle Environment Configuration Index (SECI), Software Configuration Index (SCI) and parts of the Sofware Configuration Management Records (SCMR) deliverables. For this to be possible, the CI server must have a copy of the version control database when duplicated for certification.

This means an agile setup might look as follows:

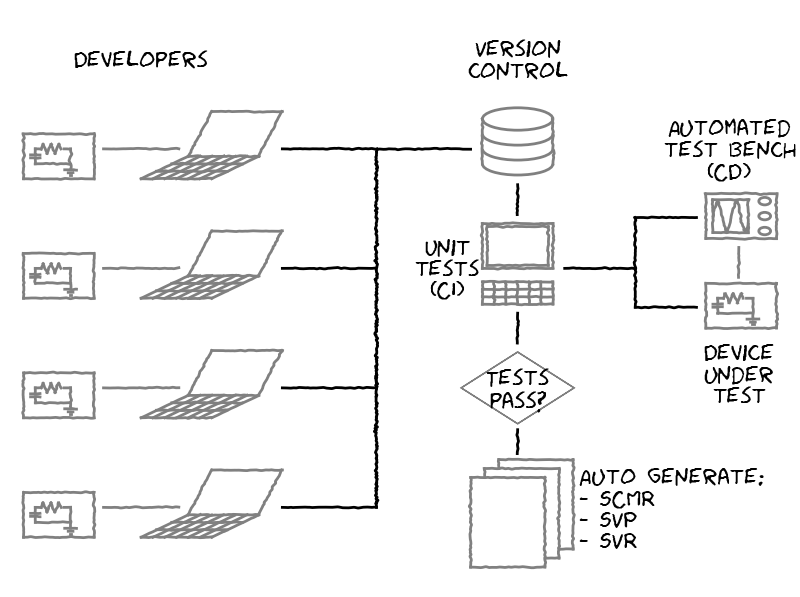


Figure : CI and CD outputs

If all the tests pass, the CI / CD will autogenerate a snapshot of itself (VM’s or some other duplication means) and the version control database to serve as the SECI and SCI. It will also generate reports of the tests run and their results to serve as the SVP and SVR’s, and generate reports that can serve as SCMR items (This is baselines, commit histories etc.).

## Test driven development

Test driven development can be used to generate the large parts of the Software Verification Cases and Procedures (SVCP’s) and Software Verification Results (SVR’s). High level requirements will be developed and tested with feature driven development, and unit tests will be used to develop and test low level requirements. But not all unit tests are really low level requirements, for instance testing if a function can handle null pointer parameters. As such we will mark which unit tests are indeed low level requirements in the unit testing code itself.

The relationship between the Continuous integration (CI) server and the Continuous deployment (CD) server is detailed in the popular test pyramid developed by Mike Cohn, where the CI is responsible for making sure the source code compiles at all times, and passes all unit tests, and the CD is responsible for making sure the automated functional tests pass at all times. One would expect to develop a lot more unit tests than functional tests, thereby limiting (but not eliminating) the need for expensive manual testing.

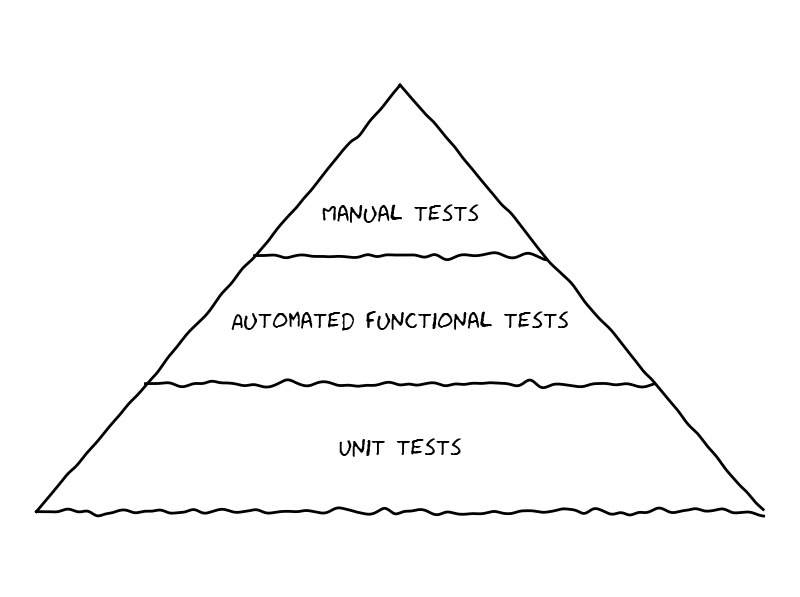


Figure : Testing triangle

# Software development environment

# Software lifecycle data

The following deliverables will be created during this project:

* **Plan for Software Aspects of Certification (PSAC)**

The Plan for Software Aspects of Certification is the primary means used by the certification authority for determining whether an applicant is proposing a software life cycle that is commensurate with the rigor required for the level of software being developed.

* **Software Development Plan (SDP)**

The Software Development Plan includes the objectives, standards and software life cycle(s) to be used in the software development processes.

* **Software Verification Plan (SVP)**

The Software Verification Plan is a description of the verification procedures to satisfy the software verification process objectives.

* **Software Requirements Data (SRD)**

Software Requirements Data is a definition of the high-level requirements including the derived requirements.

* **Software Design Description (SDD)**

The Design Description is a definition of the software architecture and the low-level requirements that will satisfy the software high-level requirements.

* **Source Code**

This data consists of code written in source language(s) and the compiler instructions for generating the object code from the Source Code, and linking and loading data. This data should include the software identification, including the name and date of revision and/or version, as applicable.

* **Executable Object Code**

The Executable Object Code consists of a form of Source Code that is directly usable by the central processing unit of the target computer and is, therefore, the software that is loaded into the hardware or system.

* **Software Verification Cases and Procedures (SVCP)**

Software Verification Cases and Procedures detail how the software verification process activities are implemented.

* **Software Verification Results (SVR)**

The Software Verification Results are produced by the software verification process activities.

* **Software Accomplishment Summary (SAS)**

The Software Accomplishment Summary is the primary data item for showing compliance with the Plan for Software Aspects of Certification.

# Schedule / Software development plan

# Licencing and ownership

The aim is to open-source the project which would be published on Github as forked from Linphone project. To this end the required permissions will be sought from the University of Pretoria.