SQAP (Software Quality Assurance Plan)

**Project: Awesome Alphabet**

**Professor: Yuting Zhang, Boston Univeristy**

**People: Jaleel Kazi, Levi Paul, Mark Musante, Michael Grant, Vivek Goyal**

# Purpose

This document describes the plan by which the Awesome Alphabet project will produce a quality product. This includes the maintainability of Awesome Alphabet.

# Referenced Documents

* + [Braude] The principal source of textbook reference material is "Software Engineering: an Object-Oriented Perspective" by E. Braude (Wiley, 2000). This document is based on the template used for the *Encounter* project in that book.
  + Also see section 4.b

# Management

## Organization

Each team member is responsible for the quality of his or her work. In addition, for the first three iterations of Awesome Alphabet, an individual "quality assurance leader" is designated. Within the priorities set by the team, the QA leader will take the lead for project-wide quality issues. After iteration three, the SQA organization will provide a team of engineers to perform this function. The team will include the existing quality assurance leader.

## Tasks

QA tasks shall include

* + - Documentation
    - Verification (including inspections)
    - Validation (mostly testing)
    - Activities designed to improve quality assurance process itself.

## Responsibilities

It is the quality assurance leader's responsibility to see to it that the tasks in 3.b are done, and to ensure that the prescriptions in this document are followed, including scheduling the reviews specified. Vivek Goyal will be the quality assurance leader.

The project leader will be responsible for ensuring that quality management is being performed.

The requirements leader and the design leader have responsibilities described in section 6 of this document.

# Documentation

## Purpose

The purpose of this section is to define the documentation that will be used to ensure quality

## Minimum Documentation Requirement

The following documents will be produced.

* + - SQAP: software quality assurance plan (this document)
    - SCMP: software configuration management plan
    - SPMP: software project management plan
    - SRS: software requirements specifications
    - SDD: software design document
    - STD: software test documentation
    - User's manual
    - Maintenance plan
    - Software validation and verification plan (SVVP)

In addition to these documents, the Java source code will utilize Javadoc, and will therefore be capable of generating package-, class-, and function-level documentation.

# Standards, practices, conventions and metrics

## Purpose

This section describes the standards, practices, conventions and metrics to be used for the Awesome Alphabet project. These are intended not only to ensure quality for Awesome Alphabet, but also to obtain quantitative metric data on the SQA process itself.

## Content

The following metrics will be maintained:

* + - Time spent by individuals on tasks.
    - Measure size of code in KLOC. Count number of KLOC at the end of project.
    - Quality Level (#defects per KLOC). At the end of project we can count number of KLOC and number of defects raised in GitHub issue tracker and come up with this number.
    - Project Effort (# of person months). We can go through weekly time sheet of each person, add it up and come up with # of person months consumed by the project.
    - Defect Repair Rate. We should be able to come up with this rate by determining when did we start fixing issues and divide number of issues by number of weeks.

The following coding guidelines will be used for this project:

<http://www.oracle.com/technetwork/java/codeconv-138413.html>

# Review and Audits

Formal technical review will be conducted for most effective software quality control. The objective of this review is to uncover errors in the stage of development process before the Changes are submitted to the Main source control branch. Each developer has his own development branch in which most the code is developed. When its time to commit the code changes to the main branch, a formal review either via code review meeting or via email will be conducted. Review is a combination of assessment methods of walkthrough, inspection, round robin review. Each developer does complete unit testing and publishes the work and the other team member tests out the functionality.

## Purpose

The purpose of reviews and audits is to provide a means of focusing the attention of engineers on quality of the application as it develops. Reviews carry this out in a scheduled and thorough manner. Audits do so on the basis of random sampling with short notice.

## Minimum Requirements

### Software requirements review

This is a walkthrough of the requirements document in the presence of the entire team. The review will be lead by the project leader. It is expected that the requirements will not have been inspected prior to this review. This review is not intended to replace inspections of the requirements. The requirements leader (see SPMP) will be responsible for seeing to it that these inspections are carried out.

### Design Review

This is review and inspection of design with entire team.

### Code Review

We will perform peer code review. The minimum requirement is that at least one peer reviews the code before the end of iteration.

## Inspections

All artifacts of the Awesome Alphabet project will be inspected

# Test

Individual developers will perform unit testing and the QA team will do the functional black box

testing to make sure customer requirements are met. Individual developers will perform unit

testing and the QA team will do the functional black box testing to make sure customer

requirements are met.

Testing activities are carried out during the entire development cycle, the effective and through

testing detects errors at the early stage and prevents it pass to the next stage. The new defects

are maintained in a separate list (bug tracker) and reviewed to be fixed before the iteration.

Documentation records are maintained and the work product of every iteration during the

development process, and the keep track of the required specification. Any change in the

project scope will result in modification of the documentation. They will be kept updated all the

time.

# Problem Reporting And Corrective Action

Issue tracker provided by GitHub will be used for reporting and keeping track of issues. Issues will be categorized by priority and severity.

# Code Control

GitHub will be used for committing the code. Before the testing phase, development will tag a particular release of code with a version number or label. QA will test the code and report any issues.

<https://drive.google.com/?tab=mo&authuser=0#activity>

# Risk Management

Refer to SPMP - section 3.3

<https://docs.google.com/document/d/13U4gFwoOZ8JL4_2qrHle30kB0cW8hILY-tCepgTuLhA/edit>

# Acronyms

KLOC - Kilo lines of code