Software Quality Assurance Plan for Java Air

IEEE 730-2002

Team: Avian Limited

Prepared by: Matthew D Moscatel and Xu Wu

Table of Contents

1. Purpose

This document describes the means by which the Java Air project will produce and maintain a high quality product. The Java Air project is Java-based software that simulates the management software of an airliner. It is also intended to describe mechanisms for improving the quality assurance process itself. The scope of this document comprises the artifacts of all releases.

2. Reference documents

2.1. Software Engineering Modern Approaches, 2nd ed. By Eric J. Braude, Michael E. Bernstein.
2.2. highered.mheducation.com/sites/dl/free/0073375977/673802/chapter16.doc
2.3. IEEE Standard 730-2002, IEEE Standard for Software Quality Assurance Plans.

3. Management

3.1 Organization

This tool is developed as a team project as part of CS-59000 'Software Engineering' course at Purdue Northwest Calumet Campus. A team of 7 graduate students is involved, it will be a collaboration to review the product's quality, including usability, efficiency, reliability, and accuracy, etc. The major professor will however give guidance to students on the project.

The quality assurance team (testing team) with two members, Xu Wu and Matthew Moscatel, is responsible for preparing and maintain the Software Quality Assurance Plan, potentially with the help from other team members.

The project leader, with the help from the quality assurance team (testing team) and the rest of the team members, will evaluate and monitor the software quality together. Thus, the quality assurance team has full freedom to control the quality. While the project team is inspecting its own product, as one team, it does not totally separate members into the software development team and the software testing team. The result would be better.

3.2 Tasks

The SQAP document covers the whole software lifecycle, including requirement engineering, design, implementation, testing, deployment and maintenance.

The following tasks shall be executed for Software Quality Assurance purposes. To access the list of complete tasks involved in the whole software lifecycle, refer to the Software Project Management Plan document.

a. Prepare the SQAP document.
The SQAP document shall be written in the standard format defined in IEEE Std 730-2002.
Entry Criteria: The Software Quality Assurance Role is taken by some team member(s).
Exit Criteria: Version 1.3 of SQAP document is reviewed and approved.

b. Prepare the Verification & Validation Plan document.
The V&V document shall be written in the standard format defined in IEEE Std. 1012-2004.
Entry Criteria: SQAP document task is completed.
Exit Criteria: Version 1.0 of V&V Plan document is reviewed and approved.

c. Prepare the Software Testing Description document.
The STD document shall be written in the standard format defined in IEEE Std. 829-1998.
Entry Criteria: SQAP document task is completed; V&V Plan document task is completed.
Exit Criteria: Version 1.1 of STD document is reviewed and approved.

d. Implement Unit Tests, Integration Tests, System Tests and Acceptance Tests.
The tests shall be implemented in chosen programming language and frameworks such that they can be easily executed against the software product code.
Entry Criteria: STD document task is completed.
Exit Criteria: All tests are implemented in code.

e. Execute tests, collect and report results.
Ideally this task should be automated with the help from Continuous Integration System.
Entry Criteria: Other members from the project team request to validate the quality of the current implemented software or to get a list of unimplemented features / attributes. (Test-Driven Development)
Exit Criteria: Test Report is delivered to requesting parties.

f. Communicate with software development team to correct test problems.
Tests can suffer from certain problems such as bugs and mismatching test cases, etc. That needs to be corrected.

Entry Criteria: Software Development Team questions the correctness of tests.
Exit Criteria: Software Development Team reviews and agrees with the updated tests.

g. Communicate with software development team to correct faults.
Faults may include code style, design quality, functional bug, and non-functional issues such as performance issues, security issues, reliability issues, etc.
Entry Criteria: Software Development Team questions the test report.
Exit Criteria: Software Development Team successfully fixed the faults and pass tests.

3.3 Roles and responsibilities

Testing Engineer (TE) is the major role involved in the Java Air project software quality assurance processes. The TEs are responsible for preparing and maintaining documents, code implementations and reports, as well as to communicate with the other members from the project team to provide feedbacks. For a list of detailed tasks, see Section 3.2.

3.4 QA assurance estimated resources

Two Test Engineers will be allocated to the quality assurance tasks. Each engineer devotes 6 hours each week to the project. The project is estimated to be completed in 13 weeks. The total cost of QA is 2*6*13 = 156 man*hours.

4 Documentation

In addition to this document, the essential documentation will include:

- The Software Project Management Plan (SPMP)
- The Software Quality Assurance Plan (SQAP)
- The Software Configuration Management Plan (SCMP)
- The Software Requirement Specification (SRS)
- The Software Design Description (SDD)
- The Software Verification & Validation Plan (V&V)
- The Software Test Description (STD)
- The Software Integration Plan (SIP)
- The User Documentation (User Manual)
- The Source Code

5 Standards, practices, conventions, and metrics

5.1 Documentation Standards

All documents shall be written in the standard IEEE formats in technical English.

All textual documents shall be saved in Microsoft Word format (*.docx).

Documentation Metrics

Word Size (Number of word)

5.2 Design Standards

All design shall be done using industry standard technologies including:

- Unified Modeling Language
- Entity Relationship Diagram
- Data Flow Diagram

All graphical diagrams shall be exported into the .jpg format to facilitate viewing.

Design Metrics

Number of classes in the UML class diagrams

5.3 Coding Standards

- All codes shall be written in Java 1.8.

Coding Metrics

- Lines of Code
- Average number of defects per unit (line of code)

5.4 Commentary Standards

- Javadoc Comments

Commentary Metrics

Javadoc Comment Coverage Rate (Number of classes with missing javadoc comment / Total number of classes)

5.5 Testing Standards

- JUnit Test Framework
- Testing Metrics
    - Number of code files (.java) with unit tests / Number of all code files (.java).
    - Number of Use Cases with system tests / Number of all use cases.
    - Number of unit tests passed / Number of all unit tests.
    - Number of system tests passed / Number of all system tests.

All the above standards shall be enforced by either tools or human engineers. The above metrics shall be continuously measured, and the measurement results shall be accumulated in a data store which shall be periodically reviewed for the lessons learned based on the management schedule.

6. Software reviews

The following contents shall be reviewed after proposed:
6.1 Software Specifications Review (SSR)
6.2 Architecture Design Review (ADR)
6.3 Detailed Design Review (DDR)
6.4 Verification & Validation Plan Review
6.5 Functional Audit
6.6 Physical Audit
6.7 In-process Audits
6.8 Managerial Reviews
6.9 Software Configuration Management Plan Review (SCMPR)
6.10 Post-implementation review
6.11 Other reviews and audits: User Document Review

The schedule of the reviews and audits are related to the project management plan and should be defined with the help from the Project Manager.

Refer to Section 8 for problem handling process.

7. Test

Reference Software Test Documentation

8. Problem reporting & corrective action

When the required proposed documents or code are ready they shall be uploaded to the team-shared Google Drive or Git repository if it’s code so the quality assurance engineers can access them. The quality assurance engineers review the documents to find potential problems and / or offer comments, potentially with the help from other engineer(s) who are more experienced in the related technical topic.

When the reviews and audits revealed problems, the quality assurance team shall document and problem including the date, reviewer / auditor names, descriptions of the problem, name and version of the artifacts where problems were discovered and severity level of the problem(s). When the problems are reported as fixed, the quality assurance team shall verify the fix and document the verification date, result, and verifying engineers' names. All reviewers and auditors shall sign off the documents after approving them.

9. Tools, techniques and methodologies

- JUnit Test Framework shall be used as the unit-testing framework. Although Continuous Integration Systems can be a great help in automated testing, the cost of learning and configuration it is simply too much to worth it for a project of this size. Manual testing is accepted for this project.
- Mock Objects and / or Stubs shall be used for unit-testing.
  Although some frameworks can automatically generate mock objects and stubs, for the same reason the compromise is made and the mock objects / stubs are manually created.
- Google Sheet shall be used for storing review / auditing results, metrics and serve as reporting.
  Although some tools such as Crystal Report and MS Project have much richer functionalities for saving structured data and generating reports, for the same reason they are not used in this project.

10. Media control

Due to the nature of the project, the security requirement of the media control is relatively low. As long as the digital files are properly version controlled, they can be created, stored and shared either on local computer hard drives, cloud-based drives (e.g. Google Drive), or portable drives (e.g. USB). All artifacts created for the project shall have a backup copy at a cloud-based drive.

11. Supplier control

The SRS shall be reviewed by both the client (the professor) and the suppliers (the student team) and both shall approve the document by signing their names.

The acceptance tests shall be reviewed and passed in order to guarantee the product's functionality does meet the original expectations.

12. Records collection, maintenance and retention

Due to the nature of the project, the management of this document is relatively relaxed. The document shall be stored in cloud drive and grant ‘read’ access to all other team members. Only the quality assurance engineers shall have the ‘update’ permission to the document. Only the project manager shall have the ‘delete’ permission to the document.

13. Training

All of the project team members (students) have the required knowledge / skills and thus there is no need for training.

14. Risk management

Risks may be identified by any team members during any phases in the software lifecycle. Once identified, the identifier shall report the risk(s) to both the quality assurance engineers and the project manager. The QA engineers shall document the risks including the date of identification, identifier name, description of the risk and severity level of the risk. The project manager shall assemble the right skillset to analyze the risk and either conquer it or avoid it. No matter what solutions are applied, the QA Engineers shall document the resolution, including the way of resolution, date and the validation results (if possible).

15. Glossary

Term	Abbreviation	Definition
Software Quality Assurance Plan	SQAP	A software engineering document that consists of a means of monitoring the software engineering processes and methods used to ensure quality.
Software Verification & Validation Plan Document	V&V	A software engineering document containing the process of checking that a software system meets specifications and that it fulfills its intended purpose.
Software Project Management Plan	SPMP	A software engineering document in

		which software projects are planned, implemented, monitored and controlled.
Software Design Description	SDD	A software engineering document that a software designer writes in order to give a software development team overall guidance to the architecture of the software project.
Software Test Description	STD	A software engineering document that contains the schedule and design of testing methodologies for software.
Software Requirement Specification	SRS	A software engineering document that lays out functional and non-functional requirements, and may include a set of use cases that describe user interactions that the software must provide.
Software Configuration Management Plan	SCMP	A software engineering document for establishing and maintaining consistency of a product's performance, functional, and physical attributes with its requirements, design, and operational information throughout its life.
Unified Modeling Language	UML	A general-purpose, developmental, modeling language in the field of software engineering, that is, intended to provide a standard way to visualize the

		design of a system.
Entity Relationship Diagram	ERD	Describe inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types.
Data Flow Diagram	DFD	A graphical representation of the "flow" of data through an information system, modelling its process aspects.
Continuous Integration	CI	A development practice that requires developers to integrate code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early.

16. SQAP change procedure and history

Version	Date	Modifier	Changes
1.0	10/1/2016	Xu Wu	Initial
1.1	10/21/2016	Xu Wu	1. Revised section 3.1 2. Revised section 3.4 3. Revised section 3.2(a) 4. Add a new task in section 3.2(f)
1.2	11/13/2016	Xu Wu	Revised format
1.3	11/15/2016	Xu Wu	Revised section 5.3
1.4	12/04/2016	Xu Wu	Revised section 5.5