SOFTWARE QUALITY ASSURANCE PROCESS
PROJECT 2 : 6502 Debugger

Edited by

Frazer Bayley
Haley Whitman
Abdulaziz Al-Heidous
Alison Legge
Jeremy Brennan

Project 2 - Team 3

Contents

Revision History

Revision	Date	Author(s)	Description
.1	13.02.17	Alison Legge	Created initial outline of document
.2	14.02.17	Alison Legge	Filled out sections 1 and 2
.3	15.02.17	Alison Legge	Filled out sections 3, 4, and 5
.4	16.02.17	Alison Legge	Finished out draft
.5	19.02.17	Alison Legge	Added Software Architecture and Design to work products

1 Introduction

1.1 Purpose

The purpose of this document is to specify how the Software Quality Assurance Process (SQAP) will be handled in the software development life-cycle of the 6502 Debugger. These standards are derived from software requirements, architecture documents and conform to the requirements of the shareholders.

1.2 Scope

The primary audience for this document is the 6502 Debugger project team. The team members are to follow the quality standards set while developing the application, documenting the results, monitoring the project progress, and testing the quality of the project. All important aspects of software development are covered in the SQAP (i.e. requirements analysis, architecture and design, implementation, testing and verification, and user acceptance).

1.3 Background and Context

The 6502 Processor and its derivatives powered machines such as the Nintendo Entertainment System (NES), Atari 2600, and the Apple I microcomputer. It is currently still used in some embedded devices. The 6502 Processor features three 8-bit general purpose registers A, X, and Y; an 8-bit stack pointer; a 16-bit program counter; and 148 total instructions.

1.4 Project Objectives

The intended use of this software project is to emulate the functions of a 6502 processor directly on the users computer entirely through software. By eliminating the delays of transferring and testing code on a 6502 machine, this debugger can increase the speed of development processes.

1.5 Architectural Objectives

...

1.6 Technical Constraints

...

1.7 Project Management Constraints

The 6502 Debugger project is under a time constraint and must be completed in under six weeks by five team members. An absence of a team member or a missed deadline will affect the project schedule. To mitigate this risk, the team will be using an iterative software development process.

1.8 Requirements

The 6502 Debugger project requirements are documented in The Concept of Operations (ConOps) and the Software Requirements Specifications (SRS). The ConOps serves two purposes; it reflects

the needs and expectations of the customer and it works to explain the problem domain.

2 Referenced Documents

IEEE Std. 730-2014 IEEE Standard for Software Quality Assurance Processes.
6502.org The 6502 microprocessor online resource.
Effective Methods for Software Testing - William E. Perry Chapter 2 - Developing a Test Strategy

3 Documentation

3.1 Purpose

This section shall perform the following functions:

1. Identify the documentation governing the development, verification and validation, use, and maintenance of the software.
2. List which documents are to be reviewed or audited for adequacy. For each document listed, identify the reviews or audits to be conducted and the criteria by which adequacy is to be confirmed, with reference to section 5 of the SQAP.

3.2 Minimum Documentation Requirements

To ensure that the implementation of this software satisfies the technical requirements, the following documentation is required as a minimum.

3.2.1 Concept of Operations (ConOps)

The ConOps can be written by the customer, developer, or both and is intended for any and all stakeholders from the project. It is intended to be written in plain language and to be easily understood. It is meant to show understanding of requirements from the customer and that there is clear communication between developers and stakeholders. An active review process is used to ensure correctness and completeness of user requirements.

3.2.2 Software Requirements Specification (SRS)

The SRS is a formal, technical document written for software developers to answer specific technical questions about the requirements. The Software Requirements Specification review is used to check for adequacy and completeness of this documentation.

3.2.3 Software Test Reports

Software Test Reports communicate the results of executed test plans. A report should only contain the test information that pertains to the system aspect being tested.

3.2.4 Software Architecture and Design

Software Architecture and Design reviews are used for checking adequacy and completeness of the design documentation. This documentation should depict how the software will be structured to satisfy the requirements in the SRS. It should also describe the components and subcomponents of the software design.

3.2.5 User Documentation

User Documentation guides the user in installing, operating, managing, and maintaining software products. The user documentation should describe the data control inputs, input sequences, options, program limitations, and all other essential information for the software product. All error messages should be described.

3.2.6 Other Documents

- Developer Logs
- Software Project Management Plan SPMP

4 Goals

4.1 QA Goals of Each Phase

Phase	***Goals***
Planning	Plan for project around information architecture is developed. Standards and procedures are decided, test objectives are designed for requirement specifications, standards and procedures are audited.
Design	The QA plan and test plan are designed, revisions are made to the standards and procedures.
Development	The test cases are planned and the QA environment is setup.
Implementation	All test cases in the QA test plan are executed and all aspects of the system are reviewed.

5 Reviews and Audits

5.1 Work Product Reviews

The general Strategy for review is given below:

5.1.1 Formal Reviews:

1. One week prior to the release of the document, the QA team will review the document list generated by the team members on the project team.
2. The QA team will ensure that necessary revisions to documents have been made and the document will be released on time.

5.1.2 Informal Reviews:

A. **Design Walk-through**
QA will conduct design walk-through to encourage peer reviews of the design. The Project Manager will ensure all reviews are done properly and all results are recorded for reference.

B. **Code Walk-through**
QA will conduct code walk-through to ensure that peer review is conducted for all code. The Project Manager will ensure all reviews are done properly and all items have been addressed.

C. **Baseline Quality Reviews**
QA will review any document or code that is base lined. This will ensure:

(a) The testing and inspection of modules and code before release.

(b) Changes to software module design document have been recorded and made.

(c) Validation testing has been performed

(d) The functionality has been documented

(e) The design documentation conforms to the standards for the document as defined in the project plan.

5.1.3 Change Request Process:

Work Product	**How QA Reviewed**
Requirements	The Requirements document is reviewed and approved by assigned reviewer(s). The document is then presented to the customer for acceptance.
Software Architecture Design	The Design phase is carried out using an appropriate system design methodology, standards and guidelines, and taking into account the design experience from past projects. The design output is documented in a design document and is reviewed to ensure that: • The requirements as stated in the SRS are satisfied. • The acceptance criteria are met. • Appropriate information for service provision is provided. Acceptance for the design document is obtained from the customer.
Code	The project team codes the product to meet the design specifications using: • Suitable techniques, methodology, standards, and guidelines. • Reusable software components as appropriate.
Testing	Before release of the product, QA ensures all tests, reviews, approvals, and acceptances as presented in the project plan have been completed and documented.

5.2 Quality Assurance Progress Reviews

In order to remove defects from work items early and to prevent them from reoccurring, examination of software work items is conducted in the following fashion:

1. Reviews of all deliverables are carried out as stated in the Quality Plan of the project.
2. Reviews evaluate the ability of the intended product to meet customer requirements.
3. Personnel independent of the activity being tested carry out the reviews.
4. Reviews focus on the work being reviewed and not the developer.
5. The defects identified in the review are tracked to closure.

6 Tools and Techniques

The 6502 Debugger uses the following strategy for selection of the testing tool:

1. The testing tool is selected based on the core functionality of the project.
2. The usage of the tool is mapped to the life cycle phase in which the tool will be used.
3. Matching the tool selection criteria to the expertise of the QA team.

6.1 Tools and Techniques for Assuring Quality of Functional Requirements

In order to ensure requirements are being met, the project team has applied the following reviews:

1. **Peer review:** All artifacts are created and stored on a Google Drive. This allows all team members to review content online as well as provide comments. Each team member reviews specific sections that are then discussed in a team meeting. This review will help clarify the requirements and make sure everyone understands how the system should behave.

2. **Customer review:** After peer review, the requirements and documentation will be sent to the project mentor. The mentor is requested to review the document with a specific perspective as well as a customer's viewpoint. The mentor's feedback is discussed in a team meeting.

3. **Traceability checking:** Once requirements are documented and reviewed, a traceability matrix is developed for the requirements. The intended use for the matrix is to trace the source of any requirement as well as any changes.

4. **Regression Testing:** The objective of regression testing is to ensure all aspects of the application work after testing.

6.2 Tools and Techniques for Assuring the Quality Attribute Requirements

The 6502 Debugger team intends to verify and validate for quality attributes that the system must possess. During the design phase, the team has developed quality attribute scenarios and reviewed those with the customer. After development phase and during initial implementation of the system, the team will use specific tools to measure whether or not the system is up to par.

Quality Attribute	Tool/Technique Used	Rationale for using tool/technique
Unit Testing	...	...
Defects Tracking	Excel spreadsheet	It will be used to record the number of defects and the rate of defects through time.
Performance	...	...
Usability	User questionnaire and surveys	These will help understand the user specific requirements and how system is user friendly. ConOps document describes various use cases.

7 Testing Strategy

Testing the 6502 Debugger project seeks to accomplish two main goals:

1. Detect failures and defects in the system.
2. Detect inconsistency between requirements and implementation.

To achieve these goals, testing strategy for the system will consist of four testing levels. These are unit testing, integration testing, acceptance testing, and regression testing. The following subsections outline the different test levels, which development team is responsible for developing and executing them, and criteria for determining their completeness.

7.1 Unit Testing

The target of unit tests is a small piece of source code. Unit tests are useful in detecting bugs early and also in validating the system design. These tests are done one function at a time and are written and executed by the developer. Ideally line of code is tested, however it is not always cost effective. Code coverage goals will be defined to ensure the most important code is well covered by tests while still making efficient use of developer time. Unit testing will be done by developers during each of the development phases outlined in the Project Plan. All unit tests must be executed and passing before each code check-in to the source control system.

7.2 Integration Testing

Integration testing will execute several modules together to evaluate how the system as a whole will function. Integration tests will be written and executed by the testing team. Attempting to integrate and test the entire system at once will be avoided because it makes finding the root cause of issues harder to identify, making it more expensive. Instead, integration tests will be performed at specific points, ideally where there are points of interaction between components. Each test is written to verify at least one requirement using scenarios or use cases specified in the SRS.

7.3 Acceptance Testing

Acceptance testing is functional testing that the customer uses to evaluate the quality of the product and verify that it meets their requirements. Test scripts are typically smaller than integration or

unit testing due to limited time and resources of the customer. These tests cover the system as a whole and are conducted on a set of realistic data using scenarios or use cases as a guide.

7.4 Regression Testing

The purpose of regression testing is to catch any new bugs introduced into previously working code due to modifications. As such, regression tests will be run every time the system changes. They will be created and run by the testing team. It will consist of running previously written automated tests or reviewing previously prepared manual procedures. It is common for bug fixes to introduce new issues and therefore several cycles will be planned and conducted during regression testing.

7.5 Test Completetion Criteria

In each development phase tests will be conducted and their completeness will be judged by the following standards:

- *Unit Testing.* Complete when:
  - At least critical sections of code have been tested.
  - All major and minor bugs found have been logged and fixed.
- *Regression Testing.* Complete when:
  - At least 90% of functions have been covered, including all modified functions.
  - At least two test/fix cycles have been completed.
  - All issues have been logged and corrected.
- *Integration Testing.* Complete when:
  - All module interfaces have been tested.
- *Acceptance Testing.* Complete when:
  - The customer is satisfied that the product has met the agreed upon requirements criteria.

8 Organization

8.1 Available Resources that Team Intends to Devote

The 6502 Debugger team is compromised of five members, each devoting an average of ten hours per week. Most activities are dispersed among multiple team members due to the small team size. The QA activities take up ten percent of the entire team's time.

Team Members	hours/week	QA percentage	QA hours/week
5	10	10%	5 hours/week

The 5 hours per week will be divided amongst the appropriate QA activities. The strengths and weaknesses, and the availability of each term members will determine the designation of the QA activities.

8.2 Quality Assurance Team

The QA plan and guidelines will be available to all team members. Discussion of QA activities will take place at bi-weekly meetings to assure all members are aware of their roles and responsibilities. In addition, all team members will collaborate to select roles for reviews so that they are filled with team members who best fit the characteristic of the role. The QA Coordinator will be in charge of managing the QA team and will be the tie breaker if need be during decision making. They will also be responsible to ensure each team member is carryout their responsibilities correctly and on time. For each activity, team members have roles defined below.

Role	**Responsibilities**
Quality Coordinator	• Responsible for ensuring quality activities are planned and carried out accordingly. • Responsible for ensuring all team members are properly trained and equipped for their given role. • Ensures QA activities align with resources. • responsible for leading QA activities.
Quality Reviewer	• Reviews and identifies project artifacts. • Provide feedback for improved quality in software artifacts.
QA Team member	• Provide support during QA activities by carryout assigned tasks.

Throughout the QA process, each team member is responsible for knowing:

- Their roles and responsibilities
- Goals of each activity they are associated with
- Processes that are to be carried out.

8.3 Managing of the Quality of Artifacts

When changes are made to the system reviews/tests will be conducted on the affected artifacts. All testing and review activities shall have the following documentation:

Process	How a particular method or technique should be carried out.
Goals	This will state the purpose of quality activities associated with the artifacts.
Reviewer	Roles and responsibilities of QA team members in relation to artifacts.
Notes	Any comments concerning the artifact that will be useful for successfully using the artifact.

A code/document management system shall be in place that enables the team to revert to a previous version in the event that issues are discovered in connection with said changes.

8.4 Process for Prioritizing Quality Assurance Techniques

The section contains a step-by-step outline of the process employed to prioritize the QA techniques used for evaluation:

1. Create a prioritized checklist of testing characteristics of the system; these will be realtive to the requirements and quality attributes.
2. Choose techniques (e.g. design and code reviews) that seem to fit in line with the characteristics identified from common knowledge or based on research.
3. The team should engage in discussion and assign weight to each technique for each checklist item in terms of how useful each technique is to serve the purposes of testing relative to the criteria that are of interest; the rating will be 1-10 with 1 being the weakest.
4. The team conducts an assessment session of techniques that could be useful for testing purposes; the QA leader will be in charge.
5. Weighting and majority team agreement should be the deciding factor on a technique.

8.5 Quality Assurance Strategy Break Down into Tasks

Tasks	**Effort**	**Exit Criteria**	**Deliverables**
Product realization			
Requirement	2	ConOps & SRS Reviewed	ConOps & SRS
Coding	3	Code walk through and formal technical review	Source with unit tests
Verification	2	All critical and major bugs resolved.	Reports and test source code
Validation	2	Reviewed and approved by customer.	Solution deployment
Measurement, analysis, and improvement to SQAP			
Process appraisal	2	Stakeholder process concerns are addressed	Updated SQAP and SPMP
Support processes			
Planning	2	Planning for a new activity is done by team members.	Updated SPMP

8.6 Quality Assurance Process Measures

Measurements of QA processes serve to provide an evaluation criterion that will show how useful the processes are in increasing quality of the system and suggest areas in which the processes can be improved. These improvements may be a result of the extension, exclusion, or modification of current process attribute.
Quality Assurance Processes will be evaluated based on reviews, follow up and tracking, and exit criteria.

8.6.1 Reviews:

The goal of the metrics for good and healthy processes is as following:

Measurement	**Goal**
Defect Find Rate	Defect find rate should be at most 20% of defects found and should decrease over time.
Defect Fix Rate	Defect fix rate should be higher each build and should at least 80% of the number of defects.
Defect Density	Defect density should be less than one defect per one hundred lines of code and decreasing overtime.
Types of Errors identified	Percentage of types of defects each build should be: 5% critical, 20% major, and 75% minor.Critical should be 0 (as possible) with each final build.

8.6.2 Follow up and tracking:

When reviews and tests are completed, a measure of success or failure will be assigned. If successful, the process would ensure that the work product is packaged for release or documents are base-lined. If failure occurs, the bugs will be tracked in a defect repository against the artifact in question. Appropriate actions will be carried out to ensure reevaluation and corrections are made.

8.7 Exit Criteria:

The exit criteria as defined in the plan depends upon the goal set for specific sections of the plan. Thus, whenever the process of review or testing takes place, the goal, specific to a deliverable or work product being tested or reviewed, would serve as the exit criteria for that section.

9 Glossary

9.1 Definitions

Word	**Definition**
Test factor	The risk or issue that needs to be addressed as part of the test strategy.
Test phase	The phase of the software development life-cycle in ehich the testing will occur.
Artifacts	A tangible by-product produced during development of software (e.g. use cases, diagrams, requirements, documents).

9.2 Acronyms

Acronym	**Expansion**
SQAP	Software Quality Assurance Process
SDLC	Software Development Life Cycle
SPMP	Software Project Management Plan
ConOps	Concept of Operations
SRD	Systems Reference Document
SRS	Software Requirements Specification