

Corporate Software Quality Assurance Plan

Corporate Document #1000-2010

Author: Christian McArthur

November 30, 2010

Maintenance Log

Date of Change	Author of Change	Change Control Number	Description of Change
09/12/10	Christian McArthur	v0.01	Initialize document with outline and formatting.
09/20/10	Christian McArthur	v0.02	Wrote the following section(s): Purpose, Reference documents,
09/22/10	Christian McArthur	v0.03	Filled in sub-section headers. Started sections 4 & 5.
09/30/10	Christian McArthur	v0.04	Added information to section 4 subsections.
11/14/10	Christian McArthur	V0.05	Worked on section 4
11/15/10	Christian McArthur	V0.06	Finished section 4, started section 5
11/16/10	Christian McArthur	V0.07	Created corporate document index appendix, finished section 5, started section 6.
11/19/10	Christian McArthur	V0.08	Worked on section 6
11/20/10	Christian McArthur	V0.09	Worked on section 6
11/21/10	Christian McArthur	V0.10	Completed sections 6 and 7.
11/23/10	Christian McArthur	V0.11	Completed sections 8 and 9.
11/24/10	Christian McArthur	V0.12	Completed sections 10, 11 & 12
11/25/10	Christian McArthur	V0.13	Completed section 13.
11/29/10	Christian McArthur	V1.00	Last review & edits.

Table of Contents

Maintenance Log.....	ii
1 Purpose.....	1
2 Reference documents.....	1
2.1 Example Software Quality Plans or Templates.....	1
2.2 Software Engineering References.....	2
3 Management.....	2
3.1 Organization.....	2
3.2 Tasks.....	4
3.3 Roles and responsibilities.....	4
3.4 Quality assurance estimated resources.....	5
4 Documentation.....	5
4.1 Purpose.....	5
4.2 Minimum documentation requirements.....	6
4.2.1 Software Quality Assurance Plan (SQAP).....	6
4.2.2 Software requirements description (SRD).....	6
4.2.3 Software design description (SDD).....	7
4.2.4 Verification and validation plans.....	7
4.2.5 Verification results report and validation results report.....	8
4.2.6 User documentation.....	8
4.2.7 Software configuration management plan (SCMP).....	9

4.3 Other documentation.....	9
4.3.1 Software Development Standards.....	9
4.3.2 Software Project Tools.....	10
4.3.3 Traceability Matrix.....	10
5 Standards, practices, conventions, and metrics.....	11
5.1 Purpose.....	11
5.2 Documentation standards.....	11
5.3 Coding standards.....	12
5.4 Testing standards and practices.....	13
6 Software reviews.....	14
6.1 Purpose.....	14
6.2 Minimum requirements.....	14
6.3 Software specifications review (SSR).....	15
6.4 Architecture design review (ADR).....	15
6.5 Detailed design review (DDR).....	16
6.6 Verification and validation plan review.....	16
6.7 Functional audit.....	17
6.8 Physical audit.....	17
6.9 In-process audit.....	18
6.10 Managerial reviews.....	18
6.11 Software configuration management plan review (SCMPR).....	19
6.12 Post-implementation review.....	20

6.13 Other reviews and audits.....	20
7 Test.....	21
7.1 Unit tests.....	22
7.2 Integration testing.....	22
7.3 System level test.....	22
8 Problem reporting and corrective action.....	23
9 Tools, techniques, and methodologies.....	24
10 Media control.....	25
11 Supplier control.....	26
11.1 Software by Contract.....	26
11.2 “Off-The-Shelf” Components.....	27
12 Records collection, maintenance, and retention.....	28
13 Training.....	28
14 Risk Management.....	29
A)IEEE Quality Standards.....	30
B) Corporate Document Index.....	31

1 Purpose

This document describes the Corporate Quality Assurance Plan. Its goal is to establish requirements, processes, and responsibilities to effectively establish Quality Assurance Plans for each software project within the corporation. It will also be used as a template for writing an individual project's software's quality assurance plan.

This plan covers all aspects of the software life cycle and includes standards for software design and implementation, guidelines for software testing and handling of defects, tools and techniques to be used to support the software quality assurance process, provisions for protecting the integrity of physical media, policies regarding the quality of third-party provided software, software documentation requirements and record retention policies, employee training needs in regards to software quality, and risk management strategies.

2 Reference documents

This section provides a list of documents that may help in understanding this Software Quality Policy and basic Software Engineering concepts.

2.1 Example Software Quality Plans or Templates

Template for the Software Quality Assurance Plan, (n.d.) Retrieved from <http://sw->

[assurance.gsfc.nasa.gov/disciplines/quality/documents/doc/software_quality_assurance_p
lan.doc](http://assurance.gsfc.nasa.gov/disciplines/quality/documents/doc/software_quality_assurance_p
lan.doc)

JISC Software Quality Assurance Policy, (n.d.) Retrieved from
[http://www.jisc.ac.uk/uploaded_documents/draft_JISC_software_quality_assurance_poli
cy.doc](http://www.jisc.ac.uk/uploaded_documents/draft_JISC_software_quality_assurance_poli
cy.doc)

Gorman, Jason. *Quality Assurance Strategy*, (September 8, 2009) Retrieved from
[http://www.parlezuml.com/tutorials/agile_qa/Example_Agile_Quality_Assurance_Strateg
y.pdf](http://www.parlezuml.com/tutorials/agile_qa/Example_Agile_Quality_Assurance_Strateg
y.pdf)

2.2 Software Engineering References

Rumbaugh, James. *The Unified Modeling Language Reference Manual*, Addison-
Wesley, 1999.

Wieggers, K. *Software Requirements*, 2nd Ed, Microsoft Press, 2003.

3 Management

3.1 Organization

Each project team will consist of several groups of individuals. The first individual will be the Project Manager. The Project Manager will be assigned at the beginning of the project and will be responsible putting together the rest of the project

team and for ensuring all parts of the software development process, including software quality assurances, are performed.

Also at the start of the project, one or more Software Engineers will be assigned and will be referred to as the Engineering Team. The exact number depends upon the size and complexity of the software project to be produced. The Engineering Team will initially work with the customer to determine their requirements for the software producing a requirements document or requirements specification; this team will then design the software project creating documentation for use by the programmers and testers for the project. At least one Software Engineer will be available throughout the development and testing phases of the project to answer any questions or resolve any defects in the design or documentation..

Before the Engineering Team has completed the software design, one or more programmers will be assigned to the project as part of the Development Team. The number of individuals on this team will be determined based upon the size and complexity of the project. The Development Team will be responsible for implementing the software project, based upon the software design and documents produced by the Engineering Team, as well as perform unit testing of their own code.

At some point during the implementation process, one or more individuals will be assigned to the project as part of the Testing Team. The number of individuals on this team will be determined based upon the size and complexity of the project. The Testing

Team will be responsible for integration and system level tests. It will provide reports back to the Development Team in regards to the defects found during the testing process.

3.2 *Tasks*

This Software Quality Policy covers all portions of the software life cycle including:

- Software requirements elicitation.
- Software requirements analysis
- Software design & documentation
- Implementation
- Testing and validation
- Release of software to customer

The entry and exit point for each task will be defined by the individual project's Software Quality Plan. This is due to the individualistic nature of each project requiring different criteria for when to start or end a particular task.

3.3 *Roles and responsibilities*

The following table lists the required tasks and the portion of the project team

responsible for the task.

Project Task	Responsible Team
Requirements Elicitation	Engineering
Requirements Analysis	Engineering
Design and Documentation	Engineering (Development as sub responsibility)
Implementation	Development
Testing and Validation	Testing (Development as sub responsibility)
Release of Software	Entire project team

Table 1: Project Tasks and Responsible Teams

3.4 Quality assurance estimated resources

Every Software Quality Assurance Plan for individual projects should include an estimation of the costs and resources required for assuring software quality and the tasks required to make that assurance.

4 Documentation

4.1 Purpose

All software projects will require a certain level of documentation. This documentation will be used during the design, implementation, testing and verification

phases of the development process. This section describes the minimum level of documentation that is required for every software project. A project's individual software quality assurance plan will note if additional documentation is required. It will include a list of required documents, information that should be contained within these documents, and how they will be used throughout the development process.

4.2 *Minimum documentation requirements*

The following documents will be required by every software project, at minimum.

4.2.1 Software Quality Assurance Plan (SQAP)

Every project will have its own Software Quality Assurance Plan. It will be written by the lead software engineer with the help of the project manager. This plan should be modeled after this policy with similar section headings and content. The content of the plan will be focused on the individual aspects of the software project.

4.2.2 Software requirements description (SRD)

The software requirements description covers the basic functionality of the software product to be developed as well as requirements for external interfaces, performance needs, security and safety, and usability and user interfaces. More specific details on document format and contents can be found in the internal corporate document

#1005-2010, “Software Requirements Documentation”.

4.2.3 Software design description (SDD)

The software design description is a formal design of the software project such that it explains how the requirements as described in the software requirements description (4.2.2) will be implemented. The software design description can come in a number of forms including textual description, UML diagrams, and possibly mockups of user interfaces. More specific details on the acceptable document formats and their appropriate contents can be found in the internal corporate document #1010-2010, “Software Design Documentation.”

4.2.4 Verification and validation plans

There may be a number of separate documents covering verification and validation depending on the size and scope of the software project. These documents will contain information about how the project will be tested at various levels including unit testing, integration testing and full system testing. There will also be documentation discussing how the software project will be evaluated to assure it meets the documented software requirements. More specific details on testing requirements, the associated documentation for test plans and testing procedures, and for software validation can be found in the internal corporate document #1015-2010, “Corporate Software Testing and

Software Validation Plans and Policies.”

4.2.5 Verification results report and validation results report

The previous section covered the documentation for plans to perform tests and validate the software project. The documentation discussed here is for how the results of the tests and validations should be presented. More specific details on reporting of software tests and validation exercises can be found in internal corporate document #1016-2010, “Software Test and Validation Report Formats.”

4.2.6 User documentation

User documentation includes all documentation to be provided to the end-user of the software project. The documentation may be comprised in different forms, including hard-copy and soft-copy, and may consist of multiple individual documents. The topics to be covered by this documentation includes guides to install, administer, and operate the software product. It should also include information about possible error messages and how a user can recover from an error. Detailed information about user documentation can be found in corporate document #1020-2010, “End-User Documentation.”

4.2.7 Software configuration management plan (SCMP)

The “Software Configuration Management Plan” (SCMP) is a document that discusses how configuration management for a software project will be controlled. The actual plan will be tailored to the specific requirements and composition of the project team for a software project. It will discuss, at minimum: how individual parts of the software project will be identified and whether special identification numbers will need to be assigned, how project baselines will be established, the process in which changes to the overall project will be approved, and how project audits will be conducted. For specific information about the SCMP, see corporate document #1025-2010, “Software Configuration Management Plans.”

4.3 *Other documentation*

4.3.1 Software Development Standards

Corporate Document #1030-2010, “Software Development Standards,” and Section 5 of this document covers the standards, requirements, and expectations for the software development team. Should a software project require deviation from these standards a specialized development standards document should be written and then approved by the corporate standards committee.

4.3.2 Software Project Tools

Corporate Document #1032-2010, “Software Project Development and Project Management Tools,” discusses the tools available for software development and project management. If a project requires additional tools not specified in this document, a proposal should be written and submitted to the acquisitions department detailing the justification for the request of the use of specialized tools, the costs associated with acquiring and maintaining the tools, and a comparison with similar tools.

4.3.3 Traceability Matrix

Traceability matrices should be created before testing starts on the software project. The matrix will be used to show the relationship between the software requirements for the software project and the individual tests that will be performed. Traceability matrices can also be used for software projects that have many individual documents to show where requirements, design elements, tests, and other information relate from one document to another. Refer to corporate document #1050-2010, “Traceability Matrices,” for more information.

5 Standards, practices, conventions, and metrics

5.1 Purpose

This section will describe the standards, practices and conventions, quality requirements and metrics that will be applied to all software projects.

5.2 Documentation standards

All documentation related to a software project, including but not limited to requirements description, design description, test plans, and end-user documentation must adhere to general documentation standards. The specifics of these standards can be found in a variety of corporate documents including: #1005-2010, “Software Requirements Documentation”, #1010-2010, “Software Design Documentation.”, #1015-2010, “Corporate Software Testing and Software Validation Plans and Policies.”, #1020-2010, “End-User Documentation.”, and #1025-2010, “Software Configuration Management Plans.”.

All documents and files related to a software project must be given some form of identifier as per the Software Configuration Management Plan. Each software project will be given its own first-level identifier and each document for a software project will have a second-level identifier. Documents must have a version number and/or date assigned.

Generally, documents created for a software project must include the following parts, at minimum:

- Title page including: document title, document number, version number and/or date, software project name and author(s).
- Table of contents
- Overall Summary
- Main content of document

5.3 Coding standards

Corporate Document #1030-2010, “Software Development Standards,” describes specific requirements for code that is created for a software project. Generally, all code for a software project should include:

- A header for each file that includes: Software project name, module/file/code name, version number and/or date, programmer(s) names, general purpose of code, copyright statement, and publicly accessible functions.
- Each function within the code should include a function header that includes: function name, publicly accessible status, argument information, output information, and general purpose of the function.
- Every global variable, module-wide variable, and functional argument should

have a descriptive name representing its purpose. In addition, when variables are declared they should include in-line comments giving a brief description for the purpose of the variable.

5.4 *Testing standards and practices*

Testing will occur throughout the software development process. Software developers are expected to write and execute unit tests for the modules they create and work on. Formal test plans are not required for unit tests, however, the unit test code should be documented just as any other code the developer creates (see section 5.3). Formal test result reports are not required, however informal documentation regarding the success or failure of unit tests and the amount of coverage by the tests should be provided to the test team when the module is submitted for further testing.

The project's test team will be responsible for integration and system testing. Formal test plans and reports will be required for this phase of testing. See corporate documents #1015-2010, “Corporate Software Testing and Software Validation Plans and Policies,” and #1016-2010, “Software Test and Validation Report Formats,” for specific information regarding test plan requirements, testing policies, and test report documentation standards.

6 Software reviews

6.1 *Purpose*

This section will describe the types of software reviews that may be required for a project. Not all of the reviews listed in this section will be required for all software projects. Each sub-section will discuss a particular type of review and under what circumstances the review may be required for a project. A software project's individual software quality assurance plan should include a list of reviews and audits that will take place for the project along with specific, project oriented information, about the purpose for the review, how the review will be conducted, the people involved in performing the review and how the results of the review will be handled. A review may be required under circumstances other than specified below if the customer/client specifically requires the review to occur as part of the contract to produce the project.

6.2 *Minimum requirements*

All software projects will must perform the following reviews as described below: Software Specifications Review, Detailed Design Review, Verification and Validation Plan Review, Functional Audit, and Post-implementation Review. Other reviews may be required by quality assurance plans for individual projects.

6.3 *Software specifications review (SSR)*

The Software Specifications Review (SSR) is used to assure that the requirements, as stated in the Software Requirements Document (see Section 4.2.2), are correct and is acceptable to the client and/or end-users. The individuals who will participate in this review will include: the project manager, the software engineering team, and the client. This group of individuals will go over the requirements document, identify any errors or ambiguities, and discuss any changes that need to occur. Should changes to the document be required, the software engineering team will make the changes and resubmit the document for review.

6.4 *Architecture design review (ADR)*

The Architecture Design Review (ADR) is a review of the high-level overview of the detailed design. It is a subset of the Software Design Description (see section 4.2.3). The purpose of this review is to assure that the overall design is appropriate to satisfying the requirements of the project. The individuals who will participate in this review will include: the project manager, the software engineering team, the client, and, at minimum, the development team leader. This group of individuals will go over the top-level view of the software design description to identify any errors or ambiguities and to discuss any changes that need to occur. Should changes to the document be required, the software engineering team will make the changes and resubmit the document for review.

6.5 Detailed design review (DDR)

The Detailed Design Review (DDR) is a review of the full design of the software project. The purpose of this review is to assure that the entire design of the software project satisfies the requirements and only satisfies the stated requirements. This review should occur prior to the start of the development process. The individuals who will participate in this review will include: the project manager, the software engineering team, the client and, at minimum, the development team leader. This group of individuals will go over the full software design description to identify any errors or ambiguities and to discuss any changes that need to occur. Should changes to the document be required, the software engineering team will make the appropriate changes and resubmit the design document for review.

6.6 Verification and validation plan review

The Verification and Validation (V&V) Plan review's purpose is to ensure the tests devised for the software project can correctly evaluate the adequacy and completeness of the software. Since the V&V plan includes requirements for unit level testing, this review should occur prior to the start of the development of the software. The individuals who will participate in this review include the project manager, the software engineering team leader, the software development team leader, and the testing team leader. This group of individuals will review the V&V plan to determine if there are any errors or

requirements that are not tested. Should changes be required to the V&V plan, the software engineering team leader and the testing team leader will work together to make the appropriate changes and will resubmit the plan for additional review.

6.7 *Functional audit*

The functional audit is used to determine that the software that has been passed out of testing matches the requirements specification, the software design , and other project documentation. This review will occur after the development and testing of the software project but prior to distribution.. The individuals who will participate in this review will include, at minimum, the project manager, the client, software engineering team leader, the development team leader, and the testing team leader. If the software should be found to not meet the project documentation, the review team will make the determination what part of the project and/or documentation must be corrected. Once the required modifications are decided, the appropriate changes will be made. The software project will be re-reviewed via functional audit and any previous reviews normally required for the parts of the project that are modified.

6.8 *Physical audit*

The physical audit is used to ensure that the software project is ready for release and that all modules have been completed, tested and defects removed and that all

documentation, including end-user documentation, is completed. The report for this review can be obtained from the configuration management utility. The individuals who will be involved in the review include, at minimum, the project manager, the client, the software engineering team leader, the software development team leader, and the testing team leader. If any part of the software project is found to be not ready for release, those items needing modifications will be sent to the appropriate group. Once the corrections have been completed, and any reviews that may need to be performed related to the corrections, the entire project will be resubmitted for a physical audit.

6.9 *In-process audit*

In-process audits are reviews that should take place at other times through the design, development and testing process. In-process audits should be performed regularly, the exact frequency will be specified in the software project's quality assurance plan. The specific type of review that will be performed will be dependent on which phased the software project is in and will mirror the reviews listed previously in this section.

6.10 *Managerial reviews*

Managerial reviews will also take place throughout the design, development and testing process. This review is used to ensure that all aspects of the project's software

quality assurance plan (SQAP) is being fulfilled. The intent behind this review is to make sure that the SQAP is adequate for the project. The individuals responsible for performing this review is the project manager and corporate management with assistance when needed from the project's software engineering team leader, software development team leader and testing team leader. If the determination is made that the SQAP needs to be modified, the project manager will make the modifications and submit it for review. If the revised SQAP is approved, it should be distributed to the leads of the sub-groups within the project.

6.11 Software configuration management plan review (SCMPR)

The Software Configuration Management Plan Review's (SCMPR) purpose is to ensure that the policies, plans and tools used for the software project's configuration management is sufficient for the project. Similar to in-process reviews, this review should occur periodically throughout the design, development and testing process. The individuals responsible for performing this review are the software project manager, the software engineering team leader, the software development team lead, and the testing team leader. If the the project's configuration management is found to be lacking in some area this needs to be corrected. If the deficiency is with the plan itself the project manager should modify the plan so that configuration management is accurately

described for the project. If the deficiency is that the configuration management tools are not being used properly the team leaders will address this with their respective groups. If the deficiency is with the tool itself, the project manager will compile the reasons why the tools are not sufficient for the software project and provide recommendations to corporate management on how the issues are best resolved.

6.12 *Post-implementation review*

Once the software product has been delivered to the client or has been deployed into the production environment, a post-implementation review should be conducted. The purpose of this review is to evaluate the entire design, development and testing process to see what worked and what needs to be improved on for future projects. The individuals who will perform this review will be the software project manager, the software engineering team leader, the software development team leader and the testing team leaders. The final report from this review along with any recommendations for future projects will be submitted to corporate management.

6.13 *Other reviews and audits*

Other reviews or audits may be necessary for a particular software project. For example, some projects may require compliance audits for government regulators. Any reviews or audits that are required for a software project that is not described above will

be included in the software project's individual software quality assurance plan. This addition to the plan should include the purpose of the review, how the review will be executed, who will be involved in performing the review and how the results of the review will be handled.

7 Test

As described in section 4.2.4 , every software project will have a Verification and Validation plan tailored to needs and requirements of the project. The sub-sections below describe some of the tests that are required for every software project. More details on tests, verification and validation requirements can be found in internal corporate document #1016-2010.

7.1 *Unit tests*

Individual software developers are responsible for performing unit level tests for all code and modules they work on. Unit tests will be used to show that the module fulfills the requirements of the module. The software project's verification and validation plan will discuss specific requirements and details on unit testing. Generally, as part of testing the module's fulfillment of the requirements, the unit tests for a module should cover at least 90% of the module's lines of code and 80% of the logical branches within the module. The module cannot be submitted for completion and sent to the testing team unless all of the the unit tests are completed and passed.

7.2 *Integration testing*

As modules are completed and submitted to the testing team, they will be integrated into overall software project. The testing team will be responsible to testing the software as it is integrated together. Integration tests will ensure that the various modules within a project are interacting correctly as per the requirements and design specifications. How integration testing will occur will be specified in the software project's verification and validation plan.

7.3 *System level test*

Prior to release of the software product, the fully integrated product will undergo

system level testing. In this set of tests, the product will be run in an environment closely matching the final environment specified in the software requirements. The software project's specific verification and validation plan will include details about how the system level tests will take place and under what circumstances the product will pass tests.

8 Problem reporting and corrective action

Every software project will have a defect tracking system in place by the start of the development process. Acceptable defect tracking tools can be found in corporate document #1032-2010, “Software Project Development and Project Management Tools.” The defect tracking system will be used to report all issues with the software requirements, design, or implementation.

A member of the software engineering team will be responsible for monitoring the tracking system to ensure that all reported issues are routed to the appropriate group. All of the team leaders will also be monitoring the tracking system for issues assigned to their group. When an issue is assigned to a group (software engineering, development, or testing) it will be assigned to a particular member of the group to followup on the issue. This will require attempting to recreate the issue, determine if the issue works as designed or if there is a defect with the software. If the issue is the result of a defect, the person evaluating the defect will determine where the defect is originating from. If the defect originates from a group other than the group the evaluator is in, the issue will be

reassigned to the appropriate group.

Once the issue is determined to be a defect and is assigned to the group where the defect originates a plan will be created to describe how the defect will be resolved. This plan will describe the nature of the defect, the cause, what actions need to be taken to resolve the defect and what effects the resolution will have on the rest of the project.

A software project's individual software quality assurance plan will include information about the specific groups and persons responsible for the defect tracking system, tracking of issues, and resolution of issues.

9 Tools, techniques, and methodologies

Each software project's software quality assurance plan should address the specific tools, techniques and methods that will be used for designing, implementing and testing the software product. Guidelines as to what tools are approved and advised for use in software projects can be found in corporate document #1032-2010, "Software Project Development and Project Management Tools." The QA plan will detail which tool(s) will be used in the project, who is responsible for maintaining and retrieving reports from the tool, as well as how the tool will be used by members of the project team.

The process for developing the project is advised to be the V-Model. Other development methodologies, such as Agile Scrum, may be allowed if approved by corporate management. Such approval may be earned only after submitting a detailed

plan describing the development method, how it will be implemented for the particular software projects and any risks involved with using the software project. If the process is approved, then the software quality assurance plan will need to reference the document describing the development method and provide any other information needed to fully explain how it will be used for the project.

10 Media control

All software projects will be stored on the corporate file server during design, development and testing. Files may be checked out by individuals when needed and any modifications must be put on the file server when checked in. Access to files for a particular project will be limited to those individuals who are part of the project team. A project team manager may provide additional restrictions on access to the files. Any such restrictions will be stated in the project's software quality assurance plan.

All projects stored on the file server will be backed-up weekly as well as daily incremental back-ups. Backups will be verified monthly to ensure both the accuracy of the files being backed-up and that backup media itself is not damaged. These backups will be used in the event of accidental deletion of project files or in the event of a failure of the corporate file server.

A project's software quality assurance plan should discuss how the final product will be delivered to the client as well as intermediate versions, such as prototypes. The

form of the project deliverable can include, but is not limited to, being contained on physical media (CD, DVD, Blu-Ray, Tape) to being delivered online (downloadable from the corporate website, FTP from corporate server, or directly transmitted to the client). Regardless of the format of the deliverable, the software quality assurance plan will describe provisions for ensuring that only the client has access to the deliverables.

11 Supplier control

This section describes the requirements for using software, components, and/or modules from a third-party in a corporate led software project. The sub-sections below describe the procedures for handling third-party software in the form of software created under contract for the project and software obtained as previously developed modules or components (“Off-The-Shelf” components).

11.1 Software by Contract

When a third-party is contracted to develop parts of the software project, there are a number of steps that must take place to ensure the contracted software maintains the level of software quality expected. The third-party must devise a software quality assurance plan using the same or similar format as this policy. This plan will cover, at minimum, all of the same concerns addressed by this policy. If the third-party does not implement a quality assurance plan or fails to follow the plan, then the contract can be voided.

When a third-party is contracted to develop a software component or module, they will be given the requirements specification for the component. In addition, the terms of the contract with the third-party will also state what deliverables will be required from the third-party upon completion of the component, including, at minimum, the component, its testing reports and defect tracking logs.

Before software from a third-party can be used in a software project, it must be verified that it works correctly. In addition to reviewing the testing reports and other documentation from the third-party, the software project testing team will perform their own set of tests on the components. If the component passes tests by the corporate software project testing team, then it may be used in the software project.

11.2 “Off-The-Shelf” Components

Pre-written, off-the-shelf components or modules can be used in a software project if the component meets all of the requirements for a particular part of the project. To ensure that a model fulfills the requirements, the module will undergo the validation and verification like any other module written in-house for the project. If the off-the-shelf component passes the tests then it can be used in the software project.

12 Records collection, maintenance, and retention

Once a software project has been completed and delivered to the client the project's documentation, source code, and reports will be retained on the corporate file server for at least one year. Following delivery to the client, access to the project's files will be limited to the project manager and any team members who are responsible for maintaining the product.

After one year has passed with no modifications to the product, whether it be to resolve defects or to add features, the software project will be archived. The archive will be backed-up to offline storage. Should the project need to be reopened, corporate management can initiate a process to load the project archive back to the corporate file server and assign a project manager to start a team to make any necessary changes.

13 Training

The company will periodically provide training in areas related to software quality. This training will include:

- Creation and analysis of software quality assurance plans.
- Requirements and design documentation

- Coding practices
- Testing strategies

The training is not intended to be introductory training in these areas. Rather its purpose is to provide a refresher, if necessary, and to teach how these areas are used within the current corporate environment.

14 Risk Management

Every software quality assurance plan will include information on risk management. This information will include how the project manager and the project team will assess possible areas of risk with the project, how these areas will be monitored, and how they will be handled.

A) IEEE Quality Standards

This appendix provides a list of useful IEEE standards for software quality and software project management.

IEEE Std 730-2002, IEEE Standard for Software Quality Assurance Plans

IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications

IEEE Std 1016-1998, IEEE Recommended Practice for Software Design Descriptions

IEEE Std 1058-1998, IEEE Standard for Software Project Management Plans

B) Corporate Document Index

#1000-2010, “Software Quality Assurance Policy”

#1005-2010, “Software Requirements Documentation”

#1010-2010, “Software Design Documentation”

#1015-2010, “Corporate Software Testing and Software Validation Plans and Policies”

#1016-2010, “Software Test and Validation Report Formats”

#1020-2010, “End-User Documentation”

#1025-2010, “Software Configuration Management Plans”

#1030-2010, “Software Development Standards”

#1032-2010, “Software Project Development and Project Management Tools”

#1050-2010, “Traceability Matrices”