# Data Mining and Monitoring (DMM)
# Quality Assurance Plan

Apr 23, 2013

**DMM Team**
**Version 0.9**


**Team Members:**
Tom Mooney
Ahmed Osman
Shailesh Shimpi
Isaac Pendergrass

## REVISION LIST

| Revision | Date | Author | Comments |
|---|---|---|---|
| 0.1 | 4/6/2013 | Ahmed Osman | First Draft |
| 0.2 | 4/14/2013 | Shail Shimpi | Introduction section completed. |
| 0.3 | 4/16/2013 | Tom Mooney | Section 7 completed |
| 0.4 | 4/16/2013 | Ahmed Osman | Add QA strategy and Review process |
| 0.5 | 4/17/2013 | Shail Shimpi | Tool and Techniques section completed. |
| 0.6 | 4/17/2013 | Tom Mooney | Grammar and clarity revisions. Added some comments. |
| 0.7 | 4/20/2013 | Shail Shimpi | Introduction is edited as per the comments posted and added MSTest for unit testing. |
| 0.8 | 4/23/2013 | Isaac Pendergrass | Updated Documentation and Organization sections. |
| 0.9 | 4/23/2013 | Shail Shimpi | Tools and Techniques section is modified. |

## APPROVAL BLOCK

| Version | Comments | Responsible Party | Date |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |

# Table of Contents


**1 INTRODUCTION ..... 4**

**2 REFERENCED DOCUMENTS ..... 5**

**3 QUALITY ASSURANCE STRATEGY ..... 5**

**4 DOCUMENTATION ..... 6**

4.1 PURPOSE ..... 6
4.2 MINIMUM DOCUMENTATION REQUIREMENTS ..... 6
4.2.1 Concept of Operations (ConOps) ..... 6
4.2.2 Software Requirements Document (SRS) ..... 7
4.2.3 Software Test Plans ..... 7
4.2.4 Software Test Reports ..... 7
4.2.5 Software Architecture and Design ..... 7
4.2.6 User Documentation ..... 7
4.2.7 Other Documents ..... 7

**5 GOALS ..... 7**

5.1 QA GOALS OF EACH PHASE ..... 7

**6 REVIEWS AND AUDITS ..... 8**

6.1 WORK PRODUCT REVIEWS ..... 8
6.2 QUALITY ASSURANCE PROGRESS REVIEWS ..... 9

**7 TOOLS AND TECHNIQUES (SHAIL) ..... 10**

7.1 TOOLS AND TECHNIQUES FOR ASSURING QUALITY OF FUNCTIONAL REQUIREMENTS ..... 10
7.2 TOOLS AND TECHNIQUES FOR ASSURING THE QUALITY ATTRIBUTE REQUIREMENTS ..... 11

**8 TESTING STRATEGY ..... 11**

8.1 UNIT TESTING ..... 11
8.2 INTEGRATION TESTING ..... 12
8.3 ACCEPTANCE TESTING ..... 12
8.4 REGRESSION TESTING ..... 12
8.5 TEST COMPLETION CRITERIA ..... 12

**9 ORGANIZATION ..... 13**

9.1 AVAILABLE RESOURCES THAT TEAM INTENDS TO DEVOTE ..... 13
9.2 QUALITY ASSURANCE TEAM ..... 13
9.3 MANAGING OF THE QUALITY OF ARTIFACTS ..... 14
9.4 PROCESS FOR PRIORITIZING QUALITY ASSURANCE TECHNIQUES ..... 15
9.5 QA STRATEGY BREAK DOWN INTO TASKS ..... 16
9.6 QUALITY ASSURANCE PROCESS MEASURES ..... 16

**10 GLOSSARY ..... 18**

10.1 DEFINITION ..... 18
10.2 ACRONYMS ..... 19

# 1 INTRODUCTION

**Purpose:**
This document outlines the quality standards for the system "Data Mining and Monitoring" (hereafter referred to as DMM) and other project artifacts. These standards are primarily derived from software requirements, software architecture documents and conform to the requirement of the stakeholders.

**Scope:**
The primary audience for this document is the DMM project team. The team members are responsible for following the quality standards laid out while developing the application, documenting the results, monitoring the project progress, and testing the project quality. This SQAP (Software Quality Assurance Plan) covers all important aspects of software development; i.e. requirements analysis, architecture and design, implementation, testing and verification, and user acceptance.

**Background and Context**
With the growth of distributed development has come a variety of environments supporting distributed team collaboration. These environments typically provide a suite of integrated applications for communication. The use of collaboration tools such as Assembla provides a rich database of developer interactions and artifacts. This suggests that it may be possible to instrument the Assembla collaboration tool to monitor progress and compare it to the results of past projects to alert users when signs of trouble are detected.

**Project Objectives**
Assembla collaboration software allows for gathering and reporting on a plethora of metrics. Where these tools come up short are on methods for analyzing those metrics and automatically alerting stakeholders to signs of trouble based on historical project performance. The purpose of the Distributed Development Monitoring and Mining application is to do this by collecting and modeling historical project data to predict, in real time, the health of an in-progress project and to alert the project stakeholders when signs of trouble are detected.

**Architectural Objectives**
The DMM system has mainly two external interfaces; interface to Assembla Collaboration software to get project space data and Google Predictor to analyze the collected data and then determining the project's prediction of success. The architectural objective of the DDM system is to design is framework that can extended or easily modifiable to change the system's external interfaces. Thus, the system can work against a different collaboration software or an analytical engine. In this regard, different modules of the system are decoupled to achieve this architectural objective.

**Technical Constraints**
The DMM project heavily relies on the Assembla and Google Predictor APIs for fetching data and analyzing project data. If there are any changes to these APIs, the DMM application will be impacted including severe fatal errors and that may lead to the application not working or processing data. In addition to this, changes to the predictive model will impact to the analysis data and reporting.

The project is developed using Microsoft ASP.NET and deployed on Mono server environment with backend as MySQL database. All these environments are considered to work well together and any limitation may impact working of this application.

**Project Management Constraints**

The DMM project is for the OMSE final practicum course. It is time constrained and should be completed in about 6 months. Four team members are working on the project. An unplanned absence of any team member will affect the project schedule. To mitigate this risk, the team has adopted an iterative software development process. Any loss of work is prevented by using Subversion source code repository.

**Requirements**
The DMM project requirements are documented in two documents; The Concept of Operations (ConOps) and the Software Requirements Specifications (SRS). The purpose of the ConOps document is twofold; it captures the needs and expectations of the customer/user and it serves to illuminate the problem domain. The SRS describes the system's anticipated behavioral and development quality attributes in details.

## 2 REFERENCED DOCUMENTS

**IEEE Std. 730-2002**
IEEE Standard for Software Quality Assurance Plans. This document defines the standards for making the SQAP document.

## 3 QUALITY ASSURANCE STRATEGY

To assure the quality of software deliverables in each software development phase, we will use the 'test factor/test phase matrix'. The matrix has two elements. Those are the test factor and the test phase. The risks coming from software development and the process for reducing the risks should be addressed by using this strategy. The test factor is the risk or issue that is being addressed, and the test phase the phase in the software development life cycle in which the tests are conducted. The matrix should be customized and developed for each project. Thus, we will adapt the strategy to our project through four steps.

- In the first step, we will select the test factors and rank them. The selected test factors such as reliability, maintainability, portability or etc, will be placed in the matrix according to their ranks.

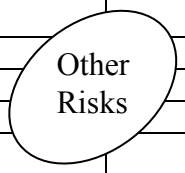
- The second step is to identify the phases of the development process. The phase should be recorded in the matrix.

- The third step is to identify the business risks of the software deliverables. The risks will be ranked into three ranks such as high, medium and low.

- The last step is to decide the test phase in which risks will be addressed. In this step, we will decide which risks will be placed in each development phase.

For example, the table given below addresses a ranked list of test factors on the project and also specifies the various lifecycle phases on the project. One risk has been highlighted and a strategy to mitigate the same is also marked. Whenever the team enters a phase, the corresponding risks associated with the phase are identified. The table below serves only as a purpose of example.

| Test phase / Test factors | Requirements | Design | Build | Dynamic test | Integrate | Maintain |
|---|---|---|---|---|---|---|
| **Correctness** | **Risk**:<br>The SRS may not be correct as per the goals of the SQAP;<br>**Strategy**:<br>Formal Technical Review of SRS | | | | | |
| **Performance** | | | | | | |
| **Availability** | | | | | | |
| **Continuity of Processing** | | | | | | |
| **Compliance** | | | | | | |
| **Ease of use** | | | | | | |
| **Coupling** | | | | | | |
| **Ease of Operations** | | | | | | |
| **Access Control** | | | | | | |
| **File Integrity** | | | | | | |




Test factors/test phase matrix [Perry 2000]

The matrix forms a part of the quality assurance strategy and as mentioned above, this matrix will be used in each of the project lifecycle phases to identify the risks associated with each of the development phases with respect to the testing factors. The risks would also be accompanied with their mitigation strategies and in case the risk materialized into a problem, the respective mitigation would be applied. It is for these reasons, that a mention is made about the matrix here in a separate section of the document and not mixed with other sections of the document to avoid repetition.

# 4 DOCUMENTATION

## 4.1 PURPOSE

This section shall perform the following functions:

a) Identify the documentation governing the development, verification and validation, use, and maintenance of the software.
b) 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 6 of the SQAP.

## 4.2 MINIMUM DOCUMENTATION REQUIREMENTS

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

### 4.2.1 Concept of Operations (ConOps)

The ConOps may be written by the supplier (internal or external), the customer, or by both. The SRD should address the basic expected feature sets and constraints imposed on the system's operation. Each requirement should be uniquely identified and defined such that its achievement is capable of being objectively measured. An active review process is to be used to ensure suitability and completeness of user requirements.

### 4.2.2 Software Requirements Document (SRS)

Software specification review is to be used to check for adequacy and completeness of this documentation. The Software Requirements Document, which defines all the functional requirements, quality attributes requirements and constraints on the DMM project.

### 4.2.3 Software Test Plans

Software Test Plans are used to determine if developed software products conform to their requirements, and whether the software products fulfill the intended use and user expectations. This includes analysis, evaluation, review, inspection, assessment, and testing of the software products and the processes that produced the products.

### 4.2.4 Software Test Reports

Software Test Reports are used to communicate the results of the executed test plans. This being the case, a particular report should contain all test information that pertains to the current system aspect being tested. The completeness of reports will be verified in walkthrough sessions.

### 4.2.5 Software Architecture and Design

Software Architecture and Design reviews are to be used for adequacy and completeness of the design documentation. This documentation should depict how the software will be structured to satisfy the requirements in the SRD. The SDD should describe the components and subcomponents of the software design, including databases and internal interfaces.

### 4.2.6 User Documentation

User documentation guides the users 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 identified and described. All corrective actions to correct the errors causing the error messages shall be described.

### 4.2.7 Other Documents

1) Software Project Management Plan (SPMP)

# 5 GOALS

## 5.1 QA GOALS OF EACH PHASE

| *Phase* | *Goals* |
|---|---|
| Requirement gathering | SRS should have no more than one defect per page as per the client's review of the SRS. |
| Architecture | The SAD should not have any defects per architectural representation during its **formal technical review** (FTR). |
| Development | Application should not have more than 10 defects per 1 KLOC found in FTR. |
| Testing | All tested work products should be checked for finding at least one defect per page or 10 defects per 1 KLOC of codes in FTR. |

# 6 REVIEWS AND AUDITS

## 6.1 WORK PRODUCT REVIEWS

The general Strategy for the review is given below:

**Formal Reviews**:

1. One week prior to the release of the document to the client, the SQA team will review the document list generated by the Software Product Engineers (team members on a project team).
2. The SQA team will ensure that the necessary revisions to the documents have been made and that the document will be released by the stated date. In case there are any shortcomings, the document will be referred to the software project management team for revision.

**Informal Reviews**:

**A. Design Walk-throughs**

SQA will conduct design walk-throughs to encourage peer and management reviews of the design. The Software Project Manager will ensure that all the reviews are done in a verifiable way and the results are recorded for easy reference. SQA will ensure that all the action items are addressed

**B. Code Walk-throughs**

SQA will conduct code walk-throughs to ensure that a peer review is conducted for the underlying code. The Software Project Management team will ensure that the process is verifiable whereas the SQA team will ensure that all the items have been addressed.

**C. Baseline Quality Reviews**

The SQA team will review any document or code that is baselined as per the revision number of the work product. This will ensure:

1. The testing and inspection of modules and code before release
2. Changes to software module design document have been recorded and made
3. Validation testing has been performed
4. The functionality has been documented
5. The design documentation conforms to the standards for the document as defined in the SPMP.
6. The tools and techniques to verify and validate the sub system components are in place.

| Work Product | When Reviewed by Quality Assurance (Status or Criteria) | How Reviewed by Quality Assurance (Standards or Method) |
|---|---|---|
| Requirements (Software Requirements Specification) | After a new release or modification | The Requirements Specification document is reviewed and approved by the assigned reviewer(s). The reviewed document is presented to the customer for acceptance. The Requirements Specification document forms the baseline for the subsequent design and construction phases. Changes, if any, to the Requirements Specification document after its release, are studied, their impact evaluated, documented, reviewed and |

| | | |
|---|---|---|
| | | approved before the same are agreed upon and incorporated. |
| Software Architecture Document (SAD) | After a new release or modification | The Architecture/Design phase is carried out using an appropriate system design methodology, standards and guidelines, taking into account the design experience from past projects. The design output is documented in a design document and is reviewed by the Reviewer to ensure that:<br>• The requirements including the statutory and regulatory requirements as stated in the Requirements Specification document, are satisfied<br>• The acceptance criteria are met<br>• Appropriate information for service provision (in the form of user manuals, operating manuals, as appropriate) is provided.<br>Acceptance for the design document is obtained from the customer.<br>The Design Document forms the baseline for the Construction phase. Changes, if any, to the Design Document after its release, are studied, their impact evaluated, documented, reviewed and approved before the same are agreed upon and incorporated. |
| Construction (Code) | After a new release or modification | The Project Team constructs the software product to be delivered to meet the design<br>specifications, using:<br>• Suitable techniques, methodology, standards and guidelines<br>• Reusable software components, generative tools, etc. as appropriate<br>• Appropriate validation and verification techniques as identified in the Project Plan.<br>Changes, if any, to the software programs after the release, are studied, their impact evaluated, documented, reviewed and approved before the same are agreed upon and incorporated. |
| Testing and Inspection | After a new release or modification | Before delivery of the product, SQA ensures that all tests, reviews, approvals and acceptances as stipulated in the Project Plan have been completed and documented. No product is delivered without these verifications. |

## 6.2 QUALITY ASSURANCE PROGRESS REVIEWS

In order to remove defects from the work products early and efficiently and to develop a better understanding of causes of defects so that defects might be prevented, a methodical examination of software work products is conducted in projects in the following framework:

1. Reviews of Project Plans and all deliverables to the customer are carried out as stated in the Quality Plan of the project. A project may identify additional work products for review.
2. Reviews emphasize on evaluating the ability of the intended product to meet customer requirements. The reviewer also checks whether the regulatory statutory and unstated requirements, if any, have been addressed.
3. Personnel independent of the activity being performed carry out the reviews.
4. Reviews focus on the work product being reviewed and not on the developer. The result of the review in no way affects the performance evaluation of the developer.

5. The defects identified in the reviews are tracked to closure. If a work product is required to be released without tracking the defects to closure, a risk analysis is carried out to assess the risk of proceeding further.

# 7 TOOLS AND TECHNIQUES (SHAIL)

The DMM project uses the following strategy for selection of the tool on the project:

1. The testing tool is selected based on 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.
4. Selection of the tool not only depends upon affordability but also depends on the quality standard requirement of the project.

## 7.1 TOOLS AND TECHNIQUES FOR ASSURING QUALITY OF FUNCTIONAL REQUIREMENTS

In order to ensure the quality of functional requirements, the DMM team has applied the following techniques:

1. Peer review: All artifacts (mainly documents, diagrams etc) are created and stored on Microsoft SkyDrive. This provides a facility for all team members to review the contents online at the same-time as well as provide comments on each other's work. Team members work on specific sections of the artifacts and then discuss related topics in a meeting. This technique helps to remove any ambiguity in the requirements and makes sure that everyone understands how the system should behave once implemented.

2. Customer review: After peer review, the DMM team sends requirements and other documentation to the project mentor. The mentor is requested to review the document with a specific perspective (role such as user) as well as an instructor's viewpoint. The mentor's feedback is discussed and included in the document and then sent again for final review.

3. Traceability Checking: Once requirements are documented and reviewed, a requirements traceability matrix is developed. The DMM team intends to use the traceability matrix to trace the source of any requirement as well any requirements changes. The traceability matrix will also help the QA team while testing the application system.

4. Regression Testing: The objective of regression testing is assuring all aspects of an application system work well after testing. Regression testing will be part of DMM's QA plan. Once the bugs are fixed, regression testing will help to ensure that bugs are correctly fixed and that new bugs do not appear.

The DMM team intends to use the following tools for verification and validation of functional requirements:

1. Excel: Microsoft Excel will be used to manage the requirements traceability matrix.

2. Redmine Collaboration Software: The DMM team uses Redmine to prioritize the requirements and assign tasks the team members. This is easily done by creating issues with specific details for the team members by the lead.

### 7.2 TOOLS AND TECHNIQUES FOR ASSURING THE QUALITY ATTRIBUTE REQUIREMENTS

The DMM team intends do verification and validation for the quality attributes that the system must possess. During the design phase, the team has developed quality attribute scenarios and reviewed those with the mentor. After the development phase and during initial implementation of the system, the team will use specific tools to measure whether or not our system meets the quality attributes. These quality attributes are derived from DMM's Software Requirements Specification (SRS) document.

| Quality Attribute | Tool/Technique Used | Rationale for using the tool/technique |
|---|---|---|
| Unit Testing | MSTest is a command line utility within Visual Studio 2012 for running automated tests. | This utility will help in executing automared unit and coded UI tests as well as to view the results from these test runs. |
| Performance | Visual Studio 2012 New Load Test Wizard for load and stress tests and Performance monitor. | These tools will help to meet the system performance requirements during development and in production. |
| Availability | Server and application availability OS commands and logs. | These commands and system logs will help to find the availability of the server and application system. |
| Usability | User questionnaire or surveys. (Note - DMM team members will act as users.) | These techniques will help to understand the user specific requirements and how the system is user friendly. Concept of Operations document that describes various use cases will be useful to refer while testing usability. |

## 8 TESTING STRATEGY

.

Testing for the DMM 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, the testing strategy for the DMM system will consist of four testing levels. These are unit testing, integration testing, acceptance testing, and regression testing. The following subsections outline these testing levels, which development team roles are responsible for developing and executing them, and criteria for determining their completeness.

### 8.1 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 architecture and design. These tests are done one function at a time and written by the developer. Ideally each logic path in the component and every line of code are tested. However, covering every line of code with unit tests is not time or cost effective in most cases. Code coverage goals will be defined to ensure that the most important code is well covered by tests while still making efficient use of developer time. See section 7.5 for specifics on code coverage goals.

Unit testing will be done by the developers during each of the three development phases outlined in the Project Plan from Jun.12 to Jul. 10. All unit tests must be executed and passing before each check-in to the source control system. Unit tests will also be run automatically as part of the continuous integration process. The results of these test runs will be stored by the continuous integration system and emailed to the development team.

## 8.2 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 all at once will be avoided as it makes finding the root cause of issues more difficult and time consuming. Instead, integration tests will be done at specific points, ideally where one component interacts with another through an interface. Integration tests will focus on these specific points of interaction between two components. This testing of interaction between two modules ultimately leads to an end-to-end system test. Each test is written to verify one or more requirements using the scenarios or use cases specified in the requirements document. Integration tests also include stress or volume testing for large numbers of users.

## 8.3 Acceptance Testing

Acceptance testing is functional testing that the customer uses to evaluate the quality of the system and verify that it meets their requirements. The test scripts are typically smaller than integration or unit testing due to the limited time resources of the customer. Acceptance tests cover the system as a whole and are conducted with realistic data using the scenarios or use cases specified in the requirements as a guide.

## 8.4 Regression Testing

The purpose of regression testing is to catch any new bugs introduced into previously working code due to modifications. As such, the regression test suite will be run every time the system changes. Regression tests will be created and run by the testing team. Regression testing 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 “test/fix” cycles will be planned and conducted during regression testing.

## 8.5 Test Completion Criteria

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

- *Unit Testing*: Complete when:
  - At least 90% of the code (including all critical sections) has been tested
  - All major and minor bugs found have been logged and fixed.
- *Regression Testing*: Complete when:
  - At least 90% of code has been covered, including all modified modules,
  - At least two test/fix cycles have been completed.
  - All issues/defects have been logged and corrected.
- *Integration Testing*: Complete when:
  - 100% of module interfaces have been tested.
- *Acceptance Testing*: Complete when:
  - The customer is satisfied that the product has met the agreed upon requirements criteria.

# 9 ORGANIZATION

## 9.1 AVAILABLE RESOURCES THAT TEAM INTENDS TO DEVOTE

The DMM team is comprised of four members, each devoting an average of 40 hours per sprint (1 sprint = 2 weeks) to the delivery of the tool. Due to the small size of the team, most activities need to be dispersed among multiple team members. The implementation of QA activities will follow the same pattern with ten percent of the entire team's time being devoted QA activities.

| Team Members | * | Average hours/sprint | * | QA Percentage | = | Total QA hours/sprint |
|---|---|---|---|---|---|---|
| (4) | * | (40) | * | (.10) | = | 16 hours/sprint |

The 16 hours per sprint will be divided amongst the QA activities as appropriate. The exact designations will depend heavily on the availability of team members and their strengths and weakness in the QA activities.

## 9.2 QUALITY ASSURANCE TEAM

All SQA team members will have access to SQA plans and guidelines to ensure that they are aware of the SQA activities and how their roles and tasks fit within these guidelines. Discussion of QA activities may be scheduled and conducted so that members can discuss roles and responsibilities. In addition, team members will collaborate to select roles for reviews so that they are filled by the team members who best fit the characteristics of the role.

The SQA Leader will be in charge of managing the SQA team and will be the tie breaker when the team hits roadblocks during decision making. The SQA leader will also have the responsibility of ensuring that all other team members are carrying out their responsibilities in the correct manner.
For each activity, team members will have defined roles. The possible roles are defined below.

| Role | Responsibilities |
|---|---|
| Quality Coordinator | Responsible for ensuring all quality activities are planned and carried out accordingly<br><br>Responsible for ensure all team member are properly trained and equipped for their given roles and responsibilities |

| | |
|---|---|
| | Ensures SQA activities align with available resources |
| Module Leader | coordinates activities related to a specific system module |
| Software Quality Leader | Responsible for leading SQA activities(i.e. coordinating reviews and walkthroughs) |
| Quality Reviewer | Reviews and identifies defects in project artifacts<br>Provides feedback for improved quality in software artifacts |
| SQA Team Member | Responsible for providing support during SQA activities by carrying out assigned tasks as they relate to quality of the system |

Throughout the SQA process each team member is responsible for knowing:

- Their Roles and Responsibilities
- Goals of each activity with which they are associated
- Processes that are to be carried out

## 9.3 MANAGING OF THE QUALITY OF ARTIFACTS

When changes are made to the system, reviews/tests will be conducted on the artifacts affected by those changes.

All testing and review activities shall have documentation indicating:

**Process** How a particular method or technique should be carried out

**Goals** This will state the purpose of quality activities associated with the artifacts.

**Results** Outputs of the methods and techniques and Analysis and Conclusions that are formed as a result of them

**Reviewer** Roles and Responsibilities of SQA 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, which enables the team to easily revert to a previous version in the event issues are discovered in connection with said changes.

## 9.4 Process for Prioritizing Quality Assurance Techniques

This section contains a step-by-step outline of the process employed to prioritize the QA techniques used for evaluation of the process artifacts.

1. Create a prioritized checklist of testing characteristics/interests of the system; these will be relative to the requirements and quality attributes.
2. Choose techniques (i.e. design and code reviews) that seem to fit in line with the characteristics identified (i.e. from common knowledge or based on research);
3. SQA team should engage in dialogue 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(in the checklist) that are of interest; the rating will be 1-10 with 1 being the weakest and 10 the strongest
4. SQA team conducts an assessment session of techniques that could be useful for testing purposes; the SQA leader will be in charge of this session
5. Team should come to an agreement about a specific technique and engage in dialogue to address any issues with a particular technique
6. Weighting and majority team agreement should be deciding factor on a technique

## 9.5 QA STRATEGY BREAK DOWN INTO TASKS

| **Tasks** | **Effort** (total hours) | **Exit criteria** | **Deliverables** |
|---|---|---|---|
| **Product realization** | | | |
| Requirement | 2 | ConOps & SRS Reviewed | ConOps & SRS |
| Design | 3 | SAD Reviewed | SAD |
| Coding | 3 | Code walkthrough and formal technical Review | Source with unit tests |
| Verification | 2 | All Critical and Major bugs resolved. | Reports and test source code (if applicable) |
| Validation | 2 | Reviewed and approved by customer | Solution Deployment |
| **Measurement, Analysis and Improvement to SQAP** | | | |
| Process appraisal | 2 | All stakeholder process concerns addressed | Updated SQAP & SPMP |
| **Support processes** | | | |
| Planning | 2 | Planning for a new activity is done by team members | Updated SPMP |

## 9.6 QUALITY ASSURANCE PROCESS MEASURES

Measurements of SQA 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:

- Number of Defects Found
- Defect Find Rate
- Defect Fix Rate
- Defect Density
- Type of Errors Identified (Critical, Major, Minor)

These measures will provide a sense of the relative performance of current methods and provide data to fuel improvement in the process.

**Follow up and Tracking:**
When reviews and testing 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.

**Exit Criteria:**
The exit criteria as defined in the plan depends upon the goals set for the 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.

# 10 GLOSSARY

## 10.1 DEFINITION

| | |
|---|---|
| Audit | An independent examination of a software product, software process, or set of software processes to assess conformance with specifications, standards, contractual agreements, or other criteria. |
| Inspection | A visual examination of a software product to detect and identify software anomalies, including errors and deviation from standards and specifications. Inspections are peer examinations led by impartial facilitators who are trained in inspection technique |
| Review | A process or meeting during which a software product is presented to project personnel, managers, users, customers, user representatives, or other interested parties for comment or approval |
| Walk-through | A static analysis technique in which a designer or programmer leads members of the development team and other interested parties through a software product, and the participants ask questions and make comments about possible errors, violation of development standards, and other problems |

## 10.2 ACRONYMS

| | |
|---|---|
| DMM | Data Mining & Monitoring |
| FTR | Formal Technical Review |
| SAD | Software Architecture Document |
| SRS | Software Requirement Specification |
| | |