

Purpose

This document describes the basic set of practices that a QA Analyst should address and be in compliance with, during the activity of planning for testing of a project. It is used to facilitate communication regarding performance expectations by providing a common frame of reference regarding definitions and expectations.

This document is arranged in the following sections:

[Section 1 – POLICY](#) – “Why” the process exists

[Section 2 – STANDARDS](#) – “What” must happen to meet the intent of the policy

[Section 3 – PROCEDURE](#) – “How” the process could be executed (step-by-step tasks)

[Section 4 – PROCESS CHECKLIST](#) – Auditing the execution of the process

Continuous Process Improvement

This document is a result of research in the industry and sampling of actual QA Analysts’ work regarding test plans. The Policy, Standards, and Procedures outlined in this document are by no means static. The success of maturing our QA organization relies heavily upon the feedback received from individual QA Analysts applying this process to their work.

Any feedback regarding the contents of this document may be sent to: [David Capocci at DAVCAP@SAFECO.COM](mailto:DAVCAP@SAFECO.COM). Please be sure to specify in your comments, the number of the section you reviewed. An advisory board will meet on a regular basis to update this document, based on the feedback and comments received.

Section 1 – Test Plan Policy

1. A test plan is compiled to supply the project team, for whom the testing is being performed, with:
 - A level of confidence in the testing approach and strategies that will be employed during testing on the project.
 - An understanding of the testing process for the particular project.
 - The set of expectations that the project team can have of the test team.
 - The testing deliverables that will be created during the test process.
 - The measurements to be collected during testing, as well as the ways that those measurements will be summarized and reported (metrics).
2. A test plan is compiled to supply the test team with:
 - A roadmap of the strategies and approaches to be executed during testing.
 - A vehicle of communication with the project team regarding the testing process.
 - A tool for allowing accurate estimation (of test effort & schedule), identification of resources, and risk analysis (of the test requirements) to be performed.
 - The set of expectations that the test team can have of the project team.
 - A contract, which can be signed off by the project team, that serves as a statement of the work to be done during the testing process as well as how it will be accomplished.
3. All QA projects (maintenance to new systems development) are required to have a test plan document.
4. All test plans are required to have sign-off by the project team for whom the work is being performed.
5. MEASUREMENTS –
 - 5.1. QA Managers and QA Leads will monitor project work to assure that test plans exist for all projects.
 - 5.2. Internal Audit will include Test Plans as an audit point.

Section 2 – Test Plan Standard

1. Introduction to the Test Plan Standard Section and its contents

1.1. Scope

1.1.1. This section on Standards describes the set of basic parts that every test plan shall address. A QA Analyst is expected to create test plans that comply to the standards described.

1.2. Rule of Scalability

1.2.1. Projects that a QA Analyst participates in can range from a small maintenance change, spanning 1-2 days, or even less, to mid-range projects, typical of enhancements, spanning days or months, to full-scale large projects, typical of new systems, application, or software development. While each of these projects may be vastly different in terms of QA effort and resources, they all have test planning in common.

1.2.2. The Rule of Scalability implies that the depth of test planning documentation is directly proportionate to the demands of the project.

1.2.3. This does not mean that it is acceptable to omit writing a test plan for a very small maintenance change. Rather, it implies that regardless of project size, the QA Analyst must demonstrate due diligence in planning for the testing activities that need to occur.

Example: The amount of small rate change maintenance projects that are worked on are numerous and frequent. They do not each require that an individual test plan be written for them. Due diligence can be shown by documenting a test plan that outlines what testing process will be performed for all rate changes that occur. This “Master Test Plan” can serve as a reference point of the testing process for all QA Analysts who are faced with the testing of a rate change.

Example: A small production fix needs to be tested. The scope is very limited and test time is within an afternoon. Due diligence can be demonstrated by addressing the core areas of test planning via a one page, simple test plan template which can be developed for such instances. In the event of even tighter timelines, demanding an Adhoc testing effort, the template can be completed after testing as a closure activity to document the effort.

1.3. A test plan shall have the following structure.

1.3.1. Table of Contents *(optional for small efforts)*

1.3.2. Project Definition: Background, QC Scope *(optional for small efforts)*

1.3.3. QC Schedule: Target Dates, Milestones & Checkpoints, Activity Date Ranges *(optional for small efforts)*

1.3.4. The Quality Control Approach for the following areas:

- Test Requirements *(Required)*
- Risk Analysis *(Required)*
- Test Environment Preparation *(optional for small efforts)*
- Test Case Design *(optional for small efforts)*
- Test Execution & Results Criteria *(optional for small efforts)*
- Test Reporting *(optional for small efforts)*
- Defect Management *(optional for small efforts)*

1.3.5. Risks & Dependencies *(optional for small efforts)*

1.3.6. Unresolved Issues *(optional for small efforts)*

- 1.3.7. Notes (*optional*)
- 1.3.8. Resources (*optional for small efforts*)
- 1.3.9. Revision History (*optional for small efforts*)

2. The Test Plan Standard

2.1. Table of Contents

- 2.1.1. Summarize the contents of the test plan and provide page number references

Usage Tip: Use the Table of Contents tool found in the Index and Tables command of Microsoft Word to build your Table of Contents.

2.2. Project Definition: Background, QC Scope

- 2.2.1. This section should provide a high-level definition of the project.
- 2.2.2. This section should describe what this project would provide to the user in terms of functionality.
- 2.2.3. This section should provide the scope of work that testing will focus on.
- 2.2.4. Questions that this section should answer include:

- What is the high level definition of the project or sub-project detailed in this document (i.e. what are you doing)?
- What is the scope? What is the benefit?
- What is the business need?
- How does this component work at a high level?
- Why is the project/component necessary? I.E. why are we doing this?
- What is expected from QA on this project?

Usage Tip: Think of reusability! Write this section so that any QA who may have to test this in the future can pick up this test plan and get a good sense of this project by reading this section. It should be written for a person who is not familiar with the application/system being tested.

2.3. QC Schedule: Target Dates, Milestones & Checkpoints, Activity Date Ranges

- 2.3.1. This section should provide specific schedule information regarding all the activities that will occur during the testing process. Dates should be set as a result of estimating based upon test requirements, risk analysis results, project scope, and typical test process activities. The QC Schedule should set expectations of when test deliverables will be provided as well as when other project team roles may be impacted by test activities.

- 2.3.2. List all milestone dates set by the project team

Example: Launch Date, Dates for Test Level Moves (unit to Integration/function testing, integration to systems testing, systems to acceptance testing), Dates for Test Environment Moves (TDV to TIN, TIN to TAC, FLEX, IDAY or PARA, etc.), Dates for test cycles

- 2.3.3. List all Test Deliverables Dates

Example of deliverables: Test Requirements Hierarchy Document, Test Cases, Test Reports

- 2.3.4. List all Test Deliverable Verification Dates

Example: Test Plan Review, Test Requirements Review, Test Cases Review

- 2.3.5. List all timeframe events. These are date ranges for activities to occur within.

Example: Test Requirements Investigation and gathering will occur from January 1st to January 30th. Test Case Development will occur from February 15th to February 28th. Others can include...automation piloting timeframe, risk analysis timeframe, test execution timeframes. Load testing in the Test Lab will occur from 12/17-12/21.

Usage Tip: Estimate dates by examining the processes that you will need to perform (i.e. the work or tasks you need to do and what it will take to do them) and estimating them individually. Don't try to give an estimate for the entire testing process, rather focus on estimate the sub-processes within it.

2.4. The Quality Control Approach

2.4.1. This section represents the core processes that will be performed during testing. Each sub-section should detail:

- What it will take (the steps required, i.e. the process you will use) to accomplish the goal or create the deliverables that are the output of that sub-section.
- What are the inputs needed in order to produce the deliverables for that section?
- What are the outputs (deliverables) that you hope to produce from that section?
- What, if any, are the measurements that will be tracked and metrics that will be generated?

2.4.2. Test Requirements Strategy (Features to be validated)

2.4.2.1. Document what process is required in order to generate the test requirements in an organized format that will allow them to be measured for test coverage.

Usage Tip: How a test requirement hierarchy will be produced, is typically, what is described in this section. The test requirement hierarchy is usually a separate document(for larger projects) that will get created as part of the testing process.

Usage Tip: A Test Requirement is also referred to as "feature to be tested", "functional decomposition", or "test inventory".

2.4.2.2. Typical inputs required for this are: Functional Requirements, Technical Requirements, flowcharts, data flow diagrams, Business Analyst or Programmer interviews, emails, meeting notes.

2.4.2.3. A typical procedure may involve:

- a) Setting the scope of the project that you will have to discover the test requirements within.
- b) Creating the repository for storing the test requirements (anything from a word document to an access database) that will allow any measurements to be tracked and traceability to test cases to be established.
- c) Reviewing available project materials
- d) Arranging interviews with various project team members
- e) Compiling the discovered test requirements in a document
- f) Having the test requirements reviewed (verified) by the project team

2.4.2.4. The typical output (deliverable) that will be created from this procedure will be a Test Requirements Hierarchy Document.

2.4.2.5. The following are possible Metrics that can be reported:

Test Coverage Percentage = The total weight of validated test requirements divided by the total weight of all test requirements.

2.4.3. Risk Analysis

2.4.3.1. The goal of performing a risk analysis on test requirements is to identify what areas have more weight and may demand more attention than others during testing.

2.4.3.2. Document what work is required in order to perform a risk analysis on the discovered test requirements.

2.4.3.3. Individual Risk Factors can be listed in this section.

Usage Tip: This is a good place to document any involvement that the project team may have in your risk analysis. For example, having a meeting to brainstorm and identify the risk factors.

2.4.3.4. The Risk Analysis method can be documented in this section.

2.4.4. Test Environment Preparation

2.4.4.1. Document all the data preparation needs that will be required in order to execute the test cases. This may include items such as how databases should be populated, or production data will be used. It may also include data that comes from upstream or downstream systems that will need to be in place.

2.4.4.2. Document any environmental conditions that need to be satisfied in order for test execution to occur. This may include additional hardware or software that may be required to mirror a production environment. It may also include setting up new test libraries in the mainframe TSO area or new server configurations in the client/server area.

2.4.4.3. Document any training needs that may be required for QA staff in order to understand what will be tested. For example, new development technology requires QA Analysts to be trained in how to test that technology.

2.4.4.4. Document any automation or automated testing tools that will be utilized during testing along with what setup may be required (test script design).

2.4.5. Test Case Design

2.4.5.1. Document any specific black box or white box approaches or other strategies that may be employed in order to design the appropriate test cases for different areas to be tested.

2.4.5.2. Document any properties that you will track for test cases. This may include:

- *Case ID*
- *Status (see Results Review section)*
- *Data values for the test case*
- *Steps, if required*
- *Expected Results*
- *Actual Results*
- *Additional notes regarding any setup or preparation issues*
- *The Test Requirement it is covering*
- *Dates (test case created, passed, failed, etc.)*

2.4.6. Test Execution & Results Criteria

2.4.6.1. This section should document any strategies for accepting a build or version of software into test, commonly referred to as “smoke” testing. The entrance criteria for determining “stable enough to perform full testing” should be outlined.

2.4.6.2. If multiple builds/versions are to be generated, or mainframe components moved to the test environment, this section should outline the schedule for performing such turnover events.

2.4.6.3. Test results should be tracked to reveal the pass/fail status of the test. The following are possible status indicators to use:

- *“PLANNED”*: Test planned, but not in fully executed form yet
- *“AVAILABLE”*: Test created, but code or logic not in place yet
- *“PASS -I”*: Where “I” = “Initial Run”, Success upon first run
- *“PASS -R#”*: Where “R” = “Re-Run”, Success upon retesting, where # is how many times it took a fix to allow this test to pass.
- *“FAIL -I”*: Defect generated
- *“FAIL -R#”*: Retest of Defect fix failed, where # is how many times this retest has failed (possible indicator of complexity or mismatched requirements)

2.4.6.4. The following are possible measurements that can be tracked and gathered from execution of test cases:

- *Total number of test cases or scenarios: Planned (but not yet created), Available (created, but application not yet available to run them against), Executed (both passed & failed)*

2.4.6.5. The following are possible Metrics that can be reported:

- *Test Progress Graph (trend analysis): A plot of the number of planned, passed, and failed test cases over time.*
- *Percentage failure rate of test cases: Number of test cases failed / # of executed test cases. (This will represent test case failures due not only to defects, but other causes like automation or execution error.)*
- *Percentage executed: Total # of Test Cases Executed/ Total# of Test Cases. (This will represent the percentage of test cases that have been executed, both passed and failed. This may be useful to show in the event that overall test coverage is low due to a high number of defects being uncovered by failed test cases.)*

2.4.7. Test Reporting

2.4.7.1. This section should outline what a test status report will include and how often it will be reported during testing.

Usage Tip: You can include a sample of the report you will use or include a link to where the report may be accessed during the testing effort.

2.4.7.2. This section should outline what metrics will be included on the test status report. Among the others listed in this standard, an additional metric to include:

- *TEST EFFICIENCY % = (Hours spent on Verification Activities + Hours spent on Planning) / (Hours spent on Verification Activities + Hours spent on Planning + Hours spent on Execution + Hours spent on Rework)*

2.4.7.3. This section should outline how QA will sign-off on the project when testing is considered complete.

2.4.8. Defect Management

2.4.8.1. Document the defect tracking process (not to be confused with the defect tool that will be used) that will be used to report and track defects during the course of the project. This section should outline what the audit trail will be like for a defect entered into the defect management tool. This section should also outline any expectations QA has regarding defect fix turn around times.

2.4.8.2. Document and define what properties will be tracked for defects.

2.4.8.2.1. All defects should be tracked for: Priority (When to fix it), and Severity (the impact of the defect)

2.4.8.2.2. Typical definitions for severity are:

- *Product – This defect affects the entire product or application*
- *Function – This defect affects the functional area being tested*
- *Scenario – This defect affects the particular scenario or case being executed*
- *No Impact – This is usually a cosmetic or spelling error type defect.*

2.4.8.2.3. Typical definitions for priority are:

- *High - Correct immediately. It is impossible to continue any testing in this area until the problem is resolved. On occurrence of a “product” level severity, the Test may be abandoned if the problem cannot be resolved rapidly. Defects with this priority must be resolved and tested prior to launch.*
- *Medium - A Test Script cannot be completed, other Test Scripts may or may not be affected, but, in general, testing can continue in this area. Defects with this priority must be formally reviewed and have management signoff if not resolved prior to launch.*
- *Low - Release and re-test can be postponed until all other test scripts are complete, higher priority defects are corrected, or can be coupled with a fix of a higher priority defect. Defects with this priority must be reviewed and have management signoff prior to launch.*
- *Track - Track for recurrence. This appears to be a non-repeatable incident. Defects receiving this priority can be placed in a postponed status regardless of severity indicator.*

The QA tester, before the entry of a newly discovered defect, will review defects with this priority.

2.4.8.2.4. Defect Origin – Where can the defect be traced back to? (Problems with Requirements, Design, Coding, or Tests?)

2.4.8.2.5. Defect Discovery – What phase was the defect discovered in? (Verification prior to coding, Test execution, Production?)

2.4.8.3. Document what authorities each project team role will have during the life of a defect.

Typical authorities are: Entry (open a new), Assign, Investigate, Resolve, Validate or Fail, Close

2.4.8.4. The following are possible measurements that can be tracked and gathered from Defect Tracking:

- *ESTIMATED and ACTUAL HOURS for the investigation of a defect, the actual coding of a fix (resolution), and the verification of the fix*
- *Number of defects by priority or severity*

2.4.8.5. The following are possible Metrics that can be reported:

- *MEAN TIME TO RESOLVE – The average time between discovery of a defect and its resolution to be incorporated into a new build. (For estimation of schedule impacts)*
- *MEAN TIME TO CLOSE – The average time between discovery of a defect and its close after retesting. (For estimation of schedule impacts)*
- *TEST EFFECTIVENESS % = (# of Defects discovered in Verification Activities + # of Defects discovered in Test) / (# of Defects discovered in Verification Activities + # of Defects discovered in Test + # of Defects discovered in Production)*

2.5. Risks & Dependencies

2.5.1. This section should list any limitations, dependencies, influences, etc. that will have an impact on the testing effort or influence the estimated schedule.

For example, “test databases will need to be cleared & reset before each automated regression test cycle”, “performance testing will need to occur in the test lab according to lab availability”.

2.5.2. State any other potential risks to testing (besides the risk factors used for performing a risk analysis on test requirements). This can be concerns, or assumptions that the test team is making about the project, project team, resources, schedule, technology being used, etc.

2.5.3. State any reasons for why areas are NOT being tested. (outside of scope, lack of resources, skill sets, proper tools, etc.)

2.6. Unresolved Issues

2.6.1. This section is useful to place issues in a “parking lot” for further discussion.

For example, “the functional requirements for the management reporting area are not yet known, if they are delivered to QA beyond 12/01/01 the testing schedule will be impacted as additional test planning time will need to be added” or “the billing area has unresolved issues dealing with how taxes will be handled, we are waiting for the business analyst to give us the tax table information”.

2.6.2. Keep track of items via a date stamp and any additional reference point (like “from design review meeting”, resolved date, TR affected)

2.7. Notes (optional)

2.7.1. This section can document any QA related notes or reminders to consider when testing this project. This is useful for future reuse of the test plan.

2.8. Resources

2.8.1. This section should document all the resources that are available to the test team during the project.

2.8.2. All documents that are sources or references for the testing team should be listed and include the following: Document title, file name, storage location, author or contact.

2.8.3. All people resources for the test team should be listed along with the following information: Name, Phone Number, Role/Responsibility description, Email ID.

Usage Tip: Don't limit the listing of people to just those that are part of the project team! Include: subject matter experts (SME's), customer or field contacts, technical support personnel, tool vendor support information (for automated tool usage), etc.

2.9. Revision History

2.9.1. This section established an audit trail for the test plan.

2.9.2. This section should document any time the test plan is updated, reviewed, edited or changed.

Section 3 – PROCEDURE

Introduction to the Test Plan Procedure Section and its contents

Scope

This section on the Test Plan Procedure describes the set of tasks that are typically executed in order to create the test plan deliverable.

Introduction

A Procedure describes the set of steps required to perform the work in order to produce a given output or deliverable. The general procedure for producing a test plan is detailed below. These steps represent the typical tasks that a QA Analyst usually performs when preparing a test plan. While shown at a great level of detail, they should be seen as a guideline for QA Analysts.

Sequential vs. Parallel steps

It is important to understand that while the following steps are placed in a given sequence, depending on the project, many of the steps may be performed independently. That is, it is not necessary to complete a prior step before the next step is started. It is possible to address several steps at the same time, or in parallel to other steps.

The steps to the procedure

1. Identify the scope of the testing effort for the project.
2. Determine the appropriate level of planning necessary & select template.
3. Fill out the appropriate test plan template with basic information.
4. Determine the strategy for identifying the test requirements.
5. Begin identifying high-level test requirements (parallel task)
6. Determine high-level, overall testing strategies.
7. Perform initial Risk Analysis on high-level test requirements (parallel task)
8. Formulate an initial estimate.
9. Refine Test Requirements (parallel task)
10. Perform detailed Risk Analysis of known test requirements (as needed)
11. Identify the Test Priorities based on risk analysis
12. Detail out testing strategies (as needed)
13. Identify resources
14. Create the schedule and final refined estimate (as needed)
15. Compile the test plan
16. Perform a desk check of the test plan and make corrections.
17. Perform a test plan review and make revisions.
18. Obtain project team signoff.

The Test Plan Procedure

1. Identify the scope of the testing effort for the project.

- 1.1 When first assigned to a project, the QA Analyst should determine what is within the scope of testing. This may include understanding what the project entails and what the expectations are regarding the involvement of QA.

- 1.2 Results of this step can be documented in the test plan section discussing Project definition, background, and QC scope.
- 2 Determine the appropriate level of planning necessary & select template.**
 - 2.1 During this step, the QA Analyst should decide the level of detail that will need to be provided in the test plan that is appropriate for the project.
 - 2.2 For large projects, an initial estimate of how much time would be required to produce a test plan for the project may be necessary.
 - 2.3 For small projects, it may be possible to use a scaled down version of a test plan template that suits the project's scope.
 - 2.4 For ongoing product maintenance, a master test plan could be produced that outlines the main processes that will be used for all areas of Quality Control, then smaller, possibly single-page test plans could be drafted for individual maintenance requests that apply the processes from the Master Test Plan.
- 3 Fill out the appropriate test plan template with basic information.**

Once an appropriate template is decided upon, the basic information should be filled out for the project.
- 4 Determine the strategy for identifying the test requirements.**

Based on the project size and team dynamics, the QA Analyst will need to determine how test requirements will be gathered. See the Test Plan Standards section for guidelines to follow.
- 5 Begin identifying high-level test requirements (parallel task)**

Upon identifying the strategy for gathering test requirements, this process can then be started as the rest of the test plan is drafted.
- 6 Determine high-level, overall testing strategies.**
 - 6.1 As test requirements are being identified, and the scope of testing is firmed up, the QA Analyst should begin to determine what approaches & strategies should be used to cover these test objectives.
 - 6.2 It is during this step that the QA Analyst begins to address the involvement of automation, the requirements for the test environment, data needs, the level and depth of testing that needs to be obtained for different areas of the project, test case design, test execution, test reporting, defect management, etc.
- 7 Perform initial Risk Analysis on high-level test requirements (parallel task)**

As testing strategies are thought out, a risk analysis on the high-level areas of test requirements may assist in ferreting out whether some areas will require a more concentrated test effort than others.
- 8 Formulate an initial estimate.**
 - 8.1 Up until this step, much of the test planning work has been at a high-level: defining scope, organizing overall test requirements, assessing overall testing strategies, and beginning to understand areas of risk.
 - 8.2 Armed with this knowledge, the QA Analyst can begin to address an initial estimate of what it will take to perform the work of testing this project.
 - 8.3 Submitting this estimate to the project team for review should encourage dialog between the project team and the QA Analyst. The QA Analyst should adjust their approaches according to the feedback received.
 - 8.4 For example, if the project team desires that a lower estimate be achieved, the QA Analyst would be able to provide information regarding what areas could be cut or diminished in terms of testing effort.
 - 8.5 It may mean that a different scaled-down version of testing is what the project team is willing to invest in. Having the information from the previous procedural steps will help the QA to explain what testing options are available to the project team.
 - 8.6 The estimate can be detailed out in the QA Schedule section of the test plan and should be used to develop the testing timeline, target dates and deliverables.
- 9 Refine Test Requirements (parallel task)**

- 9.1 As the project team provides further requirements, more granular test requirements can be documented.
- 9.2 Documenting and refining test requirements is an ongoing process that is parallel to documenting the test plan. However, knowledge of test requirements is necessary in helping to determine test priorities and strategies.

10 Perform detailed Risk Analysis of known test requirements (as needed)

If necessary, a more detailed risk analysis can be performed to gain a sense of those areas of the project that have a need for greater attention. This is useful only if the test requirements did not provide enough detail when an initial risk analysis was performed.

11 Identify the Test Priorities based on risk analysis

Based on the results of any risk analysis, the QA Analyst can now prioritize what areas of the project need to be addressed first during testing. High priority areas are usually at a higher risk of containing defects.

12 Detail out testing strategies (as needed)

Given the results of priority setting and any risk analysis, the QA Analyst may want to detail out test strategies that address particular project concerns or add more information regarding the approaches that the QA will take to develop and execute test cases.

13 Identify resources

- 13.1 As estimates are solidified, QA resources can be identified or adjusted as necessary.
- 13.2 Resources should include both QA specific personnel needed as well as other personnel that QA will need assigned in order to perform the work required to test the project.
- 13.3 It is possible that non-QA resources may be needed. For example, a programmer who is needed to create test specific utilities for QA to use in order to test certain aspects of the project.

14 Create the schedule and final refined estimate (as needed)

With resources known, the QA Analyst can map out what the test schedule will look like for the project. This should be documented in the QC schedule portion of the test plan.

15 Compile the test plan

All the information from the above procedural steps should be compiled into the appropriate areas of the test plan.

16 Perform a desk check of the test plan and make corrections.

- 16.1 Once compiled, the test plan should be submitted for review by other QA Analysts.
- 16.2 The main concern of the desk check is to make sure that the test plan addresses all the items that are typical of a test plan. Checking the test plan for compliance to standards is the focus.
- 16.3 The QA Analysts who are checking the test plan should also look for readability, reuse, and, if familiar with the project, any missing pieces that should be addressed.

17 Perform a test plan review and make revisions.

- 17.1 After correcting any issues revealed from the desk check, the QA Analyst should plan a review meeting with the project team. The Test plan should be sent to the team ahead of time to allow them to read and comment on the test plan.
- 17.2 The objective of the review is to sell the test process for the project to the project team.
- 17.3 Any input received during the review should be incorporated into the test plan. Any disagreement with the project team and the testing process for the project should be documented.

18 Obtain project team signoff.

- 18.1 Once any revisions are made, the QA Analyst should obtain final signoff by the project team.
- 18.2 The test plan should act as the contract defining the statement of work that the QA Analyst has proposed and the project team has agreed to.

Section 4 – THE PROCESS CHECKLIST

High Level Checklist

✓	Ref.	Checklist Items
	1.2	Is the test plan scaled appropriately to the project size? If the project is of a production support, maintenance, or fix nature, does a master test plan exist that addressing how testing will be accomplished?
	1.3	Does the test plan include a QC schedule with Target Dates, Milestones & Checkpoints, Activity Date Ranges?
	1.3	Does the test plan address the process that will be used for gathering test requirements?
	1.3	Does the test plan address the process that will be used for assessing risk on test requirements?
	1.3	Does the test plan address how the test environment will be prepared (data needs, architecture needs, training needs)?
	1.3	Does the test plan address the approach to how test cases will be designed?
	1.3	Does the test plan address the entrance criteria that the application/system needs to meet in order for full testing to begin?
	1.3	Does the test plan address how test reporting will be accomplished and what will be included in the test report?
	1.3	Does the test plan address how defect management will occur for the project?
	1.3	Are risks and dependencies clearly documented in the test plan?
	1.3	Are unresolved and open issues able to be clearly documented in the test plan?
	1.3	Are all the resources (documents, people) needed for the testing process identified in the test plan?
	1.3	Is there a clear way that the test plan tracks revision history when changes are made to the test plan?
	Policy – 4	Has the test plan received final sign-off by the project team?

Detail Level Checklist

✓	Ref.	Checklist Items
<i>Test Plan Size & Scale</i>		
	1.2	Is the test plan scaled appropriately to the project size? If the project is of a production support, maintenance, or fix nature, does a master test plan exist that addressing how testing will be accomplished?
	2.2	Determine if level of planning effort was appropriate for the scope and extent of the project by: <ol style="list-style-type: none"> 1) Comparing the project definition stated in the test plan against what is stated in the project charter or business case. 2) Assessing the reasonableness of how the test planning scope fits into the higher level objectives of the project. 3) Assessing the reasonableness of testing deliverables against project objectives.
	2.2	Determine if the test plan template used is appropriate for the level of planning necessary by: <ol style="list-style-type: none"> 1) Reviewing the components of the final test plan to determine if it includes all necessary/expected components best suited for the <u>size</u> and <u>nature</u> of the project. 2) Comparing usage of the template on other projects based on the following factors: <ul style="list-style-type: none"> • Time estimates • Deliverables • Total # of employees assigned to the project

✓	Ref.	Checklist Items
<i>Test Plan Dates & Estimates</i>		
	1.3	Does the test plan include a QC schedule with Target Dates, Milestones & Checkpoints, Activity Date Ranges?
	2.3	Ensure that a reasonable amount of time was estimated to fulfill test planning objectives by: <ol style="list-style-type: none"> 1) Ensuring that total budgeted time was based upon detailed analysis of estimated time it will take to test each function identified for critical and non-critical testing. 2) Providing some allowance for emergencies. 3) Considering prior experience with similar projects.
	2.3	Does the test plan list all the test deliverables that will be created and specify: <ol style="list-style-type: none"> 1) What timeframe each will be created, including start date and completion date 2) When, if applicable, the test deliverable will be reviewed and signed off by the project team or other appropriate audience
	2.3	Does the test plan specify when all testing activities will occur including:

QA Test Planning: Policy, Procedure, & Standards

Version 2.0 Last Revised: 6/12/02

		<ol style="list-style-type: none"> 1) Test Requirements gathering/organizing 2) Test Environment preparation 3) Test Case Designing 4) Test Execution
	2.3	Does the test plan specify when the application/system will move between test levels during test execution?

✓	Ref.	Checklist Items
<i>Test Plan- QC Approach: Test Requirements and Risk Analysis</i>		
	1.3	Does the test plan address the process that will be used for gathering test requirements?
	2.4.2	Ensure that the test requirements process to be used addresses: <ol style="list-style-type: none"> 1) The basis for where test requirements will be prepared from (functional requirements, technical requirements, flowcharts, data flow diagrams, meeting notes, interviews, product demos, etc.) 2) The approach for how test requirements will be documented from initial discovery to final sign-off 3) What properties or attributes will be tracked on all test requirements 4) How measurements and metrics (including test coverage) will be incorporated 5) How the test requirements will be reviewed by the project team (throughout their development? One-time at the end?) 6) The level of involvement the QA Analyst expects to have with the project team in gathering and organizing the test requirements
	1.3	Does the test plan address the process that will be used for assessing risk on test requirements?
	2.4.3	Ensure that priority is set for critical areas of testing by determining if: <ol style="list-style-type: none"> 1) Test plan includes a description of the basis and method for identifying critical test requirements (how risk analysis will be performed). 2) An explanation for gauging the significance of an item is documented in the test plan. 3) Sufficient time is available to perform the required testing for critical test requirements by comparing approved budgeted hours against the estimated cumulative number of hours to perform critical test requirements. If insufficient, revisit with project team the goals and objectives set for testing requirements.

✓	Ref.	Checklist Items
<i>Test Plan- QC Approach: Test Environment & Test Cases</i>		
	1.3	Does the test plan address how the test environment will be prepared (data needs, architecture needs, training needs)?
	2.4.4	Does the test plan ensure testing dependencies (data, databases, hardware and software, training, automation tools, etc.) have been identified and their

QA Test Planning: Policy, Procedure, & Standards

Version 2.0 Last Revised: 6/12/02

		availability assured at the time they are required? 1) Are dependencies along with their source/purpose itemized? 2) Are procedures and commitments obtained from people who agreed to provide needed dependencies at a set date for a test to run documented? 3) Has a contingency plan been identified in case any of the dependencies are not available at a set time period?
	2.4.4	Has an automation strategy or strategy for using a particular tool been specified? 1) Are there any conflicts between the tool being used and the application/system being tested that would prevent its use?
	1.3	Does the test plan address the approach to how test cases will be designed?
	2.4.5	Does the test plan specify what properties or attributes will be tracked on all test cases?
	2.4.5	Does the test plan describe any specific techniques that will be used to derive test cases? (black/white box, scenario/matrix, use case, etc.)

✓	Ref.	Checklist Items
<i>Test Plan- QC Approach: Test Execution</i>		
	1.3	Does the test plan address the entrance criteria that the application/system needs to meet in order for full testing to begin?
	2.4.6	Does the test plan ensure that test cases will be traced back to the test requirements they are covering?
	2.4.6	Does the test plan specify how the status of test cases will be tracked during their execution?
	2.4.6	Does the test plan describe what measurements will be gathered regarding test cases?
	2.4.6	Does the test plan describe how the measurements being gathered on test cases will be reported or used?

✓	Ref.	Checklist Items
<i>Test Plan- QC Approach: Test Reporting</i>		
	1.3	Does the test plan address how test reporting will be accomplished and what will be included in the test report?
	2.4.7	Does the test plan specify the frequency that test reporting will occur (daily, weekly, monthly, etc.)?
	2.4.7	Does the test plan describe what will be included in the test report?

✓	Ref.	Checklist Items
<i>Test Plan- QC Approach: Defect Management</i>		
	1.3	Does the test plan address how defect management will occur for the project?
	2.4.8	Does the test plan document and define what properties or attributes will be tracked on all defects?

QA Test Planning: Policy, Procedure, & Standards

Version 2.0 Last Revised: 6/12/02

	2.4.8	Does the test plan document what the workflow or life cycle of a defect will be from discovery to closure?
	2.4.8	Does the test plan describe the authorities that the different project team roles will have in resolving defects?
	2.4.8	Are metrics and measurements regarding defects that will be reported defined in the test plan?
	2.4.8	Does the test plan address escalation procedures or contingency plans if defects are not being fixed in a certain amount of time?

✓	Ref.	Checklist Items
<i>Test Plan- Other</i>		
	1.3	Are risks and dependencies clearly documented in the test plan?
	1.3	Are unresolved and open issues able to be clearly documented in the test plan?
	1.3	Are all the resources (documents, people) needed for the testing process identified in the test plan?
	1.3	Is there a clear way that the test plan tracks revision history when changes are made to the test plan?
	2.5	Does the test plan ensure that subsequent changes to the test plan are communicated to the whole team?
	2.5	Does the test plan provide an explanation as to why certain areas will not be tested?
	2.6	Does the test plan determine how unresolved or open issues will affect the completion of the testing effort if no closure is reached on them?

Test Plan Policy, Standards, Procedure Version History

Version #	Revised Date	Description
1.0	12/6/01	Initial version release
2.0	5/29/02	Process Checklists introduced, readability issues addressed, formatting changes made, test requirements section revised, minor corrections introduced.
2.0	6/12/02	Review Feedback incorporated