COMPANY

QE Workflow and Methodology

Quality Engineering Standards and Practices

SIGN-OFF SHEET

The following approvals are required to initiate the COMPANY Quality Engineering Workflow and Methodology document for all future quality engineering projects:

Approval Name:	Department	Approval Signature:	Date Approved:

Other Project Participants for Distribution Only:

Development Team
Quality Assurance Team
Project Managers
Program Managers

Quality Engineering Standards and Practices

PREFACE

This is the Quality Engineering Workflow and Methodology document.

Purpose

The purpose of this document is to describe COMPANY's quality engineering standards and practices.

Audience

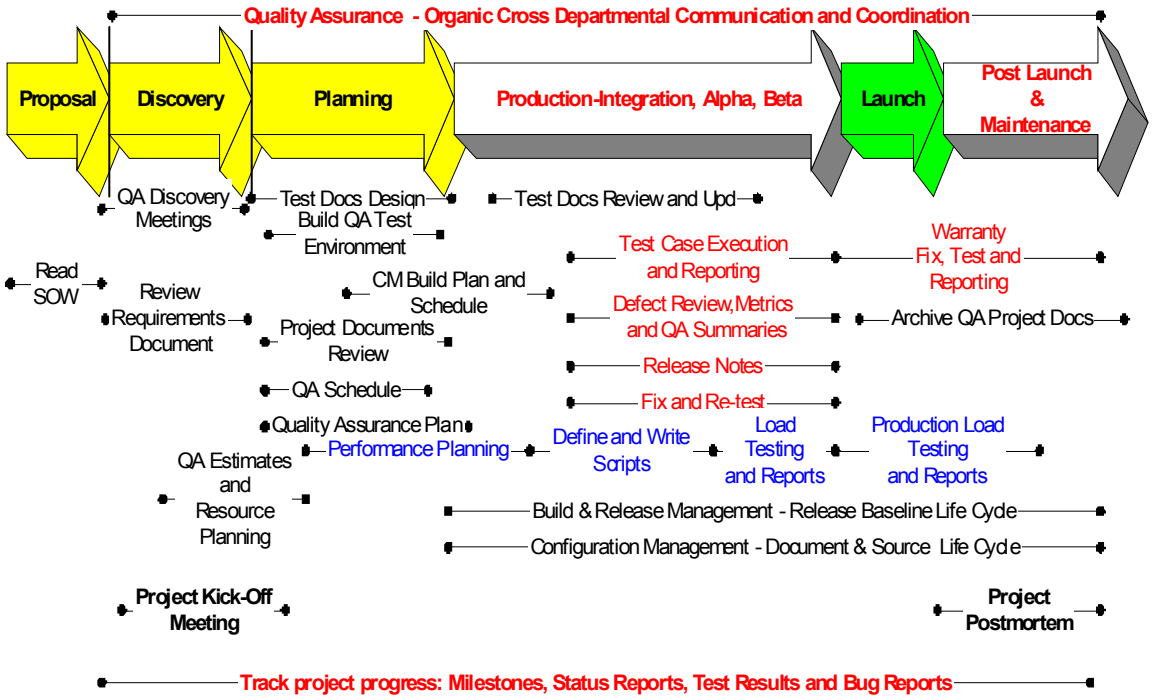
This plan is intended for use by the Engineering- Quality Assurance project teams to manage QE projects and to ensure that a quality product is developed before installing in a production client environment.

Revision History

Version	Date	Who	Revision
V1.0	12/27/00	FAL	
V1.1	4/12/01	FAL	Revisions based on QA input.

1.0 DESCRIPTION

The Quality Engineering Workflow and Methodology document describes the processes for testing and verifying COMPANY client products. The objective of this document is to provide a planned approach towards implementing specific quality assurance methods, which will be utilized across all COMPANY testing projects.



2.0 COMPANY QUALITY ENGINEERING WORK FLOW

2.1 PROPOSAL AND DISCOVERY

The Proposal is the beginning of a business relationship with the client. Selected project team members work with the client to map out what the client wants accomplished. Client information is collected, technologies and business processes are discussed, requirements start to take shape and a contract outlining the execution of the project is created.

The Discovery phase is the time for QA to participate in any project meetings. This time is used as an opportunity to work with and meet other project team members to collect client information. During this phase QA is doing research and becoming familiar with client requirements.

The Project Kick-off meeting allows each team member to get to know other team members, to collect project information, understand the tasks of the project and the time line for the execution of tasks. The QA Project Lead will work with the PE to ensure that the QA Team is involved in the Project Kick-off meeting.

Quality Assurance (QA) Tasks, Deliverables and Tools:

- Allocate time for participating in Discovery phase to collect information for the Planning phase.
- Become familiar with the project team for future communications and coordination efforts.
- Become familiar with client requirements.
- Research areas that may contribute to the successfulness of the QA effort.

Tools: None

Deliverables: From the Project QA Leader: A general statement as to what the QA requirements will be for the project. This would be a QA section in the Engineering Requirements document delivered to the client.

2.2 PLANNING

The Planning phase allows the project team to review the client's requirements and to start writing all of the required project documentation, including a project schedule with allocated resources. The QA Project Leader work with the PE to provide a QA project schedule, QA resources and time allocated for each team member. Use the QA Project Estimation form for determining hours for each QA Task. Time and resources required for the project is give to the PE and PM. A Quality Plan is written that defines the overall strategy of the QA effort. The QA Project Leader uses the QA Project Leader Guide to manage QA resources, execute the plan and provide reports to the project team.

During this phase of the project, team members work in a collaborative effort to plan out all the activities of the project. Information is shared between various disciplines and is disseminated to the project team. Communication between project members is critical to the success of this phase and every other phase of the engineering process. Status reporting, resolving project issues and keeping the project team informed will make this effort a success.

QA project team members are involved in every phase of the COMPANY engineering work flow and it is the responsibility of the QA Project Leaders to ensure that this happens by working closely with the assigned Project Engineer and other members of the team.

Quality Assurance and Release Engineering Tasks, Deliverables and Tools:

- Review client requirements.
- Determine resources and time required to meet the objectives of the project.
- Review Project Schedule and ensure that QA tasks and time lines are considered.
- Use the QA Project Estimation form for determining hours by task. This is a working sheet between the QA Project Leader and the PE.
- Review project documentation in preparation for writing QA documentation.
- Follow QA Project Leader guide in preparation for managing the project.

Quality Engineering Standards and Practices

- Write the Quality Plan for the project. See Quality Plan template.
- Start to write Test Cases based on current information. See Test Case templates.
- Use Test Director for:
  - Create a test database and framework to store all test cases
  - Add manual and automated test cases
  - Run integrated tests
  - Customize graphs and reports

- Coordinate with the Project Engineer and WebOps to build out the QA test environment.
- Secure the diagram of the client system architecture to familiarize the QA Team with the target system.
- Become familiar with the back-end database data model to determine test data requirements.
- Work with the DB Engineer to secure a data flow diagram of expected database outputs and inputs for preparation of black and white box testing requirements. TOAD will be used to query the back-end database.
- Determine client performance requirements for stress and load testing efforts.
- Write the Project Automation Plan: Performance and Functional Testing. See Project Automation Plan: Performance and Functional Testing template.
- Define the transactions and determine a plan for writing performance test scripts.
- Select the written Test Cases to be used for coding automated functional test scripts for automated regression testing efforts. Initiated when the product is stable and if project time is allowed.
- Store QA documentation into designated area.

Tools:

- Standard QA documentation templates.
- QA Project Leader Guide
- Test Director
- SQL: Toad other SQL tool
- MS Project
- Visio Diagramming

Deliverables:

- From the QA Project Leader: Quality Plan, Estimation of Project Hours and QA Schedule
- From the PE with input from Release Engineer: Build Plan and Schedule
- From the PE: All Project Documentation
- From the DB Engineer: Project data model and data flow diagram
- From the Automation Project Leader: Project Performance Plan: Load, Stress and Functional Testing
- From the Automation Project leader: Identification of transactions for Performance Test Scripts
- QA Team: Start of Test Case documentation

2.3 PRODUCTION

The Production phase of the engineering workflow is the execution of the Planning phase. This is the phase where all team members execute the various tasks assigned within a specific time frame, as described in the project schedule and Quality Plan. Code is developed, builds are created, tests are executed, defects are written and reports are generated. This is an iterative process and it is a cycle that is executed throughout the Production phase.

During this phase, the main objective of the QA testing team is to ensure that the client's requirements were met as specified. It is also possible that client requirements will change as the client product evolves. This possibility makes it imperative that project documentation gets updated, changes get documented and concise release notes, with each build, are provided to the QA team, so that changes are adequately reviewed and tested. The QA Team will also focus on making sure that client changes do not effect current functionality, by conducting a regression test of the client product. In order to ensure a quality product, QA time allocated to accomplish this task must be adhered to. If time is not available to conduct a thorough regression test, then a modified regression test may be implemented by strategically determining what functional changes will affect current functionality.

All builds created by the RE team will be smoke tested and then tested by the QA team. QA should only smoke test key milestone builds, the ones being submitted for official QA testing.

The testing of the client product may be conducted by building the product in phases. Create a little code, build and integrate the code and test the product. Then create and build more functionality, integrate and test some more, until the entire product is integrated and is ready for Launch into the client production environment. This methodology of testing will occur only for some projects and not all projects.

Quality Assurance Tasks, Deliverables and Tools:

- Complete writing Test Cases.
- Complete Performance Test Scripts.
- Implement the Quality Plan and schedule.
- Complete building of the QA test environment with WebOps help.
- Build and integrate client code to the QA test environment with RE help.
- Smoke test QA test environment for key milestone builds.
- When product is stable, use LoadRunner to record and capture functional transactions for stress testing client product.
- Modify LoadRunner automated test scripts where required to enhance test scripts.
- When product is stable and project time allowed, use WinRunner to record and capture functional transactions.
- Modify WinRunner automated test scripts, using TSL, to enhance test scripts
- Execute manual and automated (WR test scripts) Test Cases.
- If product is stable, execute LoadRunner test scripts to stress the client QA test environment.
- Write bug reports into the defect tracking system.
- Provide defect reports and statistical graphs to the assigned PE.
- Review defect reports and assigns software-engineering resources to fix defects.
- Build client code base related to fixes.

- PE writes Release Notes for QA analysis of affected functional areas to be tested for the delivered build.
- Strategically determine regression testing effort. Re-test affected functional areas fixed.
- Update Test Cases affected by changing client requirements.
- Iterative process: Fix, build, Release Notes, test, report and review new defects.
- Write final Quality Assurance Summary, statistical graphs and outstanding defect summary.
- Store QA documentation into QA designated area.
- Continue to fix, build, test and report, if appropriate, until final delivery to client site.

Tools:

- Defect tracking system.
- Project documentation
- QA Project Leader Guide
- Quality Plan
- Test Cases and Automated Test Scripts
- Test Director
- LoadRunner
- WinRunner
- SQL – Toad or other SQL tool

Deliverables:

- Defect Reports
- Statistical graphs
- Test outputs
- QA Summaries
- Release Notes
- Performance reports and graphs

2.4 LAUNCH

Launch is a critical phase of the entire COMPANY engineering workflow. This phase will determine the success of the entire project. It is during this phase that the product is delivered to the client's designated site for customer acceptance. Customer requirements are reviewed and functionality verified to make sure that what COMPANY delivered is what the customer requested.

During this phase of the project it is imperative that a hard code freeze is completed and that no fixes or changes occur during the final week or so prior to hard launch. The project team should avoid bug fixes on the day the product goes live at the client site.

In this phase, the project team supports any problems or issues the client would like to resolve.

QA team members continue to build and test fixes or changes, requested by the client. At this time, no build will be sent to the client without QA approval, after thorough smoke and regression testing is completed. The Group Director of Engineering must approve any deviation from this process.

Quality Assurance Tasks, Deliverables and Tools:

- RE delivers code to client site after QA tests it.
- Smoke test client site.
- Work with client on new defects.
- Reproduce client defects in production in the COMPANY QA test environment or development environment, to ensure that this is not a configuration problem.
- Iterative process: COMPANY fix, build, regression test, reports and analyzes defects.
- Deliver new code base to client.

Tools:

.

- Defect tracking system.
- Quality Plan
- Test Cases and Automated Test Scripts
- Test Director
- LoadRunner
- WinRunner
- SQL – Toad or other SQL tool

Deliverables:

- Defect Reports
- Statistical graphs
- Test outputs
- QA Summaries
- Release Notes
- Performance reports and graphs
- Project Launch Check List Sign-Off

2.5 POST LAUNCH

This phase is the warranty and maintenance phase of the project. During this phase COMPANY continues to service the customer to fix production problems over a 90 day warranty period.

At this phase, the project team is downsized and selected key members stay on the project to support any future client issues or problems related the product in a client production environment.

At the end of the 90 day warranty period a postmortem is conducted to review key successful and failure points during the project life cycle. The purpose of this postmortem is to improve the COMPANY engineering workflow and to increase the quality of the products COMPANY delivers to clients.

Quality Assurance and Release Engineering Tasks, Deliverables and Tools:

- Work with the PE to determine supporting requirements based on client production problems.
- Receive defect report from client regarding production problems.
- Attempt to reproduce functional defect in the QA test environment or development environment. Determine if this is a configuration problem or an actual defect.
- Iterative process: COMPANY fix, build, regression test, reports and analyzes defects.

Quality Engineering Standards and Practices

- Deliver new code base to client.
- Continue to follow the iterative process until the 90-day warranty period expires or contract is extended.
- For each build delivered to the production client site a Project Launch CheckList must be signed off.
- Store all QA documentation
- Conduct a postmortem of the project to determine what worked, what did not work and ways to improve COMPANY processes to be more successful with future projects.
- Test Plan: QA Warranty – a high level doc that has a scope limited to this post launch phase of the project. See template and example for writing this plan.

Tools:

- Defect tracking system.
- Quality Plan
- Test Cases
- Test Director
- TOAD

Deliverables:

- Test Plan: QA Warranty
- Defect Reports
- Statistical graphs
- Test outputs
- QA Summaries
- Release Notes
- Project Launch Check List

3.0 TEST METHODOLOGY

3.1 General

Testing a new client product is an iterative process. An entire suite of test cases will be executed and defects will be recorded and reported, as they are found. Defects will be fixed, another build will be created and another iteration of testing will be conducted. This testing effort will be executed again (regression tested), to ensure that a fix was confirmed and that the fix did not break some other part of the system. This iterative process will continue until the client product is considered stable enough to run in a client production environment.

All testers will use the defect tracking system to record and report defects found. The Release Engineer will handle the control and management of client software (Refer to the ESOP: Change Management document).

3.2 Functional Specification Review and Ambiguities

The QA Team will conduct a review of the functional specification and any other project documentation that will be used by the QA Team. The purpose of this review is to verify the content of the documentation and determine if all the information required to create QA documentation is clear and meaningful.

The ambiguity list will be distributed to the PE to provide answers or resolutions. This process will be iterative and will continue until all ambiguities are resolved.

3.3 Build Verification

Members of the QA Team will do a smoke test for each build created by the RE. The purpose of the smoke test is to ensure that the build is suitable enough for QA testing. A smoke test does not mean that the build can be released to the client. Defect and functional regression testing must occur prior to releasing the build to the client.

QA should only smoke test key milestone builds, the ones being submitted for official QA testing. This same methodology applies to regression testing.

The PE will maintain release notes for each build created. These release notes will contain the current version of the build and related changes to the product. These Release Notes are used by the QA Team to test fixes contained in the build.

3.5 Testing Defects Or Client Changes

The QA Team will first focus on testing specific fixes or changes to the product, before conducting any further functional testing. This will ensure that the fix or change meets specific client requirements.

3.6 Functional Regression Testing

Regression testing of current product functionality is then conducted by executing all of the baseline test cases. This type of testing will ensure that current product functionality was not corrupted by fixes or changes to the product. If problems are found during this phase of testing, defects are recorded in the defect tracking system and assigned to the development team for resolution.

In the interest of time, a functional regression test may be partially executed during the early stages of development, for critical changes to the product that may impact current product functionality. A complete regression functional test is executed on the final baseline that is identified as the final release to the client. A complete regression functional test does not occur for each release, unless changes may effect the global functionality of the product under test.

Also in the interest of time, a modified regression test may be executed, based on only the affected areas of the product. This assumes that a thorough analysis of the product and the changes was done and that affected functional areas were identified.

3.7 Automation

3.7.1 Performance: Stress and Load Testing

In order for performance testing to be successful, performance requirements must be established early on. Expected customer performance must be established and published, so that these requirements will be documented in the Performance Plan.

If performance requirements are not specifically delineated in the functional specification, then the requirements will be determined by the PE. If this is the case, then optimum performance of the integrated system is a goal of the QA Team, but not acceptance criteria.

Performance testing and improvements may require the coordinated efforts of the project team related to servers, network, products and application code as required. This is described more in the QA Automation Plan and the project Performance Plan.

LoadRunner will be used as the tool to create performance test scripts and reporting results. See the Automation Plan for a detailed explanation of automation.

3.7.2 Functional Testing

Initially all functional testing will be done manually. Functional test automation will occur after the feature has become stable enough to minimize the changes required in maintaining automated test scripts. A group of test cases will be selected, as portions of the product become stable. These test cases, along with some manual testing, will be used for the purpose of conducting regression testing. WinRunner will be the tool to be used for automating most of the functional testing effort. As changes occur to the functionality of the product, automated test scripts will be updated.

Quality Engineering Standards and Practices

The automation and maintenance of the automated test scripts is a full time effort by two FTEs, therefore this effort will be dependent upon the availability of full time resources dictated to automation and not assigned to manual testing efforts. See the Automation Plan for a detailed explanation of automation.

3.8 Test Standards

The QA Team will develop documentation standards. These standards will be used for the design of all QA documentation.

Testing standards will be developed for the following:

- QE Workflow and Methodology
- QA Project Leader Guide
- Automation Plan: General Strategy and Guidelines
- Quality Plan
- Project Performance Plan: Load, Stress and Functional Testing
- Performance Reports and Graphs
- Test Plan: Warranty/Maintenance Phase
- Test Cases
- Automated Test Scripts
- QA Summary
- Defect Summary/Graphs
- Test Case Summary: % Complete
- Project Launch Check List

See QA directory Folsom – Quality Assurance – Standards or visit the COMPANY QA web site.

4.0 TEST MANAGEMENT AND CONTROL

4.1 Project Problems

Any ambiguities, issues and defects will be entered into the defect tracking system. These problem reports will then be reviewed by the project team. The QA Project Leader will provide all the necessary reports related to project issues and defects. The QA Project Leader will facilitate the meeting to review project issues and defects with the project team. Problem reports will be reviewed and categorized. Defects will be then be assigned severity levels and priorities. All issues and defects will be assigned to someone on the team for resolution. PE will be responsible for giving status on assigned problems. Refer to the Change Management document for the process regarding defect control and management

4.2 Configuration Management

Software will be checked in and out as changes occur. The RE will create builds for each baseline. Release Notes will be created by the PE, describing defects fixed or changes made, will be provided to the QA Team. The defect tracking system will be updated based on this information. The Change Management document describes the process for managing this effort.

APPENDIX A – COMPANY Engineering Testing Strategies

This is a list of general testing strategies and definitions to be used in the overall testing and quality assurance effort.

FUNCTIONAL AND DEFECT REGRESSION TESTING

At the completion of a smoke test the QA Team will conduct regression testing efforts. This phase of testing involves the testing of defects that were fixed by the development team. Functional testing will then be done to ensure that what was fixed did not break some other part of the application.

Functional testing is testing the product under test according to the functions described in the product functional specification. Functional testing only proves that the requirements, as specified in functional specification, are met.

Functional testing also includes exercising additional various combinations or permutations of the functions under test. This is analyzing the function and determining what sort of other tests can be executed, to thoroughly test the total functionality of the product, other than meeting the functional requirements of the specification or change request. This type of testing goes beyond meeting the requirements of the function and includes negative and ad hoc testing of the function. It requires the use of your imagination to create test cases that exercises the product beyond its intended functionality, in a number of different permutations or combinations. The intent here is not to cover every possible combination or permutation, but only the most obvious combinations or permutations of the function.

NEGATIVE TESTING

A good approach to negative testing is to conduct testing by forcing errors to occur based on an error-processing document. The following negative testing are further suggestions:

1. Invalid use of keystrokes, mouse, function keys and miscellaneous keyboard operations.
2. Repeated operations between softkeys, navigation buttons and tool bars.
3. Invalid and valid interaction with Windows, if a window based product (Ex. - Maximizing and minimizing screens, navigating from one application to another, execute product-exit to windows- return to product, etc.).
4. Invalid selections of menu selections and menu selection function keys.
5. Inappropriate use of the function or set of functions.
6. Invalid data inputs or corrupted data (Ex - Put alpha characters in a numeric field or combination or blanks, or invalid length).
7. Insert more information in a field than allowed.
8. Unplug the machine or network when in the middle of executing the application.
9. Press other keys or attempt to do another operation while executing the application.
10. Decrease the amount of memory or system resources available for the application. There is software available that will allow you to do this.
11. Use the browser, under test, as it may not be intended to be used.
12. Use your imagination to break the system.

ERROR PROCESSING TESTS

Some products, especially in a client server environment, interact with other applications, systems or other platforms (Client to middle tier to back end or server) and during this process the product may traverse through many different paths. This can be an exhaustive error-processing test. It is recommended that in this case the Test Engineer should first concentrate on their assigned application error processing first and then coordinate with other projects to ensure that the error processing related to their respective areas is conducted. However, the Test Engineer should not just rely on other projects to test errors that may relate to the assigned application when it is communicating with other parts of the system. Also the Test Engineer should read the error messages returned and determine if the error makes sense to the audience it is intended for. For example, error messages are different for users versus system programmers.

LIMITED PERFORMANCE TESTING

Performance testing involves the testing of the product under stress, maximizing the load of the application or volume testing and timing the ability of product to handle capacity or load over a specific period of time. See the Automation Plan for detailed explanation of performance testing.

AD HOC TESTING

Ad Hoc testing involves informal and undocumented testing of the product. Test it any way you wish. Pretend to be an unsophisticated user. Push buttons, navigate functions and menus without purpose. Attempt to break the system. In some cases informal test cases should be documented, if it makes a good test or the steps are required to duplicate a found defect.

SYSTEM TESTING

System testing involves testing the application without focusing on the functional aspects of the application and how the application interacts with other applications or features in a fully integrated environment. It takes into consideration how the product works with other interfaces outside the functionality of the product and how the product responds to unusual exercises. The following are some of the areas covered under system testing and only selected items that are relative to the testing effort of the application will be conducted::

Volume Testing

Subject the product to heavy volumes of data. The purpose of volume testing is to show that the product can not handle the volume of data specified in its objectives. For instance, a data bases may only be handle a limited number of tables or items within a table, so a test could be created to go beyond this capacity.

Stress Testing

Stress testing involves subjecting the product to heavy loads or stresses. Unlike volume testing, a heavy stress test is a peak volume of data encountered over a short span of time. If a product can only handle 10 simultaneous users on the system, then exercise the product with 11, 15, 20 users.

Performance Testing

Products sometimes have specific performance or efficiency objectives, stating such properties as response times and throughput rates under certain workload and configuration conditions. If these requirements are defined, then it is important to exercise these performance requirements as stated and beyond. For example, it may be stated that the product will retrieve customer data within 1 second after the search key is pressed.

Interface Testing

There may be other applications or a product may be part of a larger system. In this case the product must be tested against these other interfaces its communicating with or through. For instance, in a client server environment there may be interdependencies between one product and another that communicates through some other interface via the Internet. Also in a client server environment the test team must take into consideration the interfaces between the client, middle tier and the back end. Each piece of the client server environment is integrated separately, tested separately and then tested again when all of the different pieces are integrated to make a complete operational system.

Usability Testing

This type of testing involves the usability of the system or is the product designed well enough for humans to use (Human Factors). This type of testing requires the Test Engineer to critically study the design and use of the system and determine if the system is user friendly. These type of problems are submitted as change requests and not defects. Be tactful when trekking through this area, because egos can be at stake. This type of testing also involves testing the product, as the agent in the field may actually use it. The QA Team primarily focuses on meeting the functional requirements of the release and does a limited amount of usability testing. This testing effort is best suited for the Beta testers and the User Acceptance Test Team, in determining the usability of the product for the customer.

Security Testing

Security testing is the process of attempting to devise test cases that subvert system security checks. Security testing is a very important aspect of testing, especially when it involves a bank. Sometimes it may be better to hire an outside reputable organization that has experience in this area.

Configuration Testing

Configuration testing is testing the product under a different number of hardware or software configurations. Hardware and software can sometimes have an unlimited number of configurable items and this would be too much to test, so it is advisable that the product be tested with, at least, the minimum, the maximum configuration and a few different configurations in between. The focus here should be on various operating systems and browsers against a specific hardware platform. If the QA testing configuration is not similar to the client configuration in production, then it is quite possible that configuration errors will be uncovered.

Compatibility Testing

There are many products that are dependent on a number of other software platforms. For example, the introduction of an upgraded operating system specified that it was compatible with applications previously executed under the older operating system version. This required testing a number of applications with the new operating system. It is important to know what are the compatibility requirements of a product and understanding all the interdependencies or pieces that work with the product being developed. The bottom line is does the product still work with the old requirements and will it work with the new requirements. This is what upward compatibility is all about.

Installability Testing

The successful installation and execution of the product across different software and hardware configurations is an important part of testing the product. This type of test can be combined with configuration and compatibility testing. Sometimes the product may have a set of installation diskettes or installation procedures that require testing. The installation process should be documented in the early stages of the development. This will make the install process easier for the Launch phase of the project.

Recovery Testing

This type of testing takes into consideration what will happen if there is a crash or other problem with the product. Can the system or product recover. Are there back-up procedures in place and do they work. Some of the problems can be a hardware failure, memory parity errors, I/O device errors or data errors. For example, unplug the network line in the middle of an operation, introduce noise as data input to a communication line, back up the system and introduce problems while back up is being executed.

Documentation Testing

Documentation testing involves testing any documentation the user may require. Such as, User Manuals, Reference document, Installation Procedures, etc. The Test Engineer must actually follow documentation instructions to the letter to determine if any problems occur. Documentation testing also covers any on-line Help screens, menus, tutorials or user messages.

APPENDIX B – QA Test Documentation Standards and Templates

NOTE: See Director of Quality Assurance or QA Team member for the standard. Also review the test documentation numbering system (Appendix C) for the storage and retrieval of this documentation.

APPENDIX C – QA Documentation Numbering System

Purpose

The purpose of this numbering standard is to prepare for the growth for the number of test cases being created for each project. Test Director may be used at a later date to store all QA documentation.

In addition, numbering QA test cases will facilitate managing results of executing these test cases. A test log will be created and the numbering system will be used to manage % complete and pass/fail results.

The purpose of this numbering standard is to prepare for the growth for the number of test cases being created for the COMPANY products quality assurance program. Sometime, in the future, QA will be acquiring a data base test case management tool, such as Test Diector. QA will use this tool to manage the growing number of test cases QA will write.

In addition, numbering QA test cases will facilitate managing results of executing these test cases. A test log will be created and the numbering system will be used to manage % complete and pass/fail results. If you have a problem coming up with a number, see your Manager; also make sure you do not duplicate the numbering system. Make adjustments where necessary.

Quality Plan Numbering System

1. PQP001.00 – P = Parade Project, QP = Quality Plan, 001 = Quality Plan number, .00 = versioning number when making updates to the Quality Plan.

Test Case Numbering System

A variation is using the same set of steps in a Test Case, but the set-up may be different. Sometimes it does not make sense to create an entire new Test Case, just to number a new variation. Use your best judgment and keep it simple, but follow this standard. If you are going to deviate from the standard, then you need to see your Manager.

1. PTC001X.00 – P = Parade Project, TC = Test Case, 001 = Test Case number, X=variation of the same Test Case being executed, but just a different variation (lower case a-z), .00= versioning number when making updates to Test Cases. If you do not have a variation for the Test Case do not use X.

Other Examples with P = Parade project. Change project name.

Build Plan Numbering System: PBP001.00

Release Notes Numbering System: PRN001.00

Automated Test Scripts Numbering System: PATS001X.00

Automation Plan Numbering System: AP001.00

Quality Engineering Standards and Practices

Performance Plan Numbering System: PPP001.00

Project Leader Guide Numbering System: PLG001.00

Client Architecture Diagram Numbering System: PCAD001.00

QA Summary Numbering System: PQAS001.00

Test Case Logs Numbering System: PTCL001.00