TEST STRATEGY DOCUMENT

Project Name:

Prepared by :

Version Number:
1.0

Date:
May 6, 2007

The Test Strategy Document is a living document that is created in the project's Requirements Definition phase, after the Requirements have been specified. The Test Strategy document describes the scope, approach, resources and schedule for the testing activities of the project. This includes defining what will be tested, who will perform testing, how testing will be managed, and the associated risks and contingencies. The Test Strategy document is maintained throughout the life of a project.

Table of Contents

Distribution List

Role	Name
Q/A Manager	
Q/A Test Lead	
Sponsor	
Development Manager	
Product Manager	
Product Support/Documentation Manager	
Software Quality Engineer	

Revision History

Name	Date	Reason for Changes	Ver/Rev.
		First Draft	00
			1
			2
			3
			4
			5
			6

1. Introduction

1.1 Overview

This is the Test Strategy for XXXX . This document shall be completed and used by the project test team to guide how testing will be managed for this project. The test effort will be prioritized and executed based on the project priorities as defined in the Project Plan and Requirements Specification. This is a living document that may be refined as the project progresses. The QA Manager, Test Team Lead, Product Manager, Project Manager, and Development Manager ETC. shall review and approve the final version of the Test Strategy document.

1.2 Reference Materials

XXXX, for project documentation:
XXXX Project Plan.doc
XXXX Requirements
XXXX Project Schedule

1.3 Definitions and Acronyms

- **Project name XXXX**

  Project name and description XXXX

- **Ad Hoc Testing**
  Testing contrived for only the specific purpose or problem at hand; testing not carefully planned in advance.

- **Scenario**
  Detailed description (specific instance) of a use case, including rules, exceptions, boundaries, limits, etc.

- **Test Case**
  A specific set of test data along with expected results for a particular test objective.

- **Test Coverage**
  Describes how much of a system has been tested.

- **Test Design**
  Describes how a feature or function shall be tested.

- **Test Plan**
  Test Strategy

- **Test Procedure**
  Describes the steps for executing a set of test cases and analyzing their results.

- **Test Script**
  Step by step description for specific tests.

- **Test Strategy**
  Describes the scope, approach, resources and schedule for the testing activities of the project. This includes defining what will be tested, who will perform testing, how testing will be managed, and the associated risks and contingencies. Also referred to as a Test Plan.

- **Use Case**
  Describes a sequence of interactions between a system and an external actor that results in the actor accomplishing a task that provides benefit to someone. An actor is a person or other entity external to the software system being specified who interacts with the system to accomplish tasks. Different actors often correspond to different user classes, or roles, identified from the customer community that will use the product.

2. Scope and Limitations.

2.1 Scope

The XXXX release of XXXX software is being released to bring the XXXX application on to the .Blah Blah Blah.....

Testing will cover the functional testing of the Blah Blah Blah.....Functionality for this release is detailed in the XXXX Requirements specifications documents.

Installation will be tested on the different platforms as described in the Requirements Specification. The testing for this will cover the installation on these platforms, as well as a set of critical functions to determine that the code will work on all platforms.

2.2 Limitations and Exclusions

Functionality from the XXXX (prior version) release be tested in the XXXX release through the use of the test interface designed for the release. It is possible that some functionality will be shown to be incorrect; errors of this type will be entered as a defect in the defect tracking system.

The user interface that will be used for this release will not be the final design, because of that interface-specific testing will be excluded.

3. Testing Approach.

3.1 Scope

The testing approach for this release shall be done in a fashion that will accommodate the current functionality in XXXX products being developed for XXXX on Blah Blah Blah .....

Testing will be designed to encompass the following.

- Testing will cover functionality testing for XXXX changes through the use of the test interface. This will validate base functions of the new code as it relates to the standard XXXX model of presentation for data and user entered data.

.

3.2 Test Types

3.2.1 Unit

Unit testing is testing performed to determine that individual program modules perform per the design specifications.

- **Owners**

Corresponding Lead Developers:.

- **Implementation Approach**

At the discretion of the Developer

- **Tools/Techniques**

Manual tests.

3.2.2 Assembly

Assembly testing is designed to test a related group of program modules.

- **Owners**

Corresponding Lead Developers:.

- **Implementation Approach**

At the discretion of the Developer

- **Tools/Techniques**

Manual tests.

3.2.3 System

System testing is the process of testing an integrated system to verify that it meets specified requirements. This testing will determine if the results generated by information systems and their components are accurate and that the system performs according to specifications.

- **Owners**

XXXX Test Team consisting of XXXXX and additional team members where available.

- **Implementation Approach**

The objective of system testing is to verify the correctness of the newly designed items, and their interaction with the existing functions. Testing will focus on functionality of the Blah Blah Blah...

Testing will be accomplished through an organized testing process that will have repeatable tests. This process will be accomplished by use of the scripts created and designed to match the requirements being developed for the Blah Blah Blah...

Planning the execution of test scripts for new functionality and regression tests will be done in coordination with the plan for developing XXXX . Testing and development will be executed in parallel, based on phased implementations, wherever possible.

Test scripts will be structured to give a full range of coverage to the converted functions in both a Positive and Negative fashion, simulating what a potentially unfamiliar user might do during use. Positive test cases will reflect that the application functions as expected and described in the Requirements Specification and the Project Plan. Negative test cases are tests that exercise the limits and boundaries outside the expected designs. The results of this testing will give us some idea as to the stability for the application and its components.

Additional testing beyond the scripted test may be done where feasible to exercise the application to verify error handling and system recovery due to incorrect data or entry into fields.

- **Tools/Techniques**

The information and expected results will be documented and controlled in Blah Blah Blah...

3.2.4 Usability

Usability testing tests the ease with which users can learn and use a product.

► **Owners**

N/A

► **Implementation Approach**

N/A

► **Tools/Techniques**

N/A

► **Stress**

Stress testing is conducted to evaluate a system or component at or beyond the limits of the specified requirements.

► **Owners**

N/A

► **Implementation Approach**

N/A

► **Tools/Techniques**

N/A

3.2.5 Load

Load testing simulates multi-user or multi-threaded access to an application or module to ensure that components and databases can be used to perform specified requirements with no catastrophic failures.

► **Owners**

XXXX Test Team consisting of and additional team members where available.

► **Implementation Approach**

The XXXX software will need to have some load testing done to validate the conversion to Blah Blah Blah...The approach will be to have the databases reside on a single computer in the QA lab and have multiple users access the same database through the use of other computers in the QA lab. A designed test will be written and approved prior to any testing activity for this. The script will be written by the QA team and approved by the development staff.

► **Tools/Techniques**

XXXX QA test machines and XXXX software, scripted scenarios for multiple users accessing the same database. Found defects will be logged as such in the defect tracking tool

3.2.6 Performance

Performance testing is conducted to evaluate the compliance of a system or component's response time, and the ability to function in various operating environments.

► **Owners**

XXXX Test Team consisting of XXXX and additional team members where available.

► **Implementation Approach**

The approach to this will be a manual testing of critical functions agreed on with the Development team. QA will do Blah Blah Blah....

► **Tools/Techniques**

XXXX QA Blah Blah Blah...

3.2.7 Regression

Regression testing involves re-testing a previously tested program following modification to ensure that faults have not been introduced or uncovered as a result of the changes made. In this release this will be covered by the ongoing use of manual tests being executed after each successful build of the application, prior to release of the build for general testing use.

► **Owners**

XXXX Test Team consisting of and additional team members where available.

► **Implementation Approach**

The Test Team will do a pass through all the test scripts that were developed for this project. This will encompass the re-testing of each item in each test script as well as the re-verification of each repaired defect that is decided on as an items to be regressed based on the severity of the defect and the knowledge of the XXXX development staff as to which areas of the application are the most volatile due to new features being implemented.

Additionally the re-test of the install and platforms will be done a final time. This must be done after the application is stable and considered Code Complete. Any defect found during this process must be determined to be "Must Fix" before release or deferred to the next release.

The XXXX Test Team will perform as many test scenarios as is feasible. Testing of the application will include limits and boundaries. The functional testing of the application will be covered this way.

Positive test cases will reflect that the application functions as expected. Negative test cases are tests that exercise the limits and boundaries outside the expected designs. The idea is that the application should be able to recover and / or

set error messaging as needed to accommodate this type of testing. This is important in this release as no design specification for the individual control objects exist other then the past functionality of XXXX releases not on the .Net framework. The results of this testing will give us some idea as to the design parameters and stability for the components.

3.2.8 Recovery

Recovery testing forces the failure of the software in a variety of ways to verify that recovery is properly performed.

- **Owners**

XXXX Test Team consisting of XXXX and additional team members where available.

- **Implementation Approach**

N/A.

- **Tools/Techniques**

N/A.

3.2.9 Conversion

Conversion testing involves testing programs or procedures used to convert data from existing systems for use in Replacement systems.

- **Owners**

XXXX Test Team consisting of XXXX and additional team members where available.

- **Implementation Approach**

This release will cover conversion of Existing Cases, User Defined Data and Report Templates that were used in the past XXXX applications. The testing will be executed by using old and newly created datafiles from Blah Blah Blah...All results should match. Any differences will be reported as defects and will be addressed by the development staff.

- **Tools/Techniques**

Scripted testing for the Blah Blah Blah.and use of the XXXX lab computers and desktop machines.

3.2.10 Security

Security testing evaluates whether the system meets its specified security objectives by attempting to break in or disable a system by improper acquisition of a password, bypassing security measures, browsing through insecure data, or overwhelming the system with requests.

► **Owners**

N/A

► **Implementation Approach**

N/A

► **Tools/Techniques**

N/A

3.2.11 Installation/ Configuration

Installation / Configuration testing verifies that the system will install and function on all required operating platforms, under all specified configurations.

XXXX should be able to run on any PC that has MS Windows 2000or XP running with Microsoft® Word version 7.0 or higher, Office 2000 or Office XP for report generating . See test matrix below.

OS		
Off Version		
Browser		

► **Minimum Requirements**

Hardware

- IBM-compatible computer with at least a Pentium® 166 processor (400 megahertz processor for Windows® XP)
- Pentium 2 processor / 400 / 128 meg of RAM for Windows NT®, Windows 2000, or Windows XP
- 100 MB of available disk space
- Windows-compatible mouse
- VGA or SVGA graphics card compatible with Windows
- Access to a local or network CD-ROM drive

Software

- Windows® 2000 or Windows® XP
- Microsoft® Office 2000, Office 2003, or Office XP

Preferred Configurations

- Pentium 2 processor / 400 / 128 meg of RAM for Windows 98, Windows NT or Windows 2000
- Microsoft Office 97 Service Release 2 (SR-2)
- Internet access for data updates, with Microsoft Internet Explorer version 5.0 or higher, or Netscape® Navigator version 6.0 or higher

► **Owners**

XXXX Test Team consisting of XXXX and additional team members where available.

► **Implementation Approach**

Installation will cover the installation of the application in all Windows Platforms as listed in Requirements Specification #2.4: Windows 2000, Windows XP and 2000 / 2003 Servers.

► **Tools/Techniques**

XXXX Software test workstations. Pentium 2 processor / 400 / 128 meg of ram is the base line lab machine. If possible we will use a machine that is representative of the minimum requirements listed in section 3.2 11of this document.

3.2.12 Documentation Verification

Documentation verification involves reviewing for accuracy all supporting User Documentation, Help Files, and supplemental materials.

► **Owners**

N/A

► **Implementation Approach**

N/A

► **Tools/Techniques**

N/A

3.3 Test Coverage

3.3.1 Outline

The coverage for the testing of specific areas of the Blah Blah Blah...release is detailed in the matrix below. The test coverage will include known functions that currently exist in Version 9.2 new functions as listed in the Requirements Specification for XXXX and additional test data sets designed by the XXXX Test Team. The focus of the testing will be on the new features and functionality.

3.3.2 Test Mapping

Test Mapping	Requirements	New Functionality	Tests	Test Type
XXXX Component				

3.3.3 Previously Deferred Defects

N/A.

3.3.4 Calculations

Owners

XXXX Test Team consisting of and additional team members where available.

Implementation Approach

Example might be …….For the XXXX project there are some calculations that can be validated in Blah Blah Blah.... It is however only some of the calculations that need to be checked for the entire project. Each phase will have additional items to be validated. The approach will be to create some casefiles in XXXX version Blah Blah Blah...and using that data verify that the same results can be found in the XXXX Blah Blah Blah... product. Most likely it will be manual comparison of spreadsheets that have data outputs from both applications. The actual files and variables that can be tested need to be agreed on with both QA and Development. The tests documented and approved and the results placed into XXXX for version management. Any defects that are found will be logged into the defect tracking tool.

Tools/Techniques

Scripted testing of the agreed upon casefiles and associated data, use of the XXXX lab computers and desktop machines, XXXX 9.2 and XXXX software for defect tracking and XXXX for document storage.
Manual testing of the defined scripts.

4. Organization

4.1 Testing deliverables and Milestone

Deliverables and Milestones are as follows;

- Test scripts complete, signed off by the team managers.
- Data sets to be used for testing, (spread sheets and tables with client data designed to reflect the actual use by an XXXX client)

Milestones will be decided after release of the final project timeline

4.2 Roles and Responsibilities

Role	Assigned to	Responsibilities
QA Manager		Oversees QA processes for all XXXX projects.
Build Manager		Monitors and updates the automated build procedure for XXXX applications and components.
Test Lead Software Quality Engineer		Manages and tracks software system test planning and testing.
Software Quality Engineer		Execute test scripts and log defects. Validate repaired defects.
Software Quality Engineer		Execute test scripts and log defects. Validate repaired defects.
Software Engineer		XXXX Software developer
Software Engineer		XXXX Software developer
Product Management		Product Management

5. Resources

5.1 People

Skills	Names	Constraints
Test Lead		Other projects
Experienced software tester		Other Projects
Experienced Software Tester		Other Projects
Product Manager / usability testing input		Other Projects
Test Lead		Other Projects

5.2 Software

Software needed for the Testing Effort for this project:

5.3 Other

5.3.1 SCM

- All test scripts and testing related documents will be stored in XXXX, in the following project path: XXXX, Blah Blah Blah... Projects documentation\XXXX \Testscripts or Testlog folders

- All defects found during testing will be logged in the defect tracking tool .

Any *Quality Engineer* will be required to enter the **Test Case #,** and **Test Script #** in the appropriate fields. If there is no associated Test Case or Test Script, the *Quality Engineer* will enter 0 (zero).

- Any errors found in a test script will be logged in the defect tracking tool.

Any error in logic or functionality associated with a test script will need to have both the **Test Case #,** and **Test Script #** entered into the appropriate fields. The test script will be corrected and modified prior to the next round of testing.

6. Success Factors

6.1 Objective

- Verify that the code migrated to Blah Blah Blah...functions within specified parameters as set forth in the Requirements Specification documents and the associated data spread sheets and examples. This will be accomplished through completed test scripts, and the correction and re-verification of found defects.

- Verify that all necessary defects have been repaired. Necessary defects are defined as all defects that must be repaired before the actual shipment of the product. Some defects that are found may be deemed as acceptable in XXXX . The product manager / sponsor and XXXX senior management would determine this.

6.2 Critical Success Factor.

Critical success for this project is based on delivery on time with all script passes completed and all defects closed and regressed.

Internal success will be measured by:
- Completion of application test scripts (written, reviewed, and approved) when scheduled, and
- Completion of the scheduled test cycles in a timely fashion (as scheduled in the project timeline in the project schedule)

Metrics for this are:
- The completed test scripts with notations concerning defects logged / repaired.

Impact factors for the completion of the above are as follows:
- Failure of development to return "Repaired" defects in a timely fashion.
- Failure of scheduled builds as needed to verify new functions and defects.
- Additional features being added that are outside the scope of this release.

External success factors are:
- N/A This release is internal to XXXX

6.3 Assumptions, Dependencies and Constraints

- Resources must be assigned full time to the test team in order to carry out the intended test cycles.
- Resources other than the Test team will be the Development Staff, Product manager, Research and additional testers if available from other areas within the company.
- Institutionalized knowledge of the XXXX application by some test team members is critical for successful testing.
- The following deliverables need to be in place:
- Accurate Project plans
- Complete Requirements
- Complete Test Scripts

- Quick and accurate repairs of defects logged into the defect tracking tool.
- Available resources to provide the above in the time line defined in the project plan.

6.4 Risk Management

- **High risk** must be assumed concerning the completion of new functions and the related testing within the defined time frame.

- Only XXXX SQE's are doing all the test scripting and test execution for this release and there are other ongoing projects for all.