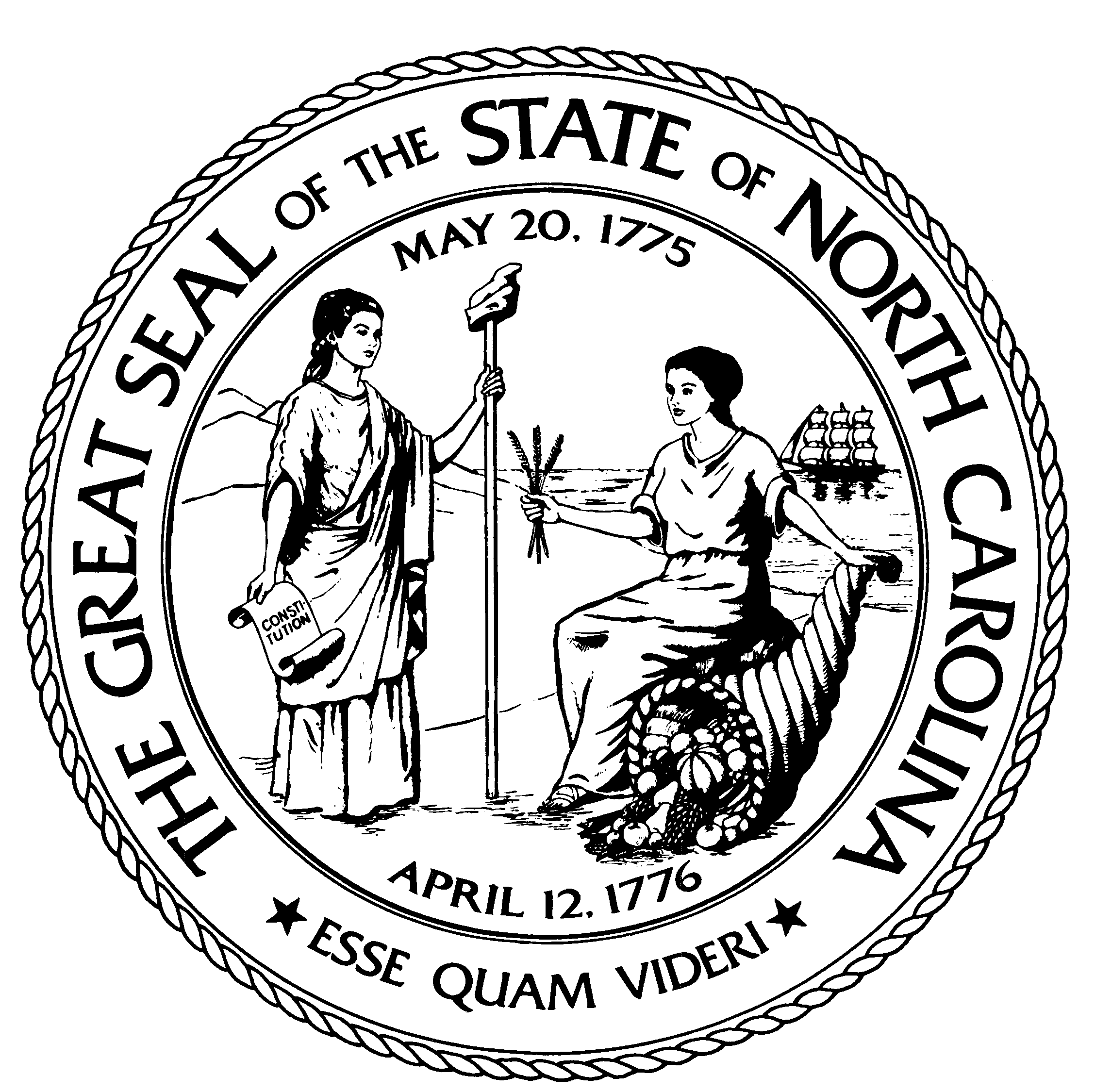
**Software Test Plan (STP) Template**

Items that are intended to stay in as part of your document are in **bold**; explanatory comments are in *italic* text. Plain text is used where you might insert wording about your project.

This document is an annotated outline for a Software Test Plan, adapted from the IEEE Standard for Software Test Documentation (Std 829-1998).

Tailor as appropriate. Where you decide to omit a section, you might keep the header, but insert a comment saying why you omit the element.

**Agency Name**



**Project Name**

**Software Quality Assurance Plan**

**Version: (n) Date: (mm/dd/yyyy)**

**Document History and Distribution**

1. Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision # | Revision Date | **Description of Change** | **Author** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Distribution

|  |  |  |
| --- | --- | --- |
| Recipient Name | Recipient Organization | **Distribution Method** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of Contents**

1. [Introduction 1](#_Toc215044574)

[2. Test Items 3](#_Toc215044575)

[3. Features To Be Tested 4](#_Toc215044576)

[4. Features Not To Be Tested 4](#_Toc215044577)

[5. Approach 4](#_Toc215044578)

[6. Pass / Fail Criteria 6](#_Toc215044579)

[7. Testing Process 6](#_Toc215044580)

[8. Environmental Requirements 7](#_Toc215044581)

[9. Change Management Procedures 8](#_Toc215044582)

[10. Plan Approvals 8](#_Toc215044583)

# Introduction

(Note 1: The Software Test Plan guidelines were derived and developed from IEEE Standard for Software Test Documentation (829-1998)).

*(Note 2: The ordering of Software Test Plan (STP) elements is not meant to imply that the sections or subsections must be developed or presented in that order. The order of presentation is intended for ease of use, not as a guide to preparing the various elements of the Software Test Plan. If some or all of the content of a section is in another document, then a reference to that material may be listed in place of the corresponding content.)*

*The Introduction section of the Software Test Plan (STP) provides an overview of the project and the product test strategy, a list of testing deliverables, the plan for development and evolution of the STP, reference material, and agency definitions and acronyms used in the STP.*

**The Software Test Plan (STP) is designed to prescribe the scope, approach, resources, and schedule of all testing activities. The plan must identify the items to be tested, the features to be tested, the types of testing to be performed, the personnel responsible for testing, the resources and schedule required to complete testing, and the risks associated with the plan**.

**1.1 Objectives**

*(Describe, at a high level, the scope, approach, resources, and schedule of the testing activities. Provide a concise summary of the test plan objectives, the products to be delivered, major work activities, major work products, major milestones, required resources, and master high-level schedules, budget, and effort requirements.)*

**1.2 Testing Strategy**

**Testing is the process of analyzing a software item to detect the differences between existing and required conditions and to evaluate the features of the software item.** (*This may appear as a specific document (such as a Test Specification), or it may be part of the organization's standard test approach. For each level of testing, there should be a test plan and an appropriate set of deliverables. The test strategy should be clearly defined and the Software Test Plan acts as the high-level test plan. Specific testing activities will have their own test plan. Refer to section 5 of this document for a detailed list of specific test plans.)*

*Specific test plan components include:*

* *Purpose for this level of test,*
* *Items to be tested,*
* *Features to be tested,*
* *Features not to be tested,*
* *Management and technical approach,*
* *Pass / Fail criteria,*
* *Individual roles and responsibilities,*
* *Milestones,*
* *Schedules, and*
* *Risk assumptions and constraints.*

**1.3 Scope**

*(Specify the plans for producing both scheduled and unscheduled updates to the Software Test Plan (change management). Methods for distribution of updates shall be specified along with version control and configuration management requirements must be defined.)*

**Testing will be performed at several points in the life cycle as the product is constructed. Testing is a very 'dependent' activity. As a result, test planning is a continuing activity performed throughout the system development life cycle. Test plans must be developed for each level of product testing.**

**1.4 Reference Material**

*(Provide a complete list of all documents and other sources referenced in the Software Test Plan. Reference to the following documents (when they exist) is required for the high-level test plan:*

* *Project authorization,*
* *Project plan,*
* *Quality assurance plan,*
* *Configuration management plan,*
* *Organization policies and procedures, and*
* *Relevant standards.)*

**1.5 Definitions and Acronyms**

*(Specify definitions of all terms and agency acronyms required to properly interpret the Software Test Plan. Reference may be made to the Glossary of Terms on the IRMC web page.)*

# Test Items

*(Specify the test items included in the plan. Supply references to the following item documentation:*

* *Requirements specification,*
* *Design specification,*
* *Users guide,*
* *Operations guide,*
* *Installation guide,*
* *Features (availability, response time),*
* *Defect removal procedures, and*
* *Verification and validation plans.)*

**2.1 Program Modules**

*(Outline testing to be performed by the developer for each module being built.)*

**2.2 Job Control Procedures**

*(Describe testing to be performed on job control language (JCL), production scheduling and control, calls, and job sequencing.)*

**2.3 User Procedures**

*(Describe the testing to be performed on all user documentation to ensure that it is correct, complete, and comprehensive.)*

**2.4 Operator Procedures**

*(Describe the testing procedures to ensure that the application can be run and supported in a production environment (include Help Desk procedures)).*

# 3. Features To Be Tested

*(Identify all software features and combinations of software features to be tested. Identify the test design specifications associated with each feature and each combination of features.)*

# 4. Features Not To Be Tested

*(Identify all features and specific combinations of features that will not be tested along with the reasons.)*

# 5. Approach

*(Describe the overall approaches to testing. The approach should be described in sufficient detail to permit identification of the major testing tasks and estimation of the time required to do each task. Identify the types of testing to be performed along with the methods and criteria to be used in performing test activities. Describe the specific methods and procedures for each type of testing. Define the detailed criteria for evaluating the test results.)*

*(For each level of testing there should be a test plan and the appropriate set of deliverables. Identify the inputs required for each type of test. Specify the source of the input. Also, identify the outputs from each type of testing and specify the purpose and format for each test output. Specify the minimum degree of comprehensiveness desired. Identify the techniques that will be used to judge the comprehensiveness of the testing effort. Specify any additional completion criteria (e.g., error frequency). The techniques to be used to trace requirements should also be specified.)*

**5.1 Component Testing**

*(Testing conducted to verify the implementation of the design for one software element (e.g., unit, module) or a collection of software elements. Sometimes called unit testing. The purpose of component testing is to ensure that the program logic is complete and correct and ensuring that the component works as designed.)*

**5.2 Integration Testing**

*(Testing conducted in which software elements, hardware elements, or both are combined and tested until the entire system has been integrated. The purpose of integration testing is to ensure that design objectives are met and ensures that the software, as a complete entity, complies with operational requirements. Integration testing is also called System Testing.)*

**5.3 Conversion Testing**

*(Testing to ensure that all data elements and historical data is converted from an old system format to the new system format.)*

**5.4 Job Stream Testing**

*(Testing to ensure that the application operates in the production environment.)*

**5.5 Interface Testing**

*(Testing done to ensure that the application operates efficiently and effectively outside the application boundary with all interface systems.)*

**5.6 Security Testing**

*(Testing done to ensure that the application systems control and auditability features of the application are functional.)*

**5.7 Recovery Testing**

*(Testing done to ensure that application restart and backup and recovery facilities operate as designed.)*

**5.8 Performance Testing**

*(Testing done to ensure that that the application performs to customer expectations (response time, availability, portability, and scalability)).*

**5.9 Regression Testing**

*(Testing done to ensure that that applied changes to the application have not adversely affected previously tested functionality.)*

**5.10 Acceptance Testing**

*(Testing conducted to determine whether or not a system satisfies the acceptance criteria and to enable the customer to determine whether or not to accept the system. Acceptance testing ensures that customer requirements' objectives are met and that all components are correctly included in a customer package.)*

**5.11 Beta Testing**

*(Testing, done by the customer, using a pre-release version of the product to verify and validate that the system meets business functional requirements. The purpose of beta testing is to detect application faults, failures, and defects.)*

# 6. Pass / Fail Criteria

*(Specify the criteria to be used to determine whether each item has passed or failed testing.)*

**6.1 Suspension Criteria**

(*Specify the criteria used to suspend all or a portion of the testing activity on test items associated with the plan.)*

**6.2 Resumption Criteria**

*(Specify the conditions that need to be met to resume testing activities after suspension. Specify the test items that must be repeated when testing is resumed.)*

**6.3 Approval Criteria**

*(Specify the conditions that need to be met to approve test results. Define the formal testing approval process.)*

# 7. Testing Process

*(Identify the methods and criteria used in performing test activities. Define the specific methods and procedures for each type of test. Define the detailed criteria for evaluating test results.)*

**7.1 Test Deliverables**

*(Identify the deliverable documents from the test process. Test input and output data should be identified as deliverables. Testing report logs, test incident reports, test summary reports, and metrics' reports must be considered testing deliverables.)*

**7.2 Testing Tasks**

*(Identify the set of tasks necessary to prepare for and perform testing activities. Identify all intertask dependencies and any specific skills required.)*

**7.3 Responsibilities**

*(Identify the groups responsible for managing, designing, preparing, executing, witnessing, checking, and resolving test activities. These groups may include the developers, testers, operations staff, technical support staff, data administration staff, and the user staff.)*

**7.4 Resources**

*(Identify the resources allocated for the performance of testing tasks. Identify the organizational elements or individuals responsible for performing testing activities. Assign specific responsibilities. Specify resources by category. If automated tools are to be used in testing, specify the source of the tools, availability, and the usage requirements.)*

**7.5 Schedule**

*(Identify the high level schedule for each testing task. Establish specific milestones for initiating and completing each type of test activity, for the development of a comprehensive plan, for the receipt of each test input, and for the delivery of test output. Estimate the time required to do each test activity.)*

*(When planning and scheduling testing activities, it must be recognized that the testing process is iterative based on the testing task dependencies.)*

# 8. Environmental Requirements

(Specify both the necessary and desired properties of the test environment including the physical characteristics, communications, mode of usage, and testing supplies. Also provide the levels of security required to perform test activities. Identify special test tools needed and other testing needs (space, machine time, and stationary supplies. Identify the source of all needs that is not currently available to the test group.)

**8.1 Hardware**

*(Identify the computer hardware and network requirements needed to complete test activities.)*

**8.2 Software**

*(Identify the software requirements needed to complete testing activities.)*

**8.3 Security**

*(Identify the testing environment security and asset protection requirements.)*

**8.4 Tools**

*(Identify the special software tools, techniques, and methodologies employed in the testing efforts. The purpose and use of each tool shall be described. Plans for the acquisition, training, support, and qualification for each tool or technique.)*

**8.5 Publications**

*(Identify the documents and publications that are required to support testing activities.)*

**8.6 Risks and Assumptions**

*(Identify significant constraints on testing such as test item availability, test resource availability, and time constraints. Identify the risks and assumptions associated with testing tasks including schedule, resources, approach and documentation. Specify a contingency plan for each risk factor.)*

# 9. Change Management Procedures

*(Identify the software test plan change management process. Define the change initiation, change review, and change authorization process.)*

# 10. Plan Approvals

*(Identify the plan approvers. List the name, signature and date of plan approval.)*

NOTE: HERE IS AN EXAMPLE OF THINGS THAT WE CAN DO……..

**Table of Contents**

1. [Objectives](http://sce.uhcl.edu/helm/RUP_course_example/courseregistrationproject/artifacts/test/plans/test_plan_arch.htm#_Toc449511154)
2. [Requirements for Test](http://sce.uhcl.edu/helm/RUP_course_example/courseregistrationproject/artifacts/test/plans/test_plan_arch.htm#_Toc449511157)
3. [Test Strategy](http://sce.uhcl.edu/helm/RUP_course_example/courseregistrationproject/artifacts/test/plans/test_plan_arch.htm#_Toc449511158)
4. [Resources](http://sce.uhcl.edu/helm/RUP_course_example/courseregistrationproject/artifacts/test/plans/test_plan_arch.htm#_Toc449511173)
5. [Project Milestones](http://sce.uhcl.edu/helm/RUP_course_example/courseregistrationproject/artifacts/test/plans/test_plan_arch.htm#_Toc449511176)
6. [Deliverables](http://sce.uhcl.edu/helm/RUP_course_example/courseregistrationproject/artifacts/test/plans/test_plan_arch.htm#_Toc449511177)
7. [Project Tasks](http://sce.uhcl.edu/helm/RUP_course_example/courseregistrationproject/artifacts/test/plans/test_plan_arch.htm#_Toc449511181)

**Test Plan**

**for the**

**Architectural Prototype**

**1.****Objectives**

**1.1 Purpose**

This document describes the plan for testing the architectural prototype of the C-Registration System. This Test Plan document supports the following objectives:

* + Identify existing project information and the software that should be tested.
  + List the recommended test requirements (high level).
  + Recommend and describe the testing strategies to be employed.
  + Identify the required resources and provide an estimate of the test efforts.
  + List the deliverable elements of the test activities.

**1.2****Scope**

This Test Plan describes the integration and system tests that will be conducted on the architectural prototype following integration of the subsystems and components identified in the Integration Build Plan for the Prototype [16].

It is assumed that unit testing already provided thorough black box testing, extensive coverage of source code, and testing of all module interfaces.

The purpose of assembling the architectural prototype was to test feasibility and performance of the selected architecture. It is critical that all system and subsystem interfaces be tested as well as system performance at this early stage. Testing of system functionality and features will not be conducted on the prototype.

The interfaces between the following subsystems will be tested:

* 1. Course Registration
  2. Finance System
  3. Course Catalog.

The external interfaces to the following devices will be tested:

* 1. Local PCs
  2. Remote PCs.

The most critical performance measures to test are:

* 1. Response time for remote login to the course registration system.
  2. Response time to access the Finance System.
  3. Response time to access the Course Catalog Subsystem.
  4. Student response time when system loaded with 200 logged in students.
  5. Student response time when 50 simultaneous accesses to the Course Catalog database.

**1.3****References**

Applicable references are:

* 1. Course Billing Interface Specification, WC93332, 1985, Wylie College Press.
  2. Course Catalog Database Specification, WC93422, 1985, Wylie College Press.
  3. Vision Document of the C-Registration System, WyIT387, V1.0, 1998, Wylie College IT.
  4. Glossary for the C-Registration System, WyIT406, V2.0, 1999, Wylie College IT.
  5. Use Case Spec - Close Registration, WyIT403, V2.0, 1999, Wylie College IT.
  6. Use Case Spec - Login, WyIT401, V2.0, 1999, Wylie College IT.
  7. Use Case Spec - Maintain Professor Info, WyIT407, Version 2.0, 1999, Wylie College IT.
  8. Use Case Spec - Register for Courses, WyIT402, Version 2.0, 1999, Wylie College IT.
  9. Use Case Spec - Select Courses to Teach, WyIT405, Version 2.0, 1999, Wylie College IT.
  10. Use Case Spec - Maintain Student Info, WyIT408, Version 2.0, 1999, Wylie College IT.
  11. Use Case Spec - Submit Grades, WyIT409, Version 2.0, 1999, Wylie College IT.
  12. Use Case Spec - View Report Card, WyIT410, Version 2.0, 1999, Wylie College IT.
  13. Software Development Plan for the C-Registration System, WyIT418, V1.0, 1999, Wylie College IT.
  14. E1 Iteration Plan, WyIT420, V1.0, 1999, Wylie College IT.
  15. Software Architecture Document, WyIT431, V1.0, 1999, Wylie College IT.
  16. Integration Build Plan for Prototype, WyIT430, V1.0, 1999, Wylie College IT.
  17. Requirements Attributes Guidelines, WyIT404, V1.0, 1999, Wylie College IT.

**2.****Requirements for Test**

The listing below identifies those items (use cases, functional requirements, non-functional requirements) that have been identified as targets for testing. This list represents *what* will be tested.

(Note: Future release of this Test Plan may use Rational RequisitePro for linking directly to the requirements in the Use Case Documents and Supplementary Specification.)

**2.1 Data and Database Integrity Testing**

Verify access to Course Catalog Database.

Verify simultaneous record read accesses.

Verify lockout during Course Catalog updates.

Verify correct retrieval of update of database data.

**2.2. Function Testing**

Vision Document, Section 12.2: "The system shall interface with the existing Course Catalog Database System. C-Registration shall support the data format as defined in [2]."

Vision Document, Section 12.2: "The system shall interface with the existing Billing System and shall support the data format as defined in [1]."

Vision Document, Section 12.2: "The server component of the system shall operate on the College Campus Server and shall run under the UNIX Operating System."

Supplementary Specification, Section 9.3: "The server component of the system shall operate on the Wylie College UNIX Server."

Vision Document, Section 12.2: "The client component of the system shall operate on any personal computer with a 486 Microprocessor or better."

Supplementary Specification, Section 9.3: "The client component of the system shall operate on any personal computer with a 486 Microprocessor or greater."

Supplementary Specification, Section 9.1: "The system shall integrate with existing legacy system (course catalog database) which operates on the College DEC VAX Main Frame."

Supplementary Specification, Section 9.2: "The system shall integrate with the existing Course Billing System which operates on the College DEC VAX Main Frame."

**2.3 Business Cycle Testing**

None.

**2.4 User Interface Testing**

Verify ease of navigation through a sample set of screens.

Verify sample screens conform to GUI standards.

Vision Document Section 10: "The System shall be easy-to-use and shall be appropriate for the target market of computer-literate students and professors."

Vision Document, Section 12.1: "The desktop user-interface shall be Windows 95/98 compliant."

Supplementary Specification, Section 5.1: "The desktop user-interface shall be Windows 95/98 compliant."

Supplementary Specification, Section 5.2: "The user interface of the C-Registration System shall be designed for ease-of-use and shall be appropriate for a computer-literate user community with no additional training on the System."

**2.5 Performance Testing**

Verify response time to access external Finance system.

Verify response time to access external Course Catalog subsystem.

Verify response time for remote login.

Verify response time for remote submittal of course registration.

Vision Document, Section 12.3: "The system shall provide access to the legacy Course Catalog Database with no more than a 10 second latency."

Supplementary Specification, Section 7.2: "The system shall provide access to the legacy Course Catalog Database with no more than a 10 second latency."

**2.6 Load Testing**

Verify system response when loaded with 200 logged on students.

Verify system response when 50 simultaneous student accesses to the Course Catalog.

**2.7 Stress Testing**

None.

**2.8 Volume Testing**

None.

**2.9 Security and Access Control Testing**

Verify Logon from a local PC.

Verify Logon from a remote PC.

Verify Logon security through user name and password mechanisms.

**2.10 Failover / Recovery Testing**

None.

**2.11 Configuration Testing**

Vision Document, Section 12.2: "The client component of the system shall run on Windows 95, Windows 98, and Microsoft Windows NT."

Supplementary Specification, Section 9.4: "The web-based interface for the C-Registration System shall run in Netscape 4.04 and Internet Explorer 4.0 browsers.

Supplementary Specification, Section 9.5: "The web-based interface shall be compatible with the Java 1.1 VM runtime environment.

**2.12 Installation Testing**

None.

**3.****Test Strategy**

The Test Strategy presents the recommended approach to the testing of the software applications. The previous section on Test Requirements described *what* will be tested; this describes *how* it will be tested.

The main considerations for the test strategy are the techniques to be used and the criterion for knowing when the testing is completed.

In addition to the considerations provided for each test below, testing should only be executed using known, controlled databases, in secured environments.

The following test strategy is generic in nature and is meant to apply to the requirements listed in Section 4 of this document.

**3.1****Testing Types**

*3.1.1**Data and Database Integrity Testing*

The databases and the database processes should be tested as separate systems. These systems should be tested without the applications (as the interface to the data). Additional research into the DBMS needs to be performed to identify the tools / techniques that may exist to support the testing identified below.

|  |  |
| --- | --- |
| Test Objective: | Ensure Database access methods and processes function properly and without data corruption. |
| Technique: | * Invoke each database access method and process, seeding each with valid and invalid data (or requests for data). * Inspect the database to ensure the data has been populated as intended, all database events occurred properly, or review the returned data to ensure that the correct data was retrieved (for the correct reasons) |
| Completion Criteria: | All database access methods and processes function as designed and without any data corruption. |
| Special Considerations: | * Testing may require a DBMS development environment or drivers to enter or modify data directly in the databases. * Processes should be invoked manually. * Small or minimally sized databases (limited number of records) should be used to increase the visibility of any non-acceptable events. |

*3.1.2 Function Testing*

Testing of the application should focus on any target requirements that can be traced directly to use cases (or business functions), and business rules. The goals of these tests are to verify proper data acceptance, processing, and retrieval, and the appropriate implementation of the business rules. This type of testing is based upon black box techniques, that is, verifying the application (and its internal processes) by interacting with the application via the GUI and analyzing the output (results). Identified below is an outline of the testing recommended for each application:

|  |  |
| --- | --- |
| Test Objective: | Ensure proper application navigation, data entry, processing, and retrieval. |
| Technique: | * Execute each use case, use case flow, or function, using valid and invalid data, to verify the following: * The expected results occur when valid data is used. * The appropriate error / warning messages are displayed when invalid data is used. * Each business rule is properly applied. |
| Completion Criteria: | * All planned tests have been executed. * All identified defects have been addressed. |
| Special Considerations: | * Access to the Wylie College UNIX Server and the existing Course Catalog System and Billing System is required to run some of the identified System Tests on the Prototype. |

*3.1.3 Business Cycle Testing*

This section is not applicable to test of the architectural prototype.

*3.1.4**User Interface Testing*

User Interface testing verifies a user’s interaction with the software. The goal of UI Testing is to ensure that the User Interface provides the user with the appropriate access and navigation through the functions of the applications. In addition, UI Testing ensures that the objects within the UI function as expected and conform to corporate or industry standards.

|  |  |
| --- | --- |
| Test Objective: | Verify the following:   * Navigation through the application properly reflects business functions and requirements, including window to window, field to field, and use of access methods (tab keys, mouse movements, accelerator keys) * Window objects and characteristics, such as menus, size, position, state, and focus conform to standards. |
| Technique: | * Create / modify tests for each window to verify proper navigation and object states for each application window and objects. |
| Completion Criteria: | Each window successfully verified to remain consistent with benchmark version or within acceptable standard |
| Special Considerations: | * Not all properties for custom and third party objects can be accessed. |

*3.1.5**Performance Profiling*

Performance testing measures response times, transaction rates, and other time sensitive requirements. The goal of Performance testing is to verify and validate the performance requirements have been achieved. Performance testing is usually executed several times, each using a different "background load" on the system. The initial test should be performed with a "nominal" load, similar to the normal load experienced (or anticipated) on the target system. A second performance test is run using a peak load.

Additionally, Performance tests can be used to profile and tune a system’s performance as a function of conditions such as workload or hardware configurations.

NOTE: Transactions below refer to "logical business transactions." These transactions are defined as specific functions that an end user of the system is expected to perform using the application, such as add or modify a given contract.

|  |  |
| --- | --- |
| Test Objective: | Validate System Response time for designated transactions or business functions under a the following two conditions:  - normal anticipated volume  - anticipated worse case volume |
| Technique: | * Use Test Scripts developed for Business Model Testing (System Testing). * Modify data files (to increase the number of transactions) or modify scripts to increase the number of iterations each transaction occurs. * Scripts should be run on one machine (best case to benchmark single user, single transaction) and be repeated with multiple clients (virtual or actual, *see special considerations below).* |
| Completion Criteria: | * Single Transaction / single user: Successful completion of the test scripts without any failures and within the expected / required time allocation (per transaction) * Multiple transactions / multiple users: Successful completion of the test scripts without any failures and within acceptable time allocation. |
| Special considerations: | * Comprehensive performance testing includes having a "background" load on the server. There are several methods that can be used to perform this, including:   + "Drive transactions" directly to the server, usually in the form of SQL calls.   + Create "virtual" user load to simulate many (usually several hundred) clients. Remote Terminal Emulation tools are used to accomplish this load. This technique can also be used to load the network with "traffic."   + Use multiple physical clients, each running test scripts to place a load on the system. * Performance testing should be performed on a dedicated machine or at a dedicated time. This permits full control and accurate measurement. * The databases used for Performance testing should be either actual size, or scaled equally. |

*3.1.6**Load Testing*

Load testing measures subjects the system-under-test to varying workloads to evaluate the system’s ability to continue to function properly under these different workloads. The goal of load testing is to determine and ensure that the system functions properly beyond the expected maximum workload. Additionally, load testing evaluates the performance characteristics (response times, transaction rates, and other time sensitive issues).

NOTE: Transactions below refer to "logical business transactions." These transactions are defined as specific functions that an end user of the system is expected to perform using the application, such as add or modify a given contract.

|  |  |
| --- | --- |
| Test Objective: | Verify System Response time for designated transactions or business cases under varying workload conditions. |
| Technique: | * Use tests developed for Business Cycle Testing. * Modify data files (to increase the number of transactions) or the tests to increase the number of times each transaction occurs. |
| Completion Criteria: | * Multiple transactions / multiple users: Successful completion of the tests without any failures and within acceptable time allocation. |
| Special Considerations: | * Load testing should be performed on a dedicated machine or at a dedicated time. This permits full control and accurate measurement. * The databases used for load testing should be either actual size, or scaled equally. |

*3.1.7**Stress Testing*

This section is not applicable to test of the architectural prototype.

*3.1.8**Volume Testing*

This section is not applicable to test of the architectural prototype.

*3.1.9 Security and Access Control Testing*

Security and Access Control Testing focus on two key areas of security:

- Application security, including access to the Data or Business Functions, and  
- System Security, including logging into / remote access to the system.

Application security ensures that, based upon the desired security, users are restricted to specific functions or are limited in the data that is available to them. For example, everyone may be permitted to enter data and create new accounts, but only managers can delete them. If there is security at the data level, testing ensures that user "type" one can see all customer information, including financial data, however, user two only sees the demographic data for the same client.

System security ensures that only those users granted access to the system are capable of accessing the applications and only through the appropriate gateways.

|  |  |
| --- | --- |
| Test Objective: | Function / Data Security: Verify that user can access only those functions / data for which their user type is provided permissions.  System Security: Verify that only those users with access to the system and application(s) are permitted to access them. |
| Technique: | * Function / Data Security: Identify and list each user type and the functions / data each type has permissions for. * Create tests for each user type and verify permission by creating transactions specific to each user type. * Modify user type and re-run tests for same users. In each case verify those additional functions / data are correctly available or denied. * System Access (see special considerations below) |
| Completion Criteria: | For each known user type the appropriate function / data are available and all transactions function as expected and run in prior Application Function tests |
| Special Considerations: | * Access to the system must be reviewed / discussed with the appropriate network or systems administrator. This testing may not be required as it maybe a function of network or systems administration. |

*3.1.10 Failover and Recovery Testing*

This section is not applicable to test of the architectural prototype.

*3.1.11 Configuration Testing*

Configuration testing verifies operation of the software on different software and hardware configurations. In most production environments, the particular hardware specifications for the client workstations, network connections and database servers vary. Client workstations may have different software loaded (e.g. applications, drivers, etc.) and at any one time many different combinations may be active and using different resources.

|  |  |
| --- | --- |
| Test Objective: | Validate and verify that the client Applications function properly on the prescribed client workstations. |
| Technique: | * Use Integration and System Test scripts * Open / close various PC applications, either as part of the test or prior to the start of the test. * Execute selected transactions to simulate user activities into and out of various PC applications. * Repeat the above process, minimizing the available conventional memory on the client. |
| Completion Criteria: | For each combination of the Prototype and PC application, transactions are successfully completed without failure. |
| pecial Considerations: | * What PC Applications are available, accessible on the clients? * What applications are typically used? * What data are the applications running (i.e. large spreadsheet opened in Excel, 100 page document in Word). * The entire systems, network servers, databases, etc. should also be documented as part of this test. |

*3.1.12**Installation Testing*

This section is not applicable to test of the C-Registration architectural prototype.

**3.2****Tools**

The following tools will be employed for testing of the architectural prototype:

|  |  |  |
| --- | --- | --- |
|  | **Tool** | **Version** |
| Test Management | Rational RequisitePro  Rational Unified Process | TBD |
| Test Design | Rational Rose | TBD |
| Defect Tracking | Rational ClearQuest | TBD |
| Functional Testing | Rational Robot | TBD |
| Performance Testing | Rational Visual Quantify | TBD |
| Test Coverage Monitor or Profiler | Rational Visual PureCoverage | TBD |
| Other Test Tools | Rational Purify  Rational TestFactory | TBD |
| Project Management | Microsoft Project  Microsoft Word  Microsoft Excel | TBD |
| DBMS tools | TBD | TBD |

**4.****Resources**

This section presents the recommended resources for testing the C-Registration architectural prototype, their main responsibilities, and their knowledge or skill set.

**4.1****Roles**

This table shows the staffing assumptions for the test of the Prototype.

|  |  |  |
| --- | --- | --- |
| **Human Resources** | | |
| **Role** | **Minimum Resources Recommended**  (number of workers allocated full-time) | **Specific Responsibilities/Comments** |
| Test Manager | 1 – Kerry Stone | Provides management oversight  Responsibilities:   * Provide technical direction * Acquire appropriate resources * Management reporting |
| Test Designer | Margaret Cox  Carol Smith | Identifies, prioritizes, and implements test cases  Responsibilities:   * Generate test plan * Generate Test Suite * Evaluate effectiveness of test effort |
| System Tester | Carol Smith | Executes the tests  Responsibilities:   * Execute tests * Log results * Recover from errors * Document defects |
| Test System Administrator | Simon Jones | Ensures test environment and assets are managed and maintained.  Responsibilities:   * Administer test management system * Install / manage worker access to test systems |
| Database Administration / Database Manager | Margaret Cox | Ensures test data (database) environment and assets are managed and maintained.  Responsibilities:   * Administer test data (database) |
| Designer | Margaret Cox | Identifies and defines the operations, attributes, and associations of the test classes  Responsibilities:   * Identifies and defines the test class(es) * Identifies and defines the test packages |
| Implementer | Margaret Cox | Implements and unit tests the test classes and test packages  Responsibilities:   * Creates the test classes and packages implemented in the Test Suite. |

**4.2****System**

The following table sets forth the system resources for the testing the C-Registration prototype.

|  |  |
| --- | --- |
| **System Resources** | |
| **Resource** | **Name / Type / Serial No.** |
| Wylie College Server | Serial No: X179773562b |
| —Course Catalog Database | Version Id: CCDB-080885 |
| —Billing System | Version Id: BSSS-88335 |
| Client Test PC's |  |
| —3 Remote PCs (with internet access) | Serial No: A8339223  Serial No: B9334022  Serial No: B9332544 |
| —3 Local PCs (connected via LAN) | Serial No: R3322411 (Registrar’s)  Serial No: A8832234 (IT Lab)  Serial No: W4592233 (IT Lab) |
| Test Repository |  |
| —Wylie College Server | Serial No: X179773562b |
| Test Development PC's - 6 | Serial No: A8888222  Serial No: R3322435  Serial No: I88323423  Serial No: B0980988  Serial No: R3333223  Serial No: Y7289732 |

**5.****Project Milestones**

Testing of the C-Registration Architectural Prototype incorporates test activities for each of the test efforts identified in the previous sections. Separate project milestones are identified to communicate project status and accomplishments.

Refer to the Software Development Plan [13] and the E1 Iteration Plan [14] for the overall phase or master project schedule.

|  |  |  |  |
| --- | --- | --- | --- |
| **Milestone Task** | **Effort (pd)** | **Start Date** | **End Date** |
| Prototype Test Planning | 2 | March 12 | March 15 |
| Prototype Test Design | 3 | March 15 | March 18 |
| Prototype Test Development | 4 | March 19 | March 23 |
| Prototype Test Execution | 3 | March 24 | March 26 |
| Prototype Test Evaluation | 1 | March 29 | March 29 |

**6.****Deliverables**

The deliverables of the test activities as defined in this Test Plan are outlined in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Deliverable** | **Owner** | **Review / Distribution** | **Due Date** |
| Test Plan | K. Stone | Senior Project Mgmt Team | March 15 |
| Test Environment | S. Jones | - | March 18 |
| Test Suite | C. Smith and M. Cox | Internal Peer Review | March 23 |
| Test Data Sets | M. Cox | Internal Peer Review | March 23 |
| Test Scripts | M. Cox | Internal Peer Review | March 23 |
| Test Scripts | M. Cox | - | March 23 |
| Test Stubs, Drivers | M. Cox | - | March 23 |
| Test Defect Reports | C. Smith | Senior Project Mgmt Team | March 26 |
| Test Results | C. Smith | - | March 26 |
| Test Evaluation Report | C. Smith | Senior Project Mgmt Team | March 29 |

**6.1****Test Suite**

The Test Suite will define all the test cases and the test scripts which are associated with each test case.

**6.2****Test Logs**

It is planned to use RequisitePro to identify the test cases and to track the status of each test case. The test results will be summarized in RequisitePro as untested, passed, conditional pass, or failed. In summary, RequisitePro will be setup to support the following attributes for each test case, as defined in the Requirements Attributes Guidelines [17]:

* + Test status
  + Build Number
  + Tested By
  + Date Tested
  + Test Notes

It will be the responsibility of the System Tester to update the test status in RequisitePro.

Test results will be retained under Configuration Control.

**6.3****Defect Reports**

Rational ClearQuest will be used for logging and tracking individual defects.

**7.****Project Tasks**

Below are the test related tasks for testing the C-Registration Architectural Prototype:

|  |
| --- |
| **Plan Test** |
| Identify Requirements for Test |
| Assess Risk |
| Develop Test Strategy |
| Identify Test Resources |
| Create Schedule |
| Generate Test Plan |
| **Design Test** |
| Workload Analysis (not applicable for Prototype) |
| Develop Test Suite |
| Identify and Describe Test Cases |
| Identify and Structure Test Scripts |
| Review and Access Test Coverage |
| **Implement Test** |
| Setup Test Environment |
| Record or Program Test Scripts |
| Develop Test Stubs and Drivers |
| Identify Test-Specific functionality in the design and implementation model |
| Establish External Data sets |
| **Execute Test** |
| Execute Test Scripts |
| Evaluate Execution of Test |
| Recover from Halted Test |
| Verify the results |
| Investigate Unexpected Results |
| Log Defects |
| **Evaluate Test** |
| Evaluate Test-Case Coverage |
| Evaluate Code Coverage |
| Analyze Defects |
| Determine if Test Completion Criteria and Success Criteria have been achieved |
| Create Test Evaluation Report |