Computerized Patient Record System (CPRS) v32

Master Test Plan

Version 1.32



October 2017

Department of Veterans Affairs

Revision History

| Date | Version | Description | Author |
| --- | --- | --- | --- |
| 10/06/2014 | 1.0 | Creation | Rishan Chandarana |
| 11/25/2014 | 1.1 | Placed Roles in Responsible Party, Updated Formatting. | Rishan Chandarana |
| 12/24/2014 | 1.2 | Monthly Update | Rishan Chandarana |
| 01/29/2015 | 1.3 | Updated IOC testing to reflect joint testing with Audio care. Updated Roles and Responsibilities. | Rishan Chandarana |
| 3/3/2015 | 1.4 | Updated with Test Environment | Rishan Chandarana |
| 4/3/2015 | 1.5 | Updated Roles/Responsibilities | Rishan Chandarana |
| 5/3/2015 | 1.6 | Updated Training | Rishan Chandarana |
| 6/3/2015 | 1.7 | Monthly Update/Corrected date formats in revision. | Rishan Chandarana |
| 7/1/2015 | 1.8 | Monthly Update | Rishan Chandarana |
| 7/31/2015 | 1.9 | Changed Test Deliverables Responsibilities to Specify HP SQA Analysts where necessary.  Changed Test Deliverables Test Environment Responsibility to John Service. | Brian Watt |
| 8/31/2015 | 1.10 | Monthly Update, Updated Responsibilities; Updated Training Needs | Rishan Chandarana, Brian Watt |
| 9/30/2015 | 1.11 | Monthly Update, Updated Test Team and Test Analysts | Brian Watt |
| 10/30/2015 | 1.12 | Monthly Update | Brian Watt |
| 11/30/2015 | 1.13 | Monthly Update, Updated Staffing | Brian Watt, Rishan Chandarana |
| 12/30/2015 | 1.14 | Monthly Updates | Brian Watt |
| 01/31/2016 | 1.15 | Monthly Updates; convert to most recent template | Rishan Chandarana  Brian Watt  Juico Bowley |
| 03/02/2016 | 1.16 | Monthly Updates | Rishan Chandarana |
| 04/04/2016 | 1.17 | Monthly Updates; Updated for 508 Compliance | Brian Watt  Rishan Chandarana |
| 04/27/2016 | 1.18 | Monthly Updates | Brian Watt |
| 05/31/2016 | 1.19 | Updated Staffing | Rishan Chandarana |
| 6/30/2016 | 1.20 | Monthly Updates | Brian Watt |
| 7/29/2016 | 1.21 | Monthly Updates | Rishan Chandarana |
| 8/31/2016 | 1.22 | Monthly Updates | Rishan Chandarana |
| 1/3/2017 | 1.23 | Monthly Updates | Brian Watt |
| 2/28/2017 | 1.24 | Monthly Updates | Brian Watt, Craig Hinton |
| 3/31/2017 | 1.25 | Monthly Updates | Brian Watt |
| 4/28/2017 | 1.26 | Monthly Updates | Brian Watt |
| 5/30/17 | 1.27 | Monthly Updates | Brian Watt |
| 6/29/17 | 1.28 | Monthly Updates |  |
| 7/31/2017 | 1.29 | Monthly Updates | Brian Watt |
| 8/30/2017 | 1.30 | Monthly Update | Brian Watt |
| 9/29/2017 | 1.31 | Monthly Update | Brian Watt |
| 10/31/2017 | 1.32 | Monthly Update | Brian Watt |

Table of Contents

[1. Introduction 1](#_Toc433725062)

[1.1. Purpose 1](#_Toc433725063)

[1.2. Test Objectives 1](#_Toc433725064)

[1.3. Roles and Responsibilities 1](#_Toc433725065)

[1.4. Processes and References 3](#_Toc433725066)

[2. Items To Be Tested 3](#_Toc433725067)

[2.1. Overview of Test Inclusions 3](#_Toc433725068)

[2.2. Overview of Test Exclusions 4](#_Toc433725069)

[3. Test Approach 4](#_Toc433725070)

[3.1. Product Component Test 4](#_Toc433725071)

[3.2. Component Integration Test 4](#_Toc433725072)

[3.3. System Tests 5](#_Toc433725073)

[3.4. User Functionality Test 5](#_Toc433725074)

[3.5. Enterprise System Engineering Testing 6](#_Toc433725075)

[3.6. Initial Operating Capability Evaluation 6](#_Toc433725076)

[4. Testing Techniques 6](#_Toc433725077)

[4.1. Risk-based Testing 6](#_Toc433725078)

[4.2. Enterprise Testing 6](#_Toc433725079)

[4.2.1. Security Testing 7](#_Toc433725080)

[4.2.2. Privacy Testing 7](#_Toc433725081)

[4.2.3. Section 508 Compliance Testing 7](#_Toc433725082)

[4.2.4. Multi-Divisional Testing 7](#_Toc433725083)

[4.3. Performance and Capacity Testing 7](#_Toc433725084)

[4.4. Test Types 8](#_Toc433725085)

[4.5. Productivity and Support Tools 10](#_Toc433725086)

[5. Test Criteria 10](#_Toc433725087)

[5.1. Process Reviews 10](#_Toc433725088)

[5.2. Pass/Fail Criteria 11](#_Toc433725089)

[5.3. Suspension and Resumption Criteria 11](#_Toc433725090)

[6. Test Deliverables 11](#_Toc433725091)

[7. Test Schedule 13](#_Toc433725092)

[8. Test Environments 13](#_Toc433725093)

[8.1. Test Environment Configurations 13](#_Toc433725094)

[8.2. Base System Hardware 14](#_Toc433725095)

[8.3. Base Software Elements in the Test Environments 15](#_Toc433725096)

[9. Staffing and Training Needs 15](#_Toc433725097)

[10. Risks and Constraints 16](#_Toc433725098)

[11. Test Metrics 16](#_Toc433725099)

[Attachment A – Approval Signatures 17](#_Toc433725100)

[Appendix A - Test Type Definitions 18](#_Toc433725101)

# Introduction

## Purpose

The purpose of this Master Test Plan for the Computerize Patient Record System v32 Development Project is to document the overall approach to validate and verify the functionality delivered in version 32 of the Computerized Patient Record System (CPRS) Graphical User Interface (GUI). CPRS v32 encompasses both new functionality as well as enhancements to existing functionality. In addition to modifying CPRS, this project/plan encompasses modifications to Text Integration Utility, Inpatient Medications, Outpatient Pharmacy, Pharmacy Data Management, Barcode Medication Administration, Adverse Reaction Tracking, Clinical Reminders, and Laboratory.

## Test Objectives

This Master Test Plan supports the following objectives:

* To provide test coverage for 100% of the documented requirements
* To provide coverage for System/ Software Design Document elements
* To execute 100% of the test cases during User Functionality Testing
* To create, maintain and control the test environment

## Roles and Responsibilities

Table 1 lists the key roles and their responsibilities for this Master Test Plan.

Table : Roles and Descriptions

| Role | Description | Team Members |
| --- | --- | --- |
| Development Team | Persons that build or construct the product/product component. | Jamie Crumley, Ty Phelps, Andrey Andriyevskiy, Andrea Freeman, Nick Costanzo, Jeff Swesky  [Previous Developers Robert Lauro, Kim Hovorka, Mike Jenkins] |
| Development Manager | Person responsible for assisting with the creation and implementation of the Master Test Plan. | Craig Hinton, Rishan Chandarana |
| Program Manager | Person that has overall responsibility for the successful planning and execution of a project; person responsible for creating the Master Test Plan in collaboration with the Development Manager. | Mike Braithwaite, Kenny, Condie, Michael Keener, April Scott |
| Stakeholders | Persons that hold a stake in a situation in which they may affect or be affected by the outcome. | End users-health care providers |
| Test Analyst | Person responsible for ensuring full execution of the test process to include the verification of technical requirements and the validation of business requirements. | Brian Watt, Juico Bowley, Rebecca Russell, Susan Scorzato |
| Test Lead | An experienced Test Analyst or member of the Test Team that leads and coordinates activities related to all aspects of testing based on an approved Master Test Plan and schedule. | Brian Watt, Juico Bowley |
| Test Team/Testers | Persons that execute tests and ensure the test environment will adequately support planned test activities. | Brian Watt, Juico Bowley, Rebecca Russell, Susan Scorzato |
| Test Environment Team | Persons that establish, maintain, and control test environments. | John Service |

## Processes and References

The processes that guide the implementation of this Master Test Plan are:

* Test Preparation
* Product Build
* Independent Test and Evaluation

The references that support the implementation of this Master Test Plan are:

* [ProPath](http://URL/process/home.aspx" \o "ProPath Home Page)
* [Section 508 Office Web Page](http://URL/index.asp)
* [Privacy Impact Assessment - Privacy Service](http://URL/Privacy_Impact_Assessment.asp)

The references that support the implementation of this Master Test Plan are:

* Business Requirement Document (BRD) Version <#.#>, Date <Month, Year>
* Requirements Specification Document (RSD) Version 1.30, Date – March 2017
* System Design Document (SDD) Version 1.16, Date – January, 2016
* Requirements Traceability Matrix (RTM) Version 1.30, Date – March, 2017
* Risk Log Version <#.#>, Date <Month, Year>

# Items To Be Tested

Items to be tested include the following:

* The CPRS GUI.
* VistA Patches for the applications being developed.
* CPRS through VistA.
* Installation Guide.
* User Guide
* Interface between affected patches.

## Overview of Test Inclusions

The following components and features and combinations of components and features will be tested:

* The CPRS GUI and installation tools and distribution methods.
* VistA Patches and installation tools and distribution methods.
* Installation Guide as well as document distribution methods.
* User Guide as well as document distribution methods.
* Interfaces between affected packages (for example, Inpatient Medications and CPRS, Audiocare).

## Overview of Test Exclusions

The following components and features and combinations of components and features will not be tested:

* Applications not included in the CPRS v32 effort, especially applications that do not interact with CPRS.

# Test Approach

## Product Component Test

The Developer performs Product Component Testing (aka Unit Testing) which includes the internal technical and functional testing of a module/component of code and is responsible for the verification of the requirements defined in the detailed design specification have been successfully applied to the module/component under test. Steps include:

* Analyze requirements to understand the application functionality and dependencies
* Identify all the routines affected by the module or object
* Specify all the routines that are called from various locations
* Execute tests on prioritized options
* Execute tests with different combinations of options and data. For example, test with minimal data entered and test with maximal data entered
* Perform exploratory testing, i.e., randomly exercise the module, object, and options based upon domain knowledge, past performance, and expertise
* Record the actual test results
* Perform static analysis of module/component source code

If a defect is identified and it is related to the code being developed that code will continue to be developed until it successfully passes the unit test.

If the defect is not related to the code being developed it will be addressed by the following methods:

* If it is a software defect that is within the scope of this project, it will be added to the project backlog.
* If it is a software defect that is outside of the scope of this project it will be referred to the existing maintenance structure.
* If the software defect is not truly a defect and requires a change in functionality it will be reported with a suggestion to enter a new service request.

## Component Integration Test

The Test Analyst installs the Product Component and performs component integration testing. Product Component Integration testing is performed to expose defects in the interfaces and interaction between integrated components as well as verifying installation instructions. Component integration testing includes testing of Identity and Access Management Integration Service Pattern changes. The Software Quality Assurance Review Checklist is started during this activity.

If a defect is identified and it is related to the code being developed the defect will be added to the project backlog.

If the defect is not related to the code being developed it will be addressed by the following methods:

* If it is a software defect that is within the scope of this project, it will be added to the project backlog.
* If it is a software defect that is outside of the scope of this project it will be referred to the existing maintenance structure.
* If the software defect is not truly a defect and requires a change in functionality it will be reported with a suggestion to enter a new service request.

## System Tests

The Test Analyst performs System Tests employing a variety of test types (i.e., compliance, regression, access control, interoperability, usability (including 508 compliance), etc.). System Tests exercise all parts of an integrated system including interfaces to external systems.

If a defect is identified and it is related to the code being developed the defect will be added to the project backlog.

If the defect is not related to the code being developed it will be addressed by the following methods:

* If it is a software defect that is within the scope of this project, it will be added to the project backlog.
* If it is a software defect that is outside of the scope of this project it will be referred to the existing maintenance structure.
* If the software defect is not truly a defect and requires a change in functionality it will be reported with a suggestion to enter a new service request.

## User Functionality Test

The VA Project Manager ensures the User Functionality Test (UFT) is conducted. UFT is a formal test conducted by the end-users to determine whether a system satisfies its acceptance criteria and enables the customer to determine whether to accept the system. The purpose of the User Functionality Test is to (1) exercise the functionality of the application using test data in a controlled test environment in order to validate functionality and (2) evaluate the usability of a component or system. Additionally, during User Functionality Testing, Enterprise Shared Service functionality, such as Identity and Access Management, are tested.

## Enterprise System Engineering Testing

The VA Project Manager reviews all testing intake assessment results, including Risk Analysis and Testing Scope Report (RATSR) or Testing Intake Analysis (TIA) closure email. The VA Project Manager then incorporates ESE Enterprise Testing Services (ETS) Independent Testing and/or Systems Quality Assurance Service (SQAS) independent testing services required in the Risk Assessment Testing Scope Report (RATSR) into project plans and schedules.

At this point a determination will be made if it is necessary to conduct independent testing, whether it is SQAS/IV&V Testing, or ESE Testing.

## Initial Operating Capability Evaluation

The Initial Operating Capability (IOC) Implementation Manager coordinates the performance of the IOC evaluation. IOC evaluation (formerly known as field testing) is when a product/system that has been modified/enhanced is placed into a limited number of production (live) environments, in order to evaluate the new features and functionality of the product/system and to ascertain if the features and functionality perform as expected and do not adversely affect the existing functionality of the product/system. Activities include:

* Distribute the product and product documentation to the Evaluation Sites
* Facilitate the timely installations at the Evaluation Sites
* Conduct formal or bi-weekly Evaluation Site calls
* Track defects identified during Initial Operating Capability Evaluation
* Address issues and questions identified during evaluation
* Obtain Site Concurrence Statements

CPRS v32 makes modifications that impact the Audiocare application. IOC testing for CPRS v32 will be conducted in tandem with IOC testing for the Audiocare application.

# Testing Techniques

## Risk-based Testing

The following table will be updated as risks are identified:

Table 2: Risks and Priorities

| Risk | Priority | Test Type/Test Case |
| --- | --- | --- |
| Configuration Management based Integration Testing | High | Details TBD, continuous integration testing will be performed to verify that modules integrate appropriately and do not cause adverse interactions with existing or previously developed software. |
| Integration of modifications from other software projects such as MOCHA (Medication Order Check Healthcare Application) | High | Details TBD, integration testing will be performed regularly and test scripts will be updated to reflect |

## Enterprise Testing

Cite how the project testing covers the enterprise requirements. Enterprise requirements include security, privacy, Section 508 Compliance requirements, and Multi-divisional requirements.

### Security Testing

Security Testing will be performed by the testing services group. Basic testing such as boundary testing, sign in procedures etc. will be performed by the Test Analysts.

### Privacy Testing

Privacy Testing will be performed by the testing services group. CPRS is a provider facing application and as such the application makes sensitive information such as Protected Health Information (PHI) and Personally Identifiable Information (PII) visible to application users. Test Analysts will perform basic testing to validate that privacy guidelines are being followed.

### Section 508 Compliance Testing

Section 508 Compliance Testing will be performed by the Test Analysts on the team. They will follow guidelines set forth by the program office to validate that the application is Section 508 compliant. Tests include utilizing JAWS (Job Access With Speech) Screen Reader software to validate that the screen reader and the visual functionality are in alignment.

### Multi-Divisional Testing

Multi-divisional testing will be conducted during the Initial Operating Capability (IOC) testing phase by an Integrated Test Site. In addition the test environment will be multi-divisional.

## Performance and Capacity Testing

TBD

Develop tests to ensure the application will perform as expected under anticipated user loads, and typical business transactions respond in a timely manner. During the test execution, the System Under Test (SUT) is actively monitored for any issues that could affect application performance, and to verify the hardware environment is adequately sized.

This type of testing covers the requirements specified in the “Performance Specifications” in the Requirements Specification Document found in the Requirements process in ProPath.

## Test Types

Table : Test Types

| Test Types | Party Responsible |
| --- | --- |
| Access control testing | HP Test Analysts, Developers |
| Build verification testing | HP Test Analysts, Developers |
| Business cycle testing | HP Test Analysts |
| Compliance testing | HP Test Analysts, Developers |
| Component integration testing | HP Test Analysts, Developers |
| Configuration testing | HP Test Analysts |
| Data and database integrity testing | HP Test Analysts, Developers |
| Documentation testing | HP Test Analysts, Developers |
| Error analysis testing | Developers |
| Exploratory testing | HP Test Analysts |
| Failover testing | HP Test Analysts, Developers |
| Installation testing | HP Test Analysts, Developers |
| Integration testing | Developers |
| Migration testing | HP Test Analysts |
| Multi-divisional testing | HP Test Analysts |
| Parallel testing | HP Test Analysts, Developers |
| Performance monitoring testing | TBD |
| Performance testing | TBD |
| Performance - Benchmark testing | TBD |
| Performance - Contention testing | TBD |
| Performance - Endurance testing | TBD |
| Performance - Load testing | TBD |
| Performance - Profiling testing | TBD |
| Performance - Spike testing | TBD |
| Performance - Stress testing | TBD |
| Privacy testing | HP Test Analysts, Developers |
| Product component testing | Developers |
| Recovery testing | TBD |
| Regression test | HP Test Analysts, Developers |
| Risk based testing | HP Test Analysts, Developers |
| Section 508 compliance testing | HP Test Analysts, Developers |
| Security testing | HP Test Analysts |
| Smoke testing | HP Test Analysts, Developers |
| System testing | HP Test Analysts, Developers |
| Usability testing | HP Test Analysts, Developers |
| User Functionality Testing | HP Test Analysts, Developers |
| User interface testing | HP Test Analysts, Developers |

## Productivity and Support Tools

Add or delete tools as appropriate.

Table 3 describes the tools that will be employed to support this Master Test Plan.

Table : Tool Category or Types

| Tool Category or Type | Tool Brand Name | Vendor or In-house | Version |
| --- | --- | --- | --- |
| Test Management | TBD |  |  |
| Defect Tracking | Rational Jazz Tool | IBM |  |
| Test Coverage Monitor or Profiler | TBD |  |  |
| Project Management | Project | Microsoft |  |
| Performance Testing | TBD |  |  |
| Configuration Management | Rational Jazz Tool | IBM |  |
| DBMS tools | Reflection for UNIX and OpenVMS | Attachmate |  |
| Document Repository | Microsoft SharePoint |  |  |
| Shared Drive | Microsoft |  |  |

# Test Criteria

## Process Reviews

The Master Test Plan under goes two reviews:

* Peer Review – upon completion of the Master Test Plan
* Formal Review – after the Development Manager approves the Master Test Plan

The Master Test Plan does serve as an input or Artifact Used for the Process Quality Gate Review for Product Build as well as for the Go No Review (Milestone) for Independent Testing.

For more information on the reviews associated with testing, see the Product Build, Test Preparation, and Independent Test and Evaluation processes.

## Pass/Fail Criteria

Incidents identified during the execution of this test plan will be evaluated to determine their severity. This impact will be recorded in the severity section of the Jazz defect.

**High** Impact Test Incident is an error or lack of functionality that:

* Jeopardizes patient or personnel safety by corrupt or incorrect data
* Has no workaround to provide similar functionality and this functionality is required to move to system, integration, or user acceptance
* Adversely affects all users or key user functionality

**Medium** Impact Test Incident is an error or lack of functionality that:

* Has a reasonable workaround to maintain functionality
* Impacts a small group of users, but has workaround
* Functionality works but not to requirements, specifications, or standards and workflow is not hampered

**Low** Impact Test Incident is an error or lack of functionality that may cause operator/user inconvenience and minimally affects operational processing.

* Spelling errors
* Minor GUI Graphical/Formatting errors that do not affect functionality/visibility

**Enhancement** Test Incident is something that would be “nice” to have in the integration piece but was not included in the specifications for this release.

All High and Medium defects shall be addressed or negotiated prior to release. Any limitation or outstanding test incident shall have an approved contingency process (workaround) in place.

## Suspension and Resumption Criteria

Testing will cease on a test item when a high impact test incident is logged. Testing will resume when the incident is resolved.

Testing will cease on the entire release when three high impact test incidents are logged. Testing will resume when the incidence are addressed.

## Acceptance Criteria

All High and Medium defects shall be addressed or negotiated prior to release. Any limitation or outstanding test incident shall have an approved contingency process (workaround) in place.

# Test Deliverables

The Test Deliverables listed below represent some possible deliverables for a testing project. The Test Deliverables table may be tailored to meet project needs. Do not include Delete any listed test deliverable that is not used by the Product Build, Test Management, and Independent Test and Evaluation processes.

Table 4 lists the test deliverables for the CPRS v32 project.

Table : Test Deliverables

| Test Deliverables | Responsible Party |
| --- | --- |
| Master Test Plan | HP SQA Analyst |
| Iteration Test Plans (when appropriate) | HP SQA Analyst |
| Test Execution Risks | VA/HP PM |
| Test Schedule | VA/HP PM |
| Test Cases/Test Scripts | HP SQA Analyst |
| Test Data | HP SQA Analyst |
| Test Environment | John Service |
| Test Evaluation Summaries | HP SQA Analyst |
| Traceability Report or Matrix | HP SQA Analyst |
| Master Test Plan | HP SQA Analyst |

# Test Schedule

List the major testing milestones. When appropriate, reference other workflow documentation or tools, such as the Project Management Plan, or Work Breakdown Structure (WBS.) Put a minimum amount of process and planning information within the Master Test Plan in order to facilitate ongoing maintenance of the test schedule.

Table : Testing Milestones

| Testing Milestones | Responsible Party |
| --- | --- |
| Approved Master Test Plan | HP SQA Analyst |
| Approved generic test cases (high level list) | HP SQA Analyst |
| Complete and stable requirements (SRS or CRs) | HP SQA Analyst |
| Creating of Test Environment(s) | HP SQA Analyst |
| Submit and manage request for Testing Services | HP SQA Analyst |
| Test Cases selected for release and entered using MS Excel Spreadsheet on SQA SharePoint | HP SQA Analyst |
| Completion of Patch verification | HP SQA Analyst |
| SQA Testing conducted (execute the selected Test Cases) in Test environment(s) | HP SQA Analyst |
| Remedy Tickets | HP SQA Analyst |
| Defects identified and entered into CQ | HP SQA Analyst |

# Test Environments

A test environment is an environment containing hardware, instrumentation, simulators, software tools, and other support elements needed to conduct a test.

## Test Environment Configurations

The party or parties responsible for configuring and maintaining the test environments are: John Service & Bay Pines Test Lab

The test environment will be hosted at the Bay Pines Test Lab, DAYT79.

## Base System Hardware

Table 6 sets forth the system resources for the test effort presented in this Master Test Plan.

The specific elements of the test system may not be fully understood in early iterations, so this section may be completed over time. The test system should simulate the production environment as closely as possible, scaling down the concurrent access and database size, and so forth, if and where appropriate. Tailor the System Hardware Resources table as required.

Table : System Hardware Resources

| Resource | Quantity | Name and Type |
| --- | --- | --- |
| Database Server |  |  |
| Network or Subnet |  | TBD |
| Server Name |  | TBD |
| Database Name |  | TBD |
| Client Test PCs |  |  |
| Include special configuration requirements |  | TBD |
| Test Repository |  |  |
| Network or Subnet |  | TBD |
| Server Name |  | TBD |
| Test Development PCs |  | TBD |

## Base Software Elements in the Test Environments

Add or delete Software Elements as appropriate. If necessary, specify software patches referenced and/or required here.

Table 7 describes the base software elements that are required in the test environment for this Master Test Plan.

Table : Software Elements

| Software Element Name | Version | Type and Other Notes |
| --- | --- | --- |
| Windows | 7 | Operating System |
| Intersystems Cache | 2014 | MUMPS environment |
| Delphi | XE3 | GUI source code |

# Staffing and Training Needs

Table 8 describes the personnel resources needed to plan, prepare, and execute this Master Test Plan.

Table : Staffing Resources

| Testing Task | Quantity of Personnel Needed | Test Process | Duration/ Days |
| --- | --- | --- | --- |
| Create the Master Test Plan |  | Test Preparation | xxx days |
| Establish the Test Environment |  | Test Preparation | xxx days |
| Perform System Tests |  | Product Build | xxx days |
| Etc. |  |  |  |

Identify training options for providing necessary skills and the estimated number of hours necessary to complete the training.

Table 9 lists the personnel that require training.

Table : Training Needs

| **Name** | **Training Need** | **Training Option** | **Estimated Training Hours** |
| --- | --- | --- | --- |
| Andrey Andriyevskiy | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Christopher Bell | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Juico Bowley | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Rishan Chandarana | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Nicholas Costanzo | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Jamie Crumley | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Andrea Freeman | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Craig O. Hinton | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Kim C. Hovorka | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Robert Lauro | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Joe Niksich | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Ty Phelps | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Blair Sanders | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Susan Scorzato | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| April Scott | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |
| Brian Watt | IBM Rational Jazz ® | Obtain IBM Rational Jazz ® training | 6 hours |

# Risks and Constraints

The risk log was taken into consideration in the development of this test plan.

The risks identified in this Master Test Plan can be found in the risk log and may be recorded and tracked in an automated tool, such as, IBM Rational Jazz®.

# Test Metrics

Metrics are a system of parameters or methods for quantitative and periodic assessment of a process that is to be measured.

Test metrics may include, but are not limited to:

* Number of test cases (pass/fail)
* Percentage of test cases executed
* Number of requirements and percentage tested
* Percentage of test cases resulting in defect detection
* Number of defects attributed to test case/test script creation
* Percentage of defects identified; listed by cause and severity
* Time to re-test

Attachment A – Approval Signatures

The Master Test Plan documents the project’s overall approach to testing and includes:

* Items to be tested
* Test strategy
* Test criteria
* Test deliverables
* Test schedule
* Test environments
* Staffing and training needs
* Risks and constraints
* Test Metrics

This section is used to document the approval of the Master Test Plan during the Formal Review. The review should be ideally conducted face to face where signatures can be obtained ‘live’ during the review however the following forms of approval are acceptable:

1. Physical signatures obtained face to face or via fax
2. Digital signatures tied cryptographically to the signer
3. /es/ in the signature block provided that a separate digitally signed e-mail indicating the signer’s approval is provided and kept with the document.

NOTE: Delete the entire section above prior to final submission.

REVIEW DATE: <Date>

Signed: Date:

< Program/Project Manager >

Signed: Date:

< Business Sponsor Representative >

Signed: Date:

<Project Team Test Manager>

Appendix A - Test Type Definitions

| Test Type | Definition |
| --- | --- |
| Access Control Testing | A type of testing that attests that the target-of-test data (or systems) are accessible only to those actors for which they are intended, as defined by use cases. Access Control Testing verifies that access to the system is controlled and that unwanted or unauthorized access is prohibited. This test is implemented and executed on various targets-of-test. |
| Benchmark Testing: | A type of performance testing that compares the performance of new or unknown functionality to a known reference standard (e.g., existing software or measurements). For example, benchmark testing may compare the performance of current systems with the performance of the Linux/Oracle system. |
| Build Verification Testing  (Prerequisite: Smoke Test) | A type of testing performed for each new build, comparing the baseline with the actual object properties in the current build. The output from this test indicates what object properties have changed or don’t meet the requirements. Together with the Smoke test, the Build Verification test may be utilized by projects to determine if additional functional testing is appropriate for a given build or if a build is ready for production. |
| Business Cycle Testing | A type of testing that focuses upon activities and transactions performed end to end over time. This test type executes the functionality associated with a period of time (e.g., one-week, month, or year). These tests include all daily, weekly, and monthly cycles, and events that are date-sensitive (e.g., end of the month management reports, monthly reports, quarterly reports, and year-end reports). |
| Capacity Testing | [Capacity](http://www.geekinterview.com/question_details/48768) testing occurs when you simulate the number of users in order to stress an application's hardware and/or network infrastructure. Capacity testing is done to determine the capacity (CPU, Data Storage, LAN, WAN, etc.) of the system and/or network under test. |
| Compliance Testing | A type of testing that verifies that a collection of software and hardware fulfills given specifications. For example, these tests will minimally include: “core specifications for rehosting – ver.1.5-draft 3.doc”, Section 508 of The Rehabilitation Act Amendments of 1998, Race and Ethnicity Test, and VA Directive 6102 Compliance. It does not exclude any other tests that may also come up. |
| Component Integration Testing | Testing performed to expose defects in the interfaces and interaction between integrated components as well as verifying installation instructions. |
| Configuration Testing | A type of testing concerned with checking the programs compatibility with as many possible configurations of hardware and system software. 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, for example, applications, drivers, and so on hand, at any one time; many different combinations may be active using different resources. The goal of the configuration test is finding a hardware combination that should be, but is not, compatible with the program. |
| Contention Testing | A type of performance testing that executes tests that cause the application to fail with regard to actual or simulated concurrency. Contention testing identifies failures associated with locking, deadlock, livelock, starvation, race conditions, priority inversion, data loss, loss of memory, and lack of thread safety in shared software components or data. |
| Data and Database Integrity Testing | A type of testing that verifies that data is being stored by the system in a manner where the data is not compromised by the initial storage, updating, restoration, or retrieval processing. This type of testing is intended to uncover design flaws that may result in data corruption, unauthorized data access, lack of data integrity across multiple tables, and lack of adequate transaction performance. The databases, data files, and the database or data file processes should be tested as a subsystem within the application. |
| Documentation Testing | Documentation testing is a type of testing that should validate the information contained within the software documentation set for the following qualities: compliance to accepted standards and conventions, accuracy, completeness, and usability. The documentation testing should verify that all of the required information is provided in order for the appropriate user to be able to properly install, implement, operate, and maintain the software application. The current VistA documentation set can consist of any of the following manual types:  Release Notes, Installation Guide, User Manuals, Technical Manual, and Security Guide. |
| Error Analysis Testing | This type of testing verifies that the application checks for input, detects invalid data, and prevents invalid data from being entered into the application. This type of testing also includes the verification of error logs and error messages that are displayed to the user. |
| Exploratory Testing | A technique for testing computer software that requires minimal planning and tolerates limited documentation for the target-of-test in advance of test execution, relying on the skill and knowledge of the tester and feedback from test results to guide the ongoing test effort. Exploratory testing is often conducted in short sessions in which feedback gained from one session is used to dynamically plan subsequent sessions. |
| Failover Testing | A type of testing test that ensures an alternate or backup system properly “takes over” (i.e., a backup system functions when the primary system fails). Failover Testing also tests that a system continually runs when the failover occurs, and that the failover happens without any loss of data or transactions. Failover Testing should be combined with Recovery Testing. |
| Installation Testing | A type of testing that verifies that the application or system installs as intended on different hardware and software configurations, and under different conditions (e.g., a new installation, an upgrade, and a complete or custom installation). Installation testing may also measure the ease with which an application or system can be successfully installed, typically measured in terms of the average amount of person-hours required for a trained operator or hardware engineer to perform the installation. Part of this installation test is to perform an uninstall. As a result of this uninstall, the system, application and database should return to the state prior to the install. |
| Integration Testing | An incremental series of tests of combinations or sub-assemblies of selected components in an overall system. Integration testing is incremental in a successively larger and more complex combinations of components tested in sequence, proceeding from the unit level (0% integration) to eventually the full system test (100% integration). |
| Load Testing | A performance test that subjects the system to varying workloads in order to measure and evaluate the performance behaviors and abilities of the system to continue to function properly under these different workloads. Load testing determines and ensures that the system functions properly beyond the expected maximum workload. Additionally, load testing evaluates the performance characteristics (e.g., response times, transaction rates, and other time-sensitive issues). |
| Migration Testing | A type of testing that follows standard VistA and Health*e*Vet (H*e*V)-VistA operating procedures and loads the latest .jar version onto a live copy of VistA and H*e*V-VistA. The following are examples of the types of tests that can be performed as part of migration testing:  Data conversion has been completed  Data tables are successfully created  Parallel test for confirmation of data integrity  Review output report, before and after migration, to confirm data integrity  Run equivalent process, before and after migration |
| Multi-Divisional Testing | A type of testing that ensures that all applications will operate in a multi-division or multi-site environment recognizing that an enterprise perspective while fully supporting local health care delivery. |
| Parallel Testing | The same internal processes are run on the existing system and the new system. The existing system is considered the “gold standard”, unless proven otherwise. The feedback (expected results, defined time limits, data extracts, etc.) from processes from the new system are compared to the existing system. Parallel testing is performed before the new system is put into a production environment. |
| Performance Monitoring Testing | Performance profiling assesses how a system is spending its time and consuming resources. This type of performance testing optimizes the performance of a system by measuring how much time and resources the system is spending in each function. These tests identify performance limitations in the code and specify which sections of the code would benefit most from optimization work. The goal of performance profiling is to optimize the feature and application performance. |
| Performance Testing | Performance Testing assesses how a system is spending its time and consuming resources. Performance testing optimizes a system by measuring how much time and resources the system is spending in each function. These tests identify performance limitations in the code and specify which sections of the code would benefit most from optimization work. Performance testing may be further refined by the use of specific types of performance tests, such as, benchmark test, load test, stress test, performance monitoring test, and contention test. |
| Performance – Benchmark Testing | A type of performance testing that compares the performance of new or unknown functionality to a known reference standard (e.g., existing software or measurements). For example, benchmark testing may compare the performance of current systems with the performance of the Linux/Oracle system. |
| Performance – Contention Testing | A type of performance testing that executes tests that cause the application to fail with regard to actual or simulated concurrency. Contention testing identifies failures associated with locking, deadlock, livelock, starvation, race conditions, priority inversion, data loss, loss of memory, and lack of thread safety in shared software components or data. |
| Performance – Endurance Testing | Endurance testing, also known as Soak testing, is usually done to determine if the system can sustain the continuous expected load. During soak tests, memory utilization is monitored to detect potential leaks. |
| Performance – Load Testing | A performance test that subjects the system to varying workloads in order to measure and evaluate the performance behaviors and abilities of the system to continue to function properly under these different workloads. Load testing determines and ensures that the system functions properly beyond the expected maximum workload. Additionally, load testing evaluates the performance characteristics (e.g., response times, transaction rates, and other time-sensitive issues). |
| Performance - ProfilingTesting | Performance profiling assesses how a system is spending its time and consuming resources. This type of performance testing optimizes the performance of a system by measuring how much time and resources the system is spending in each function. These tests identify performance limitations in the code and specify which sections of the code would benefit most from optimization work. The goal of performance profiling is to optimize the feature and application performance. |
| Performance – Spike Testing | A performance test in which an application is tested with sudden increment and decrements in the load. The focus is on system behavior during dramatic changes in load. |
| Privacy Testing | A type of testing that ensures that (1) veteran and employee data are adequately protected and (2) systems and applications comply with the Privacy and Security Rule provisions of the Health Insurance Portability and Accountability Act (HIPAA). |
| Product Component Testing | Product Component Testing (aka Unit Testing) is the internal technical and functional testing of a module/component of code. Product Component Testing verifies that the requirements defined in the detail design specification have been successfully applied to the module/component under test. |
| Recovery Testing | A type of testing that causes an application or system to fail in a controlled environment. Recovery processes are invoked while an application or system is monitored. Recovery testing verifies that application or system, and data recovery is achieved. Recovery Testing should be combined with Failover Testing. |
| Regression Test | A type of testing that validates existing functionality still performs as expected when new functionality is introduced into the system under test. |
| Risk Based Testing | A type of testing based on a defined list of project risks. It is designed to explore and/or uncover potential system failures by using the list of risks to select and prioritize testing. |
| Section 508 Compliance Testing | A type of test that (1) ensures that persons with disabilities have access to and are able to interact with graphical user interfaces and (2) verifies that the application or system meets the specified Section 508 Compliance standards. |
| Security Testing | A type of test that validates the security requirements and to ensure readiness for the independent testing performed by the Security Assessment Team as used by the Assessment and Authorization Process. |
| Smoke Test | A type of testing that ensures that an application or system is stable enough to enter testing in the currently active test phase. It is usually a subset of the overall set of tests, preferably automated, that touches parts of the system in at least a cursory way. |
| Stress Testing | A performance test implemented and executed to understand how a system fails due to conditions at the boundary, or outside of, the expected tolerances. This failure typically involves low resources or competition for resources. Low resource conditions reveal how the target-of-test fails that is not apparent under normal conditions. Other defects might result from competition for shared resources (e.g., database locks or network bandwidth), although some of these tests are usually addressed under functional and load testing. Stress Testing verifies the acceptability of the systems performance behavior when abnormal or extreme conditions are encountered (e.g., diminished resources or extremely high number of users). |
| System Testing | System testing is the testing of all parts of an integrated system, including interfaces to external systems. Both functional and structural types of testing are performed to verify that the system performance, operation and functionality are sound. End to end testing with all interfacing systems is the ultimate version. |
| Usability Testing | Usability testing identifies problems in the ease-of-use and ease-of-learning of a product. Usability tests may focus upon, and are not limited to: human factors, aesthetics, consistency in the user interface, online and context-sensitive help, wizards and agents, user documentation. |
| User Functionality Test | User Functionality Test (UAT) is a type of Acceptance Test that involves end-users testing the functionality of the application using test data in a controlled test environment. |
| User Interface Testing | User-interface (UI) testing exercises the user interfaces to ensure that the interfaces follow accepted standards and meet requirements. User-interface testing is often referred to as GUI testing. UI testing provides tools and services for driving the user interface of an application from a test. |

Template Revision History

| Date | Version | Description | Author |
| --- | --- | --- | --- |
| November 2015 | 1.18 | Expanded Section 4.3 to better describe responsibilities for 508 compliance. | Channing Jonker |
| October 2015 | 1.17 | Corrected broken link to 508 URL. | Channing Jonker |
| June 2015 | 1.16 | Updated metadata to show record retention information and required by PMAS, VHA Release Management, Enterprise Operations, and VistA Intake Program | Process Management |
| May 2015 | 1.15 | Reordered cover sheet to enhance SharePoint search results | Process Management |
| March 2015 | 1.14 | Miscellaneous updates including the addition of Performance testing. | Channing Jonker |
| November 2014 | 1.13 | Updated to latest Section 508 conformance guidelines and remediated with Common Look Office Tool | Process Management |
| August 2014 | 1.12 | Removed requirements for ESE Approval Signature | Process Management |
| October 2013 | 1.11 | Converted to Microsoft Office 2007-2010 format | Process Management |
| July 09, 2012 | 1.10 | Added System Design Document to Section 1.2 -Test Objectives as an example | Process Management |
| January 03, 2012 | 1.9 | Updated Approval Signatures for Master Test Plan in Appendix a | Process Management |
| October 13, 2011 | 1.8 | Replaced references to Test and Certification with Independent Test and Evaluation. Replaced references to Certification and Accreditation with Assessment and Authorization. | Process Management |
| October 4, 2011 | 1.7 | Repaired link to Privacy Impact Assessment | Process Management |
| August 23, 2011 | 1.6 | Changed Operational Readiness Testing (ORT) to Operational Readiness Review (ORR) | Process Management |
| April 12, 2011 | 1.5 | Updated the Signatory Authorities in Appendix A in light of organizational changes | Process Management |
| February 2011 | 1.4 | Removed Testing Service Testing and Operational Readiness Testing; added Enterprise System Engineering Testing.  Changed Initial Operating Capability Testing to Initial Operating Capability Evaluation | Process Management |
| January 2011 | 1.3 | Repaired broken link in section 1.4 | Process Management Service |
| August 2010 | 1.2 | Removed OED from template | Process Management Service |
| December 2009 | 1.1 | Removed “This Page Intentionally Left Blank” pages. | OED Process Management Service |
| July 2009 | 1.0 | Initial ProPath release | OED Process Management Service |