

# **Department of Veterans Affairs**

**Clinical Ancillary Services (CAS)**

**Development – Delivery of Pharmacy Enhancements (DDPE)**

**VA Pharmacy Interface Automation**

**Master Test Plan**



**July 2015**

## Revision History

Date	Version	Description	Author
7/29/2015	.5	Various sections updated to include Pharmacy Subject Matter Experts role.	REDACTED
7/7/2015	.4	Updated hyperlinks. Added link to PMP to Staffing and Training section. Technical edit.	REDACTED
6/26/2015	.3	Technical edit.	REDACTED
6/8/2015	.2	Modified as per Review	REDACTED
3/11/2015	.1	Draft	REDACTED

## Table of Contents

<b>1. Introduction .....</b>	<b>1</b>
1.1. Purpose .....	1
1.2. Test Objectives .....	2
1.3. Roles and Responsibilities .....	2
1.4. Processes and References .....	3
<b>2. Items to Be Tested .....</b>	<b>3</b>
2.1. Overview of Test Inclusions .....	3
2.2. Overview of Test Exclusions .....	4
<b>3. Test Approach.....</b>	<b>4</b>
3.1. Product Component Test.....	5
3.2. Component Integration Test .....	6
3.3. System Tests.....	6
3.4. User Functionality Test .....	7
3.5. Enterprise System Engineering Testing .....	7
3.6. Initial Operating Capability Evaluation .....	8
<b>4. Testing Techniques .....</b>	<b>8</b>
4.1. Risk-based Testing.....	8
4.2. Enterprise Testing .....	9
4.2.1. Security Testing.....	9
4.2.2. Privacy Testing .....	9
4.2.3. Section 508 Compliance Testing .....	9
4.2.4. Multi-Divisional Testing.....	9
4.3. Test Types .....	9
4.4. Productivity and Support Tools.....	11
<b>5. Test Criteria .....</b>	<b>11</b>
5.1. Process Reviews.....	11
5.2. Pass/Fail Criteria.....	12
5.3. Suspension and Resumption Criteria .....	13
5.4. Acceptance Criteria .....	13
<b>6. Test Deliverables .....</b>	<b>13</b>
<b>7. Test Schedule.....</b>	<b>14</b>
<b>8. Test Environments.....</b>	<b>14</b>
8.1. Test Environment Configurations .....	14
8.2. Base System Hardware .....	14
8.3. Base Software Elements in the Test Environments.....	14

<b>9. Staffing and Training Needs.....</b>	<b>14</b>
<b>10. Risks and Constraints .....</b>	<b>15</b>
<b>11. Test Metrics .....</b>	<b>15</b>
<b>Attachment A - Approval Signatures.....</b>	<b>16</b>
<b>A. Test Type Definitions .....</b>	<b>17</b>

# 1. Introduction

The Clinical Ancillary Services (CAS) Development – Delivery of Pharmacy Enhancements (DDPE) VA Pharmacy Interface Automation (PIA) aims to develop a new bidirectional interface to enhance the Department of Veterans Affairs (VA) dispensing of medications via Pharmacy Automated Dispensing Equipment (PADE).

The PIA Master Test Plan will provide the planning and control of the test effort for development testing provided by Hewlett Packard Enterprise Services (HPES) for the CAS DDPE PIA project. This test plan also defines the test approach that will be employed to test the application's software for the VA PIA and evaluate the testing results.

## 1.1. Purpose

The purpose of the Master Test Plan for the CAS DDPE VA PIA project is to:

- Provide a central artifact to govern the planning and control of the test effort for the PIA project. It defines the general testing approach the team will follow. This is the top-level plan that will be used by managers to govern and direct the detailed testing work.
- Provide visibility into the testing efforts for stakeholders.

This document will also support the following objectives:

- Identify the functional requirements and user stories utilized for testing.
- Identify the assumptions, risks, and constraints that affect this testing process.
- Outline the testing schedule.
- Describe the testing strategy, activities, and tools.
- Include the roles and responsibilities of the resources participating in the testing.
- Document the pass/fail performance of the software design parameters.

## 1.2. Test Objectives

This Master Test Plan supports the following objectives:

- Provide test coverage for 100% of the documented requirements within the approved PIA Requirements Specifications Document (RSD)
- Execute 100% of the test cases during System Testing and User Functionality Testing
- Create, maintain, and control the test environments
- Outline and document the testing approach that will be used, including the progression of testing coverage and references to quality milestones.
- Identify testing resources and needs
- List the test deliverables
- Identify the defect reporting processes
- Describe the system to be tested, the test inclusions and exclusions, and the roles and resource participation for test and configuration management

## 1.3. Roles and Responsibilities

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

**Table 1: Roles and Descriptions**

Role	Description
PIA Project Team (HPES)	Persons responsible for internal peer review of the application specific test plan, test cases/scripts, traceability of test cases to requirements in Requirements Traceability Matrix.
PIA Developers (HPES)	Persons that build or construct the product/product component.
PIA Project Manager (HPES)	Person that has overall responsibility for the successful planning and execution of a project.
Stakeholders	Persons that hold a stake in a situation in which they may affect or be affected by the outcome.
Subject Matter Experts	VHA PBM Clinical Informatics Analysts and end users that provide input to system design and testing.
SQA Test Analyst (HPES)	Person responsible for ensuring full execution of the test process to include the verification of functional requirements and the validation of business requirements.
SQA Test Lead (HPES)	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.
Test Team/Testers	Persons that execute tests and ensure the test environment will adequately support planned test activities.
Configuration Management Team	Persons that establish, maintain, and control test environments.

## 1.4. 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](#)
- [Section 508 Office Web Page](#)
- [Privacy Impact Assessment - Privacy Service](#)

## 2. Items to Be Tested

All functional requirements documented within Section 2.6 Functional Specifications of the PIA Requirements Specification Document (RSD) will be tested within the PIA Test Plan.

Please refer to the RSD in the [Technical Services Project Repository \(TSPR\)](#),

### 2.1. Overview of Test Inclusions

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

- Set up the PADE Server and PADE in VistA Pharmacy Data Management (PDM) for bidirectional HL7 transmissions
- Generate HL7 transmissions of Admission, Discharge and Transfer (ADT), Allergy, Surgery, orders, and drug file information from VistA to PADE
- Generate HL7 transmissions of dispensing and inventory information from PADE to VistA
- Display/generate Pharmacy Automated Dispensing Equipment (PADE) Activity Log, PADE On Hand Amounts Report, and PADE Transactions Report.

## 2.2. Overview of Test Exclusions

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

- Requirements that are out of scope for the VA Pharmacy Interface Automation (PIA) project.
- Testing of functionality within the VistA Pharmacy that was not impacted by the VA PIA project.
- Regression testing of existing VistA applications.
- Data flow testing of existing VistA applications.
- Development within the VistA application done by other project teams.
- Development performed by the PADE vendors (e.g. PYXIS, Omnicell, AccuDose). For instance, PADE issues that may arise while testing VistA to PADE or PADE to VistA transactions will be fixed and tested by the vendors.

## 3. Test Approach

The Pharmacy Interface Automation project utilizes a modified Agile Methodology. The epics, user stories, and requirements developed on each sprint will be documented in the Requirements Specifications Document (RSD) and in IBM Rational Team Concert (RTC). The SQA Test Analyst will work closely with the Business Analysts and Developers to develop test cases and test scripts as per sprint schedule.

The test cases and test scripts will be documented in the test deliverables and IBM Rational Quality Manager (RQM). The test cases will be mapped to the requirements in the Requirements Traceability Matrix (RTM). The test results will be recorded in the RQM, RTM, and summarized in the Test Evaluation Summary. Through this modified agile process, evolving prototypes of components will eventually stabilize in design and reach a release-worthy level to be included deployed for national release.

The following test environments will be used:

- **DVA namespace in CDDEVISTA** – Development Environment used for development of the code and for Product Component Testing conducted by the developers.
- **DVB namespace in CDDEVISTA** – SQA Test Environment used for conducting Component Integration Testing (CIT), System Testing, as well as UFT.

Testing will include the following:

- Systematic Testing – Execution of test cases designed for test coverage with expected results.
- Negative Testing – Execution of test cases designed to produce unexpected results for test coverage where practical.
- Ad-hoc/Exploratory Testing – Execution of test cases designed for test coverage of a particular feature.



- Regression Testing – A type of testing that validates existing functionality still performs as expected when new functionality is introduced into the system under test.
- Feature Testing – test performed by the SQA analyst for each sprint to verify that the feature was completed.

### **3.1. Product Component Test**

As PIA is in modified agile development, Product Component Testing will be conducted by the HP Enterprise Services (HPES) Project Team developers within the Albany test environment, DVA namespace in CDDEVISTA for each sprint. The Developer performs Product Component Testing (aka Unit Testing), which includes the internal technical and functional testing of a module/component of code, and verifies that the requirements defined in the SDD have been successfully applied to the module/component under test.

Upon successful result, the developer will inform the SQA analyst that the new feature is ready to be tested and will provide instructions for testing. Steps may 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 on domain knowledge, past performance, and expertise
- Record the actual test
- Perform M Code Secondary Developer's Review Checklist

## 3.2. Component Integration Test

The SQA Test Analyst installs the Product Component, performs the Component Integration Test (CIT) and Smoke Test within the DVB test environment as follows:

1. The SQA Test Analyst will install the PIA patch into DVB.
2. The SQA Test Analyst will perform CIT to expose defects in the interfaces and interaction between integrated components. CIT will verify the patch installation and the testing environment. The installation will be installed into the testing environment ensuring the installation process does complete and does not produce any errors.
3. At start of CIT, the SQA Test Analyst will initialize the SQA Review Checklist and complete the following sections:
  - Pre Installation (All Steps)
  - Installation (All Steps)
  - Post Installation (Steps 1-13)
4. CIT will be followed by a Smoke Test to verify stability of the environment. The Smoke test will consist of a subset of the test scripts and will cover critical path application functionality to ensure the application is functioning properly after the installation of the PIA patch software. This testing also verifies that the application is stable and not prone to errors due to the new software installation (Approximately 10 test scripts).
5. The SQA Test Analyst will record the test result. If there is a defect found, the SQA Test Analyst will send an email to the Developers with the test results and continue performing the smoke test, if possible. The severity of the defect will be assessed by the team as to whether testing can proceed to System Testing. If it is deemed necessary to resolve the defect prior to proceeding to System Test, then a new patch will be created with the fix and it will go back through product component test and CIT.

## 3.3. System Tests

The SQA Test Analyst performs System Test, employing a variety of test types (i.e., compliance, regression, access control, interoperability, etc.). System Tests exercise all parts of an integrated system including interfaces to external systems. Each application should list here which system test types they are performing. You can also refer to section 4.3 for additional info,

1. The SQA Test Analyst will perform System Test in DVB PIA SQA Test Environment.
2. The SQA Test Analyst will record test results in Rational Quality Manage(RQM).

When System Test is complete:

1. Defects written will be logged in RQM following the Work Instructions provided by the CM team.
2. Testing Results reports are generated and distributed (frequency to be determined). Same comment as above.
3. System testing completed.

4. End of testing reports are generated and distributed along with the completed SQA Review Checklist, and Test Evaluation Summary.
5. At successful completion of SQA testing by HP, HP PM works with HP Developer (MUMPS) if needed to create build/patch to send to PMO CM to load into their test account.
6. PMO CM member loads the build/patch into their account.

HPES testing ends unless defects are reported during UFT or IOC and a new product build needs to be tested. In this case, the test cycle begins again starting with product component test, followed by component integration test, and system test.

### **3.4. User Functionality Test**

User Functionality Test (UFT) will be performed upon completion of System Testing. Testing will be performed for both increments after development of the second increment by stakeholders/subject matter experts at the test sites, testing the functionality of the application using test data in the facility's test account(s). The SQA Test Team will provide the test scripts for the test sites, and the stakeholders/subject matter experts will provide results for reporting progress and completion of UFT by delivering completed test scripts back to the SQA Test Team.

The SQA Test Analyst will document UFT test results and deliver the documentation to the Development Manager and Project Manager, in the form of a UFT Test Defect Log, UFT Test Execution Log and UFT Test Evaluation Summary as per the documented process in ProPath. The content of these documents will be derived from the test results the stakeholders/subject matter experts provide from their completed test scripts.

UFT is managed by the PMO project team. The HPES Project Team will support UFT as follows:

1. Defect will be triaged by the SQA Analyst.
2. Developers review and rework code.
3. Defect correction will be retested by SQA Analyst.
4. New build will be sent to the PMO project team for retest.

### **3.5. Enterprise System Engineering Testing**

Enterprise System Engineering (ESE) testing will be conducted by the ESE Testing Services team and managed by the Program Management Office (PMO) project team. The HPES Project Team will support ESE testing as follows:

1. Defect will be triaged by the SQA Analyst.
2. Developers review and rework code.
3. Defect correction will be retested by SQA Analyst.
4. New build will be sent to will be sent to the PMO project team for retest.

### **3.6. Initial Operating Capability Evaluation**

Initial Operating Capability (IOC) testing will be performed upon completion of UFT after the PIA Project Manager has submitted the IOC entry request form and approval received. Testing will be performed at the test sites in the production accounts and PADE machines, using the same test scripts provided during UFT. IOC testing will continue for a minimum of two weeks and lasting up to 2 months. The stakeholders will provide results for reporting progress and completion of IOC by delivering completed test scripts back to the HP Software Quality Assurance (SQA) Test Team.

Each test site will respond via MS Outlook and provide written approval of the increment when IOC testing has been completed for that site. The HP SQA Test Analyst will document IOC test results and deliver the documentation to the Development Manager and Project Manager after the increment has been tested, in the form of an IOC Test Evaluation Summary as per documented process in ProPath. The content of these documents will be derived from the test results the stakeholders provide via the collaboration software portal.

## **4. Testing Techniques**

The testing for the System Test phase will be conducted as primarily manual testing by the SQA teams. The tests utilized during this portion of testing will be developed using the system requirements that are stored in the Rational Change and Configuration Management. The test scripts will cover each testable requirement that has been documented and is traceable back to the original analysis and design documentation for each specific application and as documented in the Requirements Traceability Matrix.

The testing results for each test case executed will be analyzed and recorded daily within the Rational Quality Manager tool. The results of the testing will be compiled into the required testing result reports that are to be delivered to the customer.

### **4.1. Risk-based Testing**

The PIA project will test identified risks throughout each testing cycle.

## 4.2. Enterprise Testing

### 4.2.1. Security Testing

The PIA project adheres to existing standard VistA security specifications; therefore the need to perform Security Testing is not applicable.

### 4.2.2. Privacy Testing

The PIA project adheres to existing standard VistA privacy specifications, therefore the need to perform Privacy Testing is not applicable.

### 4.2.3. Section 508 Compliance Testing

The PIA development team is responsible for ensuring that the PIA updates are usable from the keyboard and are compliant with Section 508 requirements. The PIA development team will perform 508 testing and submit the appropriate self-certification forms when necessary to support Section 508 compliance.

### 4.2.4. Multi-Divisional Testing

The PIA will not replace or modify the existing VistA or HealtheVet Application Architecture; therefore the need to perform multi-divisional tests is not applicable.

## 4.3. Test Types

**Table 2: Test Types**

Test Types	Party Responsible
Access control testing	N/A
Benchmark testing	N/A
Build verification testing	Development team
Business cycle testing	N/A
Compliance testing	ESE, 508 Office, Development Team and Test Team
Component integration testing	Development Team
Configuration testing	n/a
Contention testing	n/a
Data and database integrity testing	Test Team
Documentation testing	Test Team
Error analysis testing	Test Team

<b>Test Types</b>	<b>Party Responsible</b>
Exploratory testing	Development Team, Test Team
Failover testing	n/a
Installation testing	Development Team
Integration testing	Development, Test Team
Load testing	n/a
Migration testing	n/a
Multi-divisional testing	n/a
Parallel testing	n/a
Performance monitoring testing	n/a
Performance testing	n/a
Privacy testing	n/a
Product component testing	Development Team
Recovery testing	n/a
Regression test	Test Team
Risk based testing	Test Team
Section 508 compliance testing	Development Testing, Test Team, 508 Office
Security testing	n/a
Smoke testing	Test Team
Stress testing	n/a
System testing	Test Team
Usability testing	n/a
User Functionality Testing	Stakeholders and Subject Matter Experts
User interface testing	Test Team, Test Sites

## 4.4. Productivity and Support Tools

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

**Table 3: Tool Category or Types**

Tool Category or Type	Tool Brand Name	Vendor or In-house	Version
Test Management	Rational Quality Manage	IBM	4.0
Defect Tracking	Rational Change and Configuration Management (CCM)	IBM	4.0.5
Project Management	Rational Change and Configuration Management (CCM)	IBM	4.0.5
Configuration Management	Rational Change and Configuration Management (CCM)	IBM	4.0.5
Requirements Management	Rational Requirements Composer	IBM	4.0.5
Epic and User Story Management	Rational Team Concert (RTC)	IBM	4.0

## 5. Test Criteria

### 5.1. 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.

## 5.2. Pass/Fail Criteria

**Pass** – A Test script will be considered as passed when all of the conditions, steps, and requirements that are indicated as being covered within the test script have been executed without a variance in the described functionality and expected outcome.

**Fail** – A Test script will be considered as failed when all of the conditions, steps, and requirements that are indicated as being covered within the test script have been executed and any of the following conditions exist.

- Any defect that can compromise patient safety or system security. Examples of system security defects include breach of confidentiality requirements of the Privacy Act, Health Insurance Portability and Accountability Act (HIPAA) or Federal Tax Information guidelines. Any requirements that might potentially impact patient safety will be marked as such.
- Any defect that causes corruption of data from a result of the system (as opposed to user error) that cannot be corrected, within reason through the application.
- Any defect in which inappropriate transmissions are consistently generated or appropriate transmissions of Health Level Seven (HL7) messages fail to be generated.
- Any defects in the functionality which result in erroneous information being communicated to the user or patient.
- Any defects in the functionality resulting in a failure of all or part of the application where the expected results can temporarily be achieved by alternative means and the customer indicates the workaround is acceptable for the short term.
- A crash or hang that prevents further testing of the complete application or a section of the application.
- A loss of system functionality with no suitable workaround (i.e., there is no way to achieve the expected results using the application).
- A minor functionality is not working as intended and a workaround exists that the customer determines does not prevent the user from continuing to test the affected part of the system but may not be desirable in the long term.
- The inability of a valid user to access the system consistently with granted privileges.
- Any cosmetic issues that do not result in data entry and or data quality problems.
- A minor loss of or defect in the functionality.
- Any low-level cosmetic issues.



### 5.3. Suspension and Resumption Criteria

**Suspension** – Every effort will be made to continue testing when failures are found. However, some failures can affect product functionality to the point where further execution is not advisable. At this point, testing will be suspended or blocked until a fix or workaround is provided.

**Resumption** – In general, testing will be resumed from the point of suspension once a fix or workaround is provided. If the fix or workaround has implications outside the boundaries of the immediate test case procedure, it will be necessary to execute additional associated test procedures or test cases.

### 5.4. Acceptance Criteria

The following are the acceptance criteria that must be satisfied for the patch/build to be accepted by the customer, or other authorized entity.

- All business and functional requirements have been validated.
- All test cases/scripts are complete
- All tests have been executed.
- The test environment(s) are validated/verified.
- Test data requirements are determined and met.
- All Critical and Major defects are corrected.
- Customer approves ALL remaining open defects within the PIA project and agrees to a corrective action plan within the remaining testing phases.

## 6. Test Deliverables

Table 4 lists the test deliverables for the Pharmacy Interface Automation project.

**Table 4: Test Deliverables**

Test Deliverables	Responsible Party
Master Test Plan	Regina Devia, SQA Analyst (HPES)
Test Cases/Test Scripts	Regina Devia, SQA Analyst (HPES)
Test Evaluation Summaries	Regina Devia, SQA Analyst (HPES)
Traceability Report or Matrix	Regina Devia, SQA Analyst (HPES)
SQA Review Checklist	Regina Devia, SQA Analyst (HPES)

## 7. Test Schedule

The overall project schedule is being managed within the Project Management Plan (PMP).

Refer to TSPR for the current project plan:

<http://tspr.vista.med.va.gov/warboard/anotebk.asp?proj=1772&Type=Active>

**Table 5: Testing Milestones**

Testing Milestones	Responsible Party
Complete Master Test Plan	Regina Devia, SQA Analyst (HPES)
Complete test cases and/or scripts	Regina Devia, SQA Analyst (HPES)
Complete Requirements Traceability Matrix	Regina Devia, SQA Analyst (HPES)
Complete Systems Test	Test Team
Complete UFT Test	Stakeholders and Subject Matter Experts
Complete IOC Test	Stakeholders and Subject Matter Experts

## 8. Test Environments

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

### 8.1. Test Environment Configurations

The party responsible for configuring and maintaining the test environments is the HPES Configuration Manager.

### 8.2. Base System Hardware

The PIA project is updating existing software; therefore, no changes are required to existing base system hardware unless otherwise specified.

### 8.3. Base Software Elements in the Test Environments

Please refer to the PIA system design document located on the TSPR:

<http://tspr.vista.med.va.gov/warboard/anotebk.asp?proj=1772&Type=Active>

## 9. Staffing and Training Needs

There are no specific training needs at this time. This section will be updated as training needs arise.

Refer to the [PIA PMP](#) for additional information on staffing and training.

## 10. Risks and Constraints

The risks identified in this Master Test Plan may be recorded and tracked in an automated tool, such as, IBM Rational ClearQuest®.

## 11. 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

The Final Test Evaluation Summary Report completed by the SQA Test Analyst captures the measures specified above.



## A. Test Type Definitions

Test analysts use “test types” to validate the system or application under test. Simply put, test types are test techniques used to exercise the system or application. This table presents a listing of possible test types that may be utilized during the Product Build, Independent Testing, Operational Readiness Review (ORR) and Initial Operating Capability (IOC) Testing. The test analyst in consultation with the Development Manager selects the test types best suited to the system or application being tested. A minimum set of test types is suggested here. More tests may be added at the discretion of the Development Team.

Description	Product Build Testing	Independent Testing	IOC Testing
Types of Test			
Access Control Testing	X		
Benchmark Testing			
Build Verification Testing		X	
Business Cycle Testing			X
Compliance Testing	X		
Component Integration Testing			
Configuration Testing			
Contention Testing			
Data and Database Integrity Testing			
Documentation Testing	X		X
Error Analysis Testing			X
Exploratory Testing	X	X	X
Failover Testing			
Installation Testing	X		
Integration Testing	X	X	
Load Testing		X	
Migration Testing			
Multi-Divisional Testing			
Parallel Testing			
Performance Monitoring Testing			
Performance Testing		X	
Privacy Testing	X		

Description	Product Build Testing	Independent Testing	IOC Testing
Product Component Testing	X		
Recovery Testing			
Regression Test	X		
Risk Based Testing	X		X
Section 508 Compliance Testing	X		
Security Testing			
Smoke Testing	X	X	
Stress Testing		X	
System Testing	X	X	
Usability Testing			X
User Functionality Testing	X		
User Interface Testing			

## A.1. 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.

Test Type	Definition
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).
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.

Test Type	Definition
Documentation Testing	<p>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:</p> <p>Release Notes, Installation Guide, User Manuals, Technical Manual, and Security Guide.</p>
Error Analysis Testing	<p>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.</p>
Exploratory Testing	<p>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.</p>
Failover Testing	<p>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.</p>
Installation Testing	<p>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.</p>



Test Type	Definition
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 H <sub>e</sub> V-VistA operating procedures and loads the latest .jar version onto a live copy of VistA and H <sub>e</sub> V-VistA. The following are examples of the types of tests that can be performed as part of migration testing: <ul style="list-style-type: none"> <li>• Data conversion has been completed</li> <li>• Data tables are successfully created</li> <li>• Parallel test for confirmation of data integrity</li> <li>• Review output report, before and after migration, to confirm data integrity</li> <li>• Run equivalent process, before and after migration</li> </ul>
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.

Test Type	Definition
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.
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 required by the Assessment and Authorization Process.

Test Type	Definition
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 (UFT) 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.