

Enterprise Integration Testing

Component: Enterprise Integration Testing

Component Description:

Enterprise Integration Testing is conducted by the enterprise test group to ensure that all applications included in the release interact with one another in a functionally correct manner in a production-like environment. The emphasis is on testing functional conditions which span multiple applications. The DCII Enterprise Integration test is defined as the validation of the End-to-End processing that occurs between two or more components of the DCII that were developed by separate project teams and considered separate applications. End-to-End Integration testing is two fold. First is considered to be a technical test. The technical testing would encompass validating the input and output of each DAPI. It is not the intent of End-to-End Integration technical testing to validate the reused COE APIs that were not repackaged or modified by DFAS, nor is it the intent to validate application specific DAPIs. It will be the responsibility at the project level to validate and ensure compliance of DAPIs that are used solely within the application. DAPIs that are elevated to become COE APIs will not be validated during this test, but will be the responsibility of the DII COE.

One of two methods will be used to validate the input/output of the DAPIs. One method consists of viewing the DAPI source code (visual reading of the source code) and comparing it against the document standard for compliance, the other is to develop a driver program that calls and passes data through the DAPIs for verification of the input and output.

The second part of the DCII End-to-End Integration test is considered a functional test. The functional test would concentrate on exercising only the identified DCII critical end-to-end business processes between Type I and Type II applications, and the Type III applications. The functional test would also concentrate on those business processes that overlap between applications within the DCII. The scope of this test would be limited to end-to-end processing within DFAS, and would not include COTS and GOTS packages.

Inputs:

- 1) Enterprise Business Process Model (RD.010)
 - a) Event Catalogue
 1. Name
 2. Description
 3. Frequency and condition under which it occurs
 - b) Process Descriptions
 1. Description of process that is executed in response to each event
 2. Inputs
 3. Outputs
 - c) Process Step catalogue
 1. List of the process steps
 2. Person responsible for performing steps
 3. Degree of automation (manual, partial automation, full automation)
 - d) Process Flow Diagram
 - e) Process Re-engineering list
- 2) Enterprise Business Function Model(RD.030)
 - a) Function Hierarchy Diagram
 - b) Function Data Usage
 1. CRUD matrix
 2. Data flow diagram
- 3) Enterprise Business Data Model(RD.040)
 - a) Entities – the objects or things of significance that an organization wishes to hold information about
 - b) Synonyms – the aliases or other names of the entities
 - c) Attributes and domains – the key or descriptive properties of the entities
 - d) Relationship – the way entities are related or associated to one another

Enterprise Integration Testing

- e) Unique identifiers – the combination of attributes and relationships that uniquely identify each instance of the entities
- 4) DCII specification
 - a) Completed sections of Appendix D from the DCII Specification for each project
 - b) Remaining sections of Appendix D to be completed by this process
- 5) Project Level Software test plans
- 6) DCII Test and Evaluation Master Plan
- 7) Module Functional Documentation (MD.040)
 - a) Functional logic of the module
 - b) layouts
 - c) constrains
 - d) deviations of the standards for user interface
- 8) Module Technical Documentation
 - a) Module logic
 - b) Detailed table and column usage
 - c) Module parameters (in-going and outgoing)
 - d) Supporting modules
- 9) DCII Interface specification (TA.010)
 - a) Source system of data / target system for data
 - b) Format
 - c) Frequency
 - d) Available date of test data
 - e) Available date of production data
 - f) Point of Contact for interface data
 - g) Name of interface files
 - h) IP Location data is transmitted to or extracted from
 - i) Directory location data is transmitted to or extracted from
 - j) Processing or messaging that occurs to insure accurate transmission
 - k) Security constraints involved in the transmission
- 10) Enterprise Data Conversion specification (CV.010)
 - a) Data that needs to be converted
 - b) Functional requirements for extracting and loading the data
 - c) Requirements for validating and cleaning up both extract and load
 - d) Requirements for handling the error conditions
 - e) Scope
 - f) Objectives
 - g) Critical success factors and risks
- 11) Application test plans (TE.030) (TE.050)
- 12) Application system process test model(TE.020)
- 13) Application test scripts and data (TE.040) (TE.070)
- 14) Application test results (TE.040)
 - a) Software Integration test results
 - b) Functional Validation test results
 - c) Database conversion test results
 - d) System Interface test results
- 15) Stubs and drivers created to test the application
- 16) Glossary of terms(DO.010)

Outputs:

- 1) Enterprise Integration Test Plan
- 2) Documented Enterprise integration test cases / conditions
- 3) Documented test results – date and result of test execution associated with the test conditions
- 4) Enterprise Integration Test Report
- 5) Completed Appendix D of the DCII specification

Enterprise Integration Testing

Task Ordering:

- 1) [Strategy](#)
- 2) [Analysis](#)
- 3) [Design](#)
- 4) [Construct](#)
- 5) [Execute](#)

Project Management Tasks:

- 1) Enter task actual start / end date
- 2) Verify Enterprise Integration Testing time identified in SDP

Component Tasks:

Configuration Management Tasks:

- 1) Ensure Enterprise Integration Test environment is established
- 2) Verify Configuration Items are correct for migration to Enterprise Acceptance Testing

Component Metrics:

- 1) Data collected to determine size of the release to be tested:
 - a) Using CMIS / Requirements traceability matrix:
 1. Number of testable requirements identified in the release by category
 - (A) Requirement for a batch update process
 - (B) Requirement for a batch query process
 - (C) Requirement for a batch report process
 - (D) Requirement for an on-line update process
 - (E) Requirement for an on-line query process
 - (F) Requirement for an on-line report process
 2. Number of scripts created for each testable requirement
- 2) Data collected to determine the effort required to test the release:
 - a) Using LRS:
 1. Estimated / actual hours to perform test strategy
 2. Estimated / actual hours to perform test analysis
 3. Estimated / actual hours to perform test design
 4. Estimated / actual hours to perform test construct
 5. Estimated / actual hours to perform test execute
 6. Estimated / actual hours to perform test report / cleanup
- 3) Data collected to determine the schedule for the release:
 - a) Using MsProject:
 1. Estimated / actual hours to create or update scripts
 2. Estimated / actual hours required to receive release
 3. Planned / Actual start date (difference between planned and actual start date)
 4. Planned / Actual completion date (difference between planned and actual completion date)
 - b) Using the Software Test Report:
 1. Estimated / actual number of test scripts and data to be created or updated
 2. Estimated / actual manpower/resources required to perform test
 3. Planned / Actual Tasks performed(number of test cases executed, number not executed)
 4. Tasks performed that were identified in the plan
 5. Tasks performed that were not identified in the plan
 6. Number of changes made to the test environment and number of days required to make each change

Enterprise Integration Testing

7. Number of days required to troubleshoot the test environment
 8. Number of days spent defect tracking or troubleshooting the application
 9. Number of out-of-town trips and their duration
 10. Number and duration of meetings required during the test
- 4) Data collected to determine quality of testing release:
- a) Using CMIS:
 1. Number of TDRs written in next level of testing (Enterprise Acceptance Testing)
 - b) Using Remedy:
 1. Number of trouble calls received after implementation that identify problems that could have been found in testing (non-operator error problems)

Controls:

- 1) DCII Standards ([Web site](#))
 - a) Defense Corporate Information Infrastructure (DCII) specification
- 2) DII COE Standards ([Web site](#))
 - a) DII Strategic Enterprise Architectures
 - b) DII COE Interface and Run-Time Specification (I&RTS)
 - c) DII User Interface Specification (UIS)
- 3) DFAS Information Technology Architecture
 - a) Technical Architecture Framework for Information Management (TAFIM)
 - b) JTA standards
- 4) DoD Data Administration standards
 - a) Defense Data Dictionary System (DDDS)
 - b) Defense Finance and Accounting Data Model (DFADM)
- 5) Global Combat Support System (GCSS) (Interoperability)
- 6) IEEE/EIA J-STD-016
- 7) IEEE/EIA 12207

Enterprise Integration Testing

Task Name: Develop Test Strategy

Task Number: T-ST-###

Component: Enterprise integration Test

Category: Software Engineering

1. **Task Name:** Develop Test Strategy for Enterprise Integration testing

2. **Purpose:**

The purpose of this task is to identify the methods and techniques used to perform the Enterprise Integration validation; define the level of enterprise integration testing to be performed; identify the roles and responsibilities of the participants in the test, and to document the plan of action in a Software Test Plan.

During the strategy task, the plans and test scenarios from the project level testing, the Enterprise Business Model and the Enterprise Data Model are reviewed. The project level functions that overlap other projects or relate directly to the DAPIs of the DCD are identified. Common DCD functions and interoperability services are identified. The functionality to be tested in the Enterprise Integration test is documented in a Software Test Plan. The test plan is distributed to management for review and approval.

Information gathered during the strategy task is documented in Sections 1.3, 1.4, 2.0, 3.0 and 4.1 of the Software Test Plan.

3. **Roles:**

Test team lead for the Enterprise Integration Test.

4. **Entrance Criteria:**

- a. Application SIT software Test Plan
- b. Application Functional Validation software Test Plan
- c. DCII Architectural Design Specification
- d. DCII specification
- e. DCII Enterprise Business Model
- f. DCII Enterprise Business Data Model

5. **Procedures:**

During the strategy task, the Enterprise Business Process Model, the Enterprise Data Model, the plans and scenarios outlined in the Project Level Software Test Plans are reviewed and compared. Test scenarios identified at the project level that can be used to validate the Enterprise Business Model are flagged for use in the Enterprise Integration Testing. Processes that exist only at the Enterprise level are identified. A review of the DCII TEMP determines the standards and constraints to be incorporated in the testing.

- a. Determine the scope of the test
- b. Review the DCII TEMP
 - 1) Standards to be used in the testing process
 - 2) Necessary notification when test is complete
 - 3) Testing tools available for the test
 - 4) Critical technical parameters
 - 5) Management roles and responsibilities
 - 6) Test and evaluation resource summary
- c. Review the Application Software Test Plans for functions to be tested
 - 1) Are there functions that were deferred to the Enterprise level for testing

Enterprise Integration Testing

- 2) Are there test configurations that were deferred to the Enterprise level for testing
 - 3) Are there System Interfaces that were deferred to the Enterprise level for testing
 - 4) Are there FFMR requirements that were deferred to the Enterprise level for testing
 - 5) What are the critical Business Processes identified to be performed during the Enterprise Level testing
- d. Review the Test Architecture / Infrastructure for test environment information
- e. Review the Enterprise Business Process specification
- 1) Identify processes unique to Enterprise Level
 - 2) Identify processes that involve other applications
- f. Review the Enterprise Business Data model
- 1) Identify data required as input to each process
 - 2) Identify data relationships between processes
- g. Identify requirements to be tested
- 1) Functional Business Threads to be validated
 - 2) FFMR requirements to be validated
 - 3) Technical testing
 - 4) System Interface testing
- h. Review DCII compliance checklist provided by the projects (Appendix D)
- 1) Identify the DCII requirements to be verified during Enterprise integration testing
 - 2) Identify the APP requirements to be verified during Enterprise integration testing
 - 3) Identify the DCD requirements to be verified during Enterprise integration testing
 - 4) Identify the DCW requirements to be verified during Enterprise integration testing
 - 5) Identify the INT requirements to be verified during Enterprise integration testing
 - 6) Identify the CSA requirements to be verified during Enterprise integration testing
 - 7) Identify the INF requirements to be verified during Enterprise integration testing
 - 8) Identify the ACC requirements to be verified during Enterprise integration testing
 - 9) Identify the HW requirements to be verified during Enterprise integration testing
- i. Review DII COE standards
- 1) Identify person responsible for verifying DII COE standards
- j. Review JTA standards
- 1) Identify JTA standards to be verified during Enterprise integration testing
- k. Determine criteria for executing test (section 4.1.1, 4.1.2, 4.1.3)
- 1) Conditions under which each type of test will be performed
 - 2) Criteria for failing test
 - 3) Criteria for the re-test beginning at the Project level
 - 4) Criteria for the re-test beginning at the Enterprise Integration level
 - 5) Criteria for no re-test
- l. Confirm test resources identified in the TEMP(section 3.x.7)
- 1) Names of testing resources / organizations
 - 2) Skill level of testing resources
 - 3) Dates available
 - 4) Identify type of training required to perform test
 - 5) Identify optimum time to provide training
- m. Identify roles and responsibilities of all participating personnel (Section 3.x.6). Identify each of the following:
- 1) Test cases / scripts:
 - A. Who defines test cases / conditions

Enterprise Integration Testing

- B. Who provides support from the application level tests
 - C. Who participates in the test execution
 - D. Who executes the test cases
 - E. Who documents the results of the test cases
 - F. Who provides feedback on the Test Discrepancy Reports
 - G. Who certifies the test as complete
- 2) Problem reporting / tracking:
- A. Who tracks Test Discrepancy Reports (TDRs)
 - B. Who approves problems to be fixed for a release
 - C. Who is responsible for making the software fixes
 - D. Who provides feedback to the TDRs
 - E. Who documents the impacts (CIs) to fix a problem
 - F. Who tracks which fixes within a release
- 3) Test Data:
- A. Who provides test data for the test cases
 - B. Who pre-loads data for testing
 - C. Who controls test data
 - D. Who validates test data prior to test
- 4) Test Environment:
- A. Who establishes test environment
 - B. Who provides server support for the test environment
 - C. Who provides client support for the test environment
 - D. Who provides LAN support for the test environment
 - E. Who provides DBA support
 - Application level DBA support
 - Server DBA support
- 5) Configuration management:
- A. Who provides control over the software configuration items
 - B. Who provides the software releases to testing
 - C. Who provides control over test configuration items
- 6) Management responsibilities:
- A. Who approves the test plans
 - B. Who approves the test schedule
 - C. Who resolves ambiguities within the requirements
 - D. Who resolves discrepancies between the project and enterprise level requirements
 - E. Who is responsible for requesting the test environment
 - F. Who provides application training for test group
 - G. Who provides necessary training for test resources
 - H. Who is responsible for generating the Software test report
 - I. Who approves the application for Enterprise Acceptance level testing
- n. Notify each participant of their assigned role
- o. Establish Points of Contact(POC) for external organizations
- p. Identify test environment (Section 3)
- 1) Identify all Software items needed for the test (section 3.x.1)
 - A. OS
 - B. Database
 - C. Communication software
 - D. Compilers
 - E. Supporting software
 - 2) Identify all Hardware items needed for the test (section 3.x.2)
 - A. Client hardware / server hardware / web hardware
 - B. List type of equipment and version
 - 3) Identify printed material or processing information that must be provided to or received from participants of the test (section 3.x.3)

Enterprise Integration Testing

- 4) Identify licenses or special permission required for software used in the test(section 3.x.4)
 - 5) Identify the test sites (Section 3.x)
 - A. How many test environments
 - B. Where will the test environment(s) reside
 - C. How long is the environment(s) needed
 - D. What size is the environment(s)
 - E. Where do I get the data for the environment(s)
 - F. What security access is needed to get to the data or to perform the test
 - G. What tools are needed on the test environment to perform my test
 - H. Do I need special permission to use tool in the test environment
 - 6) Identify which organization provides the test environment (section 3.x.5)
 - 7) Identify when test environment will be created and tested (Section 3.x.5)
 - 8) Identify controls for maintaining test environment (Section 3.x.5)
- q. Identify test product ([detail](#)) storage
- 1) Identify each of the following:
 - A. Location of storage
 - B. Size required
 - C. Required retention timeframe / archive requirements
 - D. Access required for archived products
 - E. Standards for storage (naming conventions, locations, versions)
 - F. Standards for use and reuse
- r. Identify procedures for resolving problems / discrepancies (section 4.1.5)
- 1) Identify information required to use CMIS for Problem reporting
 - A. Location / address of CMIS database
 - B. Release in CMIS
 - C. Security access – type of CMIS user for each tester and participant in the test (i.e. Technical Action Manager, Employee, etc.)
 - D. Criteria used to assign a priority / criticality to a TDR ([detail](#))
 - E. Testing structure to use within CMIS (system test events, system test levels)
 - 2) Review sequence of events required for a TDR to be resolved
 - A. Sequence if everyone agrees TDR should be fixed
 - B. Sequence if developer does not agree with tester that TDR should be fixed
 - C. Sequence if TDR is to be converted to SCR
 - D. Sequence if TDR is to be canceled
- s. Identify Reporting Criteria
- 1) Report standards to be used
 - 2) Frequency required for generating the report
 - 3) Routing of the report
- t. Define Test Results procedures (section 4.1.5)
- 1) How results are documented
 - 2) Who is notified when test is complete
 - 3) Who insures all Enterprise integration tests have been validated
- u. Identify any externally mandated test standards
- 1) Test plan format standard
 - 2) Test script format standard
 - 3) Test reporting format standard
 - 4) Required reviews to be performed
- v. Document strategy in an Enterprise Integration Software Test plan
- 1) Distribute Plan to the DCII Program Manager for approval
 - 2) Modify Plan as necessary

Enterprise Integration Testing

w. Build structure to capture effort metrics

- 1) Identify testing tasks in LRS
 - A. Strategy
 - B. Analysis
 - C. Design
 - D. Construct
 - E. Execute

5. Verification:

- a. Approved initial Software Test Plan
- b. Completion of required Enterprise Integration Test reviews

6. Exit Criteria:

- a. Initial Software Test Plan ([format](#))(Sections 1.3, 1.4, 2.0, 3.0 , 4.1.1, 4.1.5)

7. Measures:

- a. Data collected to determine size of the release to be tested:
 - 1) Using CMIS / Requirements traceability matrix:
 - 2) Number of testable requirements identified in the release by category
 - 3) Requirement for a batch update process
 - 4) Requirement for a batch query process
 - 5) Requirement for a batch report process
 - 6) Requirement for an on-line update process
 - 7) Requirement for an on-line query process
 - 8) Requirement for an on-line report process
- b. Data collected to determine the effort required to test the release:
 - 1) Using LRS:
 - A. Estimated / actual hours to perform test strategy

Enterprise Integration Testing

Task Name: Analyze test

Task Number: T-ST-###

Component: Enterprise integration Test

Category: Software Engineering

1. **Task Name:** Analysis for Enterprise Integration Testing

2. **Purpose:**

The purpose of this task is to specify Test cases / conditions for each feature to be tested. The specified test cases / conditions provide the information required to build a test script and supporting transactions that insure the integration of the DCII components operates accurately. A result of the analysis task is an estimate of the time required to perform the test and the areas that cannot be tested. The results of this task are documented in sections 4.2, 4.2.x, 5.0, and 6.0 of the Software Test Plan.

3. **Roles:**

The team assigned to perform the Enterprise Integration Test

4. **Entrance Criteria:**

- a. Designed functional entity / application design for each project
 - 1) Depiction of system module hierarchy & interactions
 - 2) Screen formats & window navigation design
 - 3) Report design specifications
 - 4) Input / output design specifications
 - 5) Physical database model (database design)
 - 6) Table design
 - 7) Integrity constraints
- b. System Interface specification / Memorandum of Agreement
- c. Data Conversion specification
- d. Project level test plans
- e. Project level test scripts and data
- f. Stubs and drivers created at the project level

5. **Procedures:**

The following procedures are performed for each source of requirement or specification to be validated identified in the strategy task.

- a. Define test cases:
 - 1) Define the business or technical operational events that occur at the Enterprise level
 - A. Review the Enterprise business processes
 - B. Review DCII technical specifications
 - C. Identify business / technical processes that is not contained wholly within an application
 - D. Identify business / technical processes that span multiple applications
 - E. Identify common business / technical processes
 - 2) Divide the operational events into testable sub-events that can be assigned to resources
 - 3) Identify the functionality for DAPIs or Interoperability services requested by the applications that exist in each operational sub-events
 - A. Obtain documentation from the projects that specifies the functionality they require to be implemented in the release.
 - B. Identify the solution provided to the projects (i.e. common DAPIs the projects will use, project specific DAPIs created for the release.)
 - C. Identify the functionality to be provided by each service or DAPI scheduled for the release

Enterprise Integration Testing

- 4) Identify the critical business requirements identified by each application's test team associated with the testable sub-events
 - 5) Define the variations of test cases to be validated for each sub-event
 - 6) Identify compliance criteria to be validated and method used to validate
 - A. Review Appendix D of DCII specification
 - B. Identify requirements that have not been validated by the application
 - C. Identify which of the remaining requirements must be validated by the Enterprise Integration test
 - 7) Define test cases that will not be validated
 - 8) Identify Enterprise level documentation to be validated
- b. Document test cases: (sub-events) within the repository
- 1) Define the execution sequence for the test cases
 - 2) Identify any dependencies that may need to be resolved in order to execute test cases
 - 3) Document dependencies
 - 4) Identify naming conventions ([detail](#)) to be used for test script development
 - 5) Identify the application requesting the DAPIs created for this release, if applicable
 - 6) Identify application level test cases that match ones to be executed for Enterprise level testing
- c. Build traceability matrix ([detail](#)) linking the requirements to the test cases. Ensure all Enterprise level requirements to be validated are included within the matrix, and at minimum cross-walked to one test. If more than one test case is applicable to one requirement, crosswalk all test cases that apply. If more than one requirement is applicable to one test case, crosswalk all requirements that apply.
- d. Identify Test data needed to support test cases
- 1) Analyze the test cases to be validated
 - 2) Determine data needs for each test case
 - 3) Review available data
 - 4) Review existing application test scripts and data to be used in the Enterprise Level test
 - 5) Determine type of data that must be created versus what is available
- e. Evaluate how software will be released to testing. Coordinate with configuration management to determine the method that the software will be released. If the software components are to be released in increments, define the test cases and requirements that will be validated in each increment.
- f. Determine the work breakdown structure / tasking by resource
- 1) Identify the test cases to be assigned to each test resource
 - 2) Identify the test data to be built by each test resource to support the test cases
 - 3) Identify any additional responsibilities that will be assigned to each test resource
 - 4) Document tasks within a task form ([detail](#)) and forward to each test resource
 - 5) Confirm tasking with each test resource
- g. Estimate test duration for the component using the following factors:
- 1) Number of test cases to be developed by the number of test resources available
 - 2) Amount of test data to be built
 - 3) Determine the complexity of the test ([detail](#))
 - A. Number of critical business functions to be validated
 - B. Number of system interfaces to be validated
 - C. Number of DAPIs to be validated
 - D. Number of Interoperability Services to be validate

Enterprise Integration Testing

- 4) Review Metrics collected from previous test cycles and allow time for:
 - A. Time delays waiting for discrepancies to be fixed
 - Compare TDR initiate date to resolution date
 - B. Delays caused by changes to the functionality
 - Review number of SCRs or PTRs recorded and scheduled for the same release
 - Review time required to impact the changes
 - Review time required to build additional test scripts / data for the changes
 - C. Problems in the test environment, managing the test environment and data bases and troubleshooting automated test tools
 - Determine amount of time required to set up test environment
 - Determine amount of time required to re-set test environment between test cycles
 - Determine the amount of time required to perform tasks that were not in the plan that relate to changes to the test environment
 - Determine the amount of time required to perform tasks that were not in the plan that relate to test environment troubleshooting
 - D. Re-testing after fixes have been made
 - Determine number of TDRs fixed vs. canceled or deferred
 - Determine number TDRs recorded as “not fixed”
 - E. Time required for test support activities like version control
 - Determine the amount of time required to receive application from release management
 - F. Delays caused by troubleshooting application
 - Determine the amount of time required to perform tasks that were not in the plan that relate to Defect tracking or troubleshooting application
 - G. Effort to report and explain the testing status
 - Determine actual time required to report status
 - H. Documentation of test results
 - Determine actual time required to report test results
 - I. Time required to travel, attend meetings and coordinate activities
 - Determine actual time required to attend meetings or coordinate software testing activities
 - Determine the anticipated number and duration of trips required compare to actual number of trips and duration from previous test cycle
 - J. Determine number of tasks identified in the plan that were not performed
- i. Determine the risks and contingencies of performing the level of testing identified in the plan.
- j. Complete Software Test Plan (STP) ([format](#)) Format, validate information, document in the appropriate sections of the Software Test Plan and forward to Test Director and Project Management for review and approval. Adjust plan as necessary.
- k. Build test schedule in MS Project and provide to Test Director and Project Management

6. Verification:

- a. Review and approval of final Software Test Plan

7. Exit Criteria:

- a. Documented test cases
- b. Approved Enterprise Integration Software Test Plan

8. Measures:

- a. Data collected to determine size of the release to be tested:
 - 1) Number of scripts created for each testable requirement

- b. Data collected to determine the effort required to test the release:
 - 1) Using LRS:
 - A. Estimated / actual hours to perform test analysis

- c. Data collected to determine the schedule for the release:
 - 1) Using MsProject:
 - A. Estimated hours to create or update scripts
 - B. Estimated hours required to receive release
 - C. Planned start date
 - D. Planned completion date

Enterprise Integration Testing

Task Name: Design Enterprise Integration Test
Component: Enterprise Integration Test

Task Number: T-ST-###
Category: Software Engineering

1. Task Name: Design Enterprise Integration Test

2. Purpose:

The purpose of Enterprise Integration test design is to identify types of test transactions to be created for each test case; identify/define test data used to validate test cases and conditions; identify pre-setting of the test environment and define the sequence of events that must occur before, during and after testing.

3. Roles:

The team assigned to perform the Enterprise Integration test

4. Entrance Criteria:

- a. Enterprise Business Function Model
- b. Enterprise Business Data Model
- c. Enterprise Business Process Model
- d. Module Functional Documentation
- e. Module Technical Documentation
- f. Documented Test Cases / Conditions
- g. Access to test environment
- h. Software Test Plan

5. Procedures:

- a. Insure that all application level testing has been completed
- b. Identify the different types of test transactions required for each test case
 - 1) Review the test cases documented for the Enterprise Integration test
 - 2) Review Business data Model, Enterprise Business Function Model, Enterprise Business Process Model, Module Functional Documentation, and Module Technical Documentation to determine:
 - A. Events that occur and their characteristics
 - B. Processes that respond to or are triggered by each event
 - C. Steps that occur in each process
 - D. Processes that are new business functions (do not exist today)
 - E. Elementary functions within each business process
 - F. Relationship of processes / elementary functions to each other
 - G. Data used by the processes / elementary functions
 - H. Data types
 - I. Data attributes
 - J. Mandatory data (not null)
 - K. Optional data
 - L. Allowable values
 - M. Behavior of the process / elementary function
 - N. Database tables and columns used by the process / elementary function
 - O. User types that have access to the data
 - P. Type of access does each user type have

Enterprise Integration Testing

- c. Identify the sequence of execution that must occur within each event of the application
 - 1) Review / create a system test process and data flow of the testable events. This flow is used to identify the dependencies within the system, sequence of testing, and types of tests to be performed.
 - 2) Build a diagram or matrix that identifies the following items:
 - A. The logical order of execution of the processes / elementary functions
 - B. Roles/accesses of users that send and receive data for the processes / elementary functions
 - C. Data that is sent to and received from the process / elementary function
 - D. Dependencies between one process / elementary function and another
 - E. Concurrent processing of processes / elementary functions
 - 3) Determine the sequence of test cases to be validated. Considerations for ordering the test cases:
 - A. Normal processing sequence driven by business process dependencies
 - B. Sequence that will require the least amount of test data creation and / or manipulation when no dependency exists
 - C. Sequence to make it easier to validate test results when no dependency exists
- d. Define data for test database and non-database files
 - 1) Review the test data available to perform the test
 - 2) Build or locate the test data required to validate the test case / condition
 - A. Valid transaction data
 - B. Invalid transaction data
 - C. System interface data / in interface file format
 - D. Input test data files
 - 3) Obtain table data from interfacing software/systems, if applicable
 - 4) Build spreadsheets/tables with test data to be entered through the GUI during test
- e. Design a method that user documentation will be validated such as online Help text, user manuals, and software training material. Document within test cases.
- f. Conduct a peer review of the test design
- g. Conduct a test design review with functional and technical experts for approval
 - 1) Adjust test design as necessary

6. Verification:

- a. Review and approval of test design

7. Exit Criteria:

- a. Test design (execution order, test cases / scenarios)
- b. Test data required to validate test cases

8. Measures:

- a. Data collected to determine the effort required to test the release:
 - 1) Using LRS:
 - A. Estimated / actual hours to perform test design

Enterprise Integration Testing

Task Name: Construct Enterprise Integration Test
Component: Enterprise Integration Test

Task Number: T-ST-###
Category: Software Engineering

1. **Task Name:** Construct Enterprise Integration Test

2. **Purpose:**

The purpose of Enterprise Integration Test construct task is to build the test scripts used during the execution.

3. **Roles:**

The team assigned to perform the Enterprise Integration test.

4. **Entrance Criteria:**

- a. Documented test design
- b. Test data
- c. Access to test environment

5. **Procedures:**

- a. Build manual Test scripts. The script should include: ([format](#))
 - 1) Test case to be validated
 - 2) Data to be used in the validation
 - 3) Pre-setting or staging condition that must happen prior to the validation
 - 4) Steps required to run the test
 - 5) Method used to review results and verify what is expected
 - 6) Expected results
 - 7) Reference to requirement number being validated
- b. Build Manual test scenarios (master test script)
 - 1) Identify sequence of scripts
 - 2) Identify testing that can be executed concurrently
- c. Automate scripts (where possible)
- d. Automate scenarios to execute the same sequence as manual scripts
- e. Update requirement traceability matrix (document the following crosswalks)
 - 1) Functional/technical requirements to test scripts
 - 2) Test scripts to compliance requirements
- f. Build test environment
 - 1) Setup hardware
 - 2) Build database instance (DBA function)
 - 3) Populate data base tables with baseline data
 - 4) Build / acquire interface data files
 - 5) Establish security access
 - 6) Install application under test
 - 7) Build software support (stubs/ drivers)
 - 8) Load input test data interfaces and validate load
 - 9) Load actual input interfaces to be used for test
 - 10) Backup / export baseline test data

Enterprise Integration Testing

- g. Perform final test peer review
 - 1) Verify environment, test scripts, and test data are ready for commencement of test
 - 2) Review order of execution of test cases and simultaneous testing
 - 3) Adjust for test as necessary within the project timeline

6. Verification:

- a. Peer review of work products

7. Exit Criteria:

- a. Manual test scripts / scenarios
- b. Automated test scripts / scenarios
- c. Input interface test files
- d. Test environment
- e. Supporting software
- f. Updated Requirements traceability matrix
- g. Populated baseline test database

8. Measures:

- a. Data collected to determine the effort required to test the release using LRS:
 - 1) Estimated / actual hours to perform test construct
- b. Data collected to determine the schedule for the release using MSProject:
 - 1) Actual hours to create or update scripts
 - 2) Actual hours to build or obtain test data

Enterprise Integration Testing

Task Name: Execute Enterprise Integration Test
Component: Enterprise Integration Test

Task Number: T-ST-###
Category: Software Engineering

1. **Task Name:** Execute Enterprise Integration Test

2. **Purpose:**

The purpose of this task is to execute test scripts that validate the test cases / conditions defined in the analysis task, ordered in the design and built in the construct task.

3. **Roles:**

The team assigned to perform the Enterprise Integration test and all designated participants.

4. **Entrance Criteria:**

- a. Manual test scripts / scenarios
- b. Automated test scripts / scenarios
- c. Access to test database
- d. Access to test environment

5. **Procedures:**

- a. Document start date in CMIS and the Test scenario (Master script / detail script)
- b. Turn on Operational monitoring tools if coordinated with technical personnel
- c. Execute test scripts
 - 1) Record the start date / time for the test script
 - 2) Execute the steps within the test script
 - 3) Compare the actual results with the expected results and note any discrepancies
 - 4) During execution, capture proof of validation for compliance criteria such as GUI screen captures, database table updates, audit records
 - 5) Record completion date / time for the test script
 - 6) Document the results of the test script
 - 7) Modify / correct test script, if necessary
 - 8) Run all steps listed on master test script in order specified
- d. Generate Test Discrepancy Reports (TDRs) ([detail](#)) for any noted discrepancies from the test results
 - 1) Record TDRs in CMIS according to TDR Procedures ([web site](#))
 - 2) Document the sequence of events within the TDR
 - 3) Document the difference between the expected results and the actual test results
 - 4) Document the suspected problem
 - 5) Document any test data that was used within the test
 - 6) Document the test script identification number that the discrepancy was found
 - 7) Assign a priority level to the TDR
 - 8) Save and route TDR according to discrepancy reporting procedures
- e. Cross-walk TDR to test scripts log ([format](#))
- f. Document test complete date in CMIS and the Test scenario (Master script / detail script)

Enterprise Integration Testing

- g. Build a “Quick Look” test summary report ([format](#))
 - 1) Document work-arounds, discrepancies found, fixes, and risks
 - 2) Route to the Test Director and Project Management
 - 3) Track any open items for Project Management for decision of fixes for next release
- h. Re-test as necessary
 - 1) Reset test environment
 - 2) Coordinate with the Configuration Management Group for a new release
 - 3) Install new release
 - 4) Repeat test cycle as specified in the test plan
 - 5) Validate TDRs fixes
 - 6) Track TDRs to resolution
 - A. All TDRs should be resolved at the end of the test event (cancel, defer or fix)
- i. Document Software Test Report (STR) ([format](#))
 - 1) Document work-arounds, discrepancies found, fixes, and risks
 - 2) Include evaluation of software application quality within report
 - 3) Route to the Test Director and Project Management
 - 4) Track that any remaining TDRs are converted to SCRs or cancelled
- j. Clean up test products ([detail](#))
 - 1) Correct / update test scripts as needed
 - 2) Correct / update test data as needed
 - 3) Automate regression test scripts / scenarios. (this is a long term effort that does not have to be completed before the Enterprise Integration Test is considered complete)
 - 4) Review metrics to insure they are complete and accurate
- k. Place test products ([detail](#)) under Configuration Management control (this is a long term effort that does not have to be completed before the Enterprise Integration Test is considered complete)

6. Verification:

- a. Updated test script documentation with test completion date

7. Exit Criteria:

- a. Software Test Report
- b. Updated test scripts
- c. Automated regression test

8. Measures:

- a. Data collected to determine the schedule for the release using MSProject:
 - 1) Actual hours required to receive release
 - A. Planned / Actual start date (difference between planned and actual start date)
 - B. Planned / Actual completion date (difference between planned and actual completion date)
- b. Data collected to specify actual schedule for the release using the Software Test Report (STR):
 - 1) Estimated / actual number of test scripts and data to be created or updated
 - 2) Estimated / actual manpower/resources required to perform test
 - 3) Planned / Actual Tasks performed(number of test cases executed, number not executed)
 - 4) Tasks performed that were identified in the plan
 - 5) Tasks performed that were not identified in the plan
 - 6) Number of changes made to the test environment and number of days required to make each change

Enterprise Integration Testing

- 7) Number of days required to troubleshoot the test environment
- 8) Number of days spent defect tracking or troubleshooting the application
- 9) Number of out-of-town trips and their duration
- 10) Number and duration of meetings required during the test
- c. Data collected to determine quality of testing release:
 - 1) Using CMIS, number of TDRs written in next level of testing (Enterprise Acceptance Testing)
- d. Data collected to determine the effort required to test the release using LRS:
 - 1) Estimated / actual hours to perform test execute