

Unit / Module Test

Component: Unit / Module Test

Component Description:

A unit is the smallest testable piece of software, which can be compiled or assembled, linked, loaded, and put under the control of a test harness or driver. A unit is defined as a database trigger, stored procedure, function, report, form, batch load program or PL/SQL program. The documented specification of a unit is comprised of an input definition, processing definition, outcome definition, data base definition, and interface specification.

Unit testing is the testing that is performed to validate that the unit does satisfy its functional specification and/or that its implementation structure does match the intended design structure.

Module testing consists of units integrated into a module that performs a specific business function.

Inputs:

- 1) Application Business Function Model (RD.030)
 - a) Function Hierarchy Diagram
 - b) Function Data Usage
 1. CRUD matrix
 2. Data flow diagram
- 2) Application 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) Relationships – the way entities are related or associated to one another
 - e) Unique identifiers – the combination of attributes and relationships that uniquely identify each instance of the entities
- 3) Module Functional documentation (MD.040)
 - a) Functional logic of the module
 - b) Layouts
 - c) Constraints
 - d) Deviations of the standards for user interface
- 4) Module Technical documentation
 - a) Module logic
 - b) Detailed table and column usage
 - c) Module parameters (in-going and out-going)
 - d) Supporting modules
- 5) User Interface Style definition (TA.100)
 - a) Design conventions
 - b) Screen navigation
 - c) Standard report layouts
- 6) Program specification
- 7) Software Development Plan

Outputs:

- 1) Documented unit test procedures – updated unit test checklist
- 2) Test data
- 3) Test cases
- 4) Database audit report
- 5) Stubs / drivers

Unit / Module Test

Task Ordering:

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

Project Management Tasks:

- 1) Enter Task actual start / end date
- 2) Verify Unit testing time identified in SDP

Component Tasks:

Configuration Management Tasks:

- 1) Ensure unit testing environment is established
- 2) Verify Configuration Items are correct for migration to SIT

Component Metrics:

- 1) Data collected for each unit test:
 - a) Actual start date (note difference between planned start date and actual start date)
 - b) Actual completion date (note difference between planned completion date and actual completion date)
 - c) Test effort (manpower/resources)
 - d) Tasks performed (number of test cases executed, number not executed)
- 2) Data collected for each Configuration Item (CI):
 - a) Size of change
 - b) Number of new test cases required
 - c) Number of test cases to be modified
 - d) Test data impacted / built
 - e) Resource impact
 - f) Schedule impact

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

Unit / Module Test

Task Name: Develop Unit Test Strategy
Component: Unit / Module Test

Task Number: T-UT-001
Category: Software Engineering

1. **Task Name:** Develop Unit Test Strategy

2. **Purpose:**

The purpose of the task is to define the plan of action, resources, personnel, approach, and test methods to test individual units of software, and integrated units, referred to as modules.

3. **Roles:**

Software Engineers perform unit and module testing.

4. **Entrance Criteria:**

- a. Application Business Function Model (RD.030)
- b. Application Business Data Model (RD.040)
- c. Module Functional documentation (MD.040)
- d. Module Technical documentation
- e. User Interface Style definition (TA.100)
- f. Program specification
- g. Software Development Plan

5. **Procedures:**

- a. Review the Software Development Plan for project schedules
- b. Review the Program specification for each unit / module
- c. Define the test approach to be used ([detail](#))
 - 1) Top down
 - 2) Bottom up
 - 3) Isolation
- d. Define the depth of test coverage for unit level testing
 - 1) Determine which of the following types of unit testing ([detail](#)) will be performed, and the level of coverage of each type based upon the criticality of the requirements
 - A. Control flow coverage
 - B. Data integrity testing
 - C. Boundary value checking
 - D. Specific algorithm validation
 - E. Data definition use testing
 - 2) Validation of date algorithms handling Year 2000 dates
 - 3) Identify the 'always performed' tests specified as standard for unit testing
 - A. Maximum allowed field
 - B. Minimum allowed field
 - C. Data edits
 - D. Blank fields
 - E. Exit before normal processing complete
 - F. Open multiple windows simultaneously for user interfaces

Unit / Module Test

- e. Define the depth of test coverage for module level testing
 - 1) Determine the techniques to be used for module level testing based upon the criticality of the requirements for the module business function
 - A. Path coverage
 - B. Standards compliance
 - C. Integration of the units
 - 2) Identify to what extent private testing will be accepted as formal unit testing, and to what extent will public testing be required
 - 3) Confirm resources to perform unit / module level testing - resources identified may be programmers and/or testers
 - 4) Identify roles and responsibilities for unit level testing to identify:
 - A. Test environment
 - Specific environment requirements / separate data base schemas for each programmer
 - Who provides database support
 - Who builds the stubs / drivers for unit testing
 - B. Test management
 - Who defines when unit testing has been satisfied
 - Who is responsible for coordinating all unit level test events
 - Who controls the integration of units into modules for module level testing
 - Where are unit test cases documented
 - What method will unit test results be documented
 - C. Test setup and execution
 - Who creates the unit test cases
 - Who executes the unit test
 - Who is to provide test data for unit level testing
 - D. Problem resolution
 - How will discrepancies be documented and/or fixed
 - Who is responsible for resolving any discrepancies between the program code and the program specification
- f. Identify test environment requirements for unit level testing
 - 1) Separate data base schemas for each programmer / tester
 - 2) Unit / module testing to be performed in development environment or test environment
 - 3) Database administration of environment
- g. Identify the reporting criteria
 - 1) Identify Unit Test Checklist to be used
 - 2) Identify CMIS unit test task procedures
 - 3) Identify discrepancy reporting criteria
- h. Test results
 - 1) How test results are documented
 - 2) Who is notified when the test is complete
 - 3) Who insures that all testing has been performed
- i. Update / build unit test checklist

6. Verification:

- a. Program management review

7. Exit Criteria:

- a. Documented unit test procedures – updated unit test checklist

Unit / Module Test

8. Measures:

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

Unit / Module Test

Task Name: Analyze Unit Test Requirements
Component: Unit / Module Test

Task Number: T-UT-002
Category: Software Engineering

1. Task Name: Analyze Unit Test Requirements

2. Purpose:

The purpose of this task is to specify test cases / test data for the software unit to be tested.

3. Roles:

Software Engineers perform unit or module testing.

4. Entrance Criteria:

- a. Computer software units
- b. Unit test database
- c. Unit Test Checklist Standard ([S-SE-009](#))
- d. Test Script Standard ([S-SE-010](#))

5. Procedures:

- a. Define test case
 - 1) Analyze the specification
 - 2) Identify business or technical operational events
 - 3) Identify compliance criteria tests
 - 4) Identify documentation to be validated
- b. Identify test data
 - 1) Analyze test case
 - 2) Determine data needs

6. Verification:

- a. Program management review
- b. SQA Audit of Product

7. Exit Criteria:

- a. Documented test cases
- b. Approved unit Test Plan

8. Measures:

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

Unit / Module Test

Task Name: Design Unit Tests
Component: Unit / Module Test

Task Number: T-UT-003
Category: Software Engineering

1. Task Name: Design Unit Tests

2. Purpose:

To identify / define test data used to validate test cases / conditions.

3. Roles:

Software Engineers perform design of unit or module tests.

4. Entrance Criteria:

- a. Documented test plan
- b. Unit test criteria

5. Procedures:

- a. Review / create data flow diagram showing origin, destination, and relationships of data within the unit / module to be tested
- b. Identify the different types of test transactions for each test case
 - 1) Positive / negative
 - 2) Null / not null
 - 3) Allowable values (enforced by foreign keys, domains, list of values) & (enforced by application code)
 - 4) Boundaries
 - 5) Method of data selection (pick list behavior)
 - 6) Functional variations
 - 7) Data maintenance
- c. Define data for test database / test cases ([detail](#))
 - 1) Data in Oracle table format
 - 2) Data in GUI screen format
 - 3) Data in file format
- d. Identify the different types of test transactions required for each test case
 - 1) Valid transaction data
 - 2) Invalid transaction data
- e. Conduct peer review of test design to include:
 - 1) Test cases
 - 2) Test data
 - 3) Edits

6. Verification:

- a. Program management review
- b. SQA audit of product
 - 1) Verify that design is complete
 - 2) Verify that peer review satisfactorily completed and corrective actions have been completed

Unit / Module Test

7. Exit Criteria:

- a. Updated data flow diagram
- b. Test data required to validate test cases

8. Measures:

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

Unit / Module Test

Task Name: Construct Unit Tests
Component: Unit / Module Test

Task Number: T-UT-004
Category: Software Engineering

1. **Task Name:** Construct Unit Tests

2. **Purpose:**

The purpose of this task is to build necessary test data, and establish the test conditions to test a software unit.

3. **Roles:**

Software Engineers perform unit or module testing.

4. **Entrance Criteria:**

- a. Updated data flow diagram
- b. Test data required to validate test cases
- c. Unit test criteria

5. **Procedures:**

- a. Update unit test cases with data types, sample data, and transaction variations required to validate the case
- b. Build test environment
 - 1) Software support (stubs / drivers)
 - 2) Populate database tables with baseline data
 - 3) Build / acquire test data files

6. **Verification:**

- a. Program management review

7. **Exit Criteria:**

- a. Updated test environment
 - 1) Test data
 - 2) Software support (stubs / drivers)
- b. Updated test cases

8. **Measures:**

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

Unit / Module Test

Task Name: Execute Unit Tests
Component: Unit Test

Task Number: T-UT-005
Category: Software Engineering

1. Task Name: Execute Unit Tests

2. Purpose:

The purpose of this task is to execute test transactions to validate that the software unit meets the requirement specifications.

3. Roles:

Software Engineers perform unit or module testing.

4. Entrance Criteria:

- a. Database audit report procedures
- b. Unit Test Checklist Standard
- c. Discrepancy reporting/resolving procedures

5. Procedures:

- a. Record Start date (CMIS)
- b. Turn on database audit tool (coordinate with technical personnel)
- c. Execute test cases (modify cases and data as required)
- d. Compare test results with expected results
 - 1) Compare database audit report to expected database updates
 - 2) Compare actual screen response to expected screen response listed in test case
 - 3) Compare errors received to errors expected
 - 4) Review external files for expected result (if created)
 - 5) Verify output products
 - 6) Verify accuracy of calculations and edits
- e. Capture Proof of Validation for compliancy criteria
- f. Resolve discrepancies and repeat as necessary
- g. Complete Unit Test Checklist
- h. Record test complete date (CMIS)
- i. Place test products ([detail](#)) under Configuration Management Control
- j. Provide process improvement to process owner

6. Verification:

- a. Program management Review
- b. SQA Audit of Product (test cases and results)

Unit / Module Test

7. Exit Criteria:

- a. Tested unit
- b. Completed Unit Test Checklist
- c. Updated test data
- d. Updated test cases
- e. Test results

8. Measures:

- a. Data collected to determine the effort required to test the release using LRS:
 - 1) Estimated / actual hours to perform test execute
- b. Actual start date
- c. Actual end date
- d. Number of units tested