TEST PLAN GUIDELINES

1. GENERAL

1.01 This appendix provides guidelines for the content and format of system test plans. These guidelines are applicable to any test plan required by the tester's organization.

1.02 Whenever this appendix is reissued, the reason for reissue will be stated in this paragraph.

1.03 To include all items in each test plan, especially for maintenance and enhancement efforts on existing systems, could be counterproductive. However, exclusion of individual items should be carefully evaluated, based on the complexity of the system and the scope of the test effort as set by project management.

1.04 The following is a brief explanation of terms commonly used but which sometimes lead to confusion.

- ***Test Plan:*** An overall description of the level and type of testing effort to be performed. It establishes the testing strategy for the product. The test plan may or may not contain documentation of the test cases.

- ***Test Case:*** A distinct test unit with the objective of exercising a specific attribute or function and for which instructions/procedures are identified along with the input data requirements and the expected test results. (In the documentation which follows, test and test case are used interchangeably.)

- ***Test Instructions:*** The description of the operational procedures for running a test case.

2. TEST PLAN GUIDELINES

2.01 The test plan should include the following sections:

1. Introduction

2. Objectives

3. Components to be Tested

4. Test Environment

   —Test Site and Location
   —Hardware
   —Software
   —Data Base

5. Test Cases

   —Test Methodology
   —Test Case Description
   - Name and/or Number
   - Item Being Tested
   - Test Objective
   - Test Method (walk-through, simulation)
   - Input
   - Instructions for Performing Test Case
   - Alternate Test Methods
   - Conditions Upon Which Test is Dependent

   —Test Analysis Procedure
   —Expected Results
   —Resources Needed

6. Administrative Procedures

   —Responsibility of Testers
   —Coordination Between Developers and Testers
   —Status Control

7. Schedules

1. Introduction

This section should provide a description of the system being tested. That is, it identifies the overall system in general and describes the particular development effort under way. It further identifies the portions or aspects of that development effort to be tested.

2. Objectives

A test plan reflects the testing strategy for the system. Thus, the objectives are specified in terms of:

- —The level of testing of this test plan (Unit, Integration, System) or the type of environment (Computer Subsystem/Personnel Subsystem [CSS/PSS] Verification, Validation, Certification).
- —General objectives of exercising features/capabilities. Identify the specific developmental documentation reference (eg, System Output Specification).
- —Any dependencies or assumptions that affect the test effort or the test case in general (eg, existence of a usable test data base, relationship to other test plans, or successful completion of other test efforts).
- —Acceptance criteria for successful completion of the test effort (eg, all or selected tests successful, all major problems resolved, acceptable level of known errors remaining in system under test).

3. Components to be Tested

This section should include a list of all products and procedures being tested. For an entire system, all components should be identified. For a revision or enhancement, identify all new or revised components.

4. Test Environment

Test Site and Location: Describe the test sites and their locations.

Hardware: Describe standard and special hardware facilities needed for tests. Relate this to hardware to be used by live system. Include CPU capabilities, terminals, and other peripherals. State physical location of live and test hardware environment.

Software: Describe standard and special software facilities needed for tests in terms of operating environment, communications networks, and their versions. Relate this to software to be used by live system, eg, testing may be batch, while actual system may be interactive. Detail what, if any, special test aids are to be used and who supports them.

Data Base: Describe use, creation, structure or DBMS tools, loading, backup and recovery procedures. Specify whether these are part of the test itself, who is responsible for maintenance of the test data base, and relation of test data base to live data base.

5. Test Cases

Test Methodology: The development of the test cases is the most encompassing effort associated with testing a new system/application. The purpose of developing test cases is to demonstrate that the system/application performs according to formal documentation, ie, Design Specifications, System Requirements, etc. Test cases should be designed to test for the negative as well as positive aspects. The purpose is to uncover errors so that they can be corrected prior to turnover of the system to the end user.

Test cases should be developed for every portion of the system, subsystem(s)/module(s). They should include all possible error conditions and different possible combinations of error conditions.

Test selection is based upon identifying measurable criteria and developing appropriate test cases. This selection requires careful review and planning on the part of the individual responsible for developing the test case. The system/application should be reviewed to determine how it can be broken down into manageable and logical testing components. In the event there is more than one individual responsible for testing, this becomes even more important in identifying the logical components of the system. A goal to keep in mind is to reduce duplicate testing effort as well as developing overlapping test cases.

The individual responsible for developing the test cases should be given all formal documentation needed to develop the test cases. A thorough review of the developmental documentation is necessary before writing test cases. However, the sequential development of test cases based on the developmental documentation may result in more test cases being developed than necessary. In many instances, test objectives can be combined, thereby reducing the overall number of test cases. Some guidelines follow.

- ***Separability:*** Try to develop initial test cases which are "freestanding," in which other tests are not linked together to form a dependency. Then use a building block approach where testing is done in levels of increasing complexity and thoroughness.

- ***Repeatability:*** Tests are to be developed so that they can be repeated if for no other reason than the test may fail the first time.

- ***Measurability:*** Determine and display as test results criteria that are quantifiable so that a PASS/FAIL decision can be made. Make reference to the developmental documentation describing the feature for which the test case is being developed.

- ***Finiteness:*** Any test that does not stop is really not a test.

- ***Simplicity of Operation:*** Tests should have minimal interactive inputs and should flag anticipated problems clearly.

- ***Hierarchical Structure:*** Set up tests hierarchically with:

  —Simple tests of independent features

  —Exhaustive tests of likely combinations, volume testing

  —Tests at or beyond limits of design to simulate possible live situations (diabolical testing)

  —Random attempts based on "hunches."

From these, select base-line tests to be rerun on every version together with those that detected problems in prior versions.

Test Case Description: The following are some guidelines to follow in formatting and developing the test cases.

- ***Name and/or Number:*** Each test case should be identified by either a unique name or number. A logical breaking point for a test is whenever the test instruction results in the expected test results. Try to minimize "run-on" testing where expected results of one test lead to another test.

- ***Item Being Tested:*** Identify what function or feature is being tested. The easiest way is to reference it to a design document and note it on the test case.

- ***Test Objective:*** A quantitive statement identifying the goal of the test case. It is a statement of what the tester(s) hope to accomplish through performance of the test. Remember to test both for positive and negative (what if) results.

- ***Test Method:*** Identify the type of test being performed (eg, simulation, walk-through, machine, etc).

- ***Input:*** Identify the input data needed to achieve the objective of the test case. Determine test data, data base(s), files, any special "conditioning" needed to perform this test. The location, description, dates of creation of test files, data bases, and libraries should be documented.

- ***Instructions for Performing Test Case:*** Instructions on how to perform the test should be fully documented for each test case. Procedures may be required to operate terminals, access other system, etc. If automated tools are required, make certain that instructions/procedures are identified and incorporated into test case. Identify what other groups are required to support the performance of this test case. Instructions must be explicit since the individual who develops the test may not be the same person who performs the test. Documenting the instructions normally requires the largest amount of effort in test case development.

- ***Alternate Test Methods:*** Where applicable, identify any alternate test methods or procedures. In some instances, the primary

method/procedure may have to be modified to achieve the same results.

- ***Conditions Upon Which Test is Dependent:*** Identify the dependencies and interrelationships of all activities and groups required to perform this test. These include:

  —Dependency: Identify any dependency between this case and any previous case or "setup" required. Identify, where applicable, the impact if the prerequisite test(s) are not in place.

  —Data Condition: Identify any test data or related criteria needed to perform the test. Determine if the same expected results will be attained. If not, identify other expected results.

  —Environment: Identify the environment in which the test is to be performed. Such questions as: Can this test be performed with other ongoing activities? How does this test differ from the actual operating environment? should be considered. Identify any PSS and CSS considerations. The environment should be controlled so that the results of the test can be accurately evaluated. A load or response time test may require a stand-alone test "window" for testing.

- ***Test Analysis Procedure:*** Determine who will be responsible for reviewing the test. Identify any aids or tools required to interpret results. Identify what data is to be collected and documented as a result of executing the test.

- ***Expected Results:*** The expected results must be clearly stated. A detailed explanation of the expected results and the criteria for a successful test must be documented. If a simple PASS/FAIL cannot be identified, the acceptable criteria for this test must be stated.

- ***Resources Needed:*** Identify all resources needed to perform this test. Some of the resources follow.

  —Programming Service

  —Computer Operation Personnel

  —Software

  —People

  —Test Time.

  Along with identifying the resources required, it is also the responsibility of the testing personnel to coordinate all of the resources.

6. Administrative Procedures

The test effort may be planned cooperatively by project management, developers, designers, users, and testers. It may be reviewed, implemented, and evaluated by personnel in the same or different roles. Organizational responsibilities will vary between projects. Thus, it is necessary to clarify the roles and responsibilities of all involved in the test effort, as well as to define the testers' interfaces with the developers and the users.

Responsibility of the Testers: The degree of responsibility and authority of the testers must be spelled out in terms of:

—How far to go in tracing problems

—Responsibility for resolving problems

—Who maintains problem log

—Coordination with related systems

—Authority to delay or stop release or recommend such.

Furthermore, who the testers are must be defined, eg, is a separate group or individual performing the tests, to whom do the testers report, do testers contact a developer directly, under what type of change management process do the testers function.

Coordination Between Developers and Testers: A major development effort tested over a long period of time requires that a number of versions be planned and scheduled. As each version is tested, problems are resolved and the next version is generated and tested. Also, products may be released for testing in phases as the development effort provides testable units. How and when the versions are to be supplied must be detailed. There must be a clear distinction between different versions in order to identify which tests were run on which versions, with what results.

Problems should be reported to the developers formally under the project's change management procedures. Specific developers may be designated for contact about specific components, or contact may be through some coordinator or through project management.

Status Control: Problem reporting involves a method of recording problems and reporting them to developers for resolution using the project's change management procedures. A formal log may also be kept for problems and may contain such data as problem number, description, date found, circumstances (environment and test used), component, version, and related problems.

Results of individual tests must be filed with potential problems indentified.

Significant terminal sessions or console logs, clearly labeled, should be retained during the test effort. Those necessary for open problems or for use as samples for regression testing should be turned over to personnel responsible for maintenance. It is also helpful to maintain handwritten notes on:

- —Intention at start of session
- —Observations during session (problems encountered, confirmed, resolution verified, hunches on potential problem areas)
- —Other problems encountered (hardware, test program limitations or errors)
- —Suggested changes to system under test with reason.

A summary of the pass or fail status of individual tests must be maintained with cross-reference to problem numbers. This should include test name or number, date run, component tested, version, pass/fail, and problem numbers.

Project management must be kept informed of the overall status of the test effort in terms of percentage of tests successfully run, percentage of tests yielding problems, and percentage of tests yet to run. The tester must warn both project management and the developer of anticipated slips in test schedule that could impact system release.

7. Schedules

The test effort must have formal schedules and resource requirements similar to the development effort. This includes:

- —Personnel Schedule which details activity, person assigned, and time period (may be in the form of a GANTT chart organized by activity or person or there may be separate schedules for both)
- —Equipment/Material Schedule in the form of a list of needs or similar to the personnel schedule
- —Dependencies Schedule in the form of a PERT chart or CPM, showing interrelationships of test activities.