U.S. ARMY TEST AND EVALUATION COMMAND
TEST OPERATIONS PROCEDURE

*Test Operations Procedure (TOP) 1-1-056 9 December 1997
AD No.

SOFTWARE PERFORMANCE TESTING

* This TOP supersedes TOP 1-1-056, 15 November 1977. Approved for public release; distribution unlimited.

1. SCOPE.

a. This TOP provides procedures for a requirements-oriented approach for the test and evaluation (T&E) of battlefield automated systems that rely on software for their functionality. It is a guidance document which provides detailed procedures for the planning and execution of software performance testing for all U.S. Army Test and Evaluation Command (TECOM) test centers. The procedures are applicable throughout the system life cycle. Specific application of these procedures should be tailored to ensure that only necessary and cost-effective requirements are applied to meet the individual test center's needs. The focus of this TOP is on software performance; i.e., performance of the software's functional capabilities. It does not address other software-related issues, such as safety, security, etc.

b. The methodology to implement the concept of software performance testing discussed herein addresses software as an integral element of system T&E and is targeted at the system performance level. Accordingly, the focus is on software requirements; i.e., functional capabilities which are allocated to and implemented in software. It should be noted that the focus is deliberately not towards a further debugging of the developer's computer program code. Key elements of this approach include the allocation of system requirements to software, assessment of software performance, and assessment of the impact of software on overall system performance.

c. In order to assess overall system performance and measure software maturity, results and progress of the software development process and testing are measured and analyzed using the 12 Army Software Metrics from Department of Army (DA) Pamphlet (PAM) 73-7. The metrics are divided into the three categories of Management Metrics, Requirements Metrics, and Quality Metrics. An overview of the metrics is provided below. Metrics may be developed, collected, and used by many organizations, i.e., developers, evaluators, software engineers, testers, independent verification and validation (IV&V), etc.

(1) Management Metrics - Measure program management resources. Deals with contracting, programmatic and overall management issues.

(a) COST - Tracks software expenditures ($ spent vs. $ allocated). [Limited to local level test expenditures. Higher level expenditures require Program Manager (PM) level authorization.]

(b) SCHEDULE - Tracks progress vs. schedule (event/deliverable progress). [Limited to test schedules.]

(c) COMPUTER RESOURCE UTILIZATION - Tracks planned vs. actual size (% resource capacity utilized).

(d) SOFTWARE ENGINEERING ENVIRONMENT - Rates developer'resources and software development process maturity). [IV&V function]

(2) Requirements Metrics - Measure the quality of software requirements definition (specification and translation) and the level of change to these requirements.

(a) REQUIREMENTS TRACEABILITY - Tracks requirements down to code and test cases (% requirements traced to design, code, and test). [The TOP is limited to requirements traced from system requirements to software requirements to test procedure and back in both directions. Tracing to the design and code is an IV&V function.]

(b) REQUIREMENTS STABILITY - Tracks changes to requirements (user/developer requirements changes and effects).

(3) Quality Metrics - Measure the degree to which the software possesses a desired combination of attributes. Addresses testing and other technical characteristics of software products.

(a) COMPLEXITY - Assesses code quality. [Requires the proper tools and access to the code]

(b) BREADTH OF TESTING - Tracks testing of requirements (% functions/requirements demonstrated).

(c) DEPTH OF TESTING - Tracks testing of code (degree of testing). [IV&V function]

(d) FAULT PROFILES - Tracks open vs. closed anomalies (total faults, total number of faults resolved, and the amount of time faults are open).

(e) RELIABILITY - Monitors potential downtime (software's contribution to mission failure).

(f) DESIGN STABILITY - Tracks design changes and effects (changes to design, % design completion.) [IV&V function]

d. Policy for the Army's T&E program is established in Department of Defense Directive (DoD) 5000.1 and DoD Regulation 5000.2 which became effective on March 15, 1996. This directive and regulation rank first and second in order of precedence for providing mandatory policies and procedures for the management of acquisition programs, except when overridden by statutory requirements. The DoD policy evolved from Secretary of Defense Perry's memorandums on June 29, 1994, and December 1995, which strongly encouraged the use of non-government standards and industry practices; and required a waiver for further use of Military Standards (MIL-STDs). Secretary of Defense Perry emphasizes that to compete in the global software market, one needs to play by the global rules. Global rules are often established by the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC). The rules were established with the August 1995 approval of ISO/IEC 12207, "Information Technology-Software Life Cycle Processes". ISO/IEC 12207 provides the core software process architecture. Each country has the option of adapting the standard for their unique requirements. The two standards developing organizations, the Electronic Industries Association (EIA) and Institute of Electrical and Electronics Engineers (IEEE) implemented the non-government version of MIL-STD-498, Joint Standard (J-STD)-016, in January 1996. The EIA/IEEE J-STD-016 is essentially identical to MIL-STD-498. The Joint Industry Working Group (JIWG) is in the process of adapting ISO/IEC 12207 for use in the United States (U. S.). The latest draft was created on September 1, 1996. Currently, the plan is for U.S. 12207 to be released as three volumes: Volume I represents a verbatim copy of ISO/IEC 12207 with extra annexes added to the back for the American implementation, Volume II is to contain all the JIWG's insertions and replacements from J-STD-016, and Volume III which will contain product descriptions or Data Item Descriptions (DIDs) from J-STD-016. This U.S. 12207 must be approved by the balloting members from both IEEE and EIA. If it passes the balloting process, it will be submitted to the American National Standards Institute (ANSI), and, if accepted, be called ANSI J-STD-016. The plan is for ANSI J-STD-016 to be in place by December 1996. Since MIL-STD-498 will be officially rescinded by January 1, 1997, the DoD will have two software standards that may be invoked: EIA/IEEE J-STD-016 or ANSI J-STD-016 (if available).

e. New policy requires new mission critical software to be acquired and supported in accordance with commercial software development standard, J-STD-016. The new commercial standard establishes uniform requirements for software development and documentation. The term "software development" is used as an inclusive term encompassing new development, modification, reuse, reengineering, maintenance, and all other processes or activities resulting in software products. The J-STD-016 is invoked by citing it on a contract. It applies to each software product and to each type of software covered by the contract. The J-STD-016 defines a set of activities and documentation suitable for the development of both weapon systems and automated information systems. Other changes include improved compatibility with incremental and evolutionary developmental models; improved compatibility with non-hierarchical design methods; improved compatibility with computer-aided software engineering (CASE) tools; alternatives to, and more flexibility in, preparing documents; clearer requirements for incorporating reuseable software; introduction of software management indicators; added emphasis on software supportability; and improved links to software engineering.

f. The procedures for software T&E, defined herein, are applicable to systems for which system and software requirements documentation has been developed in consonance with

J-STD-016. J-STD-016 establishes the requirements to be applied during the acquisition, development, and support of software systems. It can be applied in any phase of the life cycle. The standard can be applied to contractors, subcontractors, or Government in-house agencies performing software development. For uniformity, the term "acquirer" is an organization that procures software products for itself or another organization. The term “developer” is an organization that develops software products. “Develops” may include new development, modification, reuse, reengineering, maintenance, or any other activity that results in software products, and includes the testing, quality assurance, configuration management, and other activities applied to these products. "Contract" is the agreement between the acquirer and the developer. "Statement of Work" (SOW) is the list of tasks to be performed by the developer.

g. The developer shall establish a software development process consistent with contract requirements. The developer shall comply with the general requirements in section 4 and the detailed requirements in section 5 of J-STD-016. The software development process shall include the following major activities, which may overlap, may be applied iteratively, may be applied differently to different elements of software, and need not be performed in the order listed. Annex B of J-STD-016 provides some examples. The developer's software development process shall be described in the software development plan.

(1) Project planning and oversight (J-STD-016 , section 5.1)

(2) Establishing a software development environment (5.2)

(3) System requirements definition (5.3)

(4) System design (5.4)

(5) Software requirements definition (5.5)

(6) Software design (5.6)

(7) Software implementation and unit testing (5.7)

(8) Unit integration and testing (5.8)

(9) Software item qualification testing (5.9)

(10) Software/hardware integration and testing (5.10)

(11) System qualification testing (5.11)

(12) Preparing for software use (5.12)

(13) Preparing for software transition (5.13)

(14) Integral processes:

(a) Software configuration management (5.14)

(b) Software product evaluation (5.15)

(c) Software quality assurance (5.16)

(d) Corrective action (5.17)

(e) Joint technical and management reviews (5.18)

(f) Risk management (5.19)

(g) Software management indicators (5.20)

(h) Administrative security and privacy protection (5.21)

(i) Managing subcontractors (5.22)

(j) Interfacing with software IV&V agents (5.23)

(k) Coordinating with associate developers (5.24)

(l) Project process improvement (5.25)

h. Annex N of J-STD-016 describes a candidate set of joint management reviews that might be held during a software development project. Annex L of J-STD-016 identifies the software products that are to undergo software product evaluations, identifies the criteria to be used for each evaluation, and contains a default set of definitions for the evaluation criteria.

i. For systems where formal requirements documentation has not been developed (e.g., nondevelopmental item (NDI) systems or systems under accelerated development), the procedures may have to be tailored to derive requirements from appropriate sources. These sources may include the system specification, informal software specifications, or other applicable requirements documentation. General tailoring guidance is provided in DoD Handbook -248 (Ref 9). Tailoring specific to J-STD-016 is provided in Annexes B and C of the standard.

Back to top

2. FACILITIES AND INSTRUMENTATION.

2.1 Facilities.

Existing test facility types, to include those developed by the materiel developer, should be used when possible. Examples of facilities to be considered to support software T&E are described below.

Item	Requirement
Software Development Facility	To support development and testing of software code
System Laboratory Facility	To support integration and testing of software in a system configuration designed to use as much of the tactical hardware as possible
Field Test Facility	To test the software in a complete system, in a tactical/near tactical, real-time environment as possible

Back to top

2.2 Instrumentation.

Instrumentation or monitoring of a software/computer system is the primary means of collecting data about that system. Data collection to support software T&E must be treated as a requirement of the system, and, therefore, should be designed in, tested in, and delivered with the system. Unfortunately, all too common with software intensive systems is the absence of concern for the data necessary to support T&E. The result is either inadequate data collection capability to support required analyses, or expensive data collection retrofits. It is necessary for the T&E community to establish data collection requirements adequate to support not only the developer, but the developmental and operational testers and evaluators as well. Also, a single data reduction capability to support this collected data should be designed to meet the collective needs of the T&E community. This capability could be developed by the test agency, or even by the system development contractor, with proper validation by the T&E community.

a. Part V of the Test and Evaluation Master Plan (TEMP) identifies the specific test range/facilities to support program testing. It identifies instrumentation that must be developed and/or procured specifically to conduct the planned test program. Testing shall be planned and conducted to take full advantage of existing investment in DoD ranges, facilities and other resources, wherever practical. The Test Facilities (TESTFACS) Register is the primary repository for information on the Army's test capabilities and is maintained by the U.S. Army TECOM, Aberdeen Proving Ground. The TESTFACS is intended to assist the Army in managing an orderly growth of test capability and serve as a basis for test planners, including Test Integration Working Groups (TIWGs), as well as providing a base of information for resource management. Testers are encouraged to survey and query existing databases (e.g., the TESTFACS register and the Operational Test and Evaluation Command (OPTEC) Instrumentation DataBase (OIDB)), catalogs, etc., to determine what additional needed resources are in inventory, where, and in what quantities. Direct coordination with documented point of contacts (see chapter 11, section IV, paragraphs 11-9 and 11-10 of DA PAM 73-1) is encouraged for the tester to gain a complete understanding of an item's capabilities, limitations, support requirements, suitability, etc., and to determine its potential availability.

b. The magnitude of data required for software T&E, generally requires that more of the internal detailed parameters for the system under test (SUT) be collected and reduced than provided for by range instrumentation. The collection of the SUT detailed parameters for assessing the performance of the software, as well as computer resource utilization requirements, can be accomplished through the use of internal software data collection programs, test hooks, or hardware monitors. Data reduction programs provide for quick look listings of selected data or can be programmed to selectively sort, merge, format, reformat, summarize, analyze, and output the data in accordance with the requirements specified in system and software specifications.

c. Modeling and simulation may be used to augment test data to assist in systems and software evaluation and assessments. Part V of the TEMP delineates how modeling and simulation is to be used for the system under test. Instrumentation, Targets and Threat Simulators (ITTS) is addressed in chapter 11 of DA PAM 73-1. Sources of information on software test tools are the Software Test Tool Report (Ref 11) and An Examination of Selected Commercial Software Testing Tools (Ref 12).

Back to top

3. REQUIRED TEST CONDITIONS.

3.1 Test Coordination.

Software T&E requirements will be developed in consonance with the project TEMP and the Computer Resource Management Plan (CRMP). Software T&E requirements will be coordinated with the test community through working groups such as TIWGs, software working groups, and computer resource working groups (CRWG's) (See AMC Reg 70-16A, Ref 13). Agencies with which test centers should interface with include the program manager, materiel developer, combat developer, user representative, independent developmental evaluator, operational tester, independent operational evaluator, Independent Verification and Validation (IV&V) agencies, and development contractors.

Back to top

3.2 Initial Activities.

There are a variety of test related technical activities which should be accomplished early, and certainly before the start of developmental testing. Early accomplishment of these activities can be of significant value towards the avoidance of schedule delays, as well as towards the avoidance of unnecessary or duplicate test expenditures. Initial activities include:

a. Perform requirements analysis to assure that the allocation of requirements to software has been established and that the requirements are testable, and traceable to functions.

b. Identify system and software test requirements. Ensure that all necessary test conditions for analysis and evaluation of these requirements have been identified. Further, ensure that planned contractor and/or government test coverage is adequate to provide data to support the T&E process. This also ensures that over testing is avoided.

c. Assist in identification and preparation of data collection and reduction to support both software and hardware T&E.

d. Provide independent assessment of the comprehensiveness of the contractor's test program. This gives the materiel developer and the contractor an opportunity to incorporate necessary additional tests into their test programs prior to the government test phase, thus increasing the likelihood for success and reducing the opportunity for problems. Software performance measurement should include review of IV&V contractor reports and/or participation in the IV&V programs where deemed appropriate by the program managers.

Back to top

3.3 Test Planning.

Test planning will be initiated through a TECOM test planning directive as described in paragraph 3-3.e and Appendix F of TECOM PAM 73-1, Ref 14. The directive delineates test center responsibilities in regard to testing an item or system. The test planning directive normally supplements the testing requirements in the Independent Assessment Plan (IAP), Independent Evaluation Plan (IEP), or the Test Design Plan (TDP). For assessed programs, the TDP is included in the IAP. A guide for preparation of an IAP and associated correspondence used by TECOM is contained in Appendix D of TECOM PAM 73-1. Ensure that software issues are represented in the test directive to establish the basis for communications between the tester, evaluator, and materiel developer. If a system requires extensive software support, a separate test directive is issued detailing the support required. Appendix I of TECOM PAM 73-1 prescribes the format for the Detailed Test Plans (DTPs) which apply to government technical testing. Prepare Software Performance Testing subtests to the DTP in accordance with Section 2 of Appendix I. The DTP is derived from and implements the TDP. Guidelines for preparing plans for customer tests is described in paragraph 11-3 of TECOM PAM 73-1.

3.3.1 Documentation Analysis. Conduct a thorough review of user requirements documentation, system specifications, subsystem specifications, and interface specifications to obtain an understanding of the user's needs, the system concept of operation, the system performance characteristics, and the system interface requirements.

a. Analyze system specifications, software specifications, interface specifications, and test documentation for adequacy, accuracy, clarity, completeness, and consistency. The documentation will be analyzed with respect to criteria that are based on appropriate military standards, development and management plans, and any applicable project directives. Comments on the documentation and recommendations for improvements to the documentation will be submitted on DA Form 2028 and provided to the materiel developer.

b. Annex K of J-STD-016 provides guidance on the format and structure of deliverable software products. Content requirements for software products associated with J-STD-016 are found in the standard's annexes E through J.

3.3.2 Requirements Identification. Identify the software requirements from the system software specifications; that is, identify the capabilities that the software is to provide. The software requirements establish a test-to-baseline and provide the discipline and structure for subsequent test planning and the other activities of test conduct and monitoring, test analysis, and test reporting. Once identified, the software requirements become the test criteria and form the basis of formal test plans.

a. Perform a detailed analysis of the SRS and IRS for each software item of a given computer system to identify the technical and operational capabilities which must be implemented in the software. The requirements should be analyzed for testability, performance, timing, and external/internal interfaces. Experience indicates that software requirements documentation contains a wide variety of information, ranging from general statements about the system, to detailed flowcharts reflecting a specific design, and even precoded routines.

(1) Perform an in-depth analysis of the requirements-level functional flow. Review and analyze flow diagrams or equivalent flow documentation generated by the developer for accuracy and consistency with the software specification. If an acceptable set of diagrams is not available, construct flow diagrams using the requirements specification.

(2) Separate requirements from design. In the process of identifying software requirements, identify the capabilities the software is to provide rather than the processes required to provide the capabilities (the what rather than the how). Requirements generated for each software item will identify both the system oriented requirements (e.g., track file maintenance, target identification, etc.) and software specific requirements (e.g., throughput processing, memory reserve, etc.).

(3) Generate a software requirements list for each software item. Requirements are broad statements of capabilities required to be provided by the software. Derive statements directly from the SRS or from other applicable requirements sources, i.e., include software requirements from the IRS when those requirements are not addressed in the SRS. The SRS generally addresses the software requirements from the IRS. The IRS provides detailed information necessary to develop the test conditions and analysis plans for each software item interface software criterion. Individually document each software requirement. An example is presented in Figure 1. Provide an identifier for each requirement. This identifier contains a sequence number and identifies the software item or the major functional area to which the requirement belongs (in Figure 1, "F" indicates firing doctrine and "D" display control). The identifier will be used to reference the software requirements in other test planning and analysis activities. In addition, provide a reference to the system or software documentation where the requirement was stated.

Requirement Number	Software Requirement	Source
F.18	Generate a combined threat order list.	SRS, para. 3.2.1.3.2.2
F.19	Evaluate engageability of targets on the combine threat order list based on their weapon control volume membership.	SRS, para. 3.2.1.3.2.4
F.20	Determine if any friendly targets in the vicinity of an engaged target might be engaged rather than the intended target.	SRS, para. 3.2.1.3.2.6
D.5	Provide a visual indicator to the tactical officer when the radar is tracking a target.	SRS, para. 3.2.4.13

Figure 1. Example of Software Requirements.

b. Generate a requirements list of the sizing and timing requirements for each software item. These requirements address the utilization of computer resources, i.e., memory/storage capacity, central processing unit (CPU) capacity, and input/output capacity. These requirements are documented in paragraph 3.6 of the SRS. This paragraph specifies the amount of internal and external memory and the amount of processing time allocated to each software item. It specifies the resources required of both memory and the CPU for the software item.

3.3.3 Requirements Traceability Analysis. The objective of requirements traceability analysis is to ensure that all requirements have their basis in the highest level specification and that no other requirements are implemented in the delivered software. The completeness of the allocation of the software requirements will be assessed through the identification of each requirement in the SRS and the subsequent mapping of these requirements to the system specification or to other applicable system requirements documentation.

a. Perform a detailed review of the contractor's requirements traceability from the system specification to the software specifications (contained in paragraph 4.2.6 of the SRS). Review the requirements list from the system specification. Beginning with this initial list, ensure each entry in the software requirements lists, generated in paragraph 3.3.2 above, is mapped to requirements from the system specification list. This mapping from the system specification to the software specifications will provide a valuable analysis tool for later use in relating software performance to system performance. An example of a mapping of software requirements to system requirements is presented in Figure 2. In the example, the mapping is from the software specification to the system specification. This example also shows software requirements from more than one functional area (firing doctrine and display control) that together support the same system requirement.

<u>System Requirement</u>

Azimuth sector for automatic threat sector of radar data will be selectable by the operator. Data processing required per radar rotation period shall be completed within 90 percent of the nominal rotation period. (A Spec, paragraph 3.2.1.4.9)

<u>Software Requirement</u>

F.17	Determine threat order numbers relative to the site and most probably threatened defended asset. (SRS, paragraph 3.2.1.3.1.2.1)
F.18	Generate a combined threat order list. (SRS, paragraph 3.2.1.3.2.2)
D.11	Provide for the processing to support operator interaction with non-real-time database through the remote control unit. (SRS, paragraph 3.2.4.3.4)

Figure 2. Example: Mapping of System Requirements to Software Requirements.

b. Document on DA Form 2028 any results of the traceability analysis which indicate material omitted from the requirements, extraneous material or inconsistencies between the system and software specifications, specifications lacking test requirements, or requirements that are simply incomplete. These comments will be presented for resolution at the SSR.

3.3.4 Test Condition Development. Identify test conditions for each software requirement. Test conditions identify the various situations under which the requirements must perform; they will be used to identify the test cases required to assess software performance. Describe test conditions in terms of system environments, target environments, weapon environments, and events which must be present to invoke software processing necessary to demonstrate the required capability. Test conditions should include stress case testing, interface and interoperability testing, and testing at the minimums and maximums of performance boundaries.

a. Generate a list of test conditions for each software requirement. Provide an identifier for each test condition which contains both the software requirement identifier and a test condition sequence number. The identifier will be used to reference the test conditions in other test planning and analysis activities.

b. An example of test conditions is presented in Figure 3. In this example, evaluating engageability of targets based on their weapon control volume membership, the test conditions are the various types of targets (friend, unknown, and hostile) and volumes (free, tight, and hold) that must be dealt with. In the example, there are ten test conditions for the single software requirement. The number of test conditions will vary depending on the number of situations, and combination of situations, in which the software is required to operate. The ten software test conditions presented in Figure 3 were generated from the sample text from a SRS, which is provided in the subparagraph below, and the functional flow diagram example in Figure 4.

Software Requirement

F.19 WEAPON CONTROL VOLUME ENGAGEABILITY EVALUATION. Evaluate engageability of targets on the combined threat order list based on their weapon control volume (WCV) membership.

Test Conditions

F.19.01	A hostile target on the combined threat order list (CTOL) is in an active FREE WCV.
F.19.02	A hostile target on the CTOL is in an active TIGHT WCV.
F.19.03	A hostile target on the CTOL is in an active HOLD WCV.
F.19.04	An unknown target on the CTOL is in an active FREE WCV.
F.19.05	An unknown target on the CTOL is in an active TIGHT WCV.
F.19.06	An unknown target on the CTOL is in an active HOLD WCV.
F.19.07	A hostile target on the CTOL is in a HOLD WCV which is inactive.
F.19.08	An unknown target on the CTOL is in a HOLD WCV which is inactive.
F.19.09	An unknown target on the CTOL is in a TIGHT WCV which is inactive.
F.19.10	The CTOL contains the maximum number of targets.

Figure 3. Example: Test Conditions.

3.3.4.1 Weapon Control Volume Engageability Evaluation (WCVEE).

a. The WCVEE function evaluates targets on the CTOL in terms of their current membership, or location, in a WCV. As a function of target identification (ID), i.e., friend, unknown, or hostile, and WCV type, i.e., weapons free, tight or hold, eligibility for automatic engagement consideration is determined as follows:

(1) Weapons Free - Weapons can fire at any aircraft not positively identified as friendly. This is the least restrictive weapons control volume.

(2) Weapons Tight - Fire only at aircraft positively identified as hostile according to prevailing hostile criteria.

(3) Weapons Hold - Do not fire except in self-defense or in response to a formal order. This is the most restrictive weapons control status.

b. The WCVEE inputs each target on the CTOL. The target's ID and Weapon Control Volume Status (WCVS) are retrieved. The Weapon Control Volume Constraint Indicator is then set or reset accordingly (0 = no restrictions, 1 = restricted) based on the target's current membership, if any, in a WCV. WCVEE will process all targets on the CTOL within each major cycle interval.

WEAPON CONTROL VOLUME ENGAGEABILITY EVALUATION
(WCVEE)

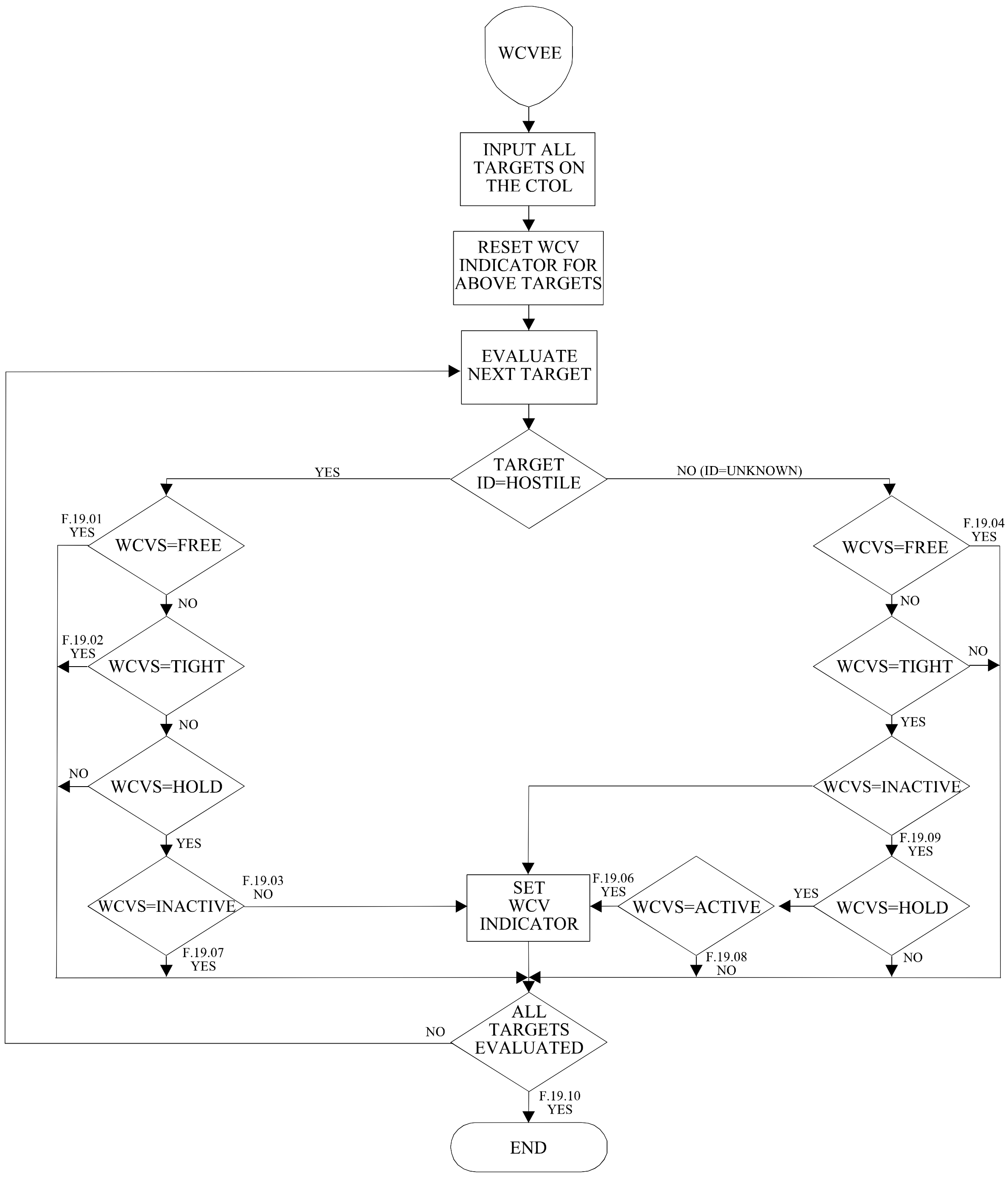

Figure 4. Example of Functional Flow Diagram.

3.3.5 Development of Analysis Plans. Develop detailed analysis plans based on software requirements, test conditions, and data collection and reduction capabilities. Analysis plans will provide a detailed procedure for analyzing the data to determine if a requirement was met. Develop an analysis plan for each test condition for each software requirement. An example of an analysis plan is presented in Figure 5. Analysis plans, as a minimum, will contain the following:

a. Information regarding the specification for the CSCI, including specification number, date of the specification or revision, and other associated documentation (e.g., system specification, interface specifications) used to develop the analysis plan.

b. The software requirement, including the reference identifier.

c. The test condition, including the reference identifier.

d. Data required to verify that the test condition exists and to verify that the software performed as required.

e. Data reduction required to support analysis.

f. Step-by-step analysis procedure for determining that the test condition existed and that the software performed as required. Be as specific as possible when writing the analysis procedure; it will greatly simplify the analysis process after testing. Since the performance of software must be assessed through available data, the following methods of assessment should be considered in developing analysis procedures:

(1) Observation. The test contains a provision for observation of and recording of whether an expected event has been attained.

(2) Direct analysis. The data collected during a test will be sufficient to assess software performance.

(3) Indirect analysis. The test will not provide data that will allow the use of either observation or direct analysis methods to assess software performance. However, software performance may be inferred based on the occurrence of specific events which would occur only if the expected software performance has been attained.

Test Condition No. F.19.03	**ANALYSIS PLAN**	
STATEMENT OF TEST CONDITION	**CRITERION SOURCES**	**DATA REDUCTION/DATA REQUIRED**
A hostile target on the CTOL is in an active HOLD WCV. **STATEMENT OF TEST CONDITION** F.19 Evaluate engageability of targets on the Combined Threat order List (CTOL) based on their Weapon Control Volume (WCV) membership.	CSCI: Firing Doctrine SPECIFICATION NUMBER: MIS-035678 DATE: 15 October 1990 SRS SPEC PARA #: 3.2.1.3.2.4 A-SPEC PARA #: 3.2.4 OTHER RELATED DOCUMENTS: IRS-32543, para 3.2.1.1	Track Data Record (Program (0801) • Target Number (TGTN) • Target Identification (ID) • CTOL Indicator (CTOLI) • Cell Membership (CELLTYP) Combined Threat Order List (Program 080B) • Target Number (TGTN) • WCV Indicator (WCVI) System Status File (Program 0801A) • WCV Identification (WCVI) • WCV Status (WCVS)
INITIAL CONDITIONS	**EXPECTED RESULTS**	**ANALYSIS PROCEDURE**
1. ID of selected target (TGTN=NNN) is HOSTILE (ID=HOS) 2. Target is on the CTOL (CTOLI=1) 3. Target is within a HOLD WCV (CELLTY1=HOLD, WCVID=HN) 4. The status for the HOLD WCV is active (WCVS=1 for WCVID=HN)	The WCV indicator is set (WCVI=1) in the CTOL for the selected target (TGTN=NNN).	1. Verify that the target ID is HOSTILE for the selected target, i.e., TGTN=134. 2. Verify the target is on the CTOL. 3. Verify target is located within a HOLD WCV, I.e., within WCVID=HI. 4. Verify the status of the WCV, i.e., HI, is active (WCVS=1). 5. To ensure the software criterion is met under the test condition, verify the WCV Indicator (WCVI) is set in the CTOL for the selected target, i.e., TGN=134.

Figure 5. Example of Analysis Plan.

3.3.6 Test Coverage Mapping. The intent of the test planning activity is to assure, to the extent possible, that opportunities exist during testing to assess each software requirement. A tool to assess completeness of testing is the test coverage matrix (TCM), a mapping of the total set of software requirements and associated test conditions to planned developer (contractor) tests and planned government tests. To develop this tool requires the existence of detailed test scenarios or test procedures. The mapping is then accomplished by analyzing the scenarios or procedures for each test to determine which software capabilities should be demonstrated by that test. An example of the TCM is presented at Figure 6. Sample test phases, both contractor and government, are listed across the top. For each test phase, the test cases (or test scenarios) are then listed. This completes construction of the TCM. It then remains to fill in the blanks by assessing the test cases (test scenarios) to determine the conditions indicated by each. When software requirements not having adequate test coverage are identified (e.g., F.18.02 in Figure 6), adjustments can often be made to already scripted tests, or if not, additional tests can be scripted to increase coverage.

a. Determine the completeness of testing of each software requirement under each of its test conditions. This is done by determining whether the test condition will exist during planned tests and whether the data collected during the tests will allow for assessment of the analysis parameters listed in the analysis plan (see paragraph 3.3.5). A test condition should be considered to exist only if the test will produce data sufficient to verify that the test condition does, in fact, occur. Coverage of each of the test conditions is determined as follows:

(1) Assign a "C" (complete) if the test condition will exist during the test and it will be possible to completely assess the performance of the software requirement.

(2) Assign a "P" (partial) if the test condition will exist during the test and it will be possible to partially, but not completely, assess the performance of the software requirement.

(3) Assign an "N" (not tested) if the test condition will not exist during the test or it will not be possible to assess the performance of the software requirement.

b. Once the test coverage of each test condition has been assessed for each test, determine the anticipated cumulative coverage of all testing. Enter the cumulative completeness of test (COT) in the "CUM COT" column of the TCM. The cumulative coverage is established as follows:

SOFTWARE REQUIRE-MENT & TEST CONDITION	CONTRACTOR															GOVERNMENT								
	SOFTWARE INTEGRATION TEST									INTEGRATED SYSTEM TEST						SEARCH TRACK & FIRING TEST								
	1	2	3	4	5	6	7	8	9	1	2	3	4	5	6	1	2	3	4	5	6	7	CUM COT	COT (%)
F.01																								83%
F.01.01	N	N	N	P	N	N	N	C	N	N	P	P	N	N	N	P	N	N	N	N	C	N	C	
F.01.02	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	
F.01.03	P	N	N	N	N	N	P	N	N	N	N	N	N	N	N	N	N	N	N	N	P	N	P	
F.02																								100%
F.02.01	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	C	N	C	

•
•
•

F.18																								75%
F.18.01	C	N	N	N	C	N	N	N	N	N	C	N	N	N	N	N	N	N	N	N	C	N	C	
F.18.02	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	
F.18.03	P	N	N	N	N	N	N	N	N	N	N	N	N	P	N	N	N	N	N	N	P	P	P	
F.18.04	N	C	N	C	N	N	N	C	N	N	N	N	N	P	N	N	N	N	N	N	C	N	C	
F.18.05	N	P	P	N	N	N	N	N	N	N	N	P	N	N	N	N	N	N	N	N	P	N	C	
F.18.06	N	C	N	N	N	N	N	N	N	N	N	N	P	N	N	N	N	N	N	N	C	N	C	
F.18.07	N	N	P	N	N	N	N	N	N	N	N	P	N	N	N	N	N	N	N	N	P	N	P	
F.18.08	N	N	N	N	N	N	C	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	C	
F.19																								50%
F.19.01	N	N	P	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	P	N	P	

Figure 6. Example: Test Coverage Matrix.

(1) If any of the tests under consideration have tested the software under a test condition, then the cumulative coverage of that test condition is also "C."

(2) If none of the tests under consideration have tested the software under a test condition, then the cumulative coverage of that test condition is "N."

(3) If some of the tests under consideration provide partial coverage of a test condition but none provide complete coverage, then the coverage provided for each test must be

considered in detail to find the cumulative coverage. This is done by determining if through a combination of tests, the software performance can be completely assessed. If so, consider the cumulative coverage of that test condition coverage to be "C"; otherwise, consider the coverage to be "P".

c. After assessing test coverage at a test condition level, calculate a projected coverage at the software requirements level. Compute the test coverage indicators for each software requirement as follows:

(1) Assign each test condition a score as follows:

1 if the cumulative test coverage was "C"

1/2 if the cumulative test coverage was "P"

0 if the cumulative test coverage was "N"

(2) For each software requirement, find the sum of the scores of its test conditions, multiply the sum by 100, and divide that product by the number of test conditions. The results are percentages which indicate the projected completeness of testing for each software requirement. Enter the percentage in the "COT (%)" column of the TCM.

d. The TCM is used to show projected test coverage at both the requirement and test condition level. Columns that are not applicable to the requirement level or test condition level are shaded as appropriate. One additional point about the TCM. As will be addressed below under Test Analysis, the same matrix format can later be filled in based on test results to determine comprehensiveness or extent of actual testing. This extent of testing (EOT), besides being a metric in itself, can serve as an indicator of confidence in the overall software evaluation. Such an after-test matrix would provide a view of how many ways each condition was tested and under how many different conditions each requirement was demonstrated. It is also an easy matter to annotate within the matrix where problem areas surfaced or software changes were made during testing.

3.3.7 Development of System/Software Test Scenarios. Develop test scenarios for software requirements not having adequate test coverage. Design scenarios such that they will represent the different conditions in which the software not having adequate test coverage is required to operate. In some cases, adjustments can be made to already scripted tests, but if not, script additional tests.

3.3.8 Identification of Data Collection and Data Reduction Requirements. Identify data collection and reduction requirements for support of the software assessment. If existing built-in data collection is not sufficient to support required analysis, more detailed test conditions and test scenarios may be required to provide for direct observation or inference of software performance.

a. In addition to built-in data collection, the following are examples of other data to support software assessment.

(1) Videotapes of display consoles.

(2) Test conductor logs.

(3) Software build and patch descriptions.

(4) Contractor test plans and procedures.

(5) Contractor test reports.

b. For tests selected for monitoring, results obtained and observations made during execution of the tests will be recorded for subsequent analysis. The test number, date, mission recording tape number, time of day, events, descriptions of any anomalies, etc., will be recorded. This information will be used to determine what data will be required for follow-on analysis and will provide reference information for location of the data on videotapes, data listings, etc. An example of a form for the recording of test monitor notes is presented in Figure 7.

TEST MONITOR NOTES

DATE ________________________ MISSION CODE _______________
NAME ________________________ TEST NUMBER________________
DATA COLLECTION TAPE NUMBER ________________________

TIME HR:MIN:SEC	EVENT
: :	
: :	
: :	
: :	
: :	

Figure 7. Sample Test Monitor Form.

Back to top

4. TEST PROCEDURES.

4.1 Test Conduct.

TECOM authorizes initiation of testing by issuing a test execution directive after test planning has been completed. Appendix H of TECOM PAM 73-1 prescribes the format for the test execution directive. The most desirable data is from full-up integrated system tests; following that, from software integration tests, with the least desirable data from unit or code testing. It should be recognized, however, that due to limitations in test resources or limitations in providing a complete threat environment, analysis of some test conditions (and even some requirements) will have to be made based on data from the less realistic test beds, such as those used to support unit testing. Wherein this is the case, the analysis should reflect less confidence in performance, although the EOT will be improved. There are requirements and test conditions, however (e.g., display control), where data from unit test may be as valid as data from system test. Make maximum use of data from the developer's tests to minimize the test requirements of the government's developmental testing. Generate specific procedures, or recommend changes to planned procedures, for testing based on the requirements defined in test plans developed for the particular phase of testing.

Back to top

4.2 Test Monitoring and Analysis.

For tests selected for monitoring and analysis, obtain results through observation during test monitoring, through analysis of data collected during test execution, and through review of developer test reports. The test data to be acquired and recorded are described in the analysis plans developed for each test condition (see paragraph 3.3.5). Perform test analysis in accordance with each analysis plan. Determine performance assessment and test completeness measures for each test condition, and document these in test-by-test and final test reports as described in paragraph 6.2.

Back to top

5. DATA REQUIRED.

Due to the nature of software performance testing, specific data required will be tailored to the required SUT. Data required is itemized in individual analysis plans (see Figure 5). The IEP addresses data required for the system at a higher level.

Back to top

6. PRESENTATION OF DATA.

6.1 Test Analysis.

Test analysis, like test planning, focuses on software requirements. Results of tests are assessed to determine software performance. Test results are obtained both through observation during test monitoring and through analysis of data collected during test execution. The performance assessment indicates the degree to which the software's functional characteristics are demonstrated for each software requirement. In addition to the analysis of test results, software changes incorporated between tests are assessed to determine impact on both the validity of current test results and the continuing validity of prior test results. Consistent with the philosophy of focusing on required capabilities, software change analysis is focused at the requirements as opposed to the code.

6.1.1 Performance Assessment and Test Coverage at Test Condition Level. Analyze data on a test-by-test basis in accordance with analysis plans. Document performance, assessment method, actual test coverage achieved, and relevant test or test plan and procedure for each test condition. An example is presented in Figure 8.

TEST ANALYSIS RESULTS

Test: Integration Test 2
Date: 15 October 1990
Analyst: John Doe

TEST CONDITION	PERFORMANCE ASSESSMENT	METHOD OF ASSESSMENT	TEST COVERAGE	TEST PLAN OR TEST PROCEDURE
F.01.01	P	I	C	ED-9
F.01.02	P	D	C	ED-1
F.01.03	F	D	C	ED-2
F.02.01	P	D	C	ED-2

•
•
•

F.18.01	P	O	C	ED-9
F.18.02	NA			
F.18.03	P	I	C	ED-9
F.18.04	P	D	C	BT-10
F.18.05	P	I	C	BT-2
F.18.06	F	D	P	ED-2
F.18.07	F	I	C	BT-2
F.18.08	NA			
F.19.01	P	D	C	ED-14

Figure 8. Example: Test by Test Analysis Results.

a. Performance assessment is an evaluation of whether a software requirement was demonstrated. It is indicated as follows:

(1) "Passed" (P) indicates that the software requirement was successfully demonstrated under the specified test condition.

(2) "Failed" (F) indicates that the software requirement was not successfully demonstrated under the specified test condition; a Test Incident Report (TIR) or Software Problem Report (SPR) was written to document this failure.

b. Method of assessment refers to the analysis technique used to assess the performance:

(1) "Observed" (O) indicates that the performance was assessed by observation of the events which occurred during test operations.

(2) "Direct analysis" (D) indicates that performance was assessed by direct analysis of test data.

(3) "Indirect analysis" (I) indicates that performance was assessed indirectly. That is, data to assess the performance by direct analysis were not available, and software performance can only be inferred.

c. The TC for each test condition is determined based on the following:

(1) "Complete" (C) indicates that data were available to assess all of the expected results.

(2) "Partial" (P) indicates that data were not available to assess all of the expected results.

(3) "Not Tested" (NT) indicates that data were not available to assess any initial conditions or expected results, or that the test condition did not exist.

d. Test/procedure indicates the test or test plan and procedure where the software requirement was tested, e.g., Engagement Decision Test 9 (ED-9), Benign Tracking Mission 10 (BT-10), etc.

6.1.2 Performance Assessment and EOT for Software Requirement. Performance assessment and EOT will be determined for each software requirement based on an accumulation of results at the test condition level. An example is presented in Figure 9.

CUMULATIVE TEST ANALYSIS RESULTS Test: Integration Test 2 Date: 15 October 1990 Analyst: Jane Doe		
SOFTWARE REQUIREMENT	PERFORMANCE ASSESSMENT	EXTENT OF TEST
F.01	Met	Exercised
F.02	Met	Limited
F.03	Partially Met	Exercised
F.04	Met	Significant
F.05	Not Met	Exercised
F.06	Partially Met	Limited
F.07	Not Met	Not Tested
F.08	Not Met	Significant
F.09	Not Met	Limited
F.10	Not Met	Significant

Figure 9. Example: Cumulative Test Analysis Results

a. The performance assessment is a measure of the degree to which the software meets the requirements. Use the following measures to assess performance:

(1) "Met" (M) indicates that the software requirement was successfully demonstrated (i.e., performance was rated as "passed" for all test conditions demonstrated for that test requirement).

(2) "Partially met" (PM) indicates that the software requirement was not successfully demonstrated under all test conditions (i.e., performance was rated as "failed" for at least one test condition demonstrated for that test requirement).

(3) "Not met" (NM) indicates that the software requirement failed to be demonstrated under any test condition (i.e., performance was rated as "failed" for all test conditions demonstrated for that software requirement).

b. The EOT is the degree to which each software requirement has been tested and provides a confidence measure of overall performance assessment. The EOT for each software requirement falls into one of four levels as defined below.

(1) "Significant" (S) indicates that performance for that software requirement has been demonstrated under a variety of test conditions and that the performance is projected as representative for all conditions.

(2) "Exercised" (E) indicates that performance for that software requirement has been demonstrated under a variety of test conditions and that the performance is projected as representative for other similar conditions.

(3) "Limited" (L) testing indicates that performance for that software requirement has been demonstrated under at least one set of conditions and that the performance is projected as representative for only those conditions.

(4) "Not tested" (NT) indicates that no capability has been demonstrated for that requirement.

c. An option to these qualitative guidelines for EOT is a quantitative measure based on number or percentage of test conditions observed, weighting test conditions with respect to relative impact, or other such factors. Guidelines for EOT based on percentage of test conditions observed is as follows:

(1) "S" for 90% to 100% coverage

(2) "E" for 50% to 89% coverage

(3) "L" for 1% to 49% coverage

(4) "NT" for 0% coverage

6.1.3 Software Change Analysis.

a. Changes to SRS's, software patches, and corrections to problems documented in SPRs and TIRs will be considered in the assessment of the impact of software changes on software requirements, software performance, and the validity of test data. Any changes to the functional and allocated baselines, i.e., requirements documentation, are required to be documented in Engineering Change Proposals (ECPs). Changes that effect the product baseline will generally require changes to the design and test documentation, but will not require changes to the requirements documents. Specific software change requests are documented on ECPs-Software. Change packages are approved and scheduled for implementation by the appropriate configuration control board in accordance with MIL-STD-973. Software configuration management responsibilities are defined in section IV.B of the CRMP. This section also addresses the maintenance of computer software baseline integrity and the identification, control, and release of all products.

b. For new SRS's or changes to SRS's, the SRS or change will be analyzed to determine what, if any, additions, deletions, or modifications are necessary to the software requirements. Software changes made during software or system tests will be analyzed to determine the validity of the changes with respect to software requirements, the impact or potential impact of the changes on software performance, and the impact of the changes on the validity or utility of test results. Due to the number of changes made, it may not be feasible to assess every software change. In that case, software changes will be assessed according to the following priorities:

(1) Changes that affect or change software test requirements.

(2) Changes that make regression testing necessary.

(3) Changes that correct coding errors in the software.

6.1.4 Software T&E Metrics. The methodology described within this TOP provides data for measurement of some of the 12 DA PAM 73-7 army software metrics, or portions of the metrics. The specific metrics addressed are computer resource utilization, requirements traceability, requirements stability, and breadth of testing. The software T&E metrics are used to identify potential problem areas and provide insight into both the acquisition process and its products. They can be used in assessing the maturity of the system software. They provide measures and indicators which address the readiness of the software to proceed to the next development phase. They are used to provide technical analysis results to developers, evaluators, and decision makers. Determine measurements for the computer resource utilization, requirements traceability, requirements stability, and breadth of testing metrics as follows:

a. Computer resource utilization (CRU) metric. This metric provides estimates and measurements of CPU capacity, input/output (I/O) capacity, and memory/storage capacity to determine compliance to CRU criteria (reference paragraph 3.3.2.b). Resource utilization tends to increase over the development of the project. Therefore, adequate planning must be done to ensure that the software's operation does not put undue demands on the target hardware's capabilities. This measure tracks utilization over time to make sure that target upper bound utilization is not exceeded and that sufficient excess capacity remains for future growth and for periods of high stress loading. Actual utilization should be demonstrated at the system level during Formal Qualification Testing (FQT), and during Post Deployment Software Support) if additional capability is added. For systems where the software is being developed in increments, project the results for utilization to be the actual measurement plus the budgeted portion to be added. Actual usage should be measured during peak operational loading periods and should include the operating system and non-developer supplied software as well as the development software.

(1) Obtain CPU and I/O resource utilization results from the contractor's timing tests for each processor in the system. Determine compliance for each CSCI by comparing actual usage to the requirements specified in the SRS. The data requirements include:

(a) For each CPU in the system:

- CPU identifier
- Target (Upper bound) CPU usage (% of capacity)
- Actual CPU usage (% of capacity)
- Projected CPU usage (% of capacity)

(b) For each I/O channel in the system:

- I/O channel identifier
- Target I/O usage (% of capacity)
- Actual I/O usage (% of capacity)
- Projected I/O usage (% of capacity)

(2) Obtain memory and storage utilization from results computed by the contractor. For memory, the resource is random access memory (RAM). RAM for this metric refers to both volatile and non-volatile (including read only) memories. For storage, the resources include disk space and other mass storage. Determine compliance for each CSCI by comparing actual usage to the requirements specified in the SRS.

(a) For each RAM in the system (in bytes):

- RAM identifier
- Capacity
- Target upper bound
- Projected usage
- Actual usage

(b) For each mass storage device in the system (in bytes):

- Storage device identifier
- Capacity
- Target upper bound
- Projected usage
- Actual usage

(c) For each CSCI in the system:

- CSCI identifier
- RAM allocation budget as specified in the requirements (in bytes)
- Actual RAM usage (in bytes)
- Mass storage allocation budget (in bytes)
- Actual mass storage usage (in bytes)

b. Requirements traceability metric. In the context of the TOP, this metric measures the adherence of the SRSs to the system specifications. It aids in the assessment of the impact of software on overall system performance and can indicate those areas where requirements have not been sufficiently developed. The requirements traceability metric should be used to verify that sufficient functionally has been identified to proceed to the design phase of the software development process.

(1) Determine the number of SRS requirements, the number of SRS requirements traceable to the system specification, i.e., the System/Subsystem Specification (SSS), and the number of SRS requirements not traceable to the system specifications. SRS requirements not traceable to the system specifications may indicate incompleteness, inconsistencies, and/or extraneous requirements in the system and software specifications.

(2) Utilize the results of the requirements traceability to determine the adequacy of the requirements specifications at Joint Technical Reviews and other appropriate reviews. All software requirements should be traced to the system specifications at the time of a Joint Technical Review.

c. Requirements stability metric. In the context of the TOP, this metric measures the degree to which changes in the software requirements affect the development effort.

(1) Track cumulative requirements discrepancies versus closure of those discrepancies over time, on a monthly basis. For each CSCI, compute the cumulative number of requirements discrepancies and the total number of discrepancies for use at Joint Technical and Joint Management reviews.

(2) Good requirements stability is indicated by a leveling off of the cumulative discrepancies curve with most discrepancies having reached closure. A high level of instability at the Joint Technical Review stage indicates serious problems that must be addressed prior to proceeding to coding. Allowances should be made for higher instability in the case where rapid prototyping is utilized.

d. Breadth of testing metric. Breadth of testing measures the amount of testing which has been achieved on the required software functions. This metric measures the percentage of software functional testing which has been performed, and the percentage of software functions which has been successfully demonstrated. Breadth of testing consists of the following measures: coverage, test success, and overall success. These three measures are used in the following equation:

$$T_{COVERAGE} \times T_{SUCCESS} = \text{Overall Success}$$

where:

$$T_{COVERAGE} = \frac{\text{Number of requirements tested } (R_{TESTED})}{\text{Total number of requirements } (R_{TOTAL})}$$

$$T_{SUCCESS} = \frac{\text{Number of requirements passed } (R_{PASSED})}{\text{Number of requirements tested } (R_{TESTED})}$$

Test Coverage Test Success Overall Success

$$R_{TESTED}/R_{TOTAL} \times R_{PASSED}/R_{TESTED} = R_{PASSED}/R_{TOTAL}$$

R_{TOTAL} = Total number of requirements. Use the total number of test conditions since a requirement may have more than one test condition.

R_{TESTED} = Number of requirements tested. Use the total number of test conditions tested. If a test condition was only partially tested, give it a credit of 1/2. This provides partial credit for testing of requirements and provides additional granularity into breath of testing.

R_{PASSED} = Number of requirements passed. The total number of test conditions that have been successfully demonstrated through testing.

Multiply the values computed by 100 to express as percentages.

The coverage portion indicates how well the software was tested, without regard to success, and the success portion indicates how well the software performed. By observing these trends over time, the extent of overall testing and growth in the functionality of the software can be tracked. The government should clearly specify what functionality should be in place at each phase of development. This process is obviously system dependent. At each stage of testing (unit through system level stress testing), emphasis should be placed on demonstrating a high percentage of the functionality needed for that stage of testing. Prior to the formal Government operational test, most functions should be demonstrated under stress loading conditions.

6.1.5 Impact of Software Performance on System Performance. The process of assessing the impact of software performance on system performance requires the application of expertise in both the software and the system. The process should be the joint effort of hardware, software, and system experts. Factors in relating software performance to system performance include an understanding of system requirements, the allocation of system requirements to hardware and software subsystems, and the traceability of software requirements to the system-level requirements. A tool used to assess the impact of software performance on system performance is the mapping of software requirements to system requirements. Based on this mapping and the performance of the individual software requirements, the impact of software performance on system performance can be assessed. For software requirements not successfully demonstrated, determine how the problem affects system operation or performance. A grouping of software capabilities into categories, as discussed below, will also assist in relating overall software performance to system performance.

a. The requirements traceability mapping, defined in paragraph 3.3.3, provides a valuable tool in assessing the completeness of user and system requirements documentation, and in verifying if sufficient functionality has been demonstrated to warrant proceeding to the next stage of development or testing. Utilize the results of the traceability analyses and measures, in paragraphs 3.3.3.c and 6.1.4.a, to determine outstanding omissions from the requirements, incompleteness of requirements, extraneous requirements, and inconsistencies between system and software requirements. The results provide the basis for assessing the adequacy of the requirements specifications at the Joint Technical Review and later in the development cycle, i.e., at design reviews, if outstanding requirements errors still exist. All software requirements should be traced to the system specification at the time of the Joint Technical Review.

b. Generate a high-level summary of software test results for presentation at Joint Technical Reviews and design reviews. Combine the software requirements into functional areas to assist in relating overall software performance to system performance. Figure 10 provides an example of a software test results summary. The summary lists the functional areas, along with the software criteria which comprise each functional area. Utilize the results of the Breadth of Testing metric to address how well the software worked and how well it was tested, i.e., "overall success" and "test coverage" from paragraph 6.1.4.d. The "Overall Success" column represents the percent of test conditions successfully demonstrated. The "Test Coverage" column indicates the percentage of test conditions applicable to the functional area which were tested. The number of outstanding SPRs and/or TIRs should also be shown, along with footnotes to highlight problem areas and their impact on system performance. This summary provides a snapshot of the status of software performance and EOT at selected milestones in the test cycle, i.e., FQT and DT, or at any point this information may be required.

Functional Area	Software Criteria	Overall Success	Test Coverage	# of Open SPRs/TIRs
Executive	F.1, D.1	96	96	None
Initialization	F.2, D.2	84	95	1(1)
Track File Maintenance	T.1 - T.6	78	84	3(2)
Saturation Alleviation	T.7	88	88	None
Target Identification	F.3, F.4	100	100	None
Threat Assessment/ Ordering	F.5, F.6	100	100	None
WCV Engageablity Evaluation	F.7	92	96	1(3)
Situation Display	D.3 - D.11	95	98	2(4)
Computer Resource Utilization	C.1 - C.8	88	100	1(5)

(1) System database parameters not maintained in Non-Volatile Memory after system has been powered down and then powered up.
(2) Remote track files are not being updated at the proper rates, target identification is not properly maintained for operator overrides, and associated track files not deleted when radar track is dropped on primary target.
(3) UNKNOWN targets are eligible for engagement, even though they are located in a WEAPONS TIGHT WCV.
(4) Radar symbol improperly displayed for engaged targets and target ID symbol improperly overlaid when remote ID is received.
(5) System crash occurred when maximum track load was reached.

Figure 10. Example: Software Test Results Summary.

Back to top

6.2 Test Reporting.

6.2.1 Test Incident Reports. Prepare TIR's for software-related incidents which are observed during test monitoring and analysis. Document TIR's on DA Form XXXX-R, Figure 10-1 of Chapter 10 of DA PAM 73-1, and prepare them in accordance with Figure 10-1 and Tables 10-1 and 10-2 of the same document. Use TIR's to report incidents that result in noncompliance with requirements specifications, summarization of subtest and final report results, suggested improvements, or the achievement of important milestones in the test program. Due to the delay in the distribution of test reports, TIR's provide for the timely identification of problems and distribution of test results to the project test community.

a. It is recommended that software TIR's be written in a standard format to ensure that important data are provided for the evaluation and correction of the incidents/problems identified in the TIR's. Provide the following information in the incident description, Block 90, of DA Form XXXX-R.

(1) Enter a title which briefly describes the incident. The title provides a quick reference to the incident in discussions with project test community personnel and can be used in the preparation of TIR status summaries.

(2) Use paragraph 1 to describe the incident. Always provide a test identifier, a time reference, the software build used, and system/target information such as system configurations, target numbers, message types, parameter values, etc. This will aid the developer in identifying the causes of problems and will expedite possible solutions.

(3) Use paragraph 2 to identify software requirements and test conditions for which the incident results in a violation of specification requirements. Also identify other requirements for which the incident affects performance of the system, even though a violation of specification requirements does not occur. This information will aid in subsequent assessment of software requirements and in relating software performance to system performance.

(4) Use paragraph 3 to provide the classification of the incident reported in the TIR. The justification for the classification of the incident should be based on the effect of the software-related incident on system performance or operation. This will aid in relating software performance to system performance during test reporting activities.

b. Maintain a summary of TIR's for use in TIWG TIR reviews and for tracking the status of software problems and concerns. The status of a TIR should remain "open" until corrections are implemented in the software and are successfully demonstrated and verified through testing and subsequent analysis by the government. Open TIR's must be tracked throughout the acquisition cycle. Schedule and budgetary considerations may require delays in fixing some problems until the post-deployment phase of the system program.

6.2.2 Software Problem Reports. Write SPRs during test phases when TIRs are not required. SPRs are documents used by the development contractor to document software-related problems discovered during contractor testing prior to the release of the system and software for government testing. The preparation of SPRs aids the developer in the early identification and correction of software-related problems. SPRs must be tracked similarly to TIRs. Each open SPR will be converted to a TIR at the start of government testing.

6.2.3 Test Assessment Reports.

a. Test reports are prepared and published in accordance with paragraph 13 of the TECOM test planning directive. The types of test reports are described in paragraph 3.5 of TECOM PAM 73-1. They are normally the formal test report, the abbreviated test report, and the test record, and may also be identified as interim, partial, or final. Test report formats and associated correspondence are contained in Appendix J of TECOM PAM 73-1. The detailed test reports developed by the test centers provide the technical analysis results for the development of the Expanded Test Report (ETR) or the Independent Assessment Report (IAR) (the Independent Evaluation Report (IER) if AMSAA is the independent evaluator) developed by the Independent Assessor (IA). An ETR may be written for Acquisition Categories (ACATs) III and IV systems not requiring a full assessment (see paragraph 3-5.b and Appendix M of TECOM PAM 73-1). Since the ETR reflects a TECOM position, it the basis of the TECOM vote at IPRs. The IAR contains the IA's technical position, explanation of differences with the position of other commands or agencies, and supporting evidence for the statements. The IAR highlights those issues which have been answered and may impact on any milestone decision, presents those issues that need further assessment, contains conclusions, and makes recommendations. The IAR is presented at milestone decision reviews (reference paragraph 3-5.c and Appendix K of TECOM PAM 73-1). Guidelines for preparing reports for customer testing is described in paragraph 11-4 of TECOM PAM 73-1.

b. Prepare Software Testing Performance subtests to TECOM test reports in accordance with Section 2 of Appendix J of TECOM PAM 73-1. Prepare report inputs to document overall software performance assessment and EOT. The overall software performance determines the software requirements that have been met, plus the boundaries and limitations of the capabilities of the software. The cumulative EOT provides a confidence indicator on results of the overall performance assessment. Test findings and technical analysis for software performance are documented in paragraphs 2.x.4 and 2.x.5 of the test report.

(1) Document results of the cumulative performance assessment, assessment method, and test coverage achieved at the software test condition level in paragraph 2.x.4 of the test report. Since there are generally a large number of test conditions, results will be recorded in

Appendix B, Test Data, to the test report. These results provide the test findings upon which the performance assessment and EOT for the software criteria will be made, and are determined in accordance with paragraph 6.1.1.

(2) Document performance assessment and EOT for each software requirement based on an accumulation of results at the test condition level in accordance with paragraph 6.1.2.

(3) Determine compliance assessment for each software criterion, based on its cumulative performance assessment and EOT, and record the results in Appendix A to the test report. Compliance assessment will be determined as Met, Partially Met, Not Met, or Not Tested. An assessment of Met indicates a high confidence for that test criterion, since no problems exist and a high percentage of test conditions has been demonstrated. An assessment of Partially Met indicates a lower level of confidence due to either existing problems or lack of testing (a limited number of test conditions was demonstrated). An assessment of Not Met indicates a low confidence for that test criterion, since outstanding problems exist for all test conditions demonstrated. An assessment of Not Tested indicates that a confidence level cannot be determined, as none of the test conditions for that criterion has been demonstrated. Compliance assessment will be determined as follows:

Compliance Assessment	Criterion Performance	Criterion EOT
Met	Met	Significant
Met	Met	Exercised
Partially Met	Partially Met	Significant
Partially Met	Partially Met	Exercised
Partially Met	Met	Limited
Partially Met	Partially Met	Limited
Not Met	Not Met	Significant
Not Met	Not Met	Exercised
Not Met	Not Met	Limited
Not Tested	NA	Not Tested

(4) Utilize the Software Test Results Summary, Figure 10, as the basis for documenting a comprehensive overview of software performance in paragraph 2.x.5 of the test report. Supporting detailed analyses should include a description of each functional area, summarization of the areas tested and not tested, and a discussion of related problem areas. Also, identify test incidents and areas which are considered deficiencies and shortcomings, along with the rational for the stated classification. MIL-STD-882 (Ref 39) contains information on classifying risks into categories which should be used as a guideline for making this preliminary determination. In addition, discuss suggested improvements identified during testing.

(5) Summarize deficiencies, shortcomings, and suggested improvements, identified in paragraph 2.x.5 of the report subtest, along with "corrected previous deficiencies and

shortcomings", in Appendix C of the test report. This appendix is a prime source in the development of the IAR and the system and software assessments developed by the IA. Suggested improvements identified here can be a key element in the user's reexamination of system requirements and the revision of subsequent software requirements, design, test and code.

6.2.4 Briefing Reports. Because of compressed program schedules, it may be necessary to prepare preliminary briefing reports, with updates and formal reports to follow. Prepare briefing reports to report the results of testing at TRRs, IPRs, and Design Reviews. The Software Test Results Summary, Figure 10, provides an excellent overview of an overall assessment of software performance and EOT. A summary of problem areas and areas not tested should also be provided. To provide balanced reporting, highlight areas where the software has worked well.

Back to top

APPENDIX A. BACKGROUND.

The requirement for this TOP was established by the TECOM Software Testing Study (Ref 40). The study provided for review of the current testing policy and procedures within TECOM for testing computer software embedded in weapon systems. It also documented that software testing procedures differed among the TECOM software test centers. The study identified the need for the establishment of procedures for a requirements-oriented approach to software testing at all TECOM test centers. The original software testing TOP (Ref 41) was issued in 1977 and does not provide step-by-step procedures for the testing of software.

The study provided for review of procedures used to test software at the three TECOM test centers designated as having software missions: WSMR, U.S. Army Electronic Proving Ground, and Combat Systems Test Activity. As stated in the study, the current TECOM software testing policy is to test materiel at the system level, with further testing of components (such as software) on an as-needed basis. It was determined that the requirements-oriented method for software testing developed by WSMR provides a systematic approach for examining system compliance to requirements and should be implemented in TECOM policy and procedures.

Back to top

APPENDIX B. GLOSSARY.

Acquisition category (ACAT)
DoD programs are designated in the four categories, ACAT I-IV, based on development risk, urgency of need, Congressional interest, Joint Service involvement, and resource requirements.

Allocated Baseline
The initial, approved performance oriented specification governing the development of configuration items (CIs) that are a part of a higher level CI, in which each specification (a) defines the functional characteristics that are allocated from those of the higher level CI, (b) establishes the tests required to demonstrate achievement of its allocated functional characteristics, (c) delineates necessary interface requirements with other associated configuration items, and (d) establishes design constraints, if any, such as component standardization, use of inventory items and integrated logistics support requirements.

Baseline
A configuration identification document or a set of documents formally designated and fixed at a specific time during a configuration item's life cycle. Baseline, plus approved changes from those baselines constitute the current configuration (reference MIL-STD-973).

CASE Tools
Systems for building systems; automates the requirements analysis, design and development process.

Computer Operation Manual (COM)
The COM provides information needed to operate a given computer and its peripheral equipment. This manual focuses on the computer itself, not on particular software that will run on the computer. The COM is intended for newly developed computers, special purpose computers, or other computers for which commercial or other operation manuals are not readily available.

Computer Programming Manual (CPM)
The CPM provides information needed by a programmer to understand how to program a given computer. This manual focuses on the computer itself, not on particular software that will run on the computer. A CPM is intended for newly developed computers, special-purpose computers, or other computers for which commercial or other programming manuals are not readily available.

Computer Resource Management Plan (CRMP) or Computer Resource Life Cycle Management Plan (CRLCMP)
The primary planning document to be used by all decision levels for assessing the adequacy of the overall computer resources management efforts throughout the system life cycle. Used only for AR 70-1 developed systems (Reference AR 70-1, DoDI 5000.2).

Computer Resource Working Group (CRWG)
Established by the Materiel Developer after Milestone I for each AR 70-1 system to aid in the management of system computer resources. The CRWG assists in ensuring compliance with policy, procedures, plans and standards established for computer resources.

Computer Software Component (CSC)
A distinct part of a CSCI. CSCs may be further decomposed into other CSCs and Computer Software Units (CSUs).

Computer Software Configuration Item (CSCI)
An aggregation of software or any of its discrete portions, which satisfies an end use function and is designated by the Government for Configuration Management.

Computer Software Unit (CSU)
An element specified in the design of a CSC that is separately testable.

Configuration Item (CI)
An aggregation of hardware/software, or any of its discrete functions, which satisfy an end use function and is designated by the Government for configuration management (reference MIL-STD-973).

Configuration Management
A discipline applying technical and administrative direction and surveillance to (a) identify and document the functional and physical characteristics of a configuration item, (b) control changes to those characteristics, and (c) record and report change processing and implementation status.

Critical Design Review (CDR)
A review conducted to determine that the detail design satisfies the performance and engineering requirements of the development specifications; to establish the detail design compatibility among the item and other equipment, facilities, computer programs, and personnel; to assess producibility and risk areas; and to review the preliminary product specification.

Customer Test
Test requested by the Program Manager (PM) which does not support an evaluation/assessment.

Database Design Description (DBDD)

A DBDD describes the design of a database, that is, a collection of related data stored in one or more computerized files in a manner that can be accessed by users or computer programs via a database management system (DBMS). It can also describe the software units used to access or manipulate data. A DBDD is used as the basis for implementing the database and related software units. It provides the acquirer visibility into the design and provides information needed for software support.

Detailed Test Plan (DTP)
The DTP provides explicit instructions for the conduct of tests and subtests. It is derived from and implements the TDP. The DTP governs test control, data collection, data analysis, and the necessary administrative aspects of a test program. There may be one or several DTPs depending on the complexity of the program and the number of test sites and test organizations involved in providing data. The DTP must be coordinated with appropriate IAs and with other TIWG members to ensure that the evaluation/assessment reflects the requirements of the TEMP and TDPs.

Developer Tests
Testing, modeling, and experimentation conducted by the system developer. Formal tests normally involve system level integration and certification by the developer with formal government monitoring. Informal tests involve lower level code and unit development with internal integration between system elements. Experimentation includes a wide variety of tests, models, development techniques and simulations used to validate design concepts and theories.

Developmental Test (DT)
Tests usually conducted by an organization independent of the developer(s) in order to validate total system conformance to technical and functional specifications and ensure the system is ready for formal or limited user testing. Formal tests focus primary on total systems integration. Limited tests are used primarily with priority 1, 2, or 3 changes during post-development software support (PDSS) and focus on the specific changes being made.

Engineering Change Proposal (ECP)
A proposed engineering change and the documentation by which the change is described, justified, and submitted to the government for approval or disapproval.

Engineering Change Proposal - Software (ECP-S)
A term which includes both a proposed engineering change and the documentation by which the change is described. DA Form 5005-R is used to document proposed changes to software baselines and associated baseline documentation.

Firmware
A combination of a hardware device and computer instructions or computer data that resides as read-only software on the hardware device. The software cannot be readily modified under program control (reference MIL-STD-498).

Firmware Support Manual (FSM)
The FSM provides the information necessary to program and reprogram the firmware devices of a system. It applies to read only memories (ROMs), Programmable ROMs (PROMs), and Erasable PROMs (EPROMs), and other firmware devices. The FSM describes the firmware devices, load software into the firmware devices, verify the load process, and mark the loaded firmware devices.

Formal Qualification Test (FQT)
A process that allows the contracting agency to determine whether a configuration item (subsystem) complies with the allocated requirements for that item.

Functional Baseline
The initial, approved technical documentation for a configuration item (CI) which prescribes (a) all necessary functional characteristics, (b) the tests required to demonstrate achievement of specified functional characteristics, (c) the necessary interface characteristics with associated CIs, (d) the CIs key functional characteristics and its key lower level CIs, if any, and (e) design constraints, such as, envelope dimensions, component standardization, use of inventory items, and integrated logistics support policies.

Functional Capability
A capability that specifies a function that a system or system component must be capable of performing.

Functional Configuration Audit (FCA)
A formal audit to validate that the development of a CI has been completed satisfactorily and that the CI has achieved the performance and functional characteristics specified in the functional or allocated configuration identification. In addition, the complete operation and support documents are required.

Hardware Monitor
A hardware monitor consists of sensors (probes), control logic, accumulators, and recording units for computing or timing events, e.g., CPU busy-wait, channel busy, disk-seek units and times.

Independent Assessment Plan/Independent Evaluation Plan (IAP/IEP)
The IAP/IEP details all aspects of developmental evaluation responsibilities relative to the system throughout its acquisition cycle. The IAP/IEP supports the TEMP by addressing the

issues for testing; describing evaluation of issues which require data from sources other than tests; stating the technical parameters; identifying data sources; providing the approach to the evaluation; and identifying program constraints.

Independent Assessment Report/Independent Evaluation Report (IAR/IER)
The IAR/IER provides the independent evaluation of the systems and is based on test data, reports, studies, simulations, and other appropriate sources. It also contains the independent assessor's assessment of the parameters, conclusions, and position on the future capability of the system to fulfill the approved requirements. The IAR/IER will contain an assessment of the adequacy of testing and the need for additional testing, and will identify program constraints and their impact on the evaluation.

Independent Verification and Validation (IV&V)
Verification and validation performed by a contractor or Government agency that is not responsible for developing the product or performing the activity being evaluated. IV&V is tailored to the risk associated with the system. IV&V is conducted separately from the software development activities.

In-Process Review (IPR)
Review body for ACAT III and IV programs. Convened at each formal milestone and at other critical points to evaluate status and make recommendations to the decision authority.

Integration Testing (IT)
An orderly progression of testing in which software elements, hardware elements, or both are combined and tested until the entire system has been integrated.

Interface
The process of passing data between systems.

Interface Design Description (IDD)
The IDD describes the interface characteristics of one or more systems, subsystems, Hardware Configuration Items (HWCIs), CSCIs, manual operations, or other system components. An IDD may describe any number of interfaces. The IDD can be used to supplement the system/subsystem design description (SSDD) and software design description (SDD) in describing the design of systems and CSCIs. The IDD and its companion IRS serve to communicate and control interface design decisions.

Interface Requirements Specification (IRS)
The IRS specifies the requirements imposed on one or more systems, subsystems, HWCIs, CSCIs, manual operations, or other system components to achieve one or more interfaces among these entities. An IRS can cover any number of interfaces. The IRS can be used to supplement the SSS and SRS as the basis for design and qualification testing of systems and CSCIs.

Interoperability
The ability of systems, units, or forces to provide services to accept services from other systems, units or forces and to use the services to enable them to operate effectively together.

Metrics
A quantitative value, procedure, methodology, and/or technique which allows one the ability to measure various aspects and characteristics of software.

Nondevelopmental Item (NDI)
Deliverable items that are not developed under the contract but are provided by the contractor, the government or a third party. NDI may be referred to as reusable, government furnished, or commonly available software, hardware or total systems, depending on the source.

Operational Concept Document (OCD)
The OCD describes a proposed system in terms of the user needs it will fulfill, its relationship to existing systems or procedures, and the ways it will be used. An OCD is used to obtain consensus among acquirer, developer support, and user agencies on the operational concept of a proposed system. Depending on its use, an OCD may focus on communicating the user's needs to the developer or the developer's ideas to the user and other interested parties. The term "system" may be interpreted to apply to a portion of a system.

Operational Test (OT) or User Test
A system-level test performed by at test activity independent of the developer or the PM. The objective of the OT is to test the entirety of the system and is performed by users in an operational environment.

OPTEC Instrumentation Database (OIDB)
The OIDB is an automated inventory program that includes all ITTS assets owned and operated by OPTEC test activities. It identifies instrumentation by category, class/subclass, quantity, and location.

Physical Configuration Audit (PCA)
The technical examination of a designated CI to verify that the CI "as-built" conforms to the technical documentation which defined the CI.

Preliminary Design Review (PDR)
This review is conducted for each CI or aggregate of CIs to (a) evaluate the progress, technical adequacy, and risk resolution (on a technical, cost, and schedule basis) of the selected design approach, (b) determine its compatibility with performance and engineering specialty requirements of the Hardware CI (HWCI) development specification, and (c) evaluate the degree of definition and assess the technical risk associated with the selected manufacturing

compatibility of the physical and functional interfaces among the CI and other items of equipment, facilities, computer software, and personnel. For CSCIs, this review focuses on (a) the evaluation of the progress, consistency, and technical adequacy of the selected top-level approach, (b) compatibility between software requirements and preliminary design, and (c) on the preliminary version of the operation and support documents.

Qualification Testing
Testing performed to demonstrate to the acquirer that a CSCI or a system meets its specified requirements.

Regression Testing
Testing of a computer program to assure correct performance after changes are made to code that previously performed correctly, or the testing or retesting those areas/aspects of a system which will or could be affected by a change.

Required Operational Characteristics
Qualitative and quantitative system performance parameters, proposed by the user and approved by the Army, that are primary indicators of a system's capability to accomplish its mission (operational effectiveness) and to be supported (operational suitability). Required operational characteristics are usually tested and evaluated by operational testing and evaluation to ascertain achievement of approved goals and thresholds for these characteristics.

Required Technical Characteristics
Quantitative system performance parameters approved by the Army management that are selected as primary indicators of technical achievement. These might not be direct measures of, but always should relate to a system's capability to perform its required mission function and to be supported. Required technical characteristics usually are tested and evaluated to ascertain approval goals and thresholds for these characteristics.

Requirements Trace
Assuring requirements flow from the user specifications through design and implementation of the product.

Software Center Operator Manual (SCOM)
The SCOM provides personnel in a computer center or other centralized or networked software installation information on how to install and operate a software system. A SCOM is developed for software systems that will be installed in a computer center or other centralized software installation, with users accessing the system via terminals or personal computers or submitting and receiving inputs and outputs in batch or interactive mode.

Software Design Description (SDD)

The SDD describes the design of a CSCI. It describes the CSCI-wide design decisions, the CSCI architecture design, and the detailed design needed to implement the software. The SDD may be supplemented by the IDDs and DBDDs. Design pertaining to interfaces may be presented in the SDD or IDDs. Design pertaining to databases may be presented in the SDD or in DBDDs. The SDD, with its associated IDDs and DBDDs, is used as the basis for implementing the software. It provides the acquirer visibility into the design and provides information needed for software support.

Software Development File (SDF)
A repository for material pertinent to the development of a particular body of software. Contents typically include (either directly or by reference) considerations, rationale, and constraints related to requirements analysis, design, and implementation; developer-internal test information; and schedule and status information.

Software Development Plan (SDP)
The SDP describes a developer's plans for conducting a software development effort. The term "software development" in this DID is meant to include new development, modification, reuse, reengineering, maintenance, and all other activities resulting in software products. The SDP is organized to correspond to MIL-STD-498. It provides the acquirer insight into, and a tool for monitoring, the processes to be followed for software development, the methods to be used, the approach to be followed for each activity, and project schedules, organizations, and resources.

Software Input/Output Manual (SIOM)
The SIOM tells a user how to access, submit inputs to, and interpret output from, a batch or interactive software system that is run by personnel in a computer center or other centralized or networked software installation. A SIOM is developed for software systems that will be installed in a computer center or other centralized or networked software installation. A SIOM is developed for software systems that will be installed in a computer center or other centralized or networked software installation, with users accessing the system via terminals or personal computers or submitting and receiving inputs and outputs in batch mode.

Software Installation Plan (SIP)
The SIP is a plan for installing software at user sites, including preparations, user training, and conversion from existing systems. A SIP is developed when developer will be involved in the installation of software at user sites and when the installation process will be sufficiently complex to require a documented plan. For software embedded in a hardware-software system, a fielding or deployment plan for the hardware-software system may make a separate SIP unnecessary.

Software Monitor
A software monitor is program, or code, resident in the computer to monitor and collect internal system data.

Software Product Specification (SPS)
The SPS contains or references the executable software, source files, and software support information, including "as built" design information and compilation, build, and modification procedures, for a CSCI. The SPS can be used to order the executable software and/or source files for a CSCI and is the primary software support document for the CSCI. Note: Different organizations have different policies for ordering delivery of software. These policies should be determined before applying this DID.

Software Requirements Specification (SRS)
The SRS specifies the requirements for a CSCI and the methods to be used to ensure that each requirement has been met. Requirements pertaining to the CSCI's external interfaces may be presented in the SRS or in one or more IRSs reference from the SRS. The SRS, possibly supplemented by IRSs, is used as the basis for design and qualification testing of a CSCI.

Software Specification Review (SSR)
A review of the finalized software configuration item requirements and operational concept. The SSR is conducted when software requirements have been sufficiently defined to evaluate the developer's responsiveness to and interpretation of the system, segment or prime item (reference MIL-STD-1521B). The SSR is held to determine the adequacy of the software requirements allocation efforts and the adequacy of the software specifications.

Software Test Description (STD)
The STD describes the test preparations, test cases, and test procedures to be used to perform qualification testing of a CSCI or a software system or subsystem. The STD enables the acquirer to assess the adequacy of the qualification testing to be performed.

Software Test Plan (STP)
The STP describes plans for qualification testing of CSCIs and software systems. It describes the software test environment to be used for the testing, identifies the tests to be performed, and provides schedules for test activities. There is usually a single STP for a project. The STP enables the acquirer to assess the adequacy of planning for CSCI and, if applicable, software system qualification testing.

Software Test Report (STR)
The STR is a record of the qualification testing performed on a CSCI, a software system or subsystem, or other software-related item. An STR enables the acquirer to assess the testing and its results.

Software Transition Plan (STrP)
The STrP identifies the hardware, software, and other resources needed for life cycle support of deliverable software and describes the developer's plans for transitioning deliverable items to the

support agency. The STrP is developed if the software support concept calls for transition of responsibility from the developer to a separate support agency. The STrP may also be used by the acquirer for updating the Computer Resources Life Cycle Management Plan.

Software User Manual (SUM)
The SUM tells a hands-on user how to install and use a CSCI, a group of related CSCIs, or a software system or subsystem. It may also cover a particular aspect of software operation, such as instructions for a particular position or task. A SUM is developed for software that is run by the user and has a user interface requiring on-line user input or interpretation of displayed output. If the software is embedded in a hardware-software system, user manuals or operation procedures for that system may make separate SUMs unnecessary.

Software Version Description (SVD)
The SVD identifies and describes a software version consisting of one or more CSCIs. It is used to release, track, and control software versions. The term "version" may be applied to the initial release of the software, to a subsequent release of that software, or to one of multiple forms of the software released at approximately the same time (for example, to different sites).

Stress Test
A test that exercises code up to, including and beyond all stated limits in order to exercise all aspects of the system (i.e., to include hardware, software and communications). The purpose is to ensure that response times and storage capacities meet requirements.

System Design Review (SDR)
This review is conducted to evaluate the optimization, correlation, completeness, and risks associated with the allocated technical requirements. Also included is a summary review of the system engineering process which produced the allocated technical requirements and of the engineering planning for the next phase of effort. This review is conducted when the system definition effort has proceeded to the point where system characteristics are defined and the CIs are identified.

System Requirements Review (SRR)
The objective of this review is to ascertain the adequacy of efforts in defining system requirements. It is conducted when a significant portion of the system functional requirements has been established.

System/Subsystem Design Description (SSDD)
The SSDD describes the system- or subsystem-wide design and the architecture design of a system or subsystem. The SSDD may be supplemented by IDDs and DBDDs. Design pertaining to interfaces may be presented in the SSDD or IDDs. Design pertaining to databases may be presented in the SSDD or DBDDs. The SSDD, with its associated IDDs and DBDDs, is used as the basis for further system development. Throughout this DID, the term "system" may

be interpreted to mean "subsystem" as applicable. The resulting document should be titled System Design Description or Subsystem Design Description.

System/Subsystem Specification (SSS)
The SSS specifies the requirements for a system or a subsystem and the methods to be used to ensure that each requirement has been met. Requirements pertaining to the system or subsystems external interfaces may be presented in the SSS or in one or more IRSs referenced from the SSS. The SSS, possibly supplemented by IRSs, is used as the basis for design and qualification testing of a system or subsystem. Throughout this DID, the term "system" may be interpreted to mean "subsystem" as applicable. The resulting document should be titled System Specification or Subsystem Specification.

System Testing
The process of testing an integrated hardware and software system to verify that the system and software meet their specified requirements.

System Test Description (STD)
The STD contains the test cases and test procedures necessary to perform testing of a CSCI, an integrated group of CSCIs or CSCIs/HWCIs, or a software system or segment. The STD enables the government to assess the adequacy of the test cases and test procedures to be performed.

Test and Evaluation Master Plan (TEMP)
The key management tool for control of the integration of all T&E requirements for each acquisition effort and is used by decision review bodies (reference DoD 5000.2-M).

Test Bed
A test environment containing the hardware, instrumentation tools, simulators, and other support software necessary for testing a system or system component.

Test Design Plan (TDP)
The TDP is responsive to the technical parameters. It includes a complete test design; a description of required tests; the conditions under which the system will be tested; and a statement of test criteria and test methodology. The TDP also specifies data requirements and includes plans for data collection and analysis. Developmental TDPs will address Government, and may address contractor, testing/modeling/simulation efforts, if appropriate. For assessed programs, the TDP is included in the IAP.

Test Facilities (TESTFACS) Register
TESTFACS is an automated program which lists and describes the Army's test capabilities. The TESTFACS database lists and describes major test facilities and major instrumentation and test equipment with values of $75,000 or more, including targets and threat simulators/actuals.

Test Hooks
Software logic which is integrated into the system in order to facilitate extraction of data to support test and analysis.

Test Integration Working Group (TIWG)
Established by the program sponsor upon receipt of the draft ORD or MNS. This is the primary group which facilitates integration of T&E requirements and prepares the TEMP.

Test Readiness Review (TRR)
A review conducted for each CSCI to determine whether the software procedures are complete and to assure that the contractor is prepared for formal CSCI testing.

Test Report (TR) and Expanded Test Report (ETR)
The TR is a formal document of record which reports the data and information obtained from the conduct of test and describes conditions which actually prevailed during test execution and data collection. For ACAT IV systems which do not have DOT&E oversight, an ETR may be written. An ETR is a test report with evaluative content which is endorsed by the evaluator in lieu of a separate evaluation.

Unit or Code Test
The lowest level developer test. The process of testing each unit in isolation.

Validation
The process of evaluating software to determine compliance with specified requirements.

Verification
The process of evaluating the products of a given software development activity to determine correctness and consistency with respect to the products and standards provided as an input to that activity.

Back to top

APPENDIX C. ABBREVIATIONS.

ACAT	-	acquisition category
AIS	-	automated information systems
AMC	-	U.S. Army Materiel Command
AMSAA	-	U.S. Army Materiel Systems Analysis Activity
AR	-	army regulation
ATTN	-	attention
BT	-	benign tracking
C	-	complete
CASE	-	computer-aided software engineering
CDR	-	critical design review
CDRL	-	contract data requirements list
CELLTYP	-	cell type
CI	-	configuration item
COM	-	computer operation manual
COT	-	completeness of test
CPM	-	computer programming manual
CPU	-	computer processing unit
CRLCMP	-	computer resource life cycle management plan
CRMP	-	computer resources management plan
CRU	-	computer resource utilization

CRWG	-	computer resource working group
CSC	-	computer software component
CSCI	-	computer software configuration item
CTOL	-	combined threat order list
CTOLI	-	combined threat order list indicator
CUM	-	cumulative
D	-	display control, direct analysis
DA	-	U.S. Department of the Army
DBDD	-	database design description
DBMS	-	database management system
DI	-	data item
DID	-	data item description
DoD	-	U.S. Department of Defense
DoDI	-	Department of Defense Instruction
DOT&E	-	Director, Operational Test and Evaluation
DT	-	developmental test
DTP	-	detailed test plan
E	-	exercised
ECP	-	engineering change proposal
ECP-S	-	engineering change proposal - software
ED	-	engagement decision

EOT	-	extent of test
EPROM	-	erasable programmable read only memory
ETR	-	expanded test report
F	-	firing doctrine, failed
FCA	-	functional configuration audit
FIG	-	figure
FIT	-	formal integration test
FSM	-	firmware support manual
FQT	-	formal qualification test
HDBK	-	handbook
HR	-	hour
HWCI	-	hardware configuration item
I	-	indirect analysis
IA	-	independent assessor
IAP	-	independent assessment plan
IAR	-	independent assessment report
ID	-	indentification
IDA	-	Institute for Defense Analysis
IDD	-	interface design description
IEP	-	independent evaluation plan
IER	-	independent evaluation report

I/O	-	input/output
IPR	-	in-process review
IRS	-	interface requirements specification
ITTS	-	instrumentation, targets and threat simulators
IV&V	-	independent verification and validation
L	-	limited
M	-	met
MCCR	-	mission-critical computer resource
MCI	-	major cycle interval
MIL	-	military
MIN	-	minute
MNS	-	mission needs statement
N	-	not tested
NA	-	not applicable
NDI	-	nondevelopmental item
NM	-	not met
NT	-	not tested
O	-	observed
OCD	-	operational concept document
OIDB	-	OPTEC Instrumentation Database
OPTEC	-	U.S. Army Operational Test and Evaluation Command

ORD	-	operational requirements document
OT	-	operational test
P	-	partial, passed
PAM	-	pamphlet
PCA	-	physical configuration audit
PDR	-	preliminary design review
PDSS	-	post deployment software support
PM	-	partially met, program manager
PROM	-	programmable read only memory
RAM	-	random access memory
REF	-	reference
REG	-	regulation
ROM	-	read only memory
S	-	significant
SDD	-	software design description
SDF	-	software development folder
SDP	-	software development plan
SDR	-	system design review
SEC	-	second
SCOM	-	software center operator manual
SIOM	-	software input/output manual

SIP	-	software installation plan
SOW	-	statement of work
SPP	-	software support plan
SPR-		software problem report
SPS	-	software product specification
SRR	-	system requirements review
SRS-		software requirements specification
SSDD	-	system/subsystem design description
SSR-		software specification review
SSS	-	system/subsystem specification
STD	-	software test description, standard
STP	-	software test plan
STR	-	software test report
STrP	-	software transition plan
SUM	-	software user manual
SUT	-	system under test
SVD	-	software version description
SWG	-	software working group
T&E	-	test and evaluation
TBD	-	to be determined
TC	-	test coverage

TCM	-	test coverage matrix
TDP	-	test design plan
TECOM	-	U.S. Army Test and Evaluation Command
TEMP	-	test and evaluation master plan
TESTFACS	-	test facilities
TGTN	-	target number
TIR	-	test incident report
TIWG	-	test integration working group
TOP	-	test operations procedure
TR	-	test report
TRR	-	test readiness review
WCV	-	weapon control volume
WCVCI	-	weapon control volume constraint indicator
WCVID	-	weapon control volume identification
WCVEE	-	weapon control volume engageability evaluation
WCVS	-	weapon control volume status
WSMR	-	U.S. Army White Sands Missile Range

Back to top

APPENDIX D. REFERENCES.

1. Test and Evaluation, Test and Evaluation Policy, AR 73-1, 15 Oct 92.

2. Test and Evaluation in Support of System Acquisition, DA PAM 73-1 (DRAFT), 27 Mar 95.

3. Research, Development, and Acquisition, Army Acquisition Policy, AR 70-1, 31 Mar 93.

4. Military Standard, Software Development and Documentation, MIL-STD-498, 5 Dec 94.

5. Military Standard, Defense System Software Development, DoD-STD-2167A, 29 Feb 88.

6. Military Standard, DoD Automated Information Systems (AIS) Documentation Standards, DoD-STD-7935A, 31 Oct 88.

7. Military Standard, Software Product Standards, DoD-STD-1703 (NS), 12 Feb 87.

8. Military Standard, Configuration Management, MIL-STD-973, 1 Dec 92 (supercedes MIL-STD-480, MIL-STD-481, MIL-STD-482, MIL-STD-483, MIL-STD-1456, and Appendices G, H, and I of MIL-STD-1521.

9. Military Handbook, Acquisition Streamlining, MIL-HDBK-248B, 9 Feb 89.

10. WSMR Range Customers Handbook, White Sands Missile Range, 1993.

11. Software Test Tool Support, Software Technology Support Center, 1991.

12. Examination of Selected Commercial Software Testing Tools, Institute for Defense Analyses (IDA), IDA Paper P-2628, Oct 91.

13. Acquisition Management Systems and Data Requirements Control List, AMC Reg 70-16A, 5 Apr 90.

14. Test and Evaluation, Developmental Test and Assessment Guide, TECOM PAM 73-1, 30 Sep 93.

15. Defense Acquisition Management Documentation and Reports, DoD 5000.2-M, Feb 91.

16. Defense Acquisition Management Policies and Procedures, DoD 5000.2, 23 Feb 91.

17. Software Development Plan, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81427 (supercedes DID DI-MCCR-80030A)

18. Software Test Plan, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81438 (supercedes DID DI-MCCR-80014A)

19. Software Transition Plan, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81429 (supercedes DID DI-MCCR-80024A).

20. Software Installation Plan, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81428 (supercedes DID DI-IPSC-80699).

21. Operational Concept Document, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81430 (supercedes DID DI-IPSC-80689).

22. System/Subsystem Specification, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81431 (supercedes DID DI-CMAN-80008A).

23. System/Subsystem Design Document, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81432 (supercedes DID DI-CMAN-80534).

24. Software Requirements Specification, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81433 (supercedes DID DI-MCCR-80025A).

25. Interface Requirements Specification, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81434 (supercedes DID DI-MCCR-80026A).

26. Software Design Description, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81435 (supercedes DID DI-MCCR-80012A).

27. Database Design Description, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81437 (supercedes DID DI-IPSC-80692).

28. Interface Design Description, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81436 (supercedes DID DI-MCCR-80027A).

29. Software Test Description, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81439 (supercedes DID DI-MCCR-80015A).

30. Software Test Report, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81440 (supercedes DID DI-MCCR-80017A).

31. Software User Manual, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81443 (supercedes DID DI-MCCR-80019A).

32. Software Center Operator Manual, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81444 (supercedes DID DI-IPSC-80695).

33. Software Input/Output Manual, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81445 (supercedes DID DI-IPSC-80693).

34. Computer Operation Manual, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81446 (supercedes DID DI-MCCR-80018A).

35. Computer Programming Manual, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81447 (supercedes DID DI-MCCR-80021A).

36. Firmware Support Manual, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81448 (supercedes DID DI-MCCR-80022A).

37. Software Product Specification, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81441 (supercedes DID DI-MCCR-80029A).

38. Version Description Document, MIL-STD-498, 5 Dec 94, DID DI-IPSC-81442 (supercedes DID DI-MCCR-80013A).

39. System Safety Program Requirements, MIL-STD-882, 19 Jan 93.

40. Memorandum, TECOM, AMSTE-TA-S, subject: Software Testing Study, 1 Sep 89.

41. Software Testing, TECOM TOP 1-1-056, 15 Nov 77.

Back to top

Back to top