Enclosure to:
SSD/PL-95-07 I 1

DRAFT

SAFETY AND FITNESS ELECTRONIC RECORDS (SAFER) SYSTEM

MASTER TEST PLAN

DRAFT

Safety and Fitness Electronic Records (SAFER) System Master Test Plan

Table of Contents

DRAFT

Safety and Fitness Electronic Records (SAFER) System Master Test Plan

1. Purpose

The purpose of this plan is to establish a formal set of guidelines and activities to be adhered to and performed by JHU/APL and the developer to ensure that the SAFER System has been tested successfully and is fully compliant with the SAFER System requirements.

The initial release of this document, submitted in draft form, provides a general framework for establishing the testing environment and provides general guidelines for performing unit, integration, system, and acceptance testing of the SAFER System. Several draft versions of this document will be issued, following review and comment by the developer, as details regarding the software design evolve.

2. Reference Documents

Software Development and Documentation, Military Standard, MIL-STD-498, 5 December 1994, AMSC NO. N7069.

IEEE Standard for Software Quality Assurance Plans, ANSI/IEEE Std 730-1984, June 14, 1984

POR-5804, Trident II Data processing Plan, Vol. 1, Johns Hopkins University/Applied Physics Laboratory, January 1991

3. Definitions

The information presented below represents a definition of terms used throughout this document.

- Approval. Written notification by an authorized representative of the acquirer that a developer's plans, design, or other aspects of the project appear to be sound and can be used as the basis for further work. Such approval does not shift responsibility from the developer to meet contractual requirements.

- Computer program. A combination of computer instructions and data definitions that enable computer hardware to perform computational or control functions.

- Computer Software Configuration Item (CSCI). An aggregation of software that satisfies an end use function and is designated for separate configuration management by the acquirer. CSCIs are selected based on tradeoffs among software function, size, host or target computers, developer, support concept, plans for reuse, criticality, interface considerations, need to be separately documented and controlled, and other factors.

- Developer. An organization that develops software products ("develops" may include new development, modification, reuse, reengineering, maintenance, or any other activity that results in software products.

- Pass/Fail Criteria. Decision rules used to determine whether a software item or a software feature passes or fails a test.

- 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 Feature. A distinguishing characteristic of a software item, e.g., performance, portability, functionality.

- Software Item. Source code, object code, job control code, control data, or a collection of these items.

- Software test environment. The facilities, hardware, software, firmware, procedures, and documentation needed to perform qualification, and possibly other, testing of software. Elements may include but are not limited to simulators, code analyzers, test case generators, and path analyzers, and may also include elements used in the software engineering environment.

- Software unit. An element in the design of a CSCI; for example, a major subdivision of a CSCI, a component of that subdivision, a class, object, module, function, routine, or database. Software units may occur at different levels of a hierarchy and may consist of other software units. Software units in the design may or may not have a one-to-one relationship with the code and data entities (routines, procedures, databases, data files, etc.) that implement them or with the computer files containing those entities.

- Test item. A software item which is an object of testing.

- Test Log. A chronological record of relevant details about the execution of the tests.

- Test Summary Report. A document or set of documents summarizing testing activities and results.

Testing. The process of analyzing a software item to detect the differences between existing and required conditions, i.e., bugs, and to evaluate the features of the software item.

4. Scope

This test plan covers general guidelines for performing unit, integration, system, and acceptance testing of the SAFER System. These guidelines will be expanded to include specific test design, case, and procedure specifications as details regarding the software design evolve. Testing will ultimately cover operator and user procedures, as well as programs and processing control. In addition to comprehensively testing multi-process functionality including multi-threading, inter-process communications and multi-processor utilization, external interfaces, security, recovery and performance will also be evaluated.

5. Features to be Tested

The SAFER System, as specified in the SAFER Physical Architecture Document, is partitioned into seven CSCIs. These are:

- Input Message Handler
- Administrative Manager
- Subscriber Processor
- External Request Processor
- Safety Data Manager
- Output Message Handler
- OPCON Manager

The features of each CSCI will be fully defined and documented in the SAFER Detailed Software Design Document. In subsequent versions of the Master Test Plan, the specific features of each CSCI and their inter-relationships will be explicitly identified for testing purposes.

6. Features Not to be Tested

Features of the SAFER System which are not to be tested initially are TBD.

7. Approach

7.1 Program Phases

The SAFER Project will be conducted in two phases:

SAFER 96 Development & Test
SAFER 97 Development, Test & Production

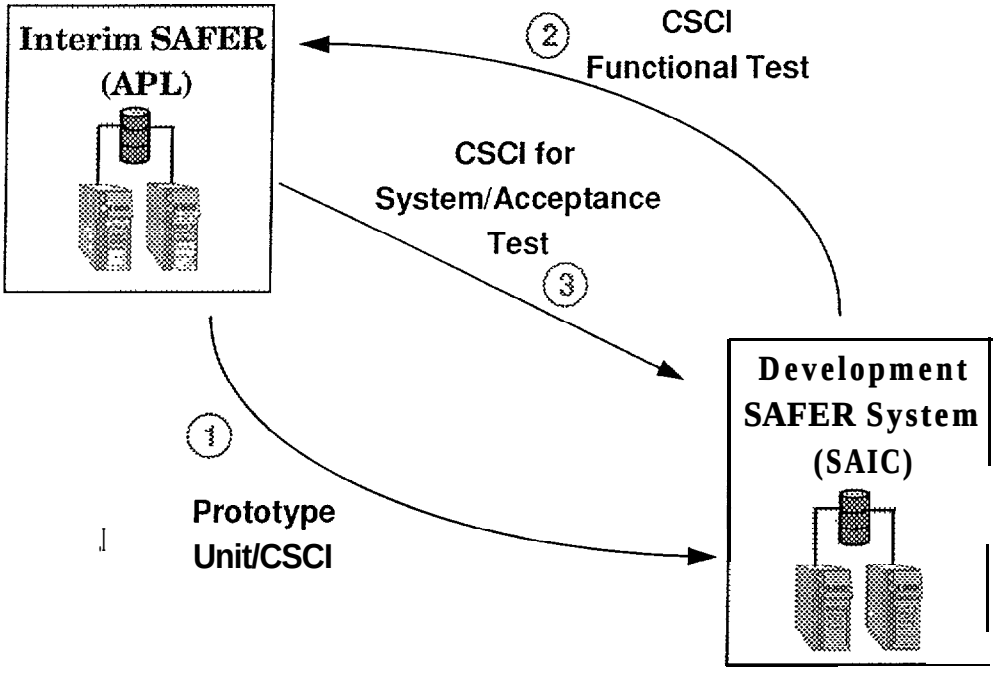
In the SAFER 96 Development & Test Phase, the SAFER 96 System will be designed, developed and tested. Deployment and operation and maintenance activities will be initiated in December of 1996. Deployment will be completed, i.e., the System deployed to support no fewer than 200 MCSAP sites, by June 1997. A preliminary system analysis and design of the second build of the SAFER System, termed SAFER 97, will also be performed.

In the SAFER 97 Development, Test & Production Phase, the SAFER 97 System will be developed based on additional requirements identified during the development effort of the SAFER 96 and 100/200 MCSAP Site Projects while the SAFER 96 System supports production operations. The SAFER 97 System will be deployed to support the 100/200 MCSAP Site Project no later than December 1997.

7.2 General Approach

Software development and testing shall be a collaborative effort between the JHU/APL and the developer. A graphical depiction of the 1996 development and test approach is shown in Figure 1.

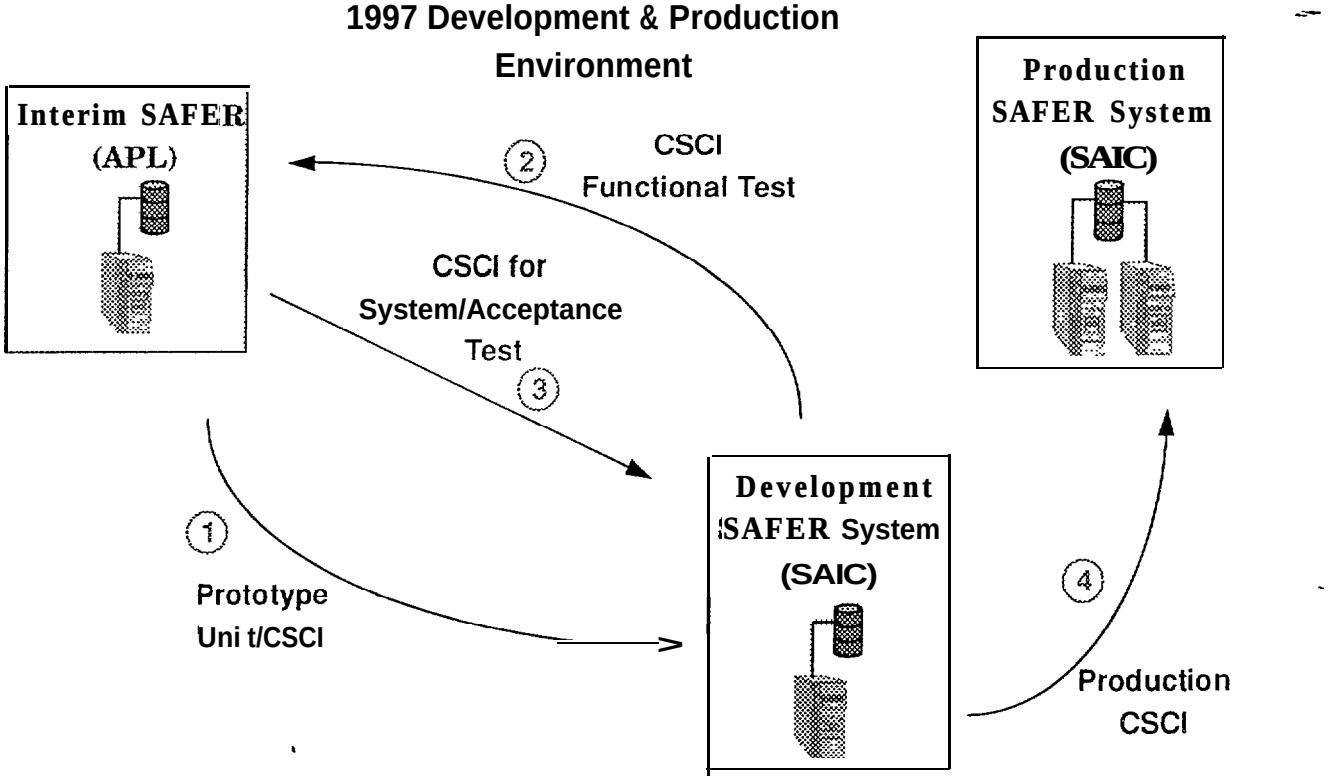
Figure 1. SAFER 96 Software Development & Testing Approach



During the 1996 development and test cycle, JHU/APL shall be responsible for developing and maintaining an interim version of the SAFER System at its facility. This system is henceforth referred to as the Interim SAFER System and shall be used for software prototype development and testing. Prototype software shall be developed by JHU/APL in accordance with the specifications of the SAFER Logical and Physical Architecture, and supplied to the developer.

The developer shall be responsible for developing and maintaining a development version of the production SAFER System, henceforth referred to as the SAFER System. The developer shall review and modify, where applicable, prototype software supplied by JHU/APL and incorporate it into the SAFER System. The developer shall provide JHU/APL with builds (releases) of the production software, at the CSCI level or above, to be installed on the Interim SAFER System, replacing prototype code, for testing in accordance with the Master Test Plan. Concurrent testing shall also be conducted at the developer's facility. Software deficiencies, detected during testing, shall be corrected by the developer and the corrected code re-installed on the Interim SAFER System for further testing. This iterative cycle shall continue until all detected deficiencies have been eliminated.

Figure 2. SAFER Software Development & Testing Approach



During the 1997 development and test cycle, production operations must also be supported at the SAIC facility. This environment is graphically depicted in Figure 2. At the start of the 1997 development and test cycle, one of the Interim SAFER processors will be migrated to SAIC to serve as their development processor while their original development system will become the production SAFER System.

APL and the developer will use the system documentation to prepare all test design, case, and procedure specifications to support unit, integration, system, and acceptance testing. This approach will verify the accuracy and comprehensives of the information in the documentation in those areas covered by the tests.

7.3 Unit Testing

Unit testing means ensuring that all aspects of each software unit's detailed design are comprehensively tested.

7.3.1 Preparing for unit testing

The developer shall establish test cases (in terms of inputs, expected results, and evaluation criteria), test procedures, and test data for testing the software corresponding to each software unit. The test cases shall cover all aspects of the unit's detailed design. The developer shall record this information in the appropriate software development files (SDFs).

7.3.2 Performing unit testing.

--

The developer shall test the software corresponding to each software unit. The testing shall be in accordance with the unit test cases and procedures.

7.3.3 Revision and retesting.

The developer shall make all necessary revisions to the software, perform all necessary retesting, and update the software development files tSDFs) and other software products as needed, based on the results of unit testing.

7.3.4 Analyzing and recording unit test results

The developer shall analyze the results of unit testing and shall record the test and analysis results in appropriate software development files (SDFs).

7.4 Integration Testing

The developer shall perform integration testing in accordance with the following requirements.

Note 1: Integration testing means integrating the software corresponding to two or more software units, testing the resulting software to ensure that it works together as intended, and continuing this process until all software in each CSCI is integrated and tested.

Note 2: If a CSCI is developed in multiple builds, integration testing of that CSCI will not be completed until the final build. Integration testing in each build should be interpreted to mean integrating software developed in the current build with other software developed in that and previous builds, and testing the results.

7.4.1 Preparing for integration testing

The developer shall establish test cases (in terms of inputs, expected results, and evaluation criteria), test procedures, and test data for conducting integration testing. The test cases shall cover all aspects of the CSCI architectural design. This information shall be recorded by the developer in the appropriate software development files (SDFs).

7.4.2 Performing integration testing

The developer shall perform integration testing. The testing shall be in accordance with the integration test cases and procedures. JHU/APL shall also perform independent functional tests of each CSCI.

7.4.3 Revision and retesting.

The developer shall make all necessary revisions to the software, perform all necessary retesting, in conjunction with JHU/APL, and update the software development files (SDFs) and other software products as needed, based on the results of integration testing.

7.4.4 Analyzing and recording integration test results.

The developer shall analyze the results of integration testing. Testing and analysis results shall be recorded in the Integration Test Results Document by the developer and be reviewed and approved by JHU/APL.

7.5 System Testing

The developer shall participate in System testing activities in accordance with the following requirements.

Note 1: System testing means integrating CSCIs with interfacing CSCIs, testing the resulting groupings to determine whether they work together as intended, and continuing this process until all CSCIs in the system are integrated and tested.

Note 2: If a system or CSCI is developed in multiple builds, system testing may not be complete until the final build. System testing in each build should be interpreted to

mean integrating the current build of each CSCI with the current build of other and testing the results to ensure that the system requirements to be implemented in that build have been met.

7.5.1 Preparing for system testing

The developer and JHU/APL shall participate in developing and recording test cases (in terms of inputs, expected results, and evaluation criteria), test procedures, and test data for conducting system testing. The test cases shall cover all aspects of the system-wide and system architectural design. The developer shall record software-related information in appropriate software development files t SDFs).

7.5.2 Performing system testing.

The developer and JHU/APL shall participate in system testing. The testing shall be in accordance with the system test cases and procedures.

7.5.3 Revision and retesting.

The developer shall make necessary revisions to the software, participate in all necessary retesting, in conjunction with JHU/APL, and update the appropriate software development files (SDFs) and other software products as needed, based on the results of system testing.

7.5.4 Analyzing and recording system test results.

JHU/APL shall be responsible for analyzing the results of system testing. JHU/APL shall document analysis and test results in the System Test Results Document.

7.6 Acceptance Testing

The developer shall participate in system acceptance testing in accordance with the following requirements.

Note: Acceptance testing is performed to demonstrate to JHU/APL that system requirements have been met.

7.6-l Preparing for system acceptance testing.

The developer and JHU/APL shall participate in developing and recording the test preparations, test cases, test procedures and test data to be used for acceptance testing and the traceability between the test cases and the system requirements.

7.6.2 Performing acceptance testing

The developer and JHU/APL shall participate in acceptance testing. This participation shall be in accordance with the acceptance test cases and procedures.

7.6.3 Independence in acceptance testing

The person(s) responsible for performing acceptance testing shall not be the persons who performed detailed design or implementation of software in the system. This does not preclude persons who performed detailed design or implementation of software in the system from contributing to the process, for example, by contributing test cases that rely on knowledge of the system's internal implementation

7.6.4 Revision and retesting

The developer shall make necessary revisions to the software, provide JHU/APL, advance notice of retesting, participate in all necessary retesting, in conjunction with JHU/APL, and update the software development files (SDFs) and other software products as needed, based on the results of acceptance testing.

7.6.5 Analyzing and recording acceptance test results

JHU/APL shall be responsible for analyzing and recording the results of acceptance testing. The results shall be documented in the Acceptance Test Results Document.

8. Item Pass/Fail Criteria

Requirements for determining item pass/fail criteria are TBD.

9. Test Suspension and Resumption Criteria

Requirements for determining test suspension and resumption criteria are TBD.

10. Environmental Requirements

The developer shall establish, control, and maintain a software test environment to perform unit, integration, system and acceptance testing of software. The developer shall ensure that each element of the environment performs its intended functions.

11. Responsibilities

Specific testing responsibilities assigned to JHU/APL and the developer for unit, integration, functional CSCI, system, and acceptance testing are summarized in the table below.

Test Type	Participant		Facility	
	APL	SAIC	APL	SAIC
Unit Test		x		x
Integration Test		x		x
CSCI Functional Test	x		x	
System Test	x	x		x
Acceptance Test	x	x		x

12. Deliverables, Milestones & Schedules

JHU/APL has responsibility for the following software **testing** deliverables and milestones:

Phase 1 Testing Deliverables:

Dec 1995	Master Test Plan
Oct 1996	System Test Results Document
Nov 1996	Acceptance Test Results Document

Phase 2 Testing Deliverables:

Oct 1997	System Test Results Document
Nov 1997	Acceptance Test Results Document

The developer has responsibility for the following software **testing** deliverables and milestones:

Phase 1 Testing Deliverables:

May 1996	Completion of Software Coding
Sep 1996	Completion of Unit, Integration & System Testing
Sep 1996	Integration Test Results Document
Oct 1996	Completion of Field Acceptance Testing

Phase 2 Testing Deliverables:

May 1997	Completion of Software Coding
Sep 1997 .	Completion of Unit, Integration & System Testing
Sep 1997	Integration Test Results Documents
Oct 1997	Completion of Field Acceptance Testing

13. Problem Reporting and Corrective Action

Problem reporting and corrective action issues were addressed in the SAFER Quality Assurance Plan.

14. Tools, Techniques, and Methodologies

Issues related to tools, techniques, and methodologies were addressed in the SAFER Quality Assurance Plan.

15. Approvals

Approval for the satisfactory completion of unit testing is the responsibility of the developer with the concurrence of JHU/APL. Approval for the satisfactory completion of integration, system, and acceptance testing must be obtained from JHU/APL prior to the release of the SAFER System for production processing.

Appendix A

SAFER System
Test Design, Case & Procedure Specifications

Test design, case and procedure specifications for the SAFER System are to be supplied as details of the software design evolve.

Appendix B

SAFER System
Detailed Testing Schedule

A detailed SAFER System testing schedule is to be supplied as details of the software design evolve.