SOFTWARE-INTENSE TRAINERS:

A STEP FORWARD IN AUTOMATED SOFTWARE SUPPORT

by

Dr. Roger W. Johnson
Software Systems Department
Grumman Aerospace Corporation
Bethpage, New York 11714

John R. Ruckstuhl, P.E.
Engineering Change Support
Branch (Code N-413)
Naval Training Equipment Center
Orlando, Florida 32813

ABSTRACT

A Software Support Facility (SSF) designed initially for the A-6E Weapon System Trainer (WST) and located at the Naval Training Equipment Center (NAVTRAEQUIPCEN), Orlando, is seen as a significant advance in capability for updating software-intense trainers. This paper describes a software support environment that facilitates the modifications, testing, and baseline configuration management of software used in sophisticated trainers. The software support environment consolidates a variety of techniques into an effective capability for managing the implementation of modifications. Baseline configuration management is founded on the implementation of operational procedures that make it easy to adhere to configuration guidelines but difficult to get changes into the system by other means. Automated identification of global symbol generation and usage greatly reduces risk of unforeseen change impact on other routines. A trainer software structure is described that permits modification of source code without patches and without the normal complete link-edit procedure. A communication link facilitates rapid turnaround in the modification and test cycle. Potential support of trainers with different computers is identified. Further, the SSF reality makes possible the overall integrated environment of "man" and "machine" for management, control and visibility for software support.

INTRODUCTION

The maintenance of software or "software support" is an ever increasing necessity as the complexity and the number of software-intense trainers grow. Consistancy between these emergent trainers and their corresponding parent operational systems require support procedures that provide both, rapid and accurate update methodology. Digital computer software support for training devices, therefore, is to be accomplished through a field network of engineers, programmers and technicians utilizing telephone data transmissions and engineering facilities. The Software Support Facility, designed initially for the A-6E Weapon System Trainer (WST) (See Figure 1), forms the nucleus of the software support concept.

The Software Support Facility (SSF is located at the Naval Training Equipment Center (NTEC) Annex at Herndon, Orlando, Florida. Funded by NAVAIR, the SSF marks a first at NTEC for providing an in-house capability for its software support of a major trainer. Its dedication signifies recognition that support of trainer software necessitates the establishment of a "computer environment" that can facilitate the modification, testing and the management of software packages resident on current trainers. This environment can be extended to support, by judicious facility growth in accordance with the added workload, many of the software-embedded training devices planned for, or existing, in the NAVY inventory.

The new Software Support Facility or "SSF" is an outgrowth of the Software Development Facility created by Grumman Aerospace Corporation. (1) This recent innovation is designed to centralize the software support function and to act as the major element in servicing and supporting training devices located in multiple remote sites. Noteworthy is the establishment of the communication links between NTEC and the trainer sites at NAS Oceana, Virginia, and NAS Whidbey Island, Wash. via Remote Job Entry (RJE) and time-share terminals. These RJE terminals make possible the centralization of an SSF and allow the utilization of the combined resources at remote sites as well as at the SSF to accomplish the necessary rapid software changes. This centralization philosophy promotes a more cost-effective solution to the software support function and forms the catalyst for the support approach to all digital computer driven training devices. This is made possible by the availability of Software Management Tools to facilitate systematic software modifications coupled with a methodical procedure to perform the software accountability or "configuration management" function over trainer software. These tools, working in conjunction with a file oriented data base, provide flexible report generation, software modification tracking and software configuration control. The "software tools," for example, allow the automation of the process for modifying source code (without the need for patches to executable code) and the compilation and transmission of only the affected segment of the software; this segment is then installed at the site without the necessity of the typical complete link-edit procedure.

From the nucleus of this A-6E/WST SSF, extended software support is planned for existing and future digitally based trainer systems. This extension implies that continuity of SSF operating procedures

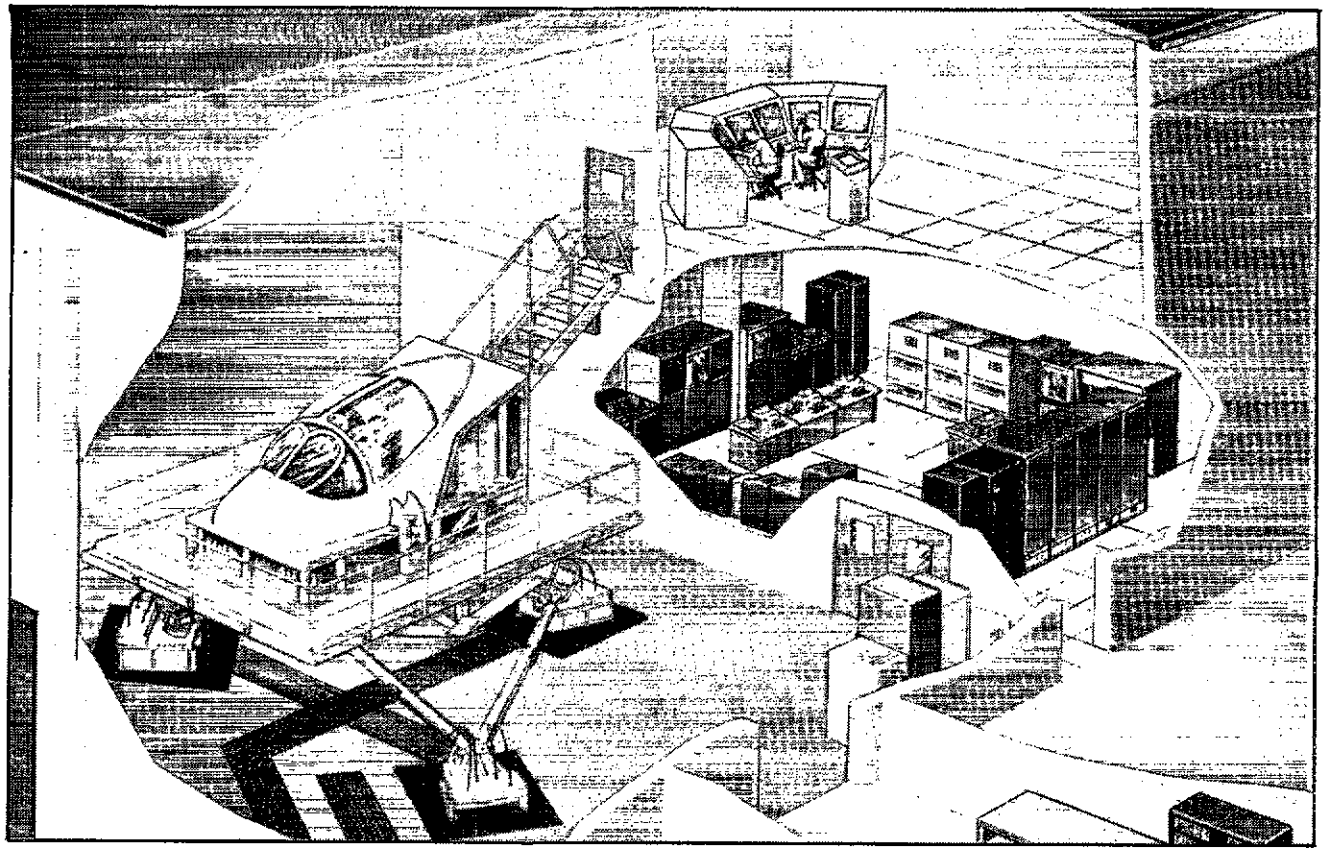

Figure 1. A-6E/WST

persist as additional digital host computers are annexed to the SSF. As long as the software support can be implemented with as few differences (computer languages, etc.) as possible, the personnel requirements for this centralized software support can be minimized.

SOFTWARE SUPPORT OBJECTIVES

The evolution of centralized software support has been motivated strongly by the realization that software contributes 70-80% of the life cycle cost of current computer systems.[2] Further, it is recognized that computer embedded systems (i.e., trainers, etc.) that remain in the inventory for long periods (5-10 yrs) also follow this same trend in resource allocation. Concurrently, the flexibility inherent in software-intense devices necessitate an "automated environment" to let the computer handle procedural functions automatically (i.e. compiliations, assemblies, memory management, etc.), while programmer modification involvement is minimized to source code change of the particular trainer program segment and/or module.

Under these considerations, the objective of the Software Support Facility is to produce a highly automated and integrated environment, in which the centralized software support function can provide service to a variety of devices. Centralization of this function implies that, in most cases, the training device will not reside at the SSF. The hardware configuration and its attendant data communication network at the SSF, therefore, provide, (1) rapid and accurate data transfer capability for modified trainer program segments through Remote Job Entry (RJE) terminals, and (2) trainer site time-share terminal access to "baseline" source code that is maintained at the SSF. These stipulations fuel the fire for a provision that rigorous "configuration management" or accounting software baseline be centrally controlled at the SSF. At the same time provisions for a single common source code "baseline" for both SSF and trainer site personnel is available for trainer software modifications - the best of both worlds...

In developing such a facility, goals have been selected to reduce the cost of development and support through centralization. This provides the opportunity to mechanize each trainer software "baseline" under a controlled configuration management procedure and to selectively control and enhance the value of the companion documentation. Subsidiary goals for life-cycle visibility, traceability and manageability are fallouts of the centralized configuration management implementation. Finally, the SSF computer configuration coupled with the trainer software "baseline", provides a known departure point, or platform for technological enhancements or growth whether it is accomplished in-house or contractually.

The SSF has been designed initially for A-6E/WST software support and by design follows the above guidelines. The practices and procedures are common to most training devices. Additional devices are anticipated to, one-by-one, join the A-6E/WST in the SSF environment, thereby reducing substantially the cost of life-cycle support for trainers. This is based on the continuing require-

ment caused by update and change of operational equipment, enhancements to training or operational value, additional capabilities, system improvements for stability, error recovery or upgrading and finally, correction of deficiencies.

The balance between responsive trainer software support and cost-effective operation remains as the central issue in the SSF implementation. The SSF configuration and its environment are described next.

SUPPORT ENVIRONMENT

The SSF environment is a combination of digital computer, communication devices, and a set of Software Management Utilities or "Tools" to automate many of the pedestrian procedures necessary to edit and construct a trainer program. The operating procedures mold this assortment of entities into a cohesive and cost-effective support system.

Hardware Configuration

The facility currently consists of a Perkin-Elmer 8/32 digital computer, 4 disc drives, 2 tape units and associated hardware to provide direct communications with Oceana and Whidbey Island. A Versatec Printer/Plotter is used for making hard copy of the graphics data normally directed to the Instructor Console. A complement of approximately 10 Navy personnel, with the help of six (6) employees of Grumman Aerospace Corporation man the facility (See Figure 2). The computer hardware configuration is selectively tuned to the software support mission. Its core memory of one million data bytes provide sufficient space to accommodate source program storage, workload capacity, software configuration control, tracking and utility programs for the SSF and trainer site personnel to efficiently carry out their mission.

Communication Network

The communications network currently implemented uses dial-up telephone lines to provide high and low baud rate capability for RJE and CRT terminals, respectively to Oceania NAS, Va. and Whidbey Island NAS, Washington. Figure 3 summarizes the network details, while Figure 4 shows the communications network currently in place.

Operational Procedures

The hardware configuration provides an optimum mix for the support of centralized "software configuration management" and rapid access of baseline source programs via either SSF or field timeshare terminals. To balance the workload and limit communication transmission durations, an operational framework between the SSF and field sites is necessary.

An account structure has been installed on the SSF computer system to implement this operational environment. A "system" account heads the hierarchical account structure; it contains such programs as Command Substitution System (CSS) procedures, FORTRAN compiler, assembler, RJE communicator, data base file system (TOTAL) and other system software utilities to properly manage the varied workloads resident on the computer. Directly under the "system" account are "group" accounts which contain, (1) various categories of trainer

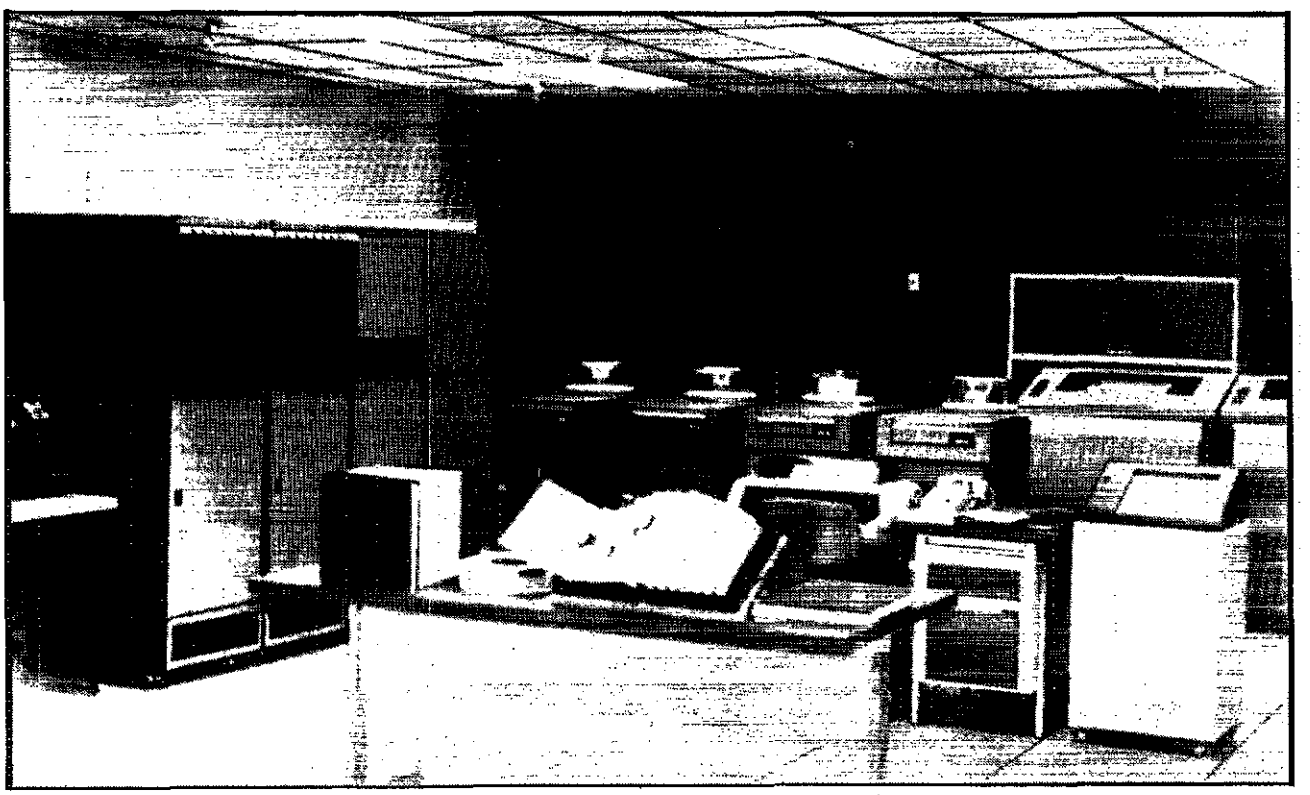

Figure 2. Software Support Facility (SSF)

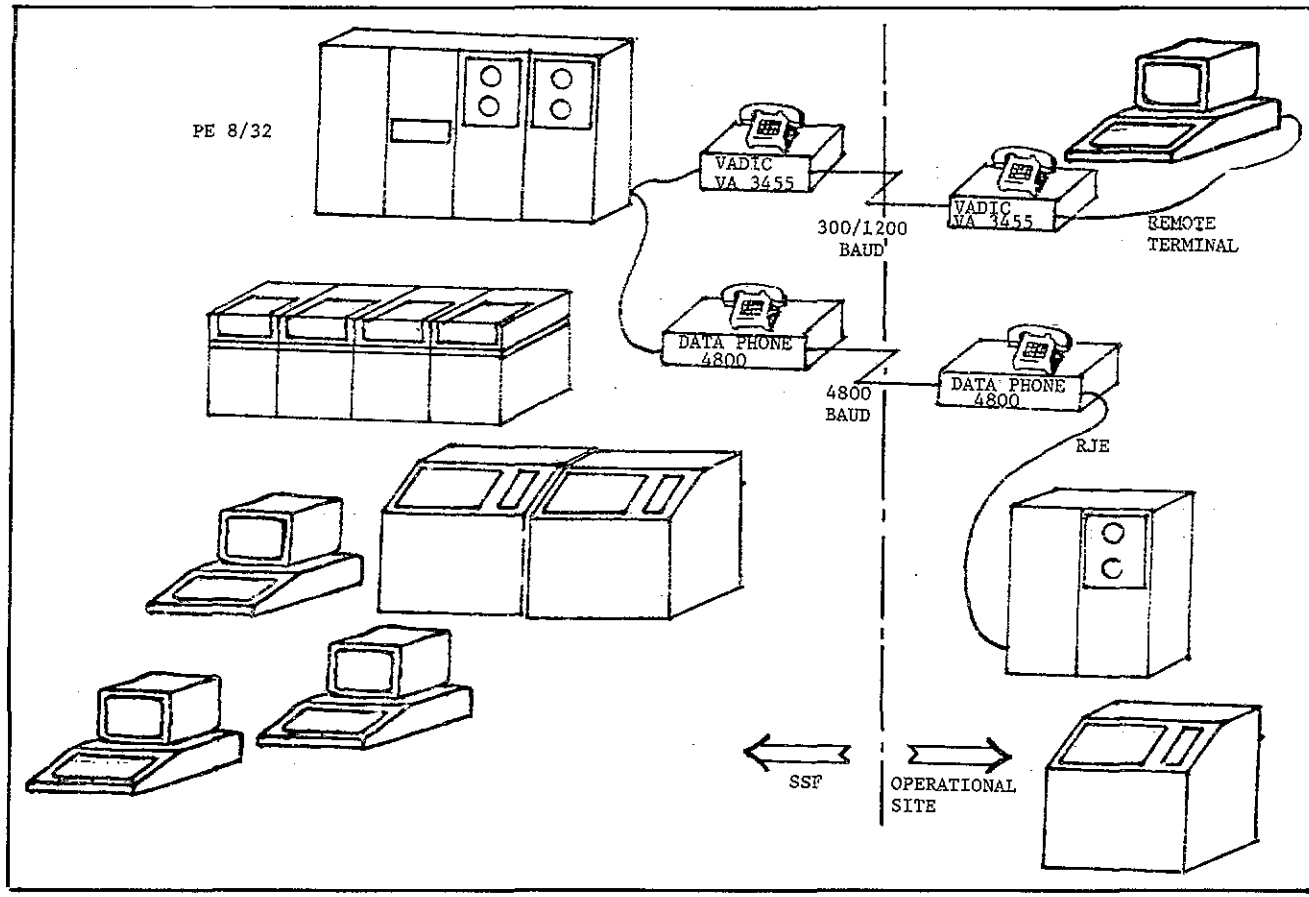

Figure 3. SSF to Site Communication Capability

"baseline" source programs, (2) software tool programs, and (3) software system component programs (i.e. I/O, Graphics pages, etc.). At the third level, private accounts are established beneath the system and group accounts, in which modifications are carried out by SSF and field personnel. The system and group accounts are protected from change by giving access through password to a selected few in the SSF; in particular, to the configuration management personnel. Private accounts can read from accounts above in hierarchy, but cannot change or replace programs in these accounts; this is reserved for the configuration management function.

In this way, modifications can be made in parallel (private accounts have access to source code baseline) and submitted to configuration management for approval. Configuration management, in turn, then processes these modifications according to a methodical procedure, including Software Review Board approval, compilation of the new module, generation of a new task segment or "overlay" (configuration management tools are available that automate much of this procedure) and the transmission of the "overlay" to the appropriate trainer sites over the communication network. Following this sequence, the site personnel install this overlay on the designated trainer and carry out the appropriate test programs to verify the software modification. The final step is then for the configuration management personnel to update the software baseline at the SSF on the successful completion of the trainer test. If deficiencies are found during the test sequence, they are recorded on the Test Discrepancy Report and modification continues until all deficiencies are corrected.

Software Management Tools

Salient features of the software design for the A-6E/WST were selected to create both a stable development and a stable support environment. This in turn, stabilized the procedures and yielded a generalized set of Software Tools that could be used to automate many of the configuration management and modification functions. In extending these tools to future trainer support, trainer software design features that are desirable to take maximum advantage of this generalized support environment are given as follows:

(1) Modularized software structure.
(2) Some categorization scheme that will permit assignment of unique identification numbers in logical manner.
(3) Descriptive information on each module.
(4) Established Mnemonic convention format for symbols, modules, etc.
(5) Higher Order Language (HOL)
(6) Partitioning of large-scale programs to implement source modifications that will not require excessive transmission time to site.

The Grumman developed Test and Configuration

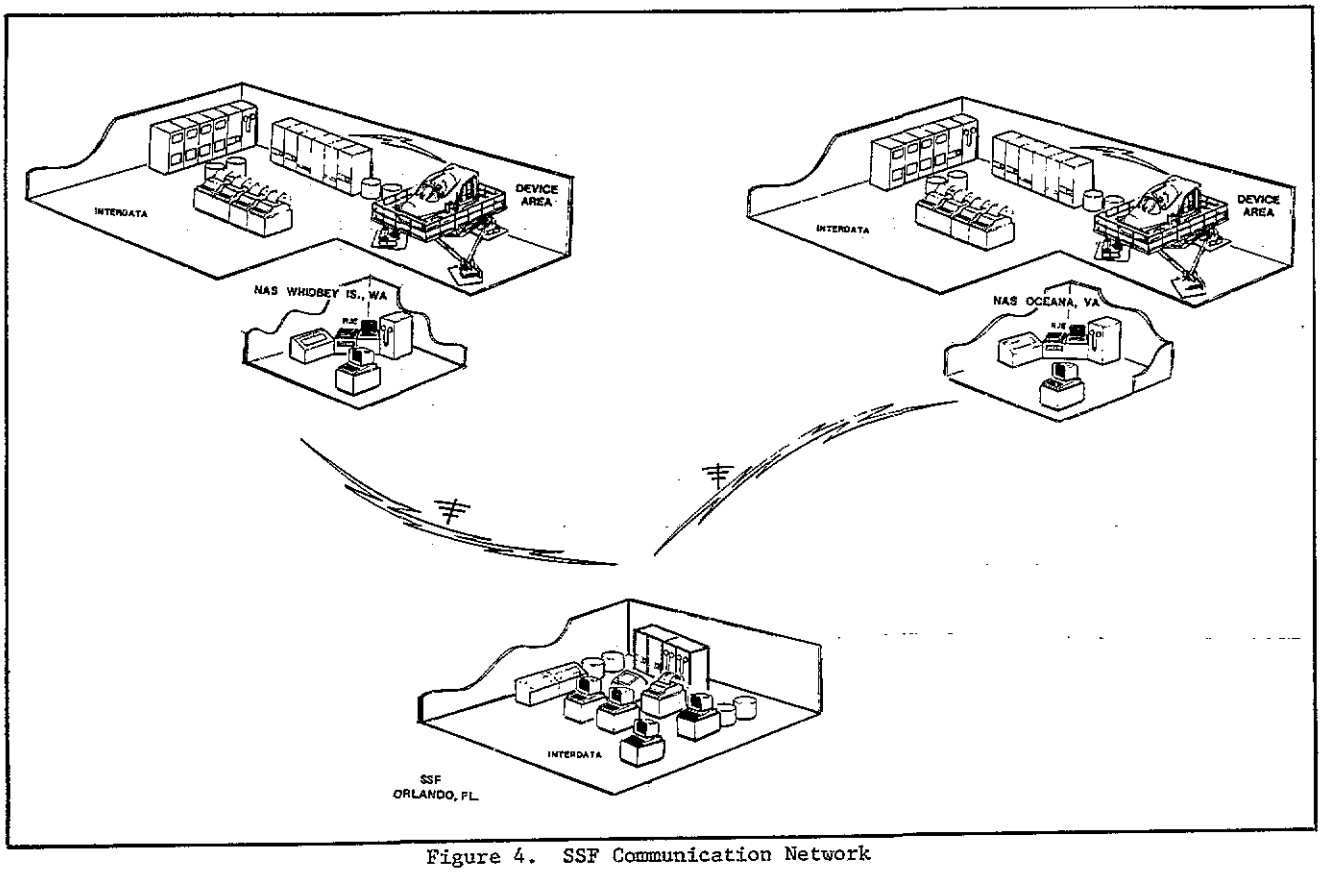

Figure 4. SSF Communication Network

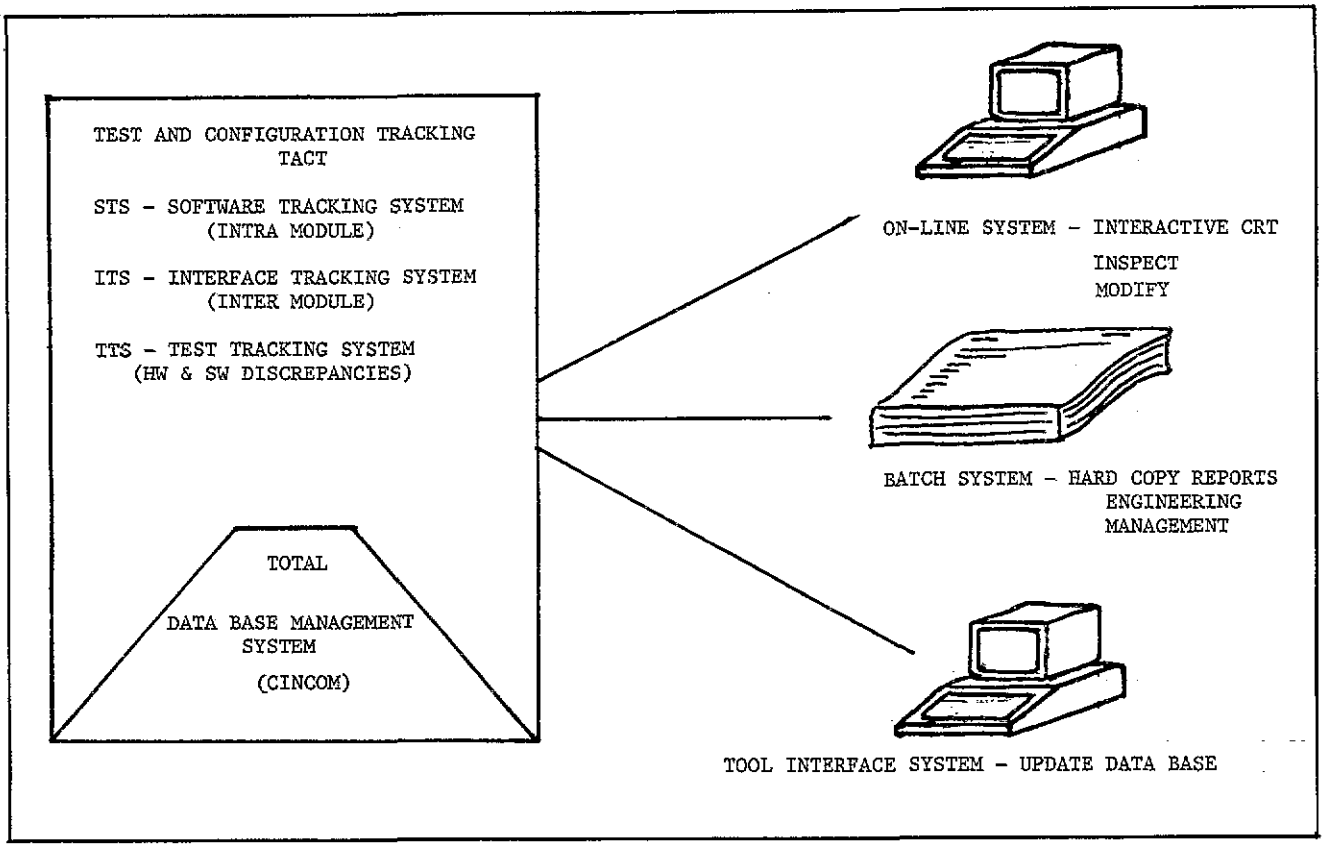

Figure 5. TACT System Configuration

Tracking System (TACT) is a file oriented data base system that combines a technical reporting system with a management information system. It is supplemented by analysis and test support tools that further enhance the flexibility for data base control, and for the analysis of "as-built" code.

The TACT system (described in more detail in reference 1) is operated in two modes, (1) on-line system, with interactive control for quick-look inquires and data base update and, (2) batch report, hard copy processing. Three major functional subsystems makeup the TACT system; they are:

(1) Software Tracking System (STS)
(2) Interface Tracking System (ITS)
(3) Test Tracking System (TTS)

The TACT system utilizes the TOTAL data base management system marketed by CINCOM, Inc. The TACT system configuration is shown in Figure 5.

The software tracked by the TACT system is organized into a hierarchical structure so that each subsystem/module is identified with a unique Hierarchical Identification (HID) code. The Software Tracking System (STS) is used to collect and track, through a module data base, various parameters of each module. Data such as module name, mode usage, execution order, execution time, core size actuals, input and output mnemonic names, version level, source language and cognizant engineer, are a few of many variables that are used to describe the anatomy of each module. In the support role, it is crucial to maintain this data base through the various modification projects for each device. The STS system serves the role of maintaining this fundemental data base as a part of the software configuration baseline.

The Interface Tracking System (ITS) automates the handling of module interconnect variables and the signal variables between the software and the trainer hardware. A Data Element Dictionary (DED) and Signal Description List (SDL) are examined by subordinating programs that supply the ITS with current interface data. These programs basically assist in the automatic generation of all the interface parameters of the system. The ITS plays a significant role in the software modification procedure by automatically identifying, for a given change of variable, the modules and parameters affected by that change. The extent or the resources needed, for a software change can be quickly assessed by the use of this portion of TACT. An ancillary program, Automatic Interface Mneumonic Extractor System (AIMS), is used to systematically dissect a software module to classify, extract and identify the attributes (global vs local or input vs output, etc.) of module variables across a spectrum of FORTRAN, assembly code or a mixture thereof. The AIMS builds the data base for the ITS program. Through this medium various reports can be gathered to identify missing inputs, unused outputs or other pertinent data necessary to insure the integrity of a modified trainer program.

The Test Tracking System (TTS) is an information management system which maintains a data base that references test procedures and the resultant Trainer Discrepancy Reports (TDR). The elements maintained are the current, relevant test procedures by module or by mode, the responsible engineer, subsystem name, description of discrepancy or change and its disposition. Reports are automatically generated (in batch mode) for the day-to-day, week-to-week tracking of these TDR's. They are to be used by the various levels of management for resource allocation, for failure analysis, for visibility of trouble spots, etc. A "quick look", as well as, update of TDR data base are available through an "on-line" TTS terminal.

Access to the TACT system via CRT terminal from the trainer sites either on-line or batch, is provided through the VADIC modem in a time-shared mode. It, in fact, gives the site personnel access to the appropriate source programs, software tools, compilers, etc., that reside on the SSF. These software utilities are summarized in Figure 6. (1)

Various software tools are also available at the SSF to maintain the baseline configuration of the trainer software. The procedure for the maintenance of the software baseline is a methodical discipline put in a centralized location (for more than one device location), to track, control and record the history of change for a given trainer. Figure 7 (1) illustrates the steps necessary to not only maintain the software baseline, but also to coordinate the configuration updates to the documentation. These software tools (1) provide complete tracking and identification of all authorized changes, (2) include header/trailer programs to emplace configuration identity and quality assurance information on each source file, (3) include configuration accounting programs for listing software components and revision levels and for automatically extracting configuration data from header/trailer blocks, and (4) generate all load modules from computerized configuration control records.

The automated configuration management tools and the procedures laid out in Figure 7 are leading toward the generation of an overlay which is ultimately transmitted to the trainer site. The overlay is a program segment which has undergone a modification under a change order, recompiled and structured into the form of an executable load module for installation on the appropriate trainer. The overlay is directed to a specific window in memory that normally contains the approved revision level of code. The overlay is structured to intercommunicate with other overlays via Fortran Labeled Common which avoids the need to link-edit the entire program. This automated sequence includes the verification of compliance to the baseline configuration prior to any processing of a change component. It concludes with the acceptance testing after installation and the revision and release of the new software configuration baseline.

EXTENDED TRAINER SUPPORT PLAN

The rationale for centralizing the resources for the trainer device software support becomes stronger as the number of training devices at different locations increase. This is substantiated by a recent study (3) prepared by Computer Sciences Corporation for NAVTRAEQUIPCEN Code N-412, Orlando, Florida.

Short-Term Extension of SSF Capability

Under this plan, (4) SSF-configured host

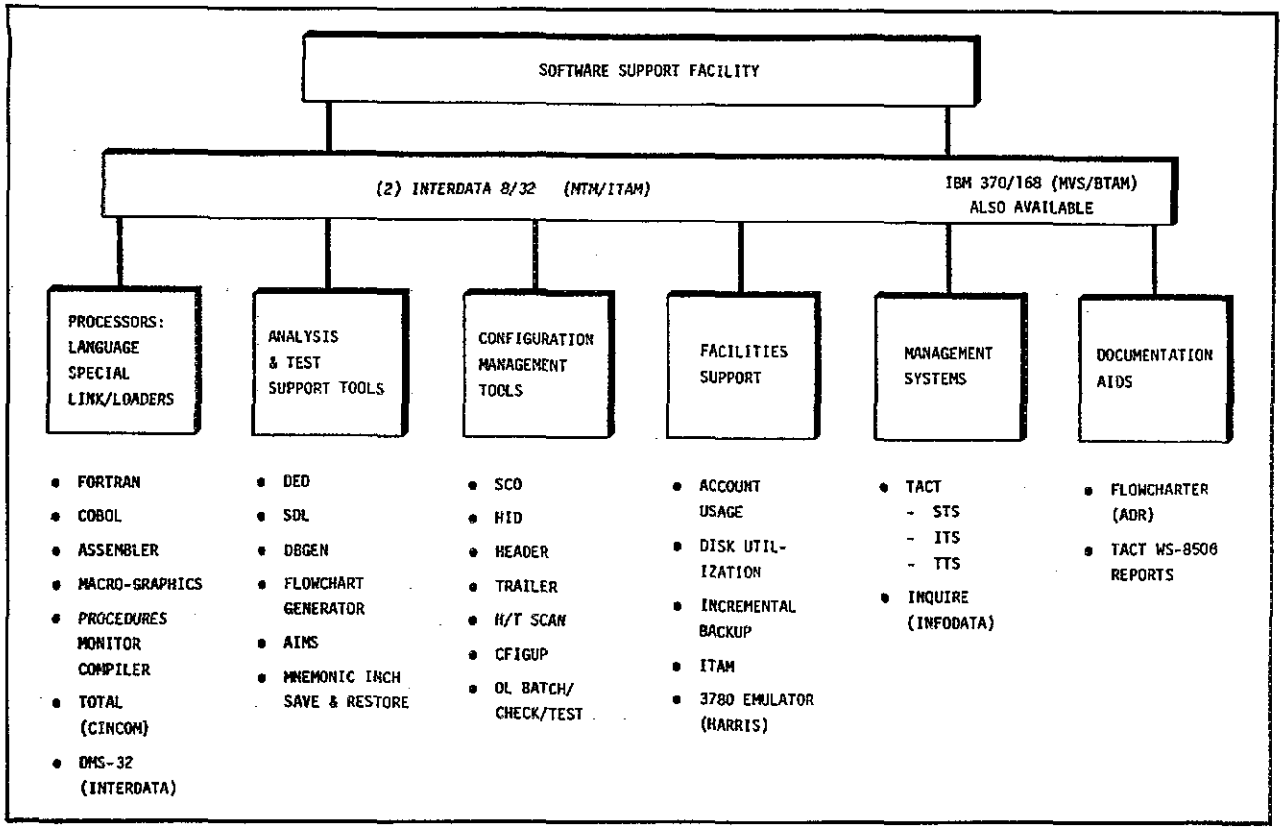

Figure 6. SSF Software Support Capability

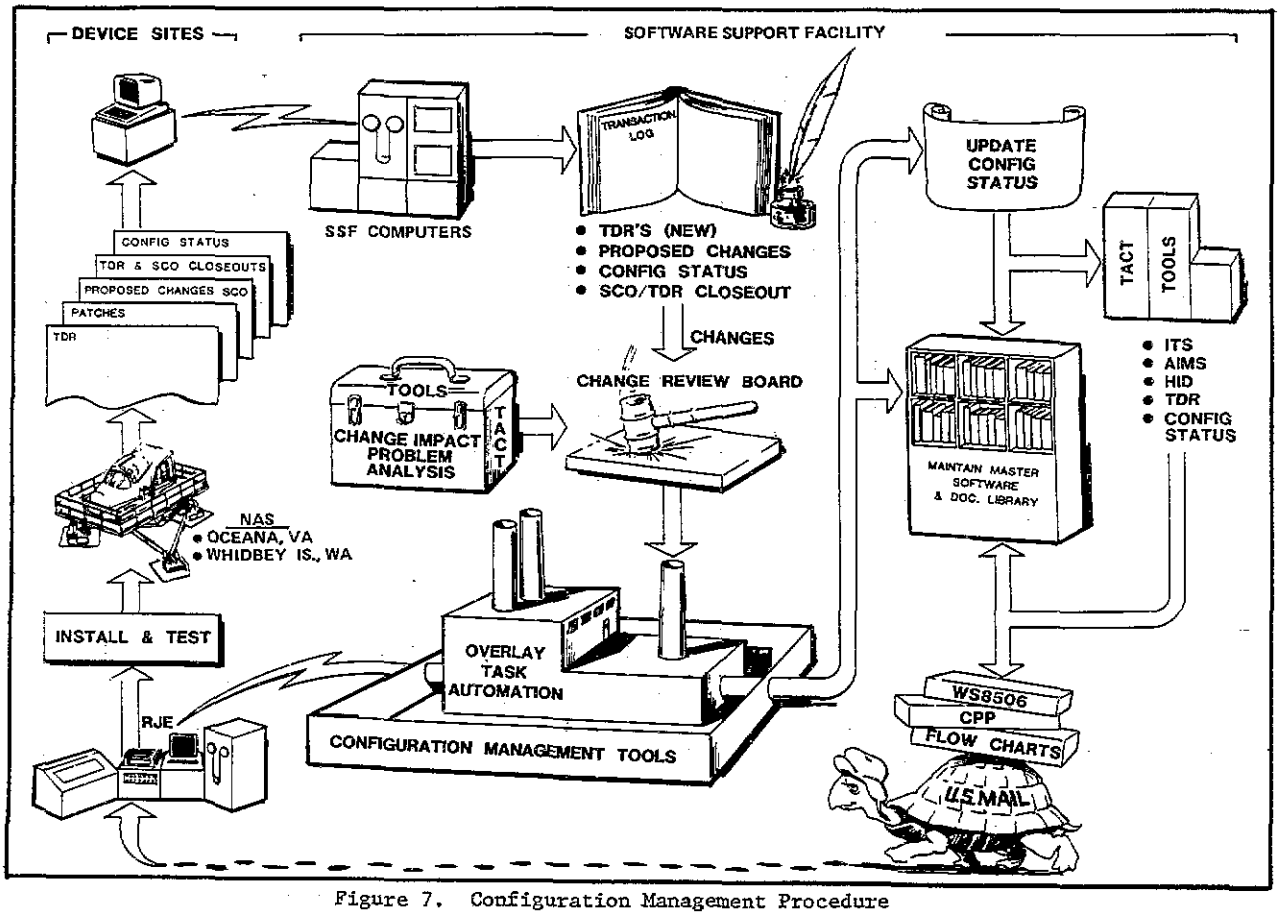

Figure 7. Configuration Management Procedure

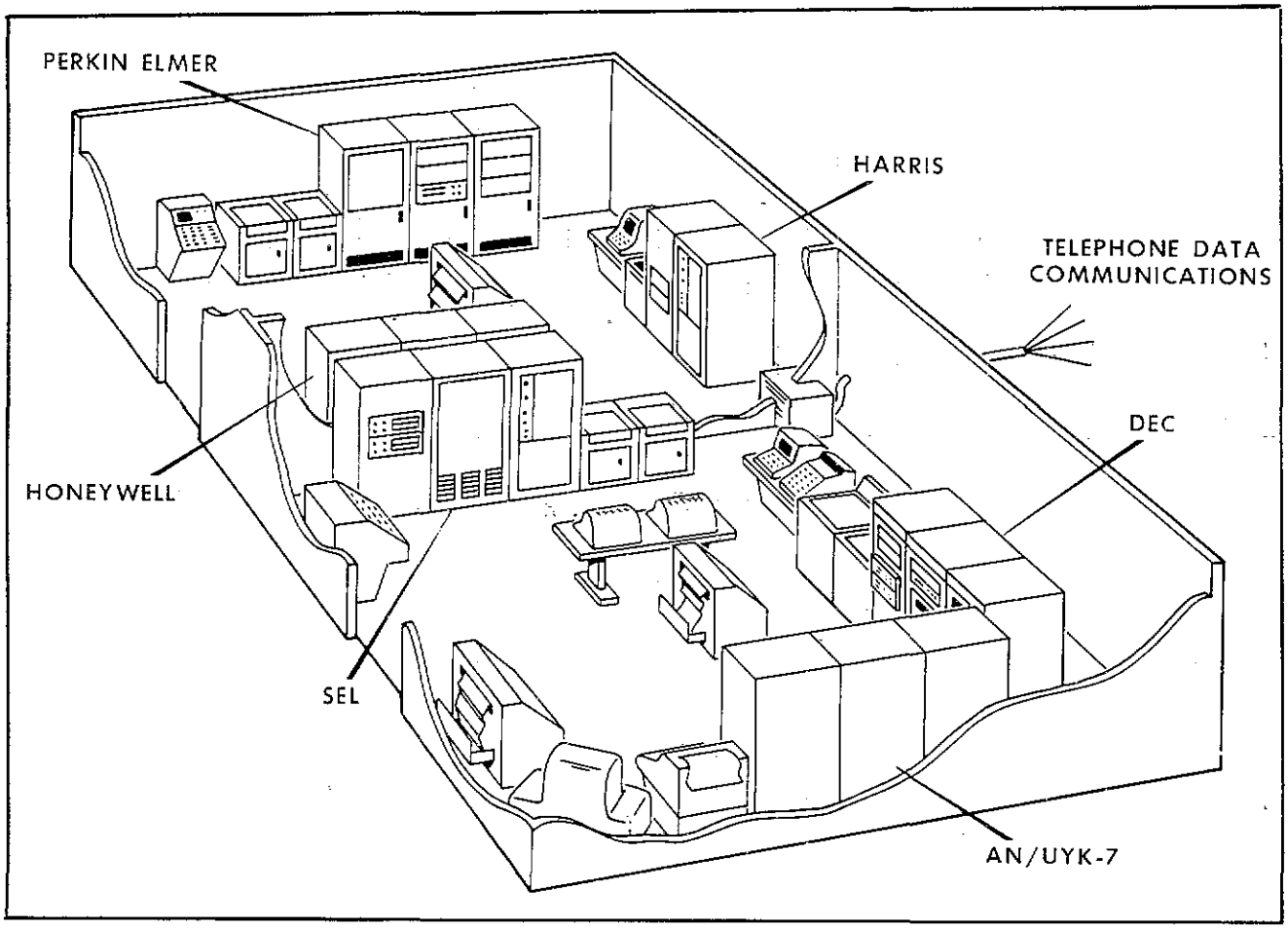

Figure 8. Proposed Orlando Software Support Expansion

computers (supporting an additional and appropriate segment of NAVY training devices) would sequentially be added to the existing SSF. This expanded facility is shown in Figure 8. The management tools already described are to be modified and be compatible with these additional host computers; the communications network would also be expanded to three main dedicated telephone circuits (shown in Figure 9) to service most of the Naval training facilities within the CONUS.

The procedures and practices now implemented at the SSF are seen to be common to the expanded SSF. This commonality, in turn, will provide a manpower cost-effective base on which to operate the centralized facility.

Long-Term SSF Implementation Goal

Greatly improved performance with minimum duplication of both effort and hardware may eventually be provided through the restructuring of the Software Support Facility to include a mainframe computer. In this configuration many of the software tools would be combined and implemented on the mainframe computer, thereby eliminating the need both for supporting multiple implementation of all the tools on separate minicomputers at the SSF and for extending implementation of those tools when additional minicomputers are brought into the SSF. The mainframe computer would be interconnected with minicomputers within the SSF to provide a capability for using compilers, assemblers, link-editors and utility packages from the OEM of the target computers. In some cases emulation of target computers would be used so that certain computers would not have to be maintained within the SSF environment. Communication links would connect with the mainframe computer rather than with each separate minicomputer. The mainframe would direct data to a minicomputer for compilation, assemblies, or link-edit as appropriate and receive the results for storage. Minimal auxiliary storage would be dedicated to each minicomputer for working storage.

The implementation thus described would greatly facilitate operations both local and from the field. Locally there would be reduced duplication of equipment and procedures among the numerous computer systems involved. The consolidation of trainer configuration baseline management into a mainframe computer not only reduces the need to learn multiple systems but also tends to provide opportunities for global analysis of multiple baselines across numerous target computers. The modification efforts, both local and field-based, can be implemented using a single text editor and common operating procedures regardless of the identity of the target computer involved. The primary aspect of modification that cannot practically be made transparent to the user are the

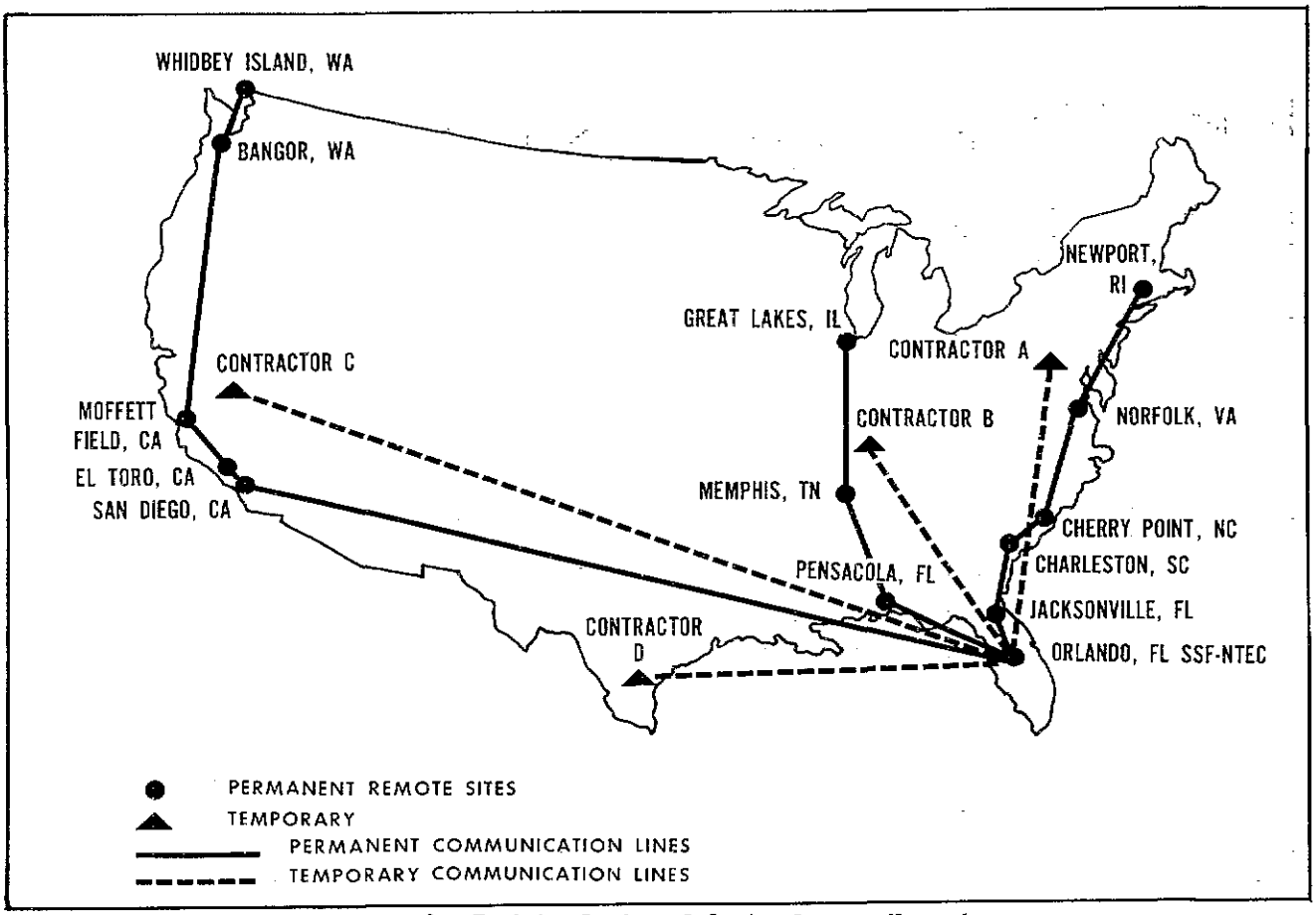

Figure 9. Training Devices Software Support Network

language differences and the software testing efforts associated with the different target computers.

The benefits of establishing a centralized Software Support Facility for use by local and field modification efforts rise to their maximum potential with the implementation of software tools on a centralized mainframe computer.

SUMMARY

The centralized SSF environment provides a cost-effective method for the implementation of software support to training devices widely distributed in location. It enhances the use of key software skills, encourages software commonality, makes use of common software maintenance tools and standards and reduces redundant support hardware.

This centralized facility provides the software baseline control or configuration management function that for so long has been missing. It accomplishes this while allowing the primary device to function on its intended mission. The flexible communication network allows maximum accessibility to authorized NAVTRAEQUIPCEN field personnel so that the most knowledgeable people on trainer operation can be used to diagnose problem areas. All the power of the SSF environment is virtually at the disposal of the field personnel.

Incremental change and growth are a way of life for large software systems. Design evaluation, aircraft updates and changes, and the addition of features to enhance training value require a well-coordinated and automated software support environment like that available at the SSF. The SSF also serves to reduce the cost of these functions and increase the life-cycle traceability and manageability of the software support; the direct accessibility to source programs, processors, tools and documentation; and direct data communication to the device sites.

The SSF serves as a "baseline" for extending central software support to other trainers with their attendant target computers. The Software Management Tools and Procedures, for example, are essentially applicable with some adjustments to those applications. The orchestration of the procedures necessary to efficiently deliver this service to the Fleet is, and will remain, the primary mission of the Software Support Facility.

REFERENCES

1. M.C. Blyseth, "A Generalized Software Development and Support Environment for the A-6E Weapon System Trainer (A-6E-WST", Proceedings, 1st Interservice/Industry Training Equipment Conference, Technical Report NAVTRAEQUIPCEN IH-316, November 1979.

2. J. Goldberg, "Proceedings of a Symposium on the High Cost of Software", ONR Contract N00014-74-C-0028, September 1973.

3. _____, "Software Support Facility Implementation Plan ", Report NAVTRAEQUIPCEN 80-D-0008-7, July 1980.

4. _____, "A Management Plan for Providing Cost Effective Software Support for Training Equipment", Prepared for NAVTRAEQUIPCEN, Code 41, January 1980.

ABOUT THE AUTHORS

Dr. Roger W. Johnson is the Director of Field Operations, Software Systems Department at Grumman Aerospace Corporation. He is presently manager of the Software Support Facility for Grumman, under contract with the Navy. Two years prior to this assignment, he was the Director of NASA Space Programs for Grumman. Before this he served in the USAF as Director of Computer Services at Vandenberg AFB and Deputy for Advanced Space Programs at SAMSO, L. A. Dr. Johnson received his B.S. at the U.S. Naval Academy, M.S. at Massachusetts Institute of Technology and, PhD at the University of California, Los Angeles.

John R. Ruckstuhl is Program Manager for the Software Support Facility within the Engineering Change Support Branch of the Modification and Maintenance Engineering Division, NAVTRAEQUIPCEN. He has been active in various aspects of software support since 1976. Previously he was involved in research concerning simulation using software models. Before joining NAVTRAEQUIPCEN in 1970, he was a Project Officer with the U.S. Army Training Device Agency, now called PM TRADE. Mr. Ruckstuhl received the degrees of B.S. in EE and M.S. from the University of Southwestern Louisiana in 1966 and 1968, respectively. He is also registered as a Professional Engineer.