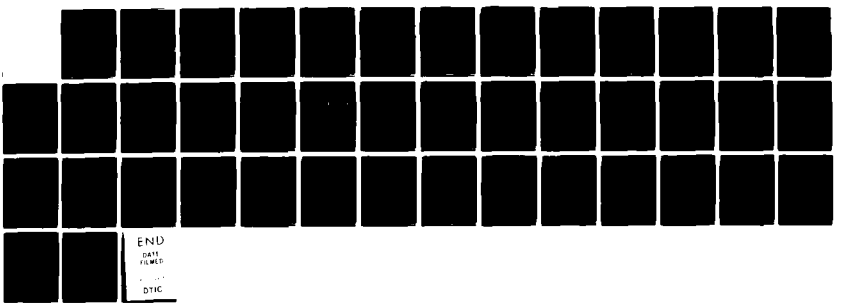
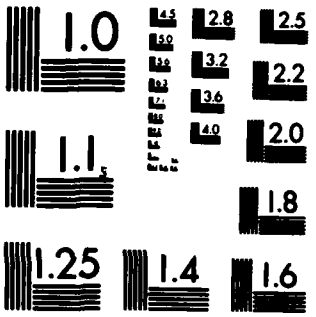
AD-A128 942 SOFTWARE TECHNOLOGY FOR ADAPTABLE RELIABLE SYSTEMS (STARS) FUNCTIONAL TAS..(U) OFFICE OF THE DEPUTY UNDER SECRETARY OF DEFENSE RESEARCH AND E.. 30 MAR 83 1/1

UNCLASSIFIED F/G 9/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A128942
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle) Software Technology For Adaptable Reliable Systems (STARS) Functional Task Area Strategy For Application Specific	5. TYPE OF REPORT & PERIOD COVERED
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) The DoD Joint Service Task Force on the Software Initiative (STARS)	8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Deputy Under Secretary of Defense Research & Advanced Technology Washington, D.C. 20301	12. REPORT DATE March 30, 1983
	13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Unclassified

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

STARS, software engineering institute, automated support environmets, mission critical military systems, Ada, software, hardware, reliable, adaptable, embedded computers, system interface standards, life-cycle costs, KIT, KAPSE, software initiative, application oriented technologies, methodologies, standardazation.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This strategy document is one of eight functional task area strategies produced by the STARS Joint Task Force. All of the documents produced by the Task Force are listed in the STARS Joint Task Force Report. This document identifies the scope, sub-objectives and strategies designed to provide the conceptual approach for accomplishment of the STARS Program objectives in the application specific functional task area.

DD FORM 1473 EDITION OF 1 NOV 65 IS OBSOLETE S-N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

SOFTWARE TECHNOLOGY FOR

ADAPTABLE RELIABLE SYSTEMS (STARS)

FUNCTIONAL TASK AREA STRATEGY FOR

APPLICATION SPECIFIC

Department of Defense

30 March 1983

FOREWORD

This strategy document is one of eight functional task area strategies produced by the STARS Joint Task Force. All of the documents produced by the Task Force, including the general STARS Program Strategy document, are listed in the STARS Joint Task Force Report.

This document identifies the scope, sub-objectives and strategies designed to provide the conceptual approach for accomplishment of the STARS Program objectives in the application specific functional task area. It identifies and describes the high-level activities, products and capabilities. In order to provide full understanding, background and rationale material is sometimes covered that is also in STARS Program Strategy.

These functional task area strategy documents do not attempt to delineate the detailed plans, costs and procedures for bringing the proposed products and capabilities into being and do not identify the form of the particular projects that will undertake the work nor the organizations in which the work will be accomplished. Instead, these strategies are intended to guide the process of such implementation planning and accomplishment.

Indeed, because of the high degree of linkage among the functional task areas, implementation plans and acquisitions may well combine related capabilities and products across areas. Individual projects may tackle only part of one subtask from a functional area or several subtasks from several functional areas.

Thus, this functional task area strategy describes broad, achievable requirements for accomplishing the relevant STARS objectives. Its main purpose is to help guide the implementation planning process.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF ILLUSTRATIONS

LIST OF TABLES

1.0 OVERVIEW

1.1 Broad Objective of the Application-Specific Area

The ultimate success of the STARS Program will be measured by its ability to be responsive to the specific needs of the DoD user. Success cannot be achieved by simply adopting a strategy of nurturing the products of the research community and outwardly adopting as the primary STARS Program objective that of seeking acceptance of such products in the user community. Success depends on the establishment of a window between the user community and the STARS Program, so that the reflected needs of the users can drive the STARS Program. The STARS Program requires the support of the user community, without the leverage from which the STARS Program cannot succeed. Such is the motivational model which drives the DoD STARS Program.

The Task Area with the responsibility to be responsive to specific needs through such an interface is the Application Specific Task Area. This is one STARS Program area where all tasking should be motivated by specific user requirements. Other Task Areas support generic applications, which, although user-driven, can be in a supporting role that is invisible to the ultimate user. It is in the Application Specific Area where the products of the other task areas should be transitioned to reflect that part of the overall solution which has been generated by specific DoD user requirements.

In the broadest sense, the overall top-level objective of the Application Specific Task Area can be stated as follows:

> The Application Specific Task Area serves as the conduit for transitioning new technologies into target-of-opportunity military system programs.

A major sub-objective of the plan is to gain leverage from cooperative support both of the military program manager and, through Government/industry cooperative efforts, of the private sector. Such

cooperation is necessary to effectively use and strategically expand the resources available to the STARS Program.

Linking the STARS Program with the user is the Application Specific Environment which would be the coordinated product of all task areas of the STARS Program, technically integrated by the STARS Program and presented for use to the application area as a user-friendly product. This environment should form part of the application infrastructure upon which the products of the application specific technologies can be designed, developed and used in large scale military systems.

Note that the tailoring of the infrastructure to become an application specific environment is a new idea for potential savings of manpower and monetary resources. However, the full benefit of this approach would be realized only when the application specific developments which build upon this base are developed in a manner which promotes reusability. Thus the leverage exerted on the military system, in the form of an application specific environment, must be reciprocated to the STARS Program in the form of reusable application specific software.

The steady state situation with respect to reusable software requires the introduction of a starting transient which adds to the cost of the first system. The resources spent on the application specific task can be thought of as the up-front costs of introducing this starting transient.

1.2 Technical Strategy for the Application-Specific Plan

The Application-Specific Task Area should encompass a number of new and/or evolving technologies. Each sub-task within the area would be motivated by a DoD operational need which could be satisfied through the use of one or more of these technologies in a major defense system environment.

However, the scope of the task area extends far beyond that of merely choosing a set of application sub-areas which encompass these technologies and allowing each application to proceed independently. These technologies, which are discussed in Section 1.4 of this plan, are for the most part immature. Thus an arm of this task area should be the research and development on the use of these technologies. There is a need to support research and technology development in Reusable Software Components and Composition Systems. These are presently the most mature of the Application Specific Technologies. For less mature technologies, such as Very High Level Languages, Application Generators, and Knowledge Based Systems, there is a need for applied research.

A thread of consistency must exist between the subtasks so that the generated products will build upon one another and interface with other elements of the STARS Program. These products must be human engineered with consistent methodology and documentation, or the potential user would find the task of learning to use the product as difficult as developing a new product. In part, this lack of human engineering has lead in the past to duplicative, non-reusable software effort. Thus, with the assistance of industry organizations, promotion is necessary of User Groups that resolve those problems in human communication which could be solved by standard methods and nomenclature.

Finally, although this is a software program, the utility of software is derived through the proper operation of the hardware. Thus the Application Specific Task Area must pursue a hardware /software synergism which recognizes a need for hardware/software tradeoff. Such synergism would be promoted within the STARS program through the consideration of tradeoffs between software and VHSIC/VLSI technologies within the subtasks of the Application Specific Task Area.

1.3 The Role Of the Application-Specific Area Within STARS

The STARS program could be characterized, as in Figure 1, as the process which contains the STARS Office to collect and focus the developments of the various task areas and mold such developments into an Ada Programming Support Environment (APSE). This APSE, along with the application specific developments form an Application-Specific Environment to be used by the STARS Program Office and the various military system project offices. The interface between the contractor and the Government would be through the STARS Program Office at the R&D and Engineering levels, and between the Contractor's Project Office and the Government Military System Project Office at the System Development level.

Standard practices and acquisition policies should assure that the contractor developments both are in accordance with Government regulations and are promoting the use of reusable software for future programs.

The efforts in Application-Specific task area should complement the task areas which are primarily generically oriented. This would promote a strategy for demonstrating Ada and the support system environment, thereby facilitating more rapid technology insertion through the STARS Program to mission-critical weapon systems. Application-oriented technologies would address the system definition problem by communicating in terminology understandable to the person in the application area. Both computer system architecture and hardware/software synergy issues suggest possibilities for application-oriented hardware to be used in productive combinations with application-oriented software. Acquisition management and procurement issues are extremely important if software reuse is ever to become common.

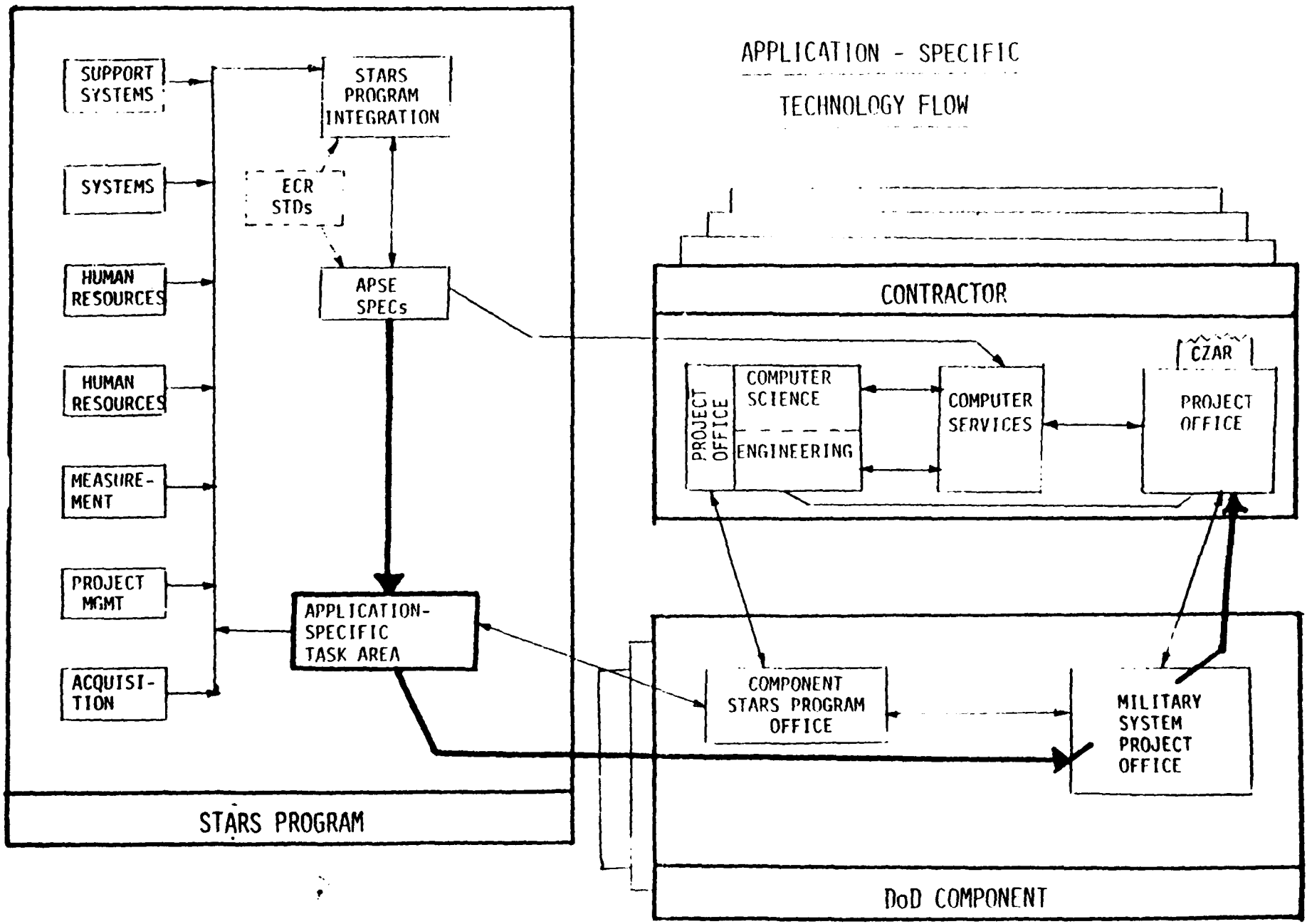

APPLICATION - SPECIFIC
TECHNOLOGY FLOW
SUPPORT SYSTEMS
SYSTEMS
HUMAN RESOURCES
HUMAN RESOURCES
MEASURE-MENT
PROJECT MGMT
ACQUISI-TION
STARS PROGRAM INTEGRATION
ECR STDs
APSE SPECs
APPLICATION-SPECIFIC TASK AREA
STARS PROGRAM
CONTRACTOR
PROJECT OFFICE
COMPUTER SCIENCE
ENGINEERING
COMPUTER SERVICES
CZAR
PROJECT OFFICE
COMPONENT STARS PROGRAM OFFICE
MILITARY SYSTEM PROJECT OFFICE
DoD COMPONENT

Figure 1

The most significant interactions with the other task areas would be:

- o Measurement: The measurement and evaluation aspects of application-specific software would provide concrete tasks for the Measurement Task Area and concrete tests of the validity of output via the demonstrations of the environments on the selected mission-critical test beds.

- o Human Resources: Training material/courses supplied by the Human Resources task area would be used to train personnel to use, maintain and rehost the environment. This should be available concurrently with the appropriate environment, and to be updated with each new release.

- o Systems: The Application-Specific task would consider system modules which are candidates for hardware/software synergistic implementation. Applications requiring very high reliability and/or critical timing would be aided by the use the integrated approaches developed.

- o Acquisition: Standardized methods to encourage the production of software suitable for reuse and to motivate such reuse are presently unavailable. Application-specific contractors should make suggestions to the Acquisition task area on this issue and should evaluate reusable software procurement strategy recommendations. These suggestions would involve such issues as (1) the responsibility for errors and continuing maintenance support, (2) the allocation of the higher initial costs due to more complete verification and testing, (3) the responsibility for production and distribution of documentation, and (4) the incentives for making reusable software available to other contractors.

- o Human Engineering: The very high level languages and the interfaces within and between Ada packages would require a heavy day-to-day labor intensive activity. The design of these application-specific technologies would benefit from the guidelines and methodologies described by the Human Engineering task area.

- o Support Systems: The application-specific task would select from the common environment libraries provided by the Support Systems Task Area in order to build and package an environment for the specific mission-critical area. Technical issues in software warehousing and distribution would be

resolved within the Support Systems Task and the appropriate capability included in the packaged environment.

1.4 Technologies for the Application-Specific Task Area

The Technologies which are candidates for development and utilization of reusable software, enhanced software development and hardware/software synergistic solutions for this task area are the following:

- Reusable Software Parts. Inventories of Ada parts could be built and a catalog system created to identify and retrieve parts from each inventory. The effectiveness of such parts inventories could be demonstrated by their reuse in multiple software development test beds. At first this retrieval and reuse may need to be done manually. A technology solution would be needed for automation.

- Parts Composition Systems. A parts composition system automates the assembly of parts retrieved from reusable parts inventories. The catalog entry for each part gives formal descriptions of its inputs, outputs, and functions that the parts composition system uses to fit parts together.

- Very High Level Languages. A very high level language (VHLL), sometimes called an application-oriented language, is a programming language with application-specific control structures, data structures and operators. An environment natural to the application area is created with the common functionalities built in. A VHLL may automate the use of software parts in programming specified application tasks.

- Application Generators. An application generator generates a program or solution from non-procedural (*what-to-do*) statements or requirements. The conceptual model is equivalent to filling out a form. By comparison, a very high level language is more likely to require procedural (how-to-do) statements. Application generators also depend on parts composition. Note that most high level application-specific systems are likely to have a combination of elements from very high level languages and application generators.

- Knowledge-Based Systems. A knowledge-based system uses codified general problem-solving techniques and codified specific application knowledge to solve problems in the

application area. It is an "expert" assistant to the application worker, reducing his need for skilled human assistance. A knowledge based system might include a combination of a VHLL with application generators. It is assumed that the general methodology for creating knowledge-based systems is developed outside the DoD STARS Program. However, it should be recognized that even given this methodology as known, it is still a formidable task to organize the application-specific knowledge properly and to insert it into a knowledge-based system framework.

- o Application-Specific Computer Architecture. A computer architecture may be tailored to perform well on the common processing in a particular application area. Examples include signal processors, arrays processors for structural engineering calculations, and LISP machines. These can be particularly powerful when combined with very high level languages, application generators and/or knowledge- based systems.

- o Standardized Methodologies. Additional benefits can be gained if technologies in this task area are continuously monitored to identify common problems and approaches. Once identified, consensus on standards for factors such as models, methodologies, and tool structures could be achieved. These standards could then result in generic technology structures that may cross application-area bountries.

1.5 Effect of Application-Specific Area on the Military Mission

Software, like other products, is composed of parts; however, unlike physical products, the cost of replicating software parts is negligible, regardless of size or complexity. The major costs are in design, development, and validation of the first version, and in maintenance (including adaptation to new requirements) of operational software over its lifetime.

A principal task in this area should be to develop sets of software parts inventories (Ada packages) for a number of specific application areas and to demonstrate their use in real systems. Parts from this inventory would be reused in new projects in the application area. Parts produced by one DoD contractor would be made

available to other contractors, greatly increasing the inventory of reusable parts and the productivity of those who use them.

Not all application areas are equally suitable for software parts; those which have matured somewhat and are suitable are characterized by repetition of function in successive system generations. Likely candidates for software parts inventories can be classified from broad DoD areas such as those on the following list:

Architecture Standards
Avionics Systems
Battlefield Automation
Command & Control Systems
Communication Systems
Intelligence Systems
Shipboard Automation
Weapons Systems.

A further breakdown of the above areas along functional lines, which was derived from a preliminary DoD R&D baseline, yielded the following functional categories:

Artificial Intelligence
Algorithm Development
Architecture
Data Base Management
Digital Signal Processing
Environment
Fault Tolerance
Flight Control
Guidance
Knowledge Based Systems

Modeling

Navigation

Very High Level Language.

Table 1 shows a matrix tabulation of the types of efforts in the preliminary baseline and notes those which are ripe to be pursued immediately by this task area, those which are potentially ripe, and those not ripe. Note that the two categorization are not orthogonal. For any breakdown of application areas, there would be overlapping of functions and consequently there would be outputs of this task area which would be common to two or more application areas.

This task area should develop, concurrently, standards, methodologies and procedures so that the products and practitioners can better fulfill military missions. The establishment of standard practices and a framework for computing, plus an inventory of reusable software parts, would yield timely, less expensive, more versatile military software.

1.6 Summary

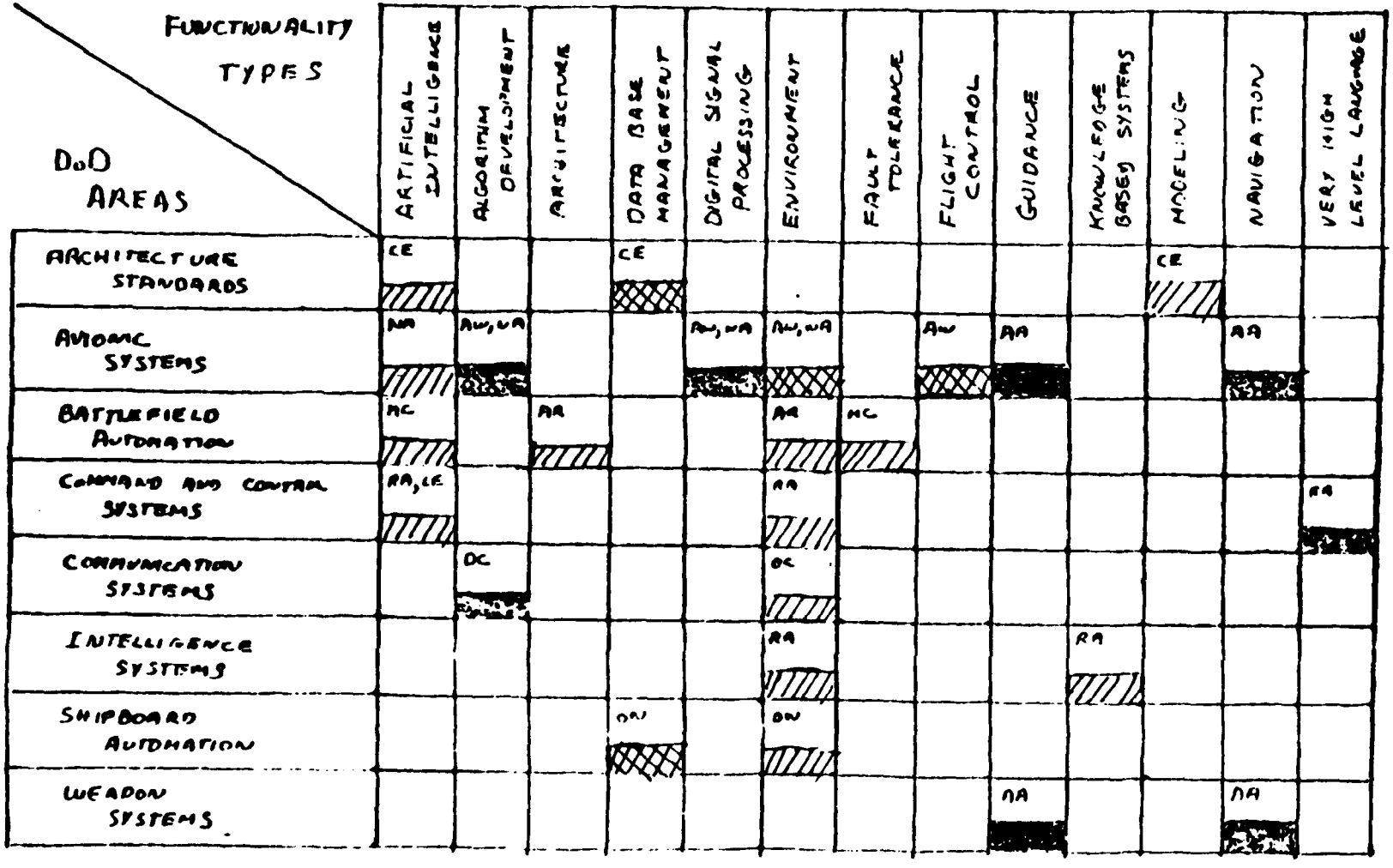
The STARS Program has three potential compatible technology activities within the Application-Specific Task Area.

(1) Inventories of reusable Ada software parts could be provided for selected military application areas. These inventories would include both simple and sophisticated parts. Collections of parts would be demonstrated by incorporating them manually into systems and by using them to build very high level application-oriented languages.

(2) A methodology could be developed to define, create, evaluate, warehouse and retrieve the software parts for systems. This activity is generic and the general methodology would be flexible and partially implemented with software tools. Each application area would adapt the methodology to its particular circumstances.

Table 1

Application-Specific Task Area Readiness Assessment



(3) Very high level languages and related advanced technologies such as Application Generators, Knowledge-based Systems and Application-Specific Computer Architectures could be developed which utilize parts from one or two of the inventories.

The long term goal should be to dramatically reduce the time and expense of developing software for military missions. Ada is a modern language for producing reliable programs. Ada has capabilities not found in its predecessors, but in those areas where it replaces such predecessors, its level or "power" is similar to languages such as Pascal, FORTRAN, or Algol. A large application still requires thousands of lines of code to be written, debugged and maintained. An approach to achieve the above goal is to dramatically reduce the number of lines of code to be written for a new application. Reusable software does this by incorporating code that has already been written, debugged and which is maintained "elsewhere". The parts composition system greatly facilitates the use of parts and, in a sense, creates a language for building software from existing code. The parts composition system user will need to be skilled both in programming and the application area. Another approach is the very high level language which greatly reduces the level of programming skill required. This either reduces the requirement for application area skills or permits more skillful personnel to concentrate on the non-software aspects of the application.

A second, longer term, approach to achieving the goal of reduction of the amount of user knowledge and skill required is the use of application generators. Ideally, one provides a short statement of the problem to be solved and a program is generated. The knowledge-based system solution would be the ultimate approach with the application engineer and the application-specific environment operating in concert as a symbiotic team of problem solvers. The knowledge-based system would provide a massive amount of factual knowledge and stan-

dard problem solving techniques. The human element would provide the creative guidance to develop a solution for a particular problem. The net result would be the rapid solution of problems with minimal programming or, perhaps, merely codifying the requirements from which a knowledge-based system would create the appropriate programs.

2.0 PROPOSED TASKS

The following nomenclature is used throughout this section of the proposed tasks:

This strategy is known as the STARS Functional Task Area Strategy for Application-Specific, which distinguishes it from other functional strategies for the STARS Program.

This section contains proposed tasks which would implement the strategies described in Section 1. The proposed tasks are divided into five subtasks. Each subtask contains several phases. Subtask 1 and Subtask 5 are each intended to be executed once. Subtasks 2, 3, and 4 are each intended to be executed several times, one (or more) time(s) for each of several application areas.

2.1 Major Subtasks

The five major subtasks for this plan are enumerated below and detailed in Sections 2.2.1 through 2.2.5. Although the titles of these subtasks are global to the STARS Program, the technical discussions address only the application-specific technologies:

- Subtask 1 -- Prepare Initial RFP: Appropriate application areas would be identified and requests for proposals issued for initial work in these areas.

- Subtask 2 -- Analyze Each Mission Critical Area: The functionalities, data types and other entities naturally occurring in the application area would be identified and coherently organized.

- Subtask 3 — Develop Each Prototype Mission Critical Environment: An initial set of software parts (Ada packages) would be produced along with the relevant catalog and retrieval information.

- Subtask 4 -- Develop Selected Production Mission Critical Environments: Some contracts would be continued to produce the entire set of capabilities, that is, everything from software parts to a knowledge-based system for application areas selected from the initial areas.

- Subtask 5 — Integration over Application Areas: The software parts development, procedures and results in application-specific projects would be monitored to identify common items, software and requirements. These would be standardized into generic parts and procedures for use in all application areas and, where appropriate, included in the software environment evolving during the STARS Program. This subtask should run concurrently with Subtasks 2, 3, and 4.

Figure 2 is a milestone chart for the subtasks. It also indicates the relationship between the tasks, their approximate schedules and some of the interaction with other areas of the DoD STARS Program.

2.2 Details of the Major Subtasks

2.2.1 Subtask 1 -- Prepare Initial RFP's

- Goal: The primary goal of this subtask should be to identify a number of application areas that are well suited for initiating a software parts technology. While it is believed that almost all areas would eventually be suitable for this technology, some are presently more suitable than others. Areas of specific interest to defense systems would be selected for technology demonstration.

- Description: This subtask should be completed in FY83. There would be five identifiable phases of this subtask, which are described in Sections 2.2.1.1 to 2.2.1.5.

- Results, Costs and Benefits: The result of this subtask would be at least six similar contracts for several application areas, each to commence with Subtask 2. This subtask is shown at the top of the milestone chart, Figure 2.

2.2.1.1 Prepare Lists of Areas and Criteria. The Joint Service Task Force should compose an initial list of six to twelve application areas thought suitable -- for example, digital avionics, communications, command and control, tactical missiles, smart munitions, ground-based air defense systems, and artillery fire control. An initial list of criteria should be developed that considers the

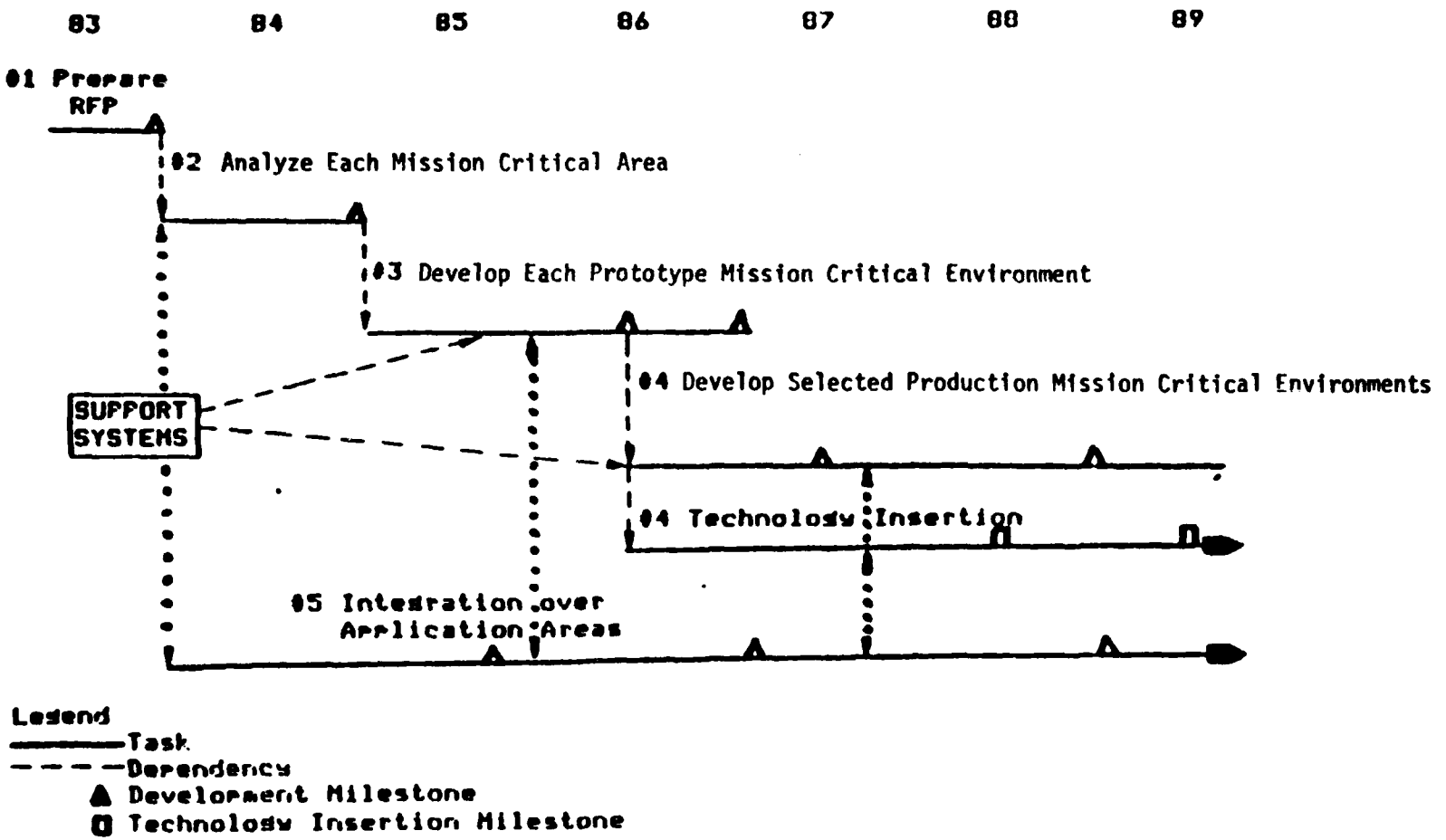


Figure 2. Milestone chart for the five major subtasks of the Application-Specific task area.

potential impact on existing or planned systems, the level of stability and understanding of the application area, the existence of related efforts, and the need for some areas requiring very high reliability. Applications with clear potential for hardware/software synergy and/or application-specific computer architectures would be included.

2.2.1.2 Identify User Group Nuclei. The ultimate success of this plan would be aided by the creation of active user groups. Such groups may be sponsored by existing public sector organizations. Individuals who have the technical and leadership qualities to eventually form such groups should be identified.

2.2.1.3 Review Lists. The initial lists should be reviewed and refined.

2.2.1.4 Prepare RFP. A single generic RFP should be prepared. The DoD component responsible for each area would be identified, and each component might enlarge the generic RFP, *without* changing the work statement, to include application area specific issues and to conform to component policies. The RFP would ask proposers to select and propose an appropriately-sized effort within a specific sub-area of one of the areas on the lists developed in Phase 1 of this subtask.

2.2.1.5 Compete and Award. The RFP's would be issued, evaluation criteria and teams established, the responses reviewed, and the contracts awarded for Subtask 2.

2.2.2 Subtask 2 -- Analyze Each Mission Critical Area

Several parallel efforts are intended in this subtask, each in a different application area and each with similar characteristics.

- Goal: The primary goal for this subtask would be to develop concrete requirements for a set of software parts and related entities. These requirement include: appropriate

data structures, relevant terminology and data, standard functions and procedures. This subtask would delineate the difficulties encountered in organizing the requirements and reaching a reasonable consensus on terminology and definition. General concepts to measure the quality of the software would be formulated and submitted to the Measurement Task Area for refinement and development.

- o Description: This subtask should be completed in FY84; the last of the six phases is to prepare proposals for the subsequent subtask.

- o Coordination: There should be one or two short workshops where each contractors would present results to date, to each other and to the Subtask 5 contractor (Integration over Application Areas).

- o Output: Each contractor would provide a substantial report with considerable detail about the entire study. Each contractor would also prepare a proposal for the next subtask based on the results from this subtask.

- o Cos/Benefit: This subtask should be executed almost entirely in FY84. The benefit would be to obtain the first definitive assessment of the potential for a software parts technology in application areas. This subtask is shown as the second line of the milestone chart, Figure 2.

2.2.2.1 Organize Functionalities and Data Types. This is the main phase of this subtask. Each contractor should identify and describe a set of functions, procedures, data types, and other relevant entities (for example, events) and their relationships. Required are the overall specifications for the set and justification that the set meets these specifications.

2.2.2.2 Perform Cost/Benefit Analysis. A preliminary cost/benefit analysis should be performed for the application area to provide justification for future investment. This analysis would suggest measures for the benefits that could be applied to later phases of the plan. These measurements would be given to the Measurement Task Area for possible use and refinement.

2.2.2.3 Suggest Approaches to General Reuse. In this phase the contractor should perform the following functions:

- Suggest general interface module (Ada package) standards
- Suggest standards for warehousing and reusing software
- Suggest terminology and information for a catalog retrieval system
- Suggest approaches to reduce non-technical obstacles to software reuse

Current DoD procurement policies do not encourage software reuse by contractors. For software reuse to succeed, DoD procurements must provide incentives for maximum production, support, and use of reusable software. DoD does not always retain complete rights to software it pays to develop. Where a contractor is free to profit from improvements he makes to such software, DoD should be entitled to benefit from the improvements if DoD paid for the development of the basic product. Otherwise, after the basic product is changed, DoD and DoD-sponsored organizations may have to pay full price for software largely conceived and originally developed with DoD funds. Procurement regulations should be studied, suggestions for improvements made, and model contracts developed and demonstrated.

Reusability must be designed in at the earliest stages of development, which may significantly increase development costs. The return from these increased costs would only be realized when and if the software is used in other projects. Analysis is needed to identify software parts for which the extra development costs are worthwhile.

2.2.2.4 Propose Tools (optional). Optionally, tools, or partial specifications for such tools, should be proposed for the following:

- An automated parts composition system to help assemble modules to work together.

- A catalog and retrieval system for the software parts inventory.

- Any special tools for this application area, for example, tools to help understand/translate existing application software, tools to interface new software with existing subsystems for checkout and demonstration, and tools to compare behaviors with existing system behavior.

2.2.2.5 Review Area for Advanced Technologies. This phase of the subtask would study feasibility and suggest advanced application specific approaches such as very high level languages, application generators, knowledge-based application systems, or special computer architectures.

2.2.2.6 Prepare Proposal for Next Subtask. Each contractor would prepare a proposal for the next subtask stating approaches and goals. Each proposal would include a detailed plan.

2.2.3 Subtask 3 -- Prototype Mission Critical Environments

o Goal: The goal of this subtask should be to create a number of actual reusable software parts and to initiate an evaluation of their use. The purpose of developing these sets of parts is not limited to exploring the problems of producing such parts, but the sets would also be of value, themselves, as reusable software parts.

o Description: Contractors with satisfactory results from the prior subtask would design and develop initial Ada package sets in their areas and perform an initial demonstration. Automated methods for software warehousing and retrieval, and for reuse would be developed. Ada support environments would be used as available.

o Coordination: There will be two workshops for contractors and representatives of Subtask 5 to meet to review progress, to discuss plans, to identify common problems and to discuss those elements which are common to more than one application area. Representatives from other task areas would participate as appropriate.

- Output: The output for this subtask from each application area should be the following:

  (a) A report detailing the design and rationale for the packages

  (b) One or more substantial software package sets plus appropriate documentation

  (c) A report describing the software parts inventory, its catalog and the retrieval procedure

  (d) A report describing the testing and evaluation results from the exercise and demonstration of each package set

  (e) An overview report that summarizes the project plus suggestions or plans for future developments (including advanced technologies).

- Cost/Benefit: This subtask is expected to be executed primarily in FY85 and FY86. The two primary benefits for this subtask would be (a) the production of useful software parts and (b) the shakedown of the methodology for the design, production and evaluation of Ada packages. Secondary benefits include (a) the shakedown of Ada and its support environment by users without a computer science background in realistic applications (b) the establishment of a set of Ada trained personnel in several application areas, (c) the collection of proposals from many different viewpoints for the future development of a software parts technology. This subtask is Number 3 on the milestone chart, Figure 2.

Some contractors may also develop prototype software parts composition systems and other special tools to aid in the generation and reuse of Ada packages. Investigation would be performed and proposals made for expansion, further demonstration, or reuse in real systems. In addition, proposals might be prepared for development and demonstration of other application-specific technologies to be considered in Subtask 4. This subtask would consist of nine phases as follows.

2.2.3.1 Refine Functionality and Data Types. Each contractor would review the product of the prior subtask and revise it to

reflect errors discovered, improved understanding, and Government guidance. This phase would insure that the description is a suitable basis to begin design of individual packages.

2.2.3.2 Plan for Ada and Environments, and Train Personnel. Each contractor would plan and begin transition to a DoD approved Ada system and, as available, support environments. A training program would be obtained from other sources or developed, and the project personnel should be trained in the use of Ada and in the use of available support environments.

2.2.3.3 Preliminary Implemenation Design. Preliminary designs for implementation ofthe software packages and tools, if any, would be prepared and reviewed.

2.2.3.4 Detailed Package Design. Detailed designs for Ada packages and tools would be prepared, reviewed and revised.

2.2.3.5 Construct Ada Packages. Ada packages would be built and tested.

2.2.3.6 Develop Software Warehousing Approach. The contractor would develop an initial approach to the software warehousing, retrieval, and reuse mechanisms. He would implement those parts appropriate for the initial demonstration.

2.2.3.7 Demonstration. Each contractor should perform an initial demonstration which shows the efficiency, reliability and adaptability of the package sets developed. See section II.B.4.5 for remarks on the requirements for a demonstration.

2.2.3.8 Preliminary Designs for Advanced Technologies. Preliminary specifications/designs for the use of advanced technologies such as VHLL, application generators and knowledge-based systems in the application area would be developed using those technologies having applicability for the respective application area.

2.2.3.9 Prepare for Next Subtask. A plan to expand the Ada package library and/or to utilize and demonstrate some set of application-specific technologies would be prepared.

2.2.4 Subtask 4 -- Develop Mission Critical Environments

Several concurrent contracts should be awarded for development, production and demonstration of a production mission critical environment which would use a subset or combination of the following technologies for the application-specific area:

- Ada packages and warehousing/distribution systems
- Software parts composition systems
- Very high level languages
- Application generators
- Knowledge-based systems
- Application-specific computer architecture
- Application methodology

The ability of a software product to deal with application complexity and evolving requirements would be a major factor in its usefulness and acceptance. In complex applications, no matter how carefully requirements are specified, some needs are discovered only after experience is gained. Evolving requirements inevitably require software adaptation or replacement. In a well-designed system, the cost of a modification should be commensurate with the magnitude of the functional change.

- Goal: The primary goals of this subtask are (a) to produce a relatively complete set of software technology products for several application areas and (b) to shakedown the methodology for producing advanced technology.

In addition, these efforts would provide ongoing demonstrations of the DoD software environment and its periodic enhancement. As

development efforts evolve to production efforts or fail to meet their goals, changes might be made to the development process for the application areas under consideration and/or new application areas might be considered.

- o Description: This subtask has six phases.
- o Coordination: There would be periodic workshops where contractors present their results to each other, to the Subtask 5 personnel and to other interested participants in the DoD STARS Program.
- o Cost/Benefits: This subtask should start in FY86. Direct support under the aegis of the DoD STARS Program would terminate in FY89. A principal goal of the entire DoD STARS Program is to dramatically improve application-specific software productivity and reliability. The results of this subtask would demonstrate the extent to which the STARS Program has been a success. Under the STARS Program, substantial sets of software would have been produced which would be of great value to DoD programs. Such value should be the measure of merit for the STARS Program.

This subtask is shown on the milestone chart, Figure 2, as Number 4.

2.2.4.1 Install Ada Environments and Train Personnel. Each contractor wouldl install the programming environment initially produced by the Support Systems Task Area along with the other elements of the STARS Program. Each would provide the necessary training in Ada, the environment, and the methods and technologies to be used.

2.2.4.2 Install Software Parts Composition System. A Software parts composition system would be installed and personnel trained in its use. Conceivably only one such system would be created either as part of some application area project, as part of Subtask 5, or as part of another task area of the STARS Program.

2.2.4.3 Specify and Design Software Products. A combination of technologies should be chosen and initial specifications prepared.

Any additional tools that might be needed (either generic or application-specific) would be identified and initial specifications developed.

2.2.4.4 Iterate to Production Quality. Initial implementation versions of technologies for which there is DoD concurrence on the need within operational systems should be developed. Initial implementations would be improved through experimentation. This might involve prototyping of tools, languages and techniques.

2.2.4.5 Demonstration. The effectiveness of developed application-oriented tools should be demonstrated. This may involve rapid prototyping of applications. Figure 3 outlines guidelines for effective demonstrations. Each contractor would develop a demonstration report that would follow this outline fairly closely and comment specifically on the relevant points. Although a totally convincing, completely thorough demonstration usually is not feasible because of the time and expense involved, a cost/benefit analysis should be made in each instance to select the appropriate parameters for a demonstration. It is particularly important that the realism of the demonstration be assessed candidly. The measures formulated in Subtask 2 and refined by the Measurements task area would be used as part of the demonstration.

Ideal demonstrations are those which use new software and technology in a real project. Such projects involve either a new system or a re-implemention of an existing system. Each contractor should strive for demonstrations as close to this ideal as feasible within the constraints of time and cost.

2.2.4.6 Technology Insertion. Initial versions of technology insertion methods should be developed and used to move this technology from the R&D environment to the DoD operational environment. These methods would include provision for assistance to DoD programs

and the dissemination of material which has been tested and perfected to provide demonstrably effective technology insertion methodology. The technology insertion activity should transition into the 1990's as a continuing activity which has been initiated by the STARS Program.

Successful projects would yield application-oriented tools that could, along with generic tools, significantly influence software development in the respective application area. Required use of Ada, if only to write the compiler for a very high level language, and continuous prompt use of new releases of the integrated support environment would ensure that the application-oriented tools and other results would be integrated into the application-specific environment.

* Realism
    - Size of developed portion
        - Complete System or Subsystem
        - Significant units of manipulation
        - 10K-200K lines of code
    - Testbeds
        - Actual or planned realistic contexts and exercises
        - Variety of sizes
    - Multiple Uses
        - Demonstrate in a variety of uses (at least two)
        - Demonstrate in unplanned (i.e. unknown/unexpected to developers) situation (optional)
    - Evolving Systems
        - Demonstrate ability to evolve as requirements change
    - Level of Complexity
        - Not toy or oversimplication
    - Schedule
        - Tight deadlines during a demonstration
        - Rapid Prototyping (optional)
    - Use by other than development team

* Design Excellence
    - High quality and disciplined plans and procedures
    - High quality approach to application
    - Use previously formulated measures of quality

* Conclusions of Demonstration
    - Quantitative evaluation
    - Conclusion on merit of approach and effort which can guide future DoD decisions
    - Conclusions on parts of approach and effort

* Chance of Success
    - Reasonable chance of full success
    - Substantial residual benefits if not full success

* Direct Technology Insertion Effects
    - Significant number of organizations and persons in DoD community obtain experience and capability

* Direct Future Benefit to DoD
    - Important or costly (application) area to DoD
    - Existing or firmly planned future system can use products (optional)

Figure 3. Checklist and guideline for design and report for a demonstration of a combination of software

2.2.5 Subtask 5 -- Integration Over Application Areas

During the period of performance of Subtasks 2, 3, and 4, of this plan, several projects would be operating in parallel with similar objectives for different application areas. Uncoordinated, these projects would result in duplication of effort both for the software written and for the approaches developed.

- o Goal: The goal of this subtask should be to successfully monitor the other application area subtasks and provide an interchange of ideas between these areas. To meet this goal it would be necessary (a) to identify software that is common to several projects which should become generic software, (b) to prompt interchange of ideas about common approaches and (c) to solve common problems. For example, the Human Engineering Task Area would focus on identifying generic requirements for the system-user interface. Insights are expected to be gained as to commonality among various application area interface requirements. Periodic workshops or other means of coordination should be established. This subtask would continue in parallel with the others, commencing in FY84.

- o Description: This subtask has four phases.

- o Coordination: The main function of this subtask would be coordination.

- o Output: This subtask would produce periodic reports documenting the results of its coordination and monitoring activities. It would also produce evaluations of the quality of the products and the viability of the projects evolving from this task area.

- o Cost/Benefit: The effort for this subtask should be spread rather uniformly over the period, FY84 through FY89. The primary benefit derived from this subtask would be the elimination of considerable duplication of effort. A second important benefit would be the feedback provided to the STARS Program top management through the reports derived from the continuous monitoring of the progress of projects over the span of this plan. Finally, a small but still very worthwhile benefit would be derived from the subtask personnel who would be familiar with all projects but should be able to offer comments and direction as a party without a

vested interest in any one particular project. This subtask is Number 5 on the milestone chart, Figure 2.

2.2.5.1 Inter-project Communications. The subtask would develop mechanisms for the exchange of information and experience between the application area projects. A series of periodic workshops would be used as the prime means to achieve this exchange, but alternatives should be explored and used if shown to be more effective.

2.2.5.2 Standardization of Methodology. Application-specific projects would have a number of elements in common. Different groups would adopt different standards and guidelines for these common elements unless some specific and ongoing coordination takes place. The contractor would continuously monitor the application-specific projects in order to identify such common elements. Once identified, the different approaches should be evaluated and a reasonable consensus standard formulated for the common elements. Examples of probable common elements are: (a) procedures for classifying and identifying software parts, (b) mechanisms for warehousing the software parts inventory, (c) a common specific need for software technology.

2.2.5.3 Standardization of Generic Software. The elements common to application-specific projects would include various software parts, for example, graphics utilities, data base facilities, statistical and tabulation procedures and basic science procedures. Projects would be continuously monitored to identify common software elements and, once identified, a reasonable consensus specification should be made for these elements. This subtask would then decide which project is best suited for actually producing the resulting generic software. Such software might well also be produced as a part of the responsibility of another task area. It is not planned that operational software would be produced under this subtask, but

performance comparison, using either specifications or actual software, of duplicative software would be part of this subtask.

2.2.5.4 Progress Monitoring. Since this effort involves continuous monitoring of the application areas, it should assist in the evaluation of the progress of the projects, and it would provide data for the evaluations of the proposals considered at various stages of the program.

3.0 EFFORT FACTORS

The duration of each of the major subtasks is indicated by the milestone chart, Figure 2. Note that there are several contractors working in parallel except during the first subtask, the preparation of initial RFPs. The effort under this plan grows steadily until FY87 and then decreases. During the final three year period there are about six major software projects underway.

4.0 RESOURCES AND RELATED ACTIVITIES

This final section indicates relations with efforts and activities outside the DoD STARS Program. The three subsections list (a) forthcomming conferences and workshops, (b) general references and (c) related activities and projects.

4.1 Conferences and Workshops

1. ICS 83: International Computing Symposium on Application Systems Development,
   Nurnberg, W. Germany, 22-24 March 1983.

2. Workshop on Software Engineering Technology Transfer,
   Miami Beach, Florida, 22-27 April 1983.
   IEEE Computer Society

3. Tools, Methods and Languages for Scientific and Engineering Computation,
   Paris, France, 17-19 May 1983.
   Four sponsors, in cooperation with SIGNUM

4. Second Software Engineering Standards Application Workshop,
   San Francisco, 17-19 May 1983.

5. Softfair: Conference on Software Development Tools, Techniques and Alternatives,
   Washington, D.C., 26-28 July 1983.
   SIGSOFT and DoD Ada Joint Program Office

6. Reusability in Programming
   Newport, Rhode Island, Sept. 7-9, 1983
   ITT Programming

7. SAFECOMP 83: Third International Workshop on Achieving Safe Real Time Computer Systems,
   Cambridge, England, 20-22 September 1983.
   IFAC and IEEE

4.2 References

1. Department of Defense, "Application-Oriented Technologies and Reuse," in Strategy for a DoD Software Initiative, Vol. II: Appendices (DTIC/NTIS ADA 121738), 1 October 1982, pp. 186-200.

2. Gremillion, L. L., "Systems Development and Implementation Costs Using Standardized Application Systems," in R. E. Goldberg and H. Lorin (eds.), The Economics of Information Processing, Vol. 2, John Wiley and Sons, 1982, pp. 89-97.

3. Hasse, W. H., and G. R. Koch (eds.), Special Issue on "Application-Oriented Specifications," Computer, Vol. 15, No. 5, May 1982.

4. Martin, J., "Application Development Without Programmers," Prentice-Hall, 1982.

5. Mauro, P. A. and R. J. Morreale (Chairpersons), "Report on the Panel on Software Reusability," in Proceedings of the Joint Logistics Commanders Joint Policy Coordinating Group on Computer Resource Management, Computer Software Management Subgroup Second Software Workshop, 1 November 1981, Tab E.

6. Weisberg, L. R., "A New Approach to Lowering DoD Software Costs," Honeywell Aerospace and Defense Group, March 1982.

4.3 Other Related Activities

Application-specific software involves most aspects of software engineering; however, only the three activities that have the most direct impact on this plan are noted below.

4.3.1 *Software Projects for Application-specific Areas*

The most notable activity here is the effort of the American Statistical Association to organize, measure and improve statistical software. This elaborate effort ranges from measuring the efficiency and reliability of basic utilities to the aesthetics of very high level languages and application generators.

4.3.2 Classification, Warehousing and Retrieval Activities

There has been a long term effort in mathematical software to develop a flexible classification system. Instances of this are seen in the Assoc. Comput. Machinery (ACM) Algorithms classification and the classifications of the commercial libraries of IMSL, Inc. and

NAG, Ltd. These efforts show that a classification scheme is necessary, difficult to do well and, by itself, inadequate for warehousing and retrieval.

Other ideas being used or explored include KWIC and keyword indices, guide books (see the "Guide to Available Mathematical Software", Scientific Computing Division, National Bureau of Standards, internal report) and interactive inquiry systems (see "The NIT User's Manual", P. Gaffney et al, ORNL/CSD/INF-80/11, Oak Ridge National Laboratory.)

Given that one has located a potentially useful software part, one still must have its description from a program abstract, catalog or reference manual. Such documentation is notoriously difficult to prepare well; it must simultaneously be complete, be very compact, be easy to read and provide examples. There is an ANSI standard activity for program abstracts, and the major software libraries each have standard formats for this documentation. Some operating system reference manuals are collections of such descriptions arranged alphabetically.

4.3.3 Implementation of High Technology Facilities

The high technology facilities of very high level languages, application generators and knowledge based systems require extensive language processing and user interface support. Many such processors and interface environments must be created under this plan, and this must be done with reusable software and with borrowing from the experience of others in this area. Illustrative of the range of relevant ideas are compiler-compilers, syntax directed editors (or application knowledgeable editors) and segmented screen workstations. These implementation facilities are generic to the application-specific area; however, no detailed listing of activities are being

developed because of the broad areas of software engineering involved.

END

DATE
FILMED

6 — 83

DTIC